Date of Submission June 2024

Digital Object Identifier Not Applicable

A Study on the Implementation of Decision Tree Analysis

NIMESH GOPAL PRADHAN[1], RAMAN BHATTARAI[1]
[1]Department of Electronics and Computer Engineering, Thapathali Campus, Tribhuvan University, Kathmandu, Nepal

ABSTRACT Decision Tree Analysis is fundamental and widely-used in machine learning and data mining for solving classification and regression problems. It offers a structured approach to decision-making by outlining potential outcomes clearly. A decision tree consists of internal nodes representing features and leaf nodes denoting class labels or predictions. Through recursive partitioning based on informative features like Gini Index or Entropy, data is split until a stopping criterion is met. Decision trees excel in interpretability, handling of diverse feature types, and resilience to outliers. Moreover, decision trees are known for their ability to handle both numerical and categorical data effectively, making them versatile tools in practical data analysis. This report introduces Decision Tree Analysis, covering its theory, algorithms, and applications using the 'Differentiated Thyroid Cancer Recurrence' dataset. It validates prediction accuracy through preprocessing, model construction, and evaluation based on various criteria such as Gini Index and Entropy.

INDEX TERMS Classification, entropy, gini index, log loss, regression

I. INTRODUCTION

A Decision Tree is a supervised machine learning algorithm that is used for both classification and regression tasks. It is a powerful and understandable machine learning model that builds a tree-like structure by recursively splitting the data based on the most significant features to represent the relationships between features and the target variable. This recursive splitting process continues until a stopping criterion is met which results in a tree that can be used to make predictions for new instances. Each internal node represents a feature or attribute, each branch represents a decision rule, and each leaf node represents the outcome or target variable. Decision Trees are popular due to their simplicity and interpretability. They can handle both categorical and numerical data, and they are robust against outliers. Decision Trees can capture complex relationships between variables, including non-linear relationships. Decision Trees are robust to noisy data and capable of handling large datasets with numerous features. They can also manage both numerical and categorical variables effectively, and are insensitive to scaling and normalization. Despite their simplicity, Decision Trees can capture complex patterns and interactions in the data. However, they are prone to overfitting, especially when the tree depth is not properly controlled. To avoid overfitting, various pruning techniques can be applied, such as pre-pruning, which stops the tree from growing too deep, and post-pruning, which removes branches that do not provide additional power in prediction.

The construction of a Decision Tree begins with the selection of the most informative feature that best splits the dataset. This process is repeated recursively for each subset of data created by the splits until a stopping criterion is met, such as a maximum tree depth or a minimum number of data points in each leaf node. The metrics used for splitting nodes include Entropy, Gini Index, log loss, mean squared error and so on. During each split, the algorithm evaluates all potential features and thresholds, selecting the one that optimally reduces impurity or error. This ensures that the data is partitioned in a way that maximizes the homogeneity of the resulting nodes.

In classification tasks, Decision Trees use metrics

like Gini Index, Entropy, or Information Gain to determine the best feature to split the data. In regression tasks, they minimize variance within each split using techniques like mean squared error or mean absolute error, ensuring that the predicted values are as close as possible to the actual values. Basically, Decision Trees are versatile and widely used in various domains such as finance, healthcare, and marketing, where interpretability and ease of understanding the model's decisions are crucial.

In classification tasks, entropy measures the impurity or disorder of a set of examples, while information gain quantifies the reduction in entropy achieved by splitting the data on a particular feature. Decision Trees aim to maximize information gain, selecting features that best separate the classes or categories. Another metric used in classification, the Gini Index measures the impurity of a dataset. It is minimized by splitting the data into subsets where each subset predominantly contains examples from a single class. Log loss, also known as logistic loss or cross-entropy loss, is a performance metric typically used in binary and multiclass classification problems. It is used particularly in the context of probabilistic decision trees or when integrating decision trees with other models like Gradient Boosting Machines. It measures the performance of a classification model where the prediction is a probability value between 0 and 1. In regression tasks, Decision Trees use mean squared error (MSE) or mean absolute error (MAE) to evaluate the homogeneity of the target variable within each split. The split that results in the lowest MSE or MAE is chosen. By doing so, the decision tree aims to create subsets of data that are as pure as possible in terms of the target variable.

Decision Trees have limitations such as overfitting, bias towards features with many levels, and sensitivity to data variations. These issues can be mitigated through techniques like pruning and ensemble methods. Pruning, in particular, enhances the model's generalization to unseen data. The main pruning techniques include pre-pruning (early stopping), post-pruning, pessimistic pruning, cross-validation pruning and so on.

One common pre-pruning technique is setting a maximum depth for the tree, which makes the tree shallow. This involves training multiple decision tree models at different depths then calculating and comparing various evaluation metrics for each tree on validation sets to obtain the depth with the best performance on the validation set. This optimal depth balances bias and variance, thus avoiding both underfitting and overfitting.

For post-pruning, a highly effective method is cost complexity pruning (CCP). Initially, a decision tree is trained without any depth restriction to obtain effective alpha values and corresponding tree impurities. The effective alpha refers to the pruning parameter α that effectively balances the trade-off between tree complexity and its ability to generalize to new data. For each alpha value, a pruned tree is trained and evaluated on the validation set to identify the alpha that results in the best performance which is then used to retrain the tree on the entire training dataset, enhancing its performance and robustness.

II. RELATED WORKS

Decision Trees are widely recognized as a leading method for classifier representation. Experts in fields like statistics, machine learning, pattern recognition, and data mining have extensively explored techniques for constructing decision trees from given datasets. This paper "Decision Trees" provides a comprehensive review of contemporary methods for building decision tree classifiers using a top-down approach. It introduces a unified algorithmic framework for these techniques and discusses different splitting criteria such as gini index, information gain, gain ratio, chi-squared statistics, etc. and pruning strategies such as cost-complexity pruning, reduced error pruning, minimum error pruning ans so on. [1] Data mining has become integral to various fields. It involves discovering valid, novel, useful, and understandable patterns in raw data, also known as Knowledge Discovery in Databases. Among data mining techniques, classification is prominent for predicting categories within datasets. Decision trees, a key method in classification, help manage data dimensionality and predict group relationships. This study "A Study and Analysis of Decision Tree Based Classification algorithms using R" compares five decision tree algorithms: ID3, C4.5, C5.0, PART, and Bagging CART. Results indicate that Bagging CART achieves the highest classification accuracy. The evaluation uses precision, recall, F-measure, and kappa statistics. [2] This article "Understanding Decision Trees" is designed for beginners in Machine Learning who aim to grasp one of the simplest yet highly significant algorithms due to its interpretability, predictive power, and applications in various forms such as Random Forest and Gradient Boosting Trees. The article is divided into two parts: the first part focuses on setting up the dataset and model, while the second part delves into understanding the Decision Tree model. To clarify Decision Trees, the well-known iris dataset is utilized, with the gini index serving as the splitting criterion. [3] The decision tree segments a population into branch-like struc-

tures, forming an inverted tree with root, internal, and leaf nodes. As a non-parametric method, it efficiently handles large, complex datasets without requiring a complicated parametric framework. For large samples, data can be split into training and validation sets to build and refine the decision tree model. This paper "Decision tree methods: applications for classification and prediction" introduces common algorithms for developing decision trees (such as CART, C4.5, CHAID, and QUEST) and discusses the use of SPSS and SAS programs for visualizing tree structures. [4] Efficiency and scalability are major concerns in data mining for large databases. This paper "Decision Tree Induction: An Approach for Data Classification Using AVL-Tree", addresses these issues by proposing a data classification method using AVL trees, which improves quality and stability. Researchers in fields like statistics, machine learning, pattern recognition, and data mining have focused on building decision trees from available data. This study applies a multi-level mining method to the dataset, demonstrating how the proposed approach efficiently classifies large amounts of data at multiple levels. The results show improved performance of the proposed algorithm, which derives design rules from the knowledge database. [5] The study "Machine learning for risk stratification of thyroid cancer patients: a 15-year cohort study" aimed to develop machine learning models to predict recurrence risk in patients with well-differentiated thyroid cancer. Over a 15-year period, 383 patients were studied, assessing 13 clinicopathologic features. Various models, including Support Vector Machines, K-nearest neighbors, decision trees, random forest, and ANN, were trained on different feature sets. Results showed high sensitivity, specificity, and AUC for models using all features. Models excluding ATA risk data improved specificity but reduced sensitivity, whereas models trained solely on ATA risk data showed the opposite trend. These findings highlight the potential of machine learning to enhance recurrence risk stratification, guiding personalized treatment and follow-up strategies. [6] In her article, Shailey Dash explains the inner workings of decision trees, including the structure of root, sub, and leaf nodes, the splitting criteria like entropy, information gain, and Gini index, and the process of node splitting. She also discusses issues of overfitting and methods to prevent it, along with the simplicity of making predictions once the tree is trained and tested, emphasizing that the choice of impurity measurement can significantly impact the model's results. [7]

III. METHODOLOGY

A. DATASET INFORMATION

The dataset used in this paper is Differentiated Thyroid Cancer Recurrence dataset downloaded from kaggle. This data set was created aiming to predict recurrence of well differentiated thyroid cancer. The data set was collected in duration of 15 years and each patient was followed for at least 10 years. In this dataset there are 384 instances with 16 attributes and 1 target attribute. The sixteen attributes are Age, Gender, Smoking, Hx Smoking, Hx Radiothreapy, Thyroid Function, Physical Examination, Adenopathy, Pathology, Focality, Risk, T, N, M, Stage and Response while the target attribute, Recurred contains two values yes or no. The 384 instances of data is distributed as follows 275 instances of No and 108 instances of Yes. Some of the features are categorical in nature while some are continuous and there are no missing values.

