Decision Trees in Machine Learning: Tutorial

Student Id:23093730 **GitHub Link**

1. Introduction

Decision Trees (DTs) are possibly one of the simplest but widely used machine learning algorithms. They offer an easy and powerful way for classification as well as regression tasks. A Decision Tree is a tree-like chart in which each internal node is a choice regarding a feature, each branch is a consequence, and each leaf node has a terminal prediction. Due to their interpretability, efficiency, and ability to handle numerical and categorical data, Decision Trees have been widely used in many real-world problems, including healthcare, finance, and recommendation systems.

This tutorial provides an in-depth explanation of Decision Trees, covering their mechanism of working, key concepts, mathematical basis, implementation, performance estimation, advantages, real-world applications, and how they can be improved**.**

2.How Decision Trees Function

A Decision Tree functions by partitioning a dataset recursively into subsets of smaller size until it reaches a stopping criterion. It does so in a structured fashion:

Selecting the Best Feature to Split – For each node, the algorithm looks at various features and chooses the one that best splits the data. The decision is made on the basis of a chosen criterion, e.g., Gini impurity or entropy, which measures the homogeneity of the resulting subsets.

Splitting the Data – Once the optimal feature has been selected, the dataset is divided into two or more sets based on its values. This ensures that like data points are grouped together to improve predictive performance.

Recursive Partitioning – The division process repeats for every subset, generating more branches until a stopping condition is reached. Typical stopping conditions are when a maximum tree depth is reached, too few samples fall in a node, or pure classification is reached (i.e., when all the data points in a node are of the same class).

Making Predictions – In classification problems, the most common class in a leaf node is used as the predicted label. In regression problems, the mean or median of the target variable in the node is used as the ultimate prediction.

This approach, known as recursive binary splitting, is a greedy, top-down algorithm in which the algorithm makes locally optimal choices at each step. The resulting tree structure makes Decision Trees very interpretable and easy to visualize, which is why they are used so widely in many machine learning tasks**.**

3.Key Concepts and Formulas

3.1 Splitting Criteria

Decision Trees utilize measures of impurity to determine which feature is most suitable to split on.

- Gini Impurity (for classification): Calculates how often an element randomly selected would be incorrectly classified if assigned at random based on the distribution in the node. More purified splits yield smaller values of Gini.

- Entropy & Information Gain: Entropy quantifies disorder in a dataset, and Information Gain quantifies entropy decrease after splitting. Higher Information Gain represents a good split.

- Mean Squared Error (for regression): Quantifies variance in a subset to ensure splits minimize prediction errors.

3.2 Tree Depth and Stopping Criteria

In order to prevent overfitting, Decision Trees employ stopping criteria such as:

- **Maximum depth:** Limits how deep the tree will grow.

- **Minimum samples per split:** Avoids splitting nodes when they have too little data.

- **Impurity threshold:** Avoids splitting when splitting further does not decrease impurity by a large amount.

3.3 Feature Selection

Choosing the best feature at each step is critical for precision. Entropy, information gain, or Gini impurity are utilized by algorithms to determine the most informative split. Feature selection assists in correct performance and prevents unnecessary complexity in the tree**.**

---

4. Implementation of Code

This section presents a step-by-step code implementation of the Decision Tree algorithm on the Heart Disease dataset. The objective is to predict patients as having heart disease or not based on medical features.

We first import libraries required such as scikit-learn for machine learning, pandas for data handling, and matplotlib and seaborn for visualization. We import the dataset of Heart Disease from Open ML and split it into features (X) and target variable (y). We split the dataset into training set (70 percent) and test set (30 percent) through train test split.

We then train a Decision Tree Classifier without pruning such that the tree is allowed to grow as big as possible. The trained model is then used to predict on the test set and its performance evaluated using accuracy, confusion matrix, and feature importance.

For visualization purposes, we generate a decision tree plot to understand its structure, a feature importance plot to visualize significant features, and a confusion matrix to check classification accuracy. The plots help interpret the decision-making of the model.

Finally, the generated plots are saved as clear images in PNG format. This code provides a real-world perspective into Decision Trees in healthcare predictions.

---

5. Model Evaluation and Performance

To obtain a better comprehension of the model's performance, we examined it with a range of significant measures and graphs.