B. WORKING MECHANISM

1) Preprocessing of Dataset

Before training the decision tree, the dataset was preprocessed to ensure quality and consistency. The dataset contained both categorical and continuous features, so specific preprocessing steps were preformed to handle each type appropriately.

a: Encoding Categorical Features

Categorical features were converted into numerical values using the `LabelEncoder` library from Python. This step involved assigning a unique integer to each category, enabling the decision tree algorithm to process the categorical data effectively. For instance, if a categorical feature like "Color" has values such as "Red", "Blue", and "Green", the `LabelEncoder` will transform these into integers like 0, 1, and 2, respectively.

b: Splitting the Dataset

The preprocessed dataset should then be split into training and testing sets using an appropriate ratio in this work a ratio of 80:20 was used. This split must ensures that the majority of the data is used for training the model, while a smaller portion is reserved for evaluating its performance. If the class distribution is imbalanced, the `stratify` parameter need to be utilized during the splitting process. This parameter ensures that the class proportions in the training and testing sets mirror those in the original dataset, thereby maintaining the balance of classes across the splits.

2) Decision Tree

A decision tree is structured like an inverted tree with a root node at the top, several internal nodes,

branches connecting parent and child nodes, and leaf nodes at the ends. The root node marks the start of the decision tree. Each internal node features a splitting predicate, branches illustrate the results of these tests, and leaf nodes signify class labels.

The fundamental algorithm for building a decision tree is a greedy, top-down, recursive divide-and-conquer approach. During construction, the training dataset is progressively partitioned into smaller subsets. Initially, the tree consists of a single root node representing the entire training dataset. Then a root attribute is selected from the dataset, known as the split attribute. A branch is created for each value of the node attribute, labeled accordingly, and the dataset is partitioned based on these values. This process is then recursively applied to each partition to form the decision tree. Once an attribute is used in a node, it is excluded from consideration in any of that node's descendants.

a: Hunt's Algorithm

Hunt's algorithm is a widely used method for constructing decision trees. It operates recursively to partition the dataset based on attribute values, creating a tree structure where internal nodes represent decisions and leaf nodes represent outcomes. The algorithm is defined as follows:

1) **Initialization:**
   - Begin with the entire dataset D.
2) **Termination Condition:**
   - If all instances in D belong to the same class, then the current node is a leaf node labeled with that class.
   - If D is empty, then the current node is a leaf node labeled with the majority class of the parent node.
   - If there are no remaining attributes to split on, then the current node is a leaf node labeled with the majority class of the instances in D.
3) **Recursive Partitioning:**
   - Select the best attribute A to split the data using a splitting criterion (e.g., Information Gain or Gini Impurity).
   - For each value v of attribute A:
     a) Create a child node.
     b) Partition the instances in D into subsets D_v where D_v contains instances with $A = v$.
     c) Recursively apply Hunt's algorithm to each subset D_v.

The algorithm ensures that the dataset is divided into increasingly homogeneous subsets, ultimately leading to a decision tree that can be used for classification or regression tasks. The recursive nature of Hunt's algorithm allows it to handle datasets with various types of attributes and produce interpretable tree structures.

3) Splitting Criterion

In decision trees, the 'Splitting Criterion' serves as the compass guiding how our tree makes decisions. A tree's structure is influenced by the splitting method used. In a multi-way split, a parent node can have more than two children nodes, whereas in a binary split, each parent node can only have two children, dividing the dataset into just two subsets. Selecting appropriate root attributes and internal nodes is critical in decision tree creation. Poor choices can lead to an excessively deep tree, increasing complexity, computational intensity, and processing time. Proper selection can be achieved by effectively using the splitting criteria discussed in the following sections.

a: Entropy

Entropy is a measure of the uncertainty or impurity in a dataset. It quantifies the amount of disorder or randomness present, which helps in determining how mixed the class labels are at a node in a decision tree. It is widely employed as a criterion for splitting nodes in decision trees to assess the uniformity of class labels and identify the optimal attribute for data partitioning. The metric indicates how effectively an attribute can differentiate between various target classifications. Lower entropy indicates greater information content, whereas higher entropy indicates more disorder. Information gain quantifies the expected reduction in dataset entropy resulting from a split. An entropy value of 1 indicates complete disorder, while a value of 0 denotes perfect homogeneity in the dataset. Entropy $H(D)$ of a dataset D with c classes is given in Equation 1

b: Gini Index

The Gini Index, also known as Gini Impurity, is a measure of the impurity or uncertainty in a dataset. It is commonly used as a splitting criterion in decision trees to determine the best attribute for partitioning the data. The formula for the Gini Index $G(D)$ of a dataset D with c classes is given in Equation 2 The Gini Index helps quantify the level of impurity or disorder in the dataset. A lower Gini Index indicates higher purity, while a higher Gini Index indicates greater impurity.

c: Classification Error

Classification Error is a measure of the misclassification rate at a node. It represents the proportion of incorrect predictions made by a classifier. The formula for Classification Error $E(D)$ of a dataset D with c classes is given in Equation 3

d: Information Gain

Information Gain measures the reduction in entropy or impurity achieved by splitting the dataset based on an attribute. It is a commonly used criterion for selecting the best attribute for partitioning the data in decision trees. The formula for Information Gain $IG(D, A)$ of an attribute A in a dataset D is given in Equation 4

While Information Gain is useful, it tends to favor attributes with a large number of distinct values. Gain Ratio is an enhancement that addresses this bias by normalizing Information Gain using Split Information.

e: Split Information

Split Information $SI(D, A)$ measures the potential information generated by splitting the dataset D based on attribute A. The formula for Split Information is given in Equation 5

f: Gain Ratio

The Gain Ratio $GR(D, A)$ adjusts the Information Gain by the Split Information, providing a more balanced measure for attribute selection. The formula for Gain Ratio is given in Equation 6

4) Performance Metrics

When evaluating the effectiveness of a decision tree classifier, it's important to consider various performance metrics that provide various insights. These metrics help in understanding the strengths and weaknesses of the model in different contexts.

a: Accuracy

Accuracy is the ratio of correctly predicted observations to the total observations. It provides the overall effectiveness of the model and is given by the formula in Equation 7

However, accuracy can be misleading in the case of imbalanced datasets. When one class significantly outnumbers the other, a high accuracy may simply reflect the model's ability to predict the majority class correctly, while completely ignoring the minority class. For example, in a dataset where 95% of the instances are of one class, a model that predicts all instances as this majority class will achieve 95% accuracy, despite failing to identify any instances of the minority class. This highlights the need for additional metrics such as Precision, Recall, and F1 Score, which provide a more balanced evaluation of model performance, especially in imbalanced datasets.

b: Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positives. It answers the question: "Of all the instances that were predicted as positive, how many were actually positive?" Precision is given by the formula in Equation 8

c: Recall

Recall, also known as Sensitivity or True Positive Rate, is the ratio of correctly predicted positive observations to all the observations in the actual class. It answers the question: "Of all the instances that are actually positive, how many were predicted correctly?" Recall is given by the formula in Equation 9

d: F1 Score

The F1 Score is the harmonic mean of Precision and Recall. It provides a balance between precision and recall, making it a useful measure when both are important and need to be considered together. It is particularly useful when the class distribution is imbalanced. The formula for F1 Score is in Equation 10

The harmonic mean is used instead of the arithmetic or geometric mean because it effectively balances Precision and Recall, giving more weight to the smaller value. This ensures that the F1 Score is high only when both Precision and Recall are high, making it a robust measure for evaluating classification performance, especially in cases with imbalanced data.

C. PRUNING OF DECISION TREES

Pruning is a crucial technique used in decision tree algorithms to prevent overfitting and enhance the generalization capabilities of the model. By simplifying the tree structure, pruning helps in improving the model's performance on unseen data.

1) Pre-pruning

Pre-pruning, also known as early stopping, involves stopping the growth of the decision tree at an early stage before it reaches its maximum depth. This can be achieved through various strategies, such as setting a maximum depth for the tree, specifying a minimum number of samples required to split a node, or requiring a minimum number of samples in a leaf node. These constraints help in prevent-

ing the tree from becoming too complex and thus reduce the risk of overfitting.

- **Maximum Depth:** Restricting the maximum depth of the tree ensures that the tree does not grow too deep, thus limiting the complexity.
- **Minimum Samples Split:** This parameter sets the minimum number of samples required to split an internal node. If a node has fewer samples than this threshold, it is not split further.
- **Minimum Samples Leaf:** This parameter sets the minimum number of samples that a leaf node must have. Nodes with fewer samples than this threshold are not split.

Pre-pruning can significantly reduce the size of the tree, making it faster and more efficient while maintaining a balance between model complexity and performance.

a: Problems with Pre-pruning

Despite its advantages, pre-pruning has some potential drawbacks:

- **Underfitting:** Pre-pruning may stop the growth of the tree too early, leading to underfitting. The model may not capture all the underlying patterns in the data, resulting in poor predictive performance.
- **Difficulty in Setting Thresholds:** Determining the appropriate thresholds for parameters like maximum depth, minimum samples split, and minimum samples leaf can be challenging. Incorrect thresholds can either lead to an overly complex tree or an overly simplistic one.
- **Loss of Important Splits:** Important splits that could enhance the model's predictive accuracy might be missed because of the predefined stopping criteria.

Careful consideration and tuning of parameters should be done when employing pre-pruning to ensure that the model is both efficient and effective.

2) Post-pruning

Post-pruning, also known as late pruning, involves first growing the decision tree to its full depth and then removing nodes that do not provide significant power in predicting the target variable. This method allows the tree to capture the patterns in the training data fully and then prunes the less significant branches to enhance generalization.

- **Reduced Error Pruning:** This method involves removing nodes that lead to the least increase in error rate on a validation set.
- **Cost Complexity Pruning (CCP):** Also known as weakest link pruning, this method involves removing nodes in a way that minimizes the cost complexity measure, which is a trade-off between the size of the tree and its accuracy on the training data.

a: Identifying the Weakest Link

In Cost Complexity Pruning (CCP), the weakest link refers to the subtree or node that, when pruned, leads to the smallest increase in the cost complexity measure. The cost complexity measure $R_\alpha(T)$ is defined in Equation 11 The weakest link is identified by finding the subtree T_t whose removal results in the smallest increase in the cost complexity measure. Formally, for each subtree T_t Equation 12 is calculated. The subtree with the smallest $\Delta R_\alpha(T_t)$ is considered the weakest link and is pruned. This process is repeated until further pruning does not result in a decrease in the overall cost complexity.

Post-pruning usually results in a more accurate and simpler model, as it removes the branches that have little importance in predicting the target variable, thus reducing the risk of overfitting and improving the model's performance on unseen data.

b: Problems with Post-pruning

While post-pruning addresses some of the issues associated with pre-pruning, it has its own potential drawbacks:

- **Computational Complexity:** Growing the tree to its full depth before pruning can be computationally expensive, especially with large datasets.
- **Complexity of Pruning Process:** The process of identifying the weakest links and pruning them can be complex and time-consuming.
- **Dependence on Validation Set:** Post-pruning often relies on a validation set to evaluate the effect of pruning. If the validation set is not representative of the test data, the pruning process might not yield optimal results.

These challenges indicate that while post-pruning can lead to a more generalized model, it requires careful implementation and consideration of computational resources.

D. BLOCK DIAGRAM

to be doneThe Decision Tree Analysis system block diagram, illustrated in Figure 1, outlines the process for designing and utilizing decision trees to predict outcomes. It involves three primary stages: data preprocessing, decision tree design, and result prediction. Initially, the dataset undergoes preprocessing to ensure cleanliness and formatting. It includes handling missing values, encoding categorical features, and selecting relevant features in the dataset. Afterwards, the dataset is split into training and testing sets. The training set is utilized to iteratively construct the decision tree by selecting the

optimal attribute based on metrics such as entropy, information gain, Gini index, or log loss. This attribute partitions the dataset into smaller subsets until either a pure subset is achieved or a predefined stopping criteria is met, resulting in leaf nodes. The trained model is then applied to predict outcomes for the testing dataset, and evaluation metrics such as accuracy, recall, precision, and F1 score are computed using the predicted value and ground truth values. To mitigate overfitting in decision tree models, pruning techniques such as pre-pruning, post-pruning, pessimistic pruning, cross-validation pruning, and others are applied. Following pruning, the decision tree model is reconstructed using the refined dataset.Finally, the decision tree designed from the training dataset is constructed based on this process, enabling robust predictive modeling.

E. FLOWCHART

to be doneThe flowchart depicted in Figure 3, illustrates the sequential steps involved in constructing and analyzing a dataset using a decision tree. Initially, the dataset undergoes thorough cleaning and formatting to ensure reliability, addressing issues such as handling missing values, encoding categorical features, and selecting influential attributes crucial for the model's predictive accuracy. Following this, the dataset is partitioned into distinct training and testing sets, essential for validating the model's efficacy. A chosen metric such as entropy, Gini index, or log loss guides the creation of the decision tree, evaluating attribute potential and guiding effective dataset partitioning. The decision tree then iteratively selects optimal attributes to partition the dataset into smaller, manageable subsets, where each leaf node represents the majority class of its subset or meets predefined criteria to finalize leaf nodes. This recursive process continues until the entire decision tree is fully constructed and capable of predicting target classes for new data instances in the testing dataset. Following model construction, performance evaluation metrics such as accuracy, recall, precision, and F1 score are computed to assess the model's effectiveness against actual values. To ensure optimal performance, the next step involves assessing for signs of overfitting, which can degrade model accuracy. If overfitting is detected, the model undergoes pruning using techniques such as pre-pruning, post-pruning, pessimistic pruning, or cross-validation pruning. These methods refine the model by reducing complexity and enhancing its ability to generalize to new data. Once pruned, the decision tree is reconstructed using the optimized dataset, resulting in an optimal model that accurately predicts outcomes. Finally, the completed decision tree derived from the training dataset is prepared for real-world applications, providing valuable insights and predictions based on new data inputs.

IV. IMPLEMENTATION DETAILS AND RESULTS

The Differentiated Thyroid Cancer Recurrence dataset was analyzed, revealing an imbalanced class distribution as shown in Figure 2, with more instances labeled as "No" compared to "Yes." The dataset was free of empty or null values and consisted of both continuous and categorical features. Categorical data were encoded, and the dataset was split into training and testing sets.

A decision tree classifier was initialized with entropy as the splitting criterion. The resulting tree, illustrated in Figure 4, has a depth of 8. Each node in the tree represents various information the splitting condition, entropy for that split, the number of training samples at the node, the distribution of class labels, and the predicted class.

The confusion matrix, depicted in Figure 6, shows 73 correct predictions and 4 incorrect predictions. The classification report and performance metrics are detailed in Table 1.

Early stopping was applied next, and F1 scores for various tree depths were calculated, plotted in Figure 7. The optimal depth of 4 achieved an F1 score of 0.9333 on the testing dataset. A decision tree of depth 4 is shown in Figure 5, with its corresponding confusion matrix in Figure 8. After pruning the number of correct prediction was 74 and the number of incorrect prediction was 3.

Post-pruning techniques were then employed. The impurity for various alpha values was plotted in Figure 9, the relationship between number of nodes and effective alpha is shown in Figure 10, and the F1 score for different alpha values is displayed in Figure 11. The optimal alpha value of 0.0208 was determined. A pruned tree with depth 3 using this alpha is depicted in Figure 16, with its confusion matrix in Figure 12. The number of correct prediction was 75 and the number of incorrect prediction was found to be 2.

When training the decision tree with default parameter the F1 score was found to be 0.904 then on using early stopping by limiting the depth of the tree the F1 score went up to 0.933. Similarly, using post pruning technique the F1 score went up to 0.952.

Then a decision tree classifier was instantiated with the Gini Index as the splitting criterion. The resulting decision tree, shown in Figure 17, was found to have depth of 10. Each node in the tree represents critical information: the condition for splitting, Gini impurity for that split, the number

of training samples at the node, the distribution of class labels, and the predicted class.

The confusion matrix for the decision tree using Gini Index is shown in Figure 13, the number of correct classification was 74 and the incorrect classification was 3. The classification report and detailed performance metrics for the Gini Index criterion can be found in Table 2.

Early stopping technique was then applied the f1 score at various levels of depth can be seen on Figure 14. The optimum value of depth was found to be 3. A new decision tree having this depth value was initialized and the corresponding confusion matrix is shown in Figure 15. The number of correct prediction was 75 and the number of incorrect prediction was 2.

Then, Post-pruning techniques were applied. The impurity values for various alpha thresholds were plotted in Figure 20, and the relationship between the number of nodes and effective alpha is displayed in Figure 21. Additionally, the F1 scores corresponding to different alpha values are presented in Figure 22. The optimal alpha value of 0.0063 was identified. A pruned decision tree with a depth of 4 using this optimal alpha value is shown in Figure 19, along with its confusion matrix in Figure 23. The number of correct prediction was 75 and the number of incorrect prediction was 2.

When training the decision tree with default parameter the F1 score was found to be 0.933 then on using early stopping by limiting the depth of the tree the F1 score went up to 0.952. Similarly, using post pruning technique the F1 score went up to 0.952.

V. DISCUSSION AND ANALYSIS

The class distribution analysis of the Differentiated Thyroid Cancer Recurrence Dataset, as depicted in Figure 2, reveals a significant class imbalance. This necessitates the use of the stratify parameter when splitting the dataset into training and testing sets. By employing the stratify parameter, we ensure that the distribution of classes in the training and testing sets mirrors that of the original dataset.

Figure 4 illustrates the decision tree generated using entropy without setting a depth limit. The tree extends to a depth of 8. Each node in the tree provides essential information, including the splitting attribute, entropy value, number of samples, class distribution, and predicted class. The predicted class is displayed at every node, including the root. This preemptive assignment is based on the majority classes at that particular node and aids in scenarios where the tree may undergo early pruning, ensuring that samples reaching each node are appropriately classified. When not setting any depth limit the tree gets splitted so that every leaf node has an entropy value of 0. The tree is very accurate only misclassifying 4 out of 77 samples of the test dataset. The F1 score is also very high at 0.904 which means the tree has achieved a balanced of recall and precision values.

Taking a look at 7 we see that the highest F1 score is not the highest at depth of 8. In fact, the tree performs better at depths of 1,3,4,5 and 6 with the best performance being at a depth of 4. This result may be attributed to the overfitting, at a depth of 8 the tree likely becomes overly complex, capturing noise and specific details of the training data that do not generalize well to new, unseen data. Shallower depths, on the other hand, tend to create simpler decision boundaries that generalize better by focusing on broader patterns and relationships in the data. This simplicity reduces the risk of overfitting, resulting in better performance on the testing set where the tree's predictions are more robust and less sensitive to variations in the training data. This, shows the importance of balancing the model complexity and generalization capability, especially in datasets where deeper tree may not provide significant benefits and can even lead to reduced performance due to overfitting. Also, the decision tree at optimal depth shown in Figure 5 is just the subset of the original decision tree where the F1 score is 0.933.

Taking a look at Figure 9 we can observe that as the value of effective alpha is increased the total impurity in leaves also increase. These two terms have a positive correlation. Effective alpha represents a threshold where nodes with impurities above this value are considered for pruning. A higher effective value of alpha indicates a higher threshold for pruning, meaning nodes with higher impurity and potentially less contribution to overall accuracy are retained in the pruned tree. Therefore, as effective alpha increases, nodes with higher impurity levels are allowed to remain in the tree rather than being pruned. This results in a higher total impurity across the leaves of the pruned tree.

Figure 10 shows the plot of number of nodes and depth vs effective alpha we can observe that as the value of effective alpha is increased both the max depth and the number of nodes is increased. As alpha increases, more nodes with higher impurity or less significance are pruned. This reduction in nodes directly decreases the total number of nodes and the depth in the pruned tree.

Figure 11 shows the values of F1 scores at various alpha levels. The optimal alpha value where the F1 score for testing set was the highest was found to be 0.0208. The decision tree using this alpha value is shown in Figure 16 and its F1 score

was 0.952 with depth of 3. The optimal depth of 4 was determined by evaluating the F1 score at various depths, controlling the maximum depth of the tree. However, when using cost-complexity pruning with and effective alpha value of 0.0208, the tree's depth reduced to 3, resulting in a higher F1 score of 0.953. This improvement might have occurred because cost-complexity pruning not only limits the depth but also selectively removes less significant branches. The pruned tree despite being shallower, is more effctive because it balances model complexity and performance more precisely than simply setting depth limit. Also the optimal depth was calculated without changing any other parameter and only checking the F1 scores at various depth where the alpha value was the default alpha value. If the optimal depth was found for taking alpha 0.0208 then the optimal depth might have been 3 instead of 4.

Figure 6, Figure 8 and Figure 12 shows the confusion matrix of original decision tree, decision tree with optimal depth and the decision tree with optimal value of effective alpha respectively. The predicted label is along the x-axis and the actual label is along the y-axis. The diagonal values represent the number of samples where the predicted and actual class label are same and the non-diagonal values represent the wrong prediction. Table 1 presents the classification report for the trained model, detailing precision, recall, and F1 score. These metrics are provided for each class, considering both "No" and "Yes" as the positive class in turn. Additionally, the report includes accuracy, support, macro average, and weighted average values, offering a comprehensive evaluation of the model's performance. we can observe that the model has high metric scores even without any pruning which might be due to the dataset begin well-separated or due to the imbalance present in the dataset. Since the class No is significantly larger it could cause the model to appear to perform well due to it accurately predicting the majority class, which dominates the evaluation metric.

The decision tree was trained using Gini as the splitting criteria, the insights gained from the various plots and tables obtained when training the decision tree with Gini Index as the splitting criterion are similar to those discussed above for the entropy criteron. The analysis of class distribution, decision tree structure, and performance metrics, including the effects of early stopping and post-pruning techniques show consistent patterns. Specifically, the F1 score optimization, impurity versus alpha correlation and the reduction in tree complexity with increased alpha values reflect comparable trends. This consistency indicates that both splitting criteria provide robust model for this dataset.

Some differences can be observed when using the Gini Index as the splitting criterion. At a depth of 1, node 38, the Gini decision tree has used the Thyroid Function attribute as the splitting attribute, while the Entropy decision tree has selected the pathology attribute. This difference arises because the Gini Index and Entropy criteria prioritize splits differently based on their respective impurity measures. The Gini Index tends to create splits that maximize the purity of the resultant nodes by focusing on the most significant decrease in impurity, whereas Entropy focuses on maximizing information gain. Additionally, the depth of the tree using Gini is 10 compared to 8 when using Entropy, indicating that the Gini criterion may lead to more granular splits. The slightly higher F1 score with Gini suggests that the tree built using Gini might be capturing more nuances in the data, leading to marginally better classification performance. This could be because the Gini Index can be more sensitive to class distributions and better at handling imbalanced datasets, resulting in finer, more effective splits for this specific dataset

VI. CONCLUSION

In conclusion, the application of Decision Tree analysis on the 'Differentiated Thyroid Cancer Recurrence' dataset has provided valuable insights into the behavior and performance of the models. By implementing the decision tree algorithm, we successfully constructed a model that accurately classified instances based on the provided features. The obtained results highlight the predictive capabilities and effectiveness of the Decision Tree models in forecasting thyroid cancer recurrence in patients. Performance metrics such as accuracy, precision, recall, and F1 scores demonstrated the model's effectiveness in predicting the outcomes of future data, indicating a balanced between precision and recall, and suggesting satisfactory predictive capability of the model.

The high accuracy values reflect a significant level of correct predictions, while the precision and recall values further emphasize the model's ability to accurately identify and capture positive instances from the dataset. The confusion matrices offered additional insights into the model's performance, identifying potential areas for enhancement. Analyzing the decision trees enabled the interpretation of influential features and the identification of patterns and rules contributing to thyroid cancer recurrence.

Dataset pruning was implemented to mitigate overfitting. This technique effectively demonstrates the improved performance of the pruned

model in predicting classes. The pruned model exhibits satisfactory performance metrics when evaluated on testing data, highlighting its effectiveness in managing overfitting.

Decision trees present several advantages, including simplicity in understanding and interpretation, minimal data preparation requirements, and the ability to handle both numerical and categorical data. They are also effective in managing multi-output problems and provide a transparent, white-box model that explains decisions with boolean logic. They are efficient in prediction, with a cost that is logarithmic in the number of data points used to train the tree.

However, decision trees also have notable disadvantages. They can create over-complex trees that do not generalize well, known as overfitting, which necessitates mechanisms such as pruning to avoid. Decision trees can be unstable, with small variations in data potentially resulting in significantly different trees. Predictions are piecewise constant approximations and not smooth or continuous, making them poor at extrapolation. Additionally, decision trees can create biased trees if some classes dominate, so it is always better to balance the dataset prior to fitting.

This study showcases the application of decision trees in classification tasks, affirming their viability and effectiveness in predicting thyroid cancer recurrence. Future enhancements may include exploring ensemble methods to further improve the model's predictive power and robustness. Despite the limitations, decision trees remain a powerful tool in the field od machine learning, offering valuable insights and reliable performance in a variety of domains.

APPENDIX

A. EQUATIONS

1) Entropy

$$H(D) = -\sum_{i=1}^{c} p_i \log_2(p_i) \tag{1}$$

where:

- c is the number of classes in the dataset.
- p_i is the proportion of instances in class i, calculated as $p_i = \frac{|D_i|}{|D|}$, where $|D_i|$ is the number of instances in class i and $|D|$ is the total number of instances in the dataset.
- $\log_2$ denotes the logarithm to the base 2.

2) Gini Index

$$G(D) = 1 - \sum_{i=1}^{c} p_i^2 \tag{2}$$

3) Classification Error

$$E(D) = 1 - \max(p_i) \tag{3}$$

4) Information Gain

$$IG(D,A) = H(D) - \sum_{v \in \text{values}(A)} \frac{|D_v|}{|D|} H(D_v) \tag{4}$$

where:

- $IG(D,A)$ is the Information Gain of attribute A in the dataset D.
- $H(D)$ is the entropy of the entire dataset D.
- v represents each value of attribute A.
- D_v is the subset of the dataset where attribute A has value v.
- $|D_v|$ is the number of instances in subset D_v.
- $|D|$ is the total number of instances in the dataset.
- $H(D_v)$ is the entropy of subset D_v.

5) Split Information

$$SI(D,A) = -\sum_{v \in \text{values}(A)} \frac{|D_v|}{|D|} \log_2 \left(\frac{|D_v|}{|D|} \right) \tag{5}$$

where:

- $SI(D,A)$ is the Split Information of attribute A in the dataset D.
- v represents each value of attribute A.
- D_v is the subset of the dataset where attribute A has value v.
- $|D_v|$ is the number of instances in subset D_v.
- $|D|$ is the total number of instances in the dataset.

6) Gain Ratio

$$GR(D,A) = \frac{IG(D,A)}{SI(D,A)} \tag{6}$$

where:

- $GR(D,A)$ is the Gain Ratio of attribute A in the dataset D.
- $IG(D,A)$ is the Information Gain of attribute A.
- $SI(D,A)$ is the Split Information of attribute A.

7) Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

where:

- TP (True Positives) are the correctly predicted positive instances.
- TN (True Negatives) are the correctly predicted negative instances.
- FP (False Positives) are the incorrectly predicted positive instances.
- FN (False Negatives) are the incorrectly predicted negative instances.

8) Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

9) Recall

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

10) F1 Score

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

11) Cost-Complexity Pruning

$$R_\alpha(T) = R(T) + \alpha \cdot |T| \tag{11}$$

where:

- $R(T)$ is the error rate of the tree T.
- $|T|$ is the number of terminal nodes (leaves) in the tree.
- α is a complexity parameter that controls the trade-off between the tree's complexity and its accuracy.

$$\Delta R_\alpha(T_t) = \frac{R(T_t) - R(T_t')}{|T_t| - 1} \tag{12}$$

where T_t' is the tree after pruning subtree T_t.

B. FIGURES

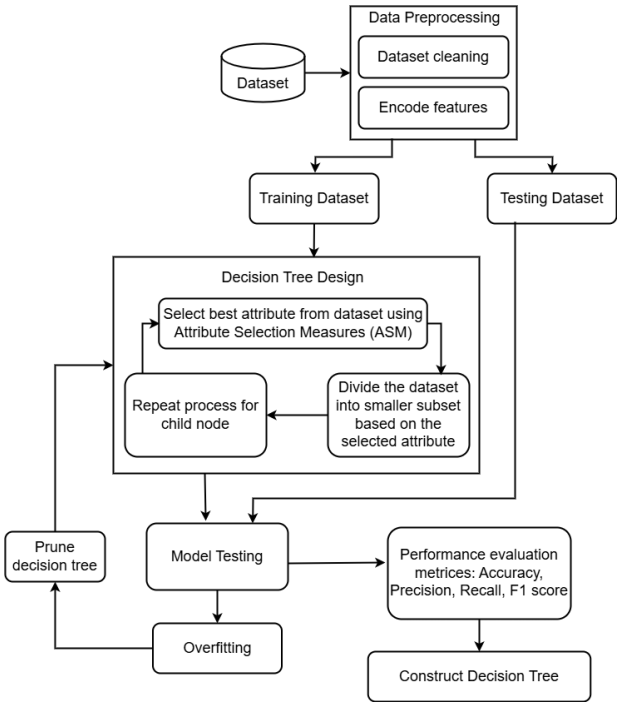

FIGURE 1. Block Diagram

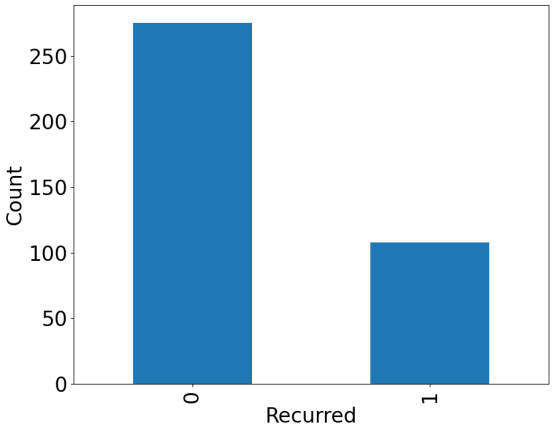

FIGURE 2. Class Distribution of Dataset

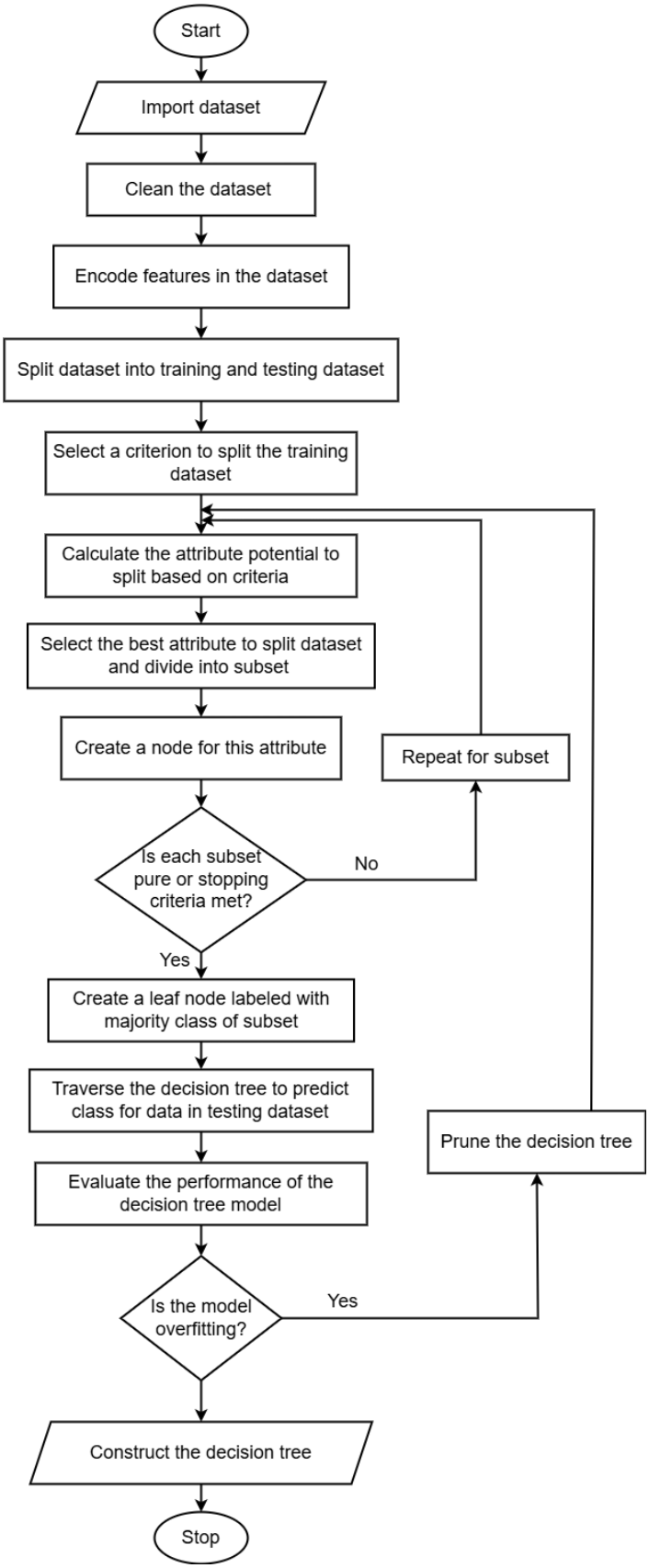

FIGURE 3. **Flowchart**

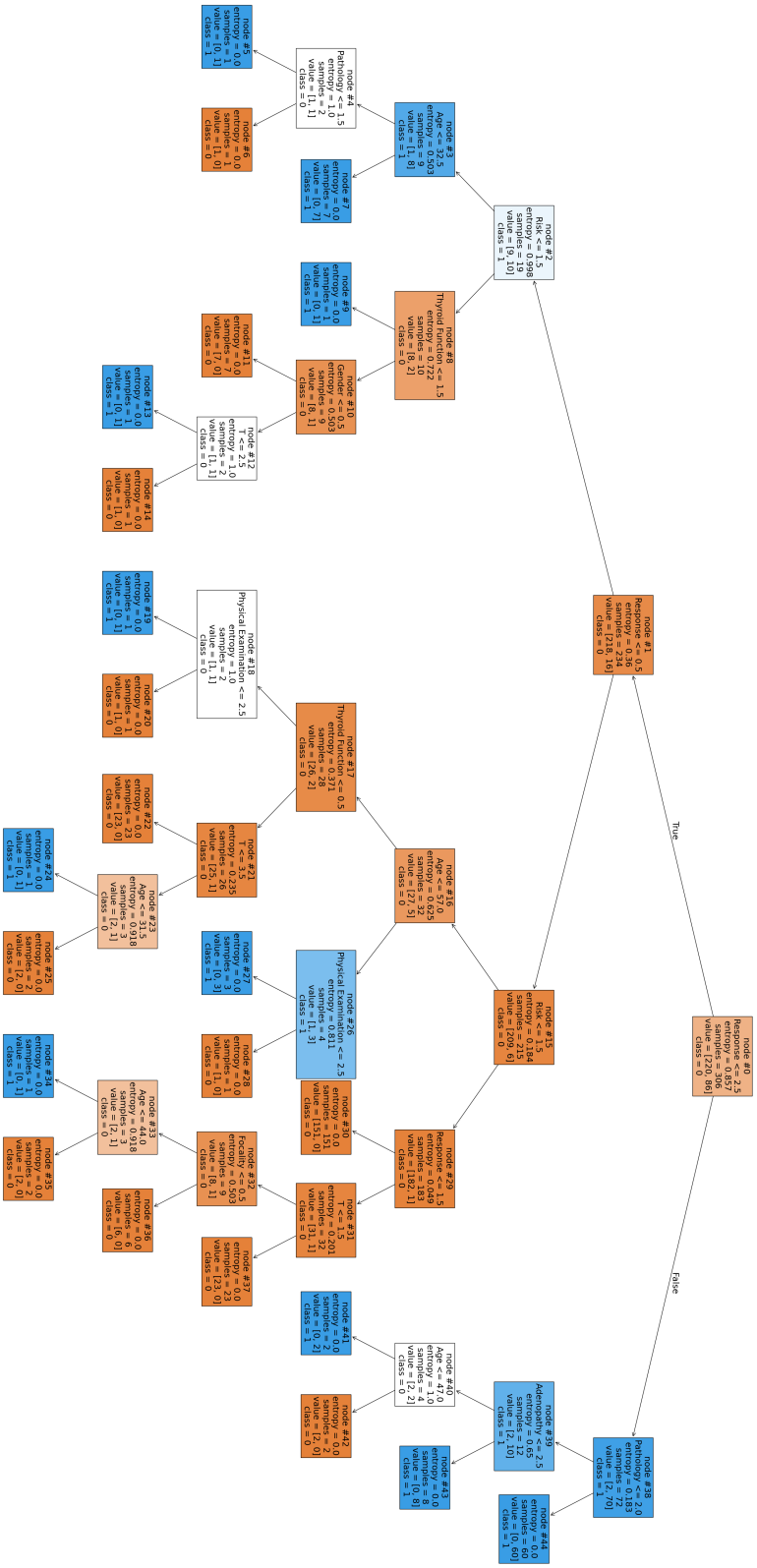

FIGURE 4. Decision Tree with Entropy as Splitting Criteria

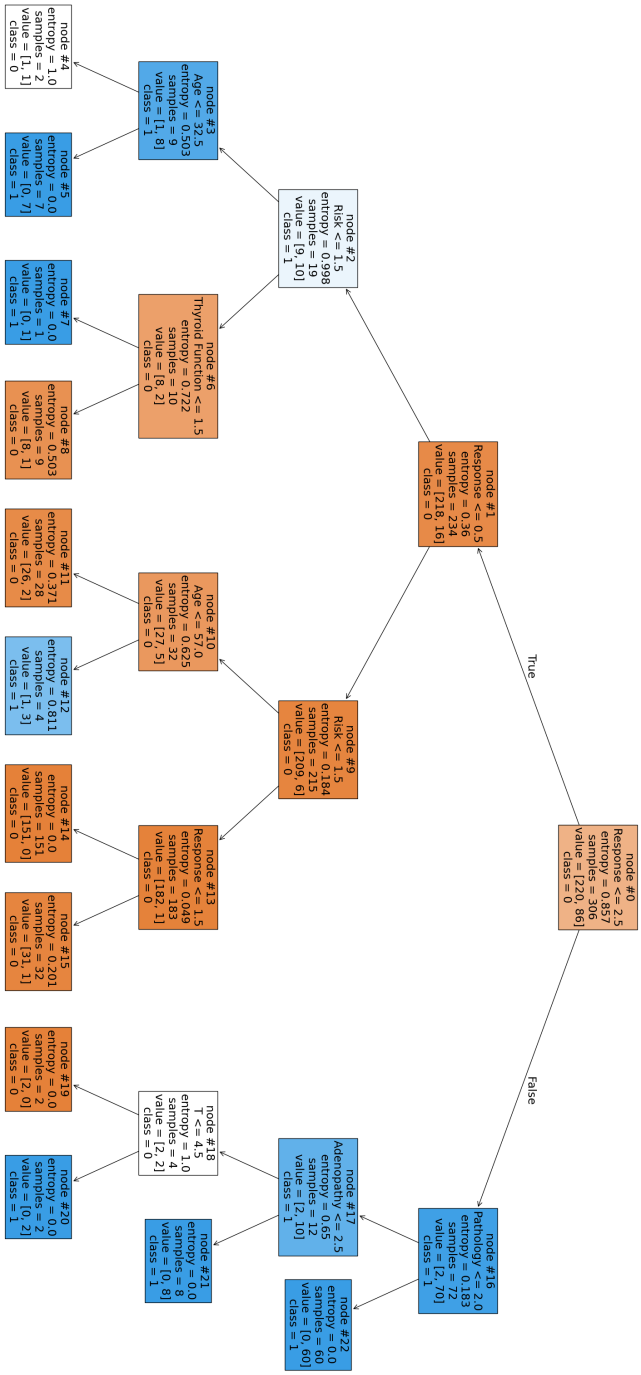

FIGURE 5. **Decision Tree with Depth 4 using Entropy**

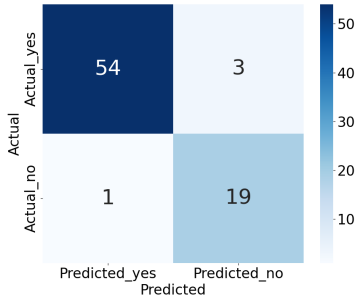

FIGURE 6. Confusion Matrix with Entropy as Splitting Criteria

Class	Precision	Recall	F1-Score	Support
No	0.95	0.98	0.96	55
Yes	0.95	0.86	0.90	22
Accuracy			0.95	77
Macro Avg	0.95	0.92	0.93	77
Weighted Avg	0.95	0.95	0.95	77

TABLE 1. Classification Report using Entropy

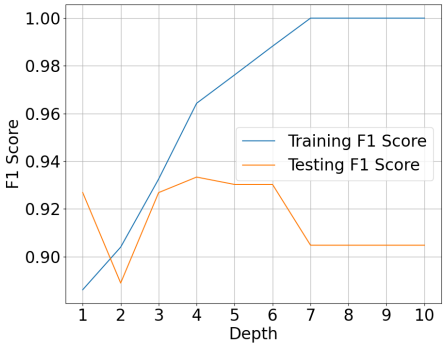

FIGURE 7. F1 Score vs Depth using Entropy

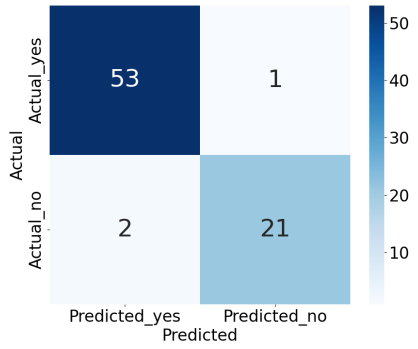

FIGURE 8. Confusion Matrix using Entropy and Depth 4

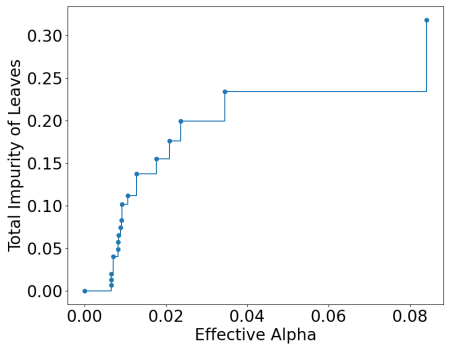

FIGURE 9. Total Impurity vs Effective Alpha using Entropy

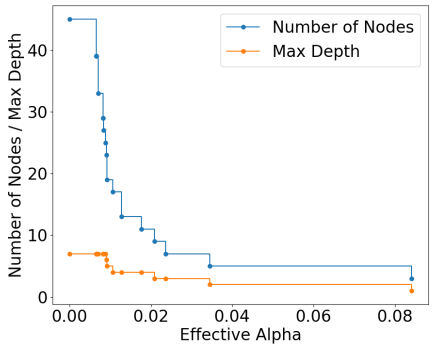

FIGURE 10. Number of Nodes and Depth vs Effective Alpha using Entropy

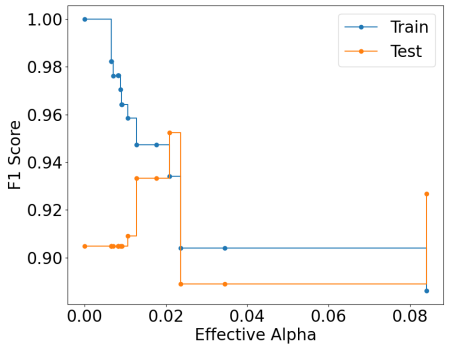

FIGURE 11. F1 Score vs Effective Alpha using Entropy

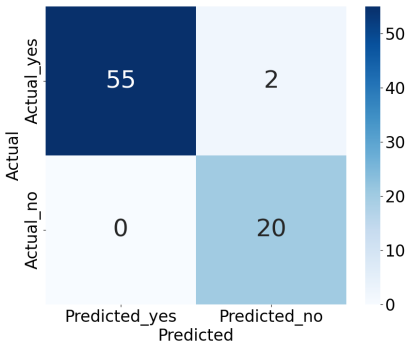

FIGURE 12. Confusion Matrix using Entropy and Optimal Alpha

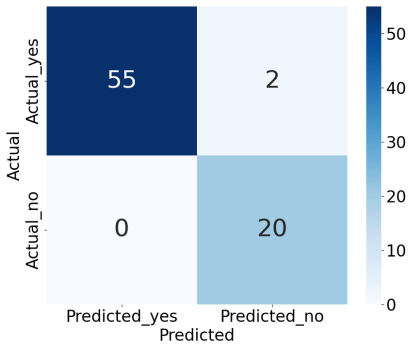

FIGURE 15. Confusion Matrix using Gini and Depth 3

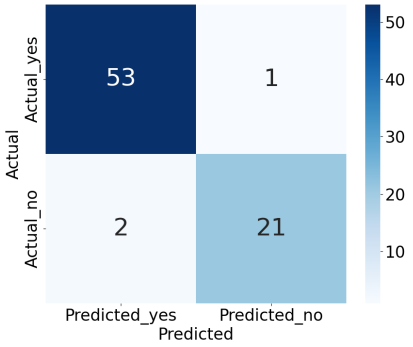

FIGURE 13. Confusion Matrix with Gini as Splitting Criteria

Class	Precision	Recall	F1-Score	Support
No	0.98	0.96	0.97	55
Yes	0.91	0.95	0.93	22
Accuracy			0.96	77
Macro Avg	0.95	0.96	0.95	77
Weighted Avg	0.96	0.96	0.96	77

TABLE 2. Classification Report using Gini

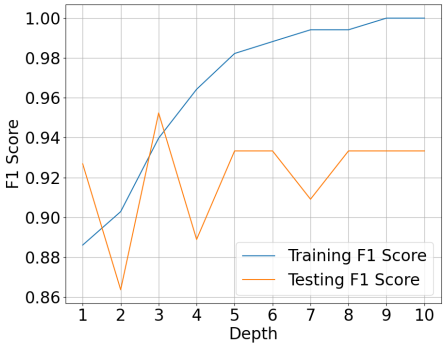

FIGURE 14. F1 Score vs Depth using Gini

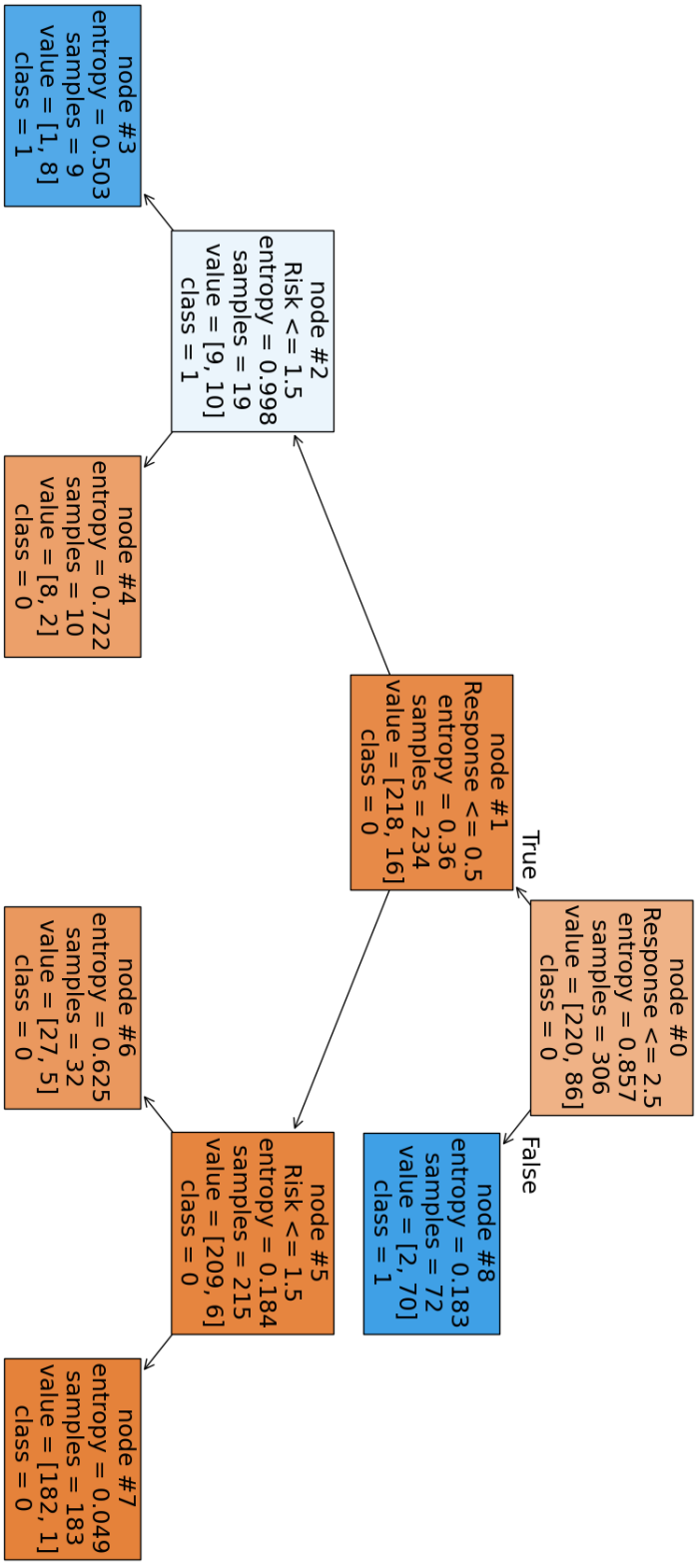

FIGURE 16. **Decision Tree using Entropy and Optimal Alpha**

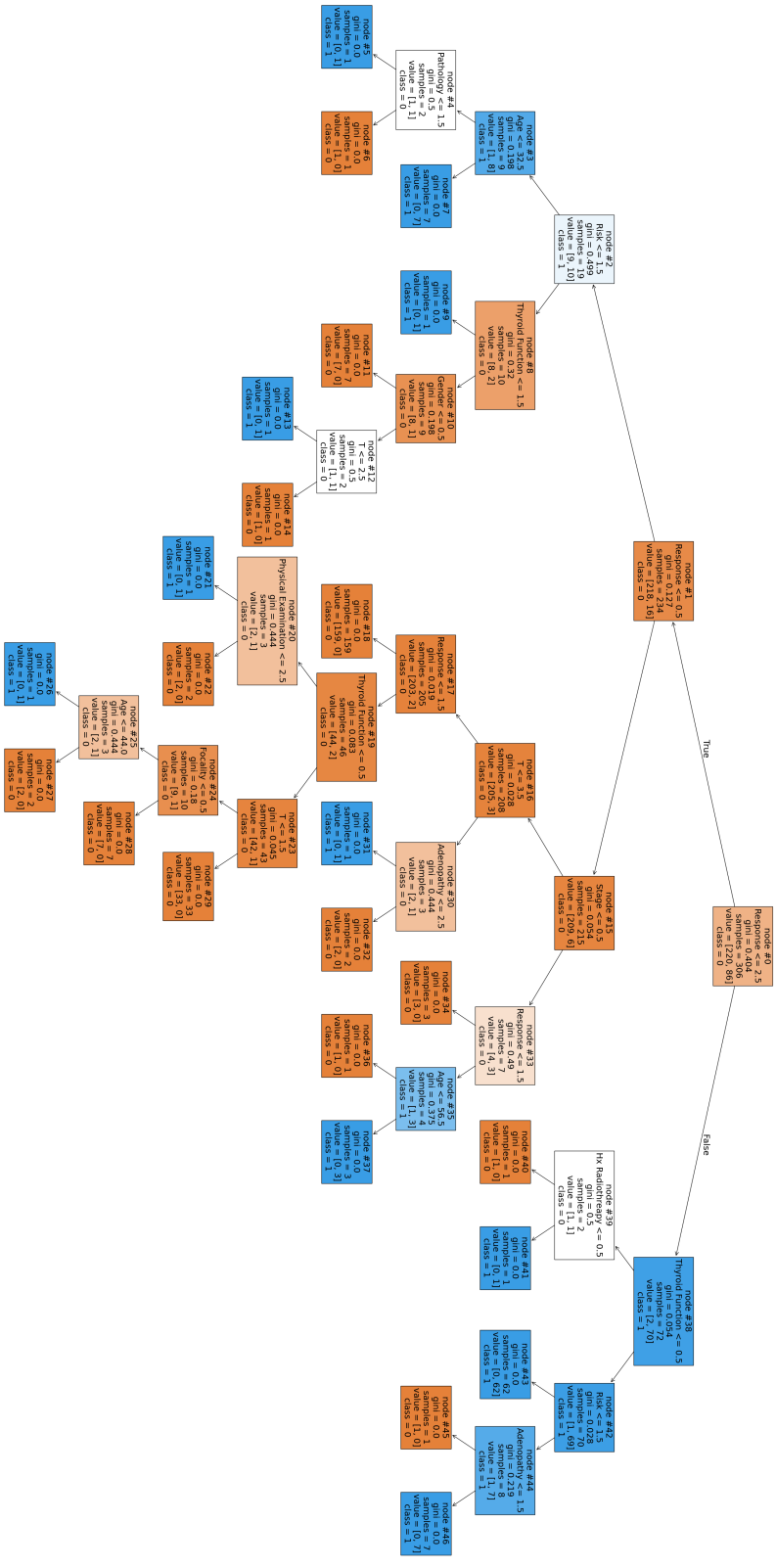

FIGURE 17. Decision Tree with Gini as Splitting Criteria*X*

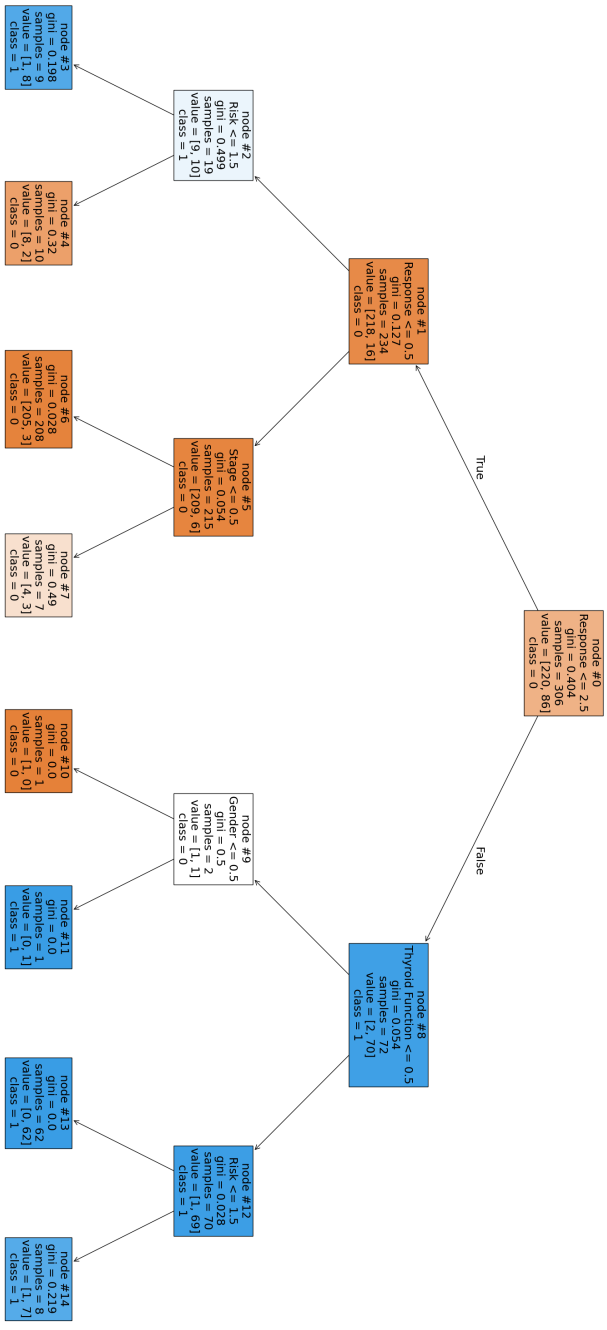

FIGURE 18. **Decision Tree with Depth 3 using Gini**

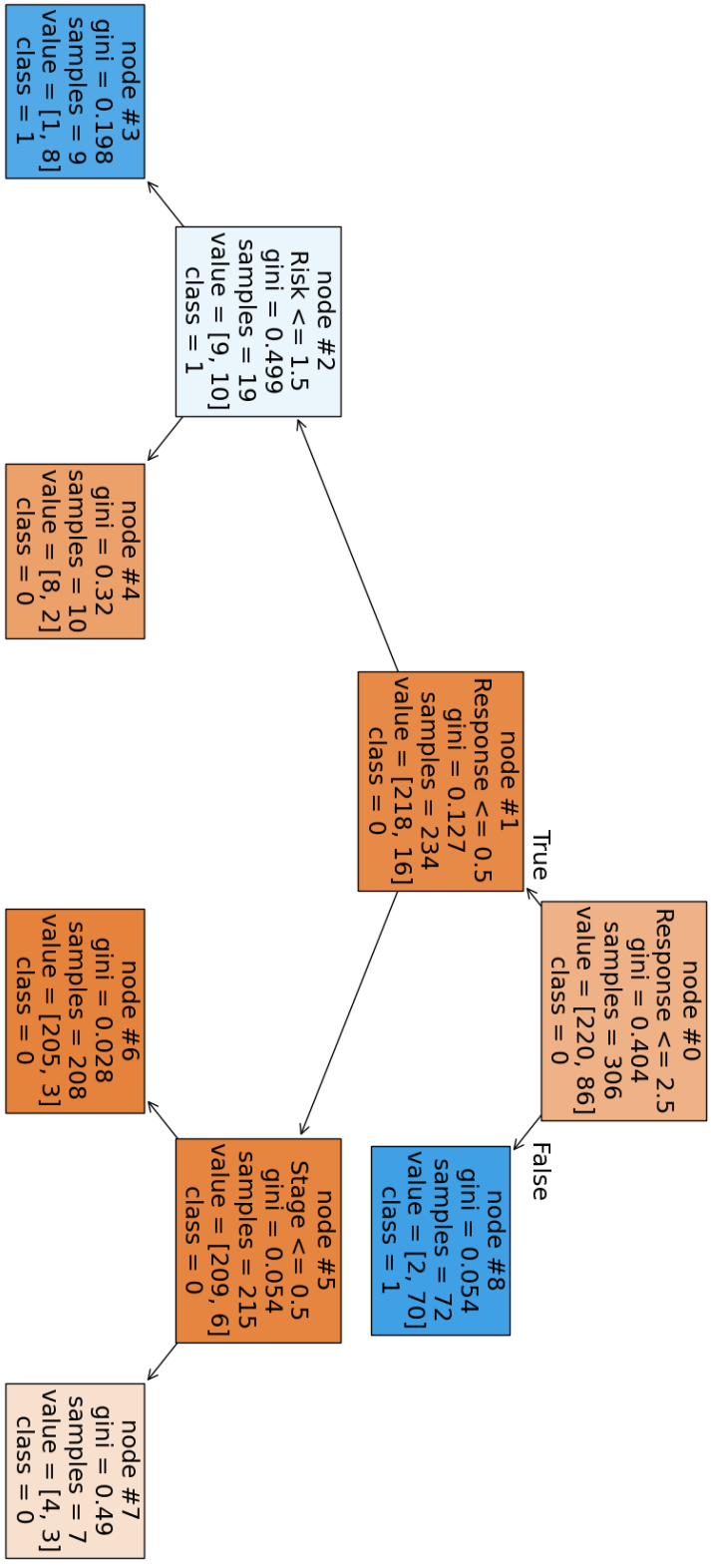

FIGURE 19. **Decision Tree using Gini and Optimal Alpha**

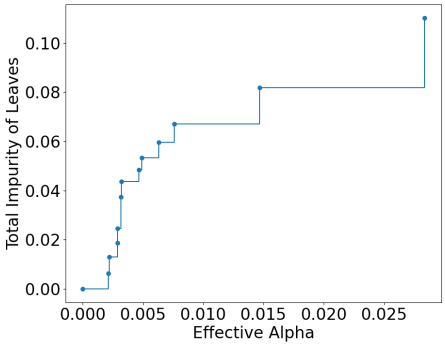

FIGURE 20. Total Impurity vs Effective Alpha using Gini

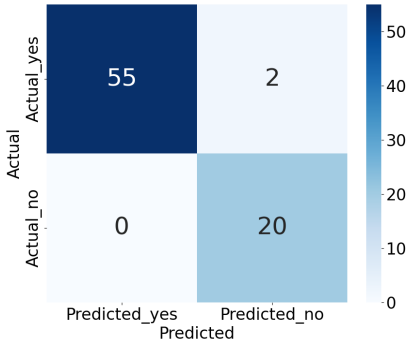

FIGURE 23. Confusion Matrix using Gini and Optimal Alpha

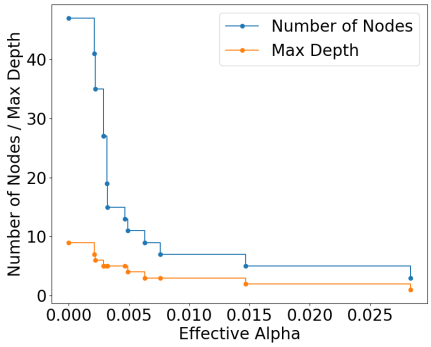

FIGURE 21. Number of Nodes and Depth vs Effective Alpha using Gini

REFERENCES

[1] L. Rokach and O. Maimon, "Decision Trees," in *The Data Mining and Knowledge Discovery Handbook*, vol. 6, pp. 165-192, 2005.

[2] V. Saranya and R. Porkodi, "A Study and Analysis of Decision Tree Based Classification algorithms using R," 2018.

[3] V. Richer, "Understanding Decision Trees (once and for all!)," *Medium*, Mar. 2, 2019. [Online]. Available: https://towardsdatascience.com/understanding-decision-trees-once-and-for-all-2d891b1be579 [Accessed: June 20, 2024].

[4] Y. Y. Song and Y. Lu, "Decision tree methods: applications for classification and prediction," *Shanghai Archives of Psychiatry*, vol. 27, no. 2, pp. 130-135, Apr. 2015.

[5] D. Bhukya and R. Sirandas, "Decision Tree Induction: An Approach for Data Classification Using AVL-Tree," *International Journal of Computer and Electrical Engineering*, vol. 2, pp. 660-665, Jan. 2010.

[6] S. Borzooei, G. Briganti, M. Golparian, et al., "Machine learning for risk stratification of thyroid cancer patients: a 15-year cohort study," *Eur Arch Otorhinolaryngol*, vol. 281, pp. 2095-2104, 2024.

[7] S. Dash, "Decision Trees Explained — Entropy, Information Gain, Gini Index, CCP Pruning," *Medium*, Nov. 2, 2022. [Online]. Available: https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-prur [Accessed: June 20, 2024].

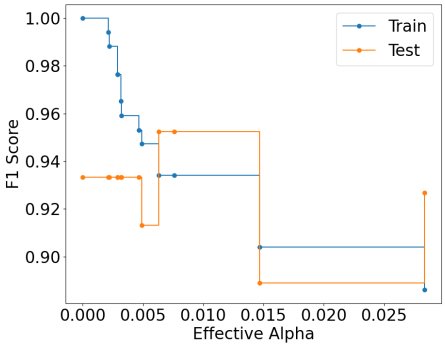

FIGURE 22. F1 Score vs Effective Alpha using Gini

NIMESH G. PRADHAN is currently pursuing a Bachelor's degree in Electronics, Communication, and Information Engineering at Thapathali Campus. He is currently in the final year of his degree. His interests lie in the fields of Data Mining, Machine Learning, and Deep Learning.

RAMAN BHATTARAI is currently pursuing a Bachelor's degree in Electronics, Communication, and Information Engineering at Thapathali Campus. He is currently in the final year of his degree. His interests lie in the fields of Data Mining, Computer Vision, and Deep Learning.