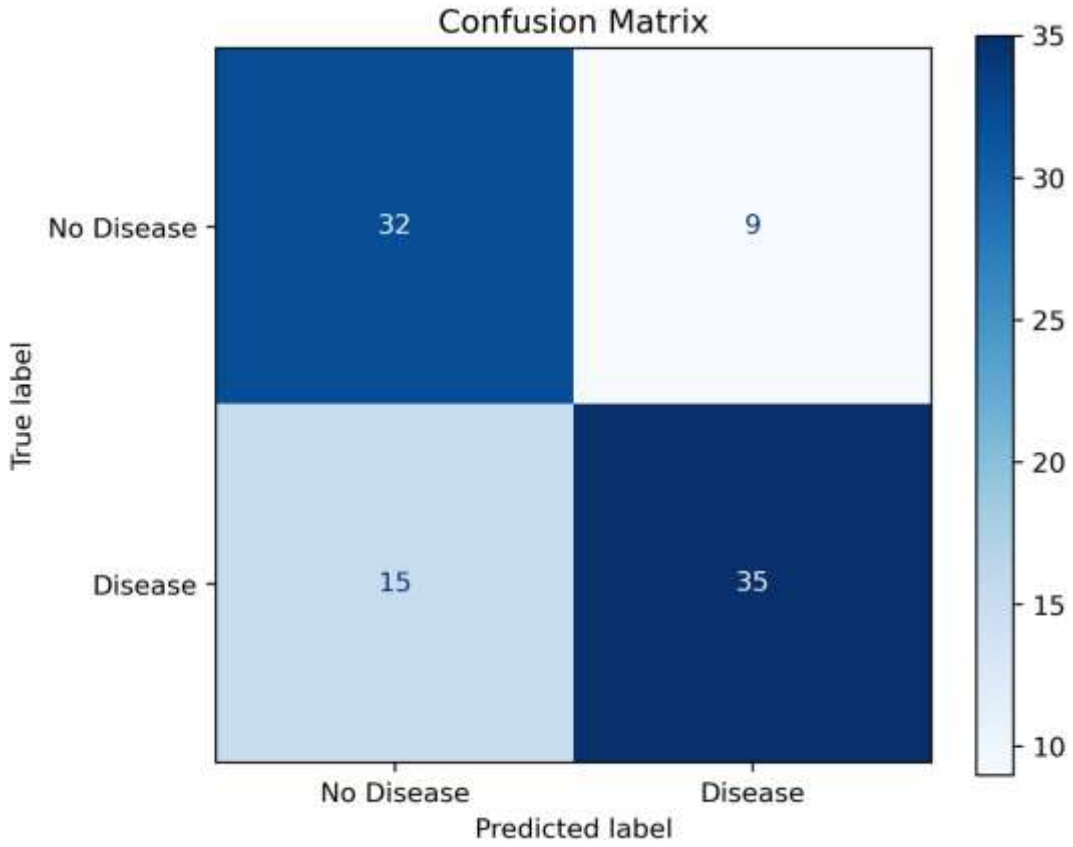
The confusion matrix

True Positives: 35 correctly predicted instances of heart disease

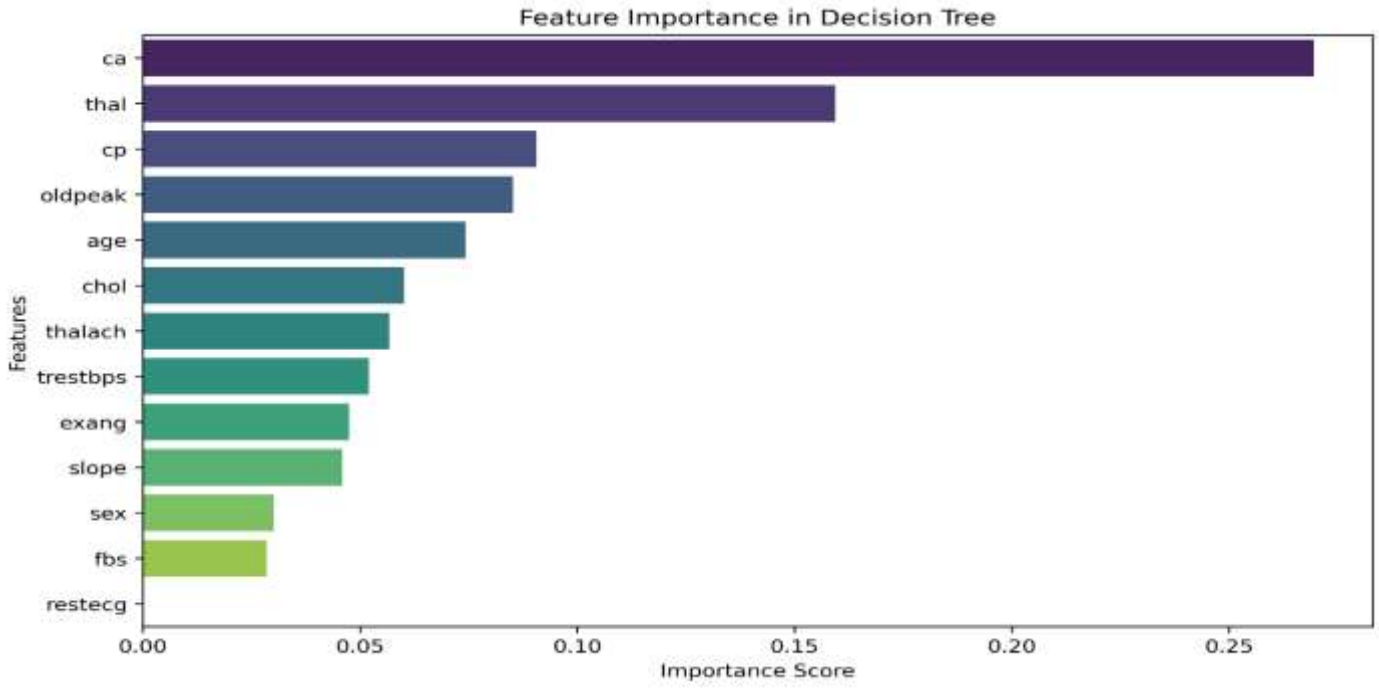
True Negatives: 32 correctly classified non-disease

False Positives: 9 wrongly predicted instances of heart disease

False Negatives: 15 missed instances of heart disease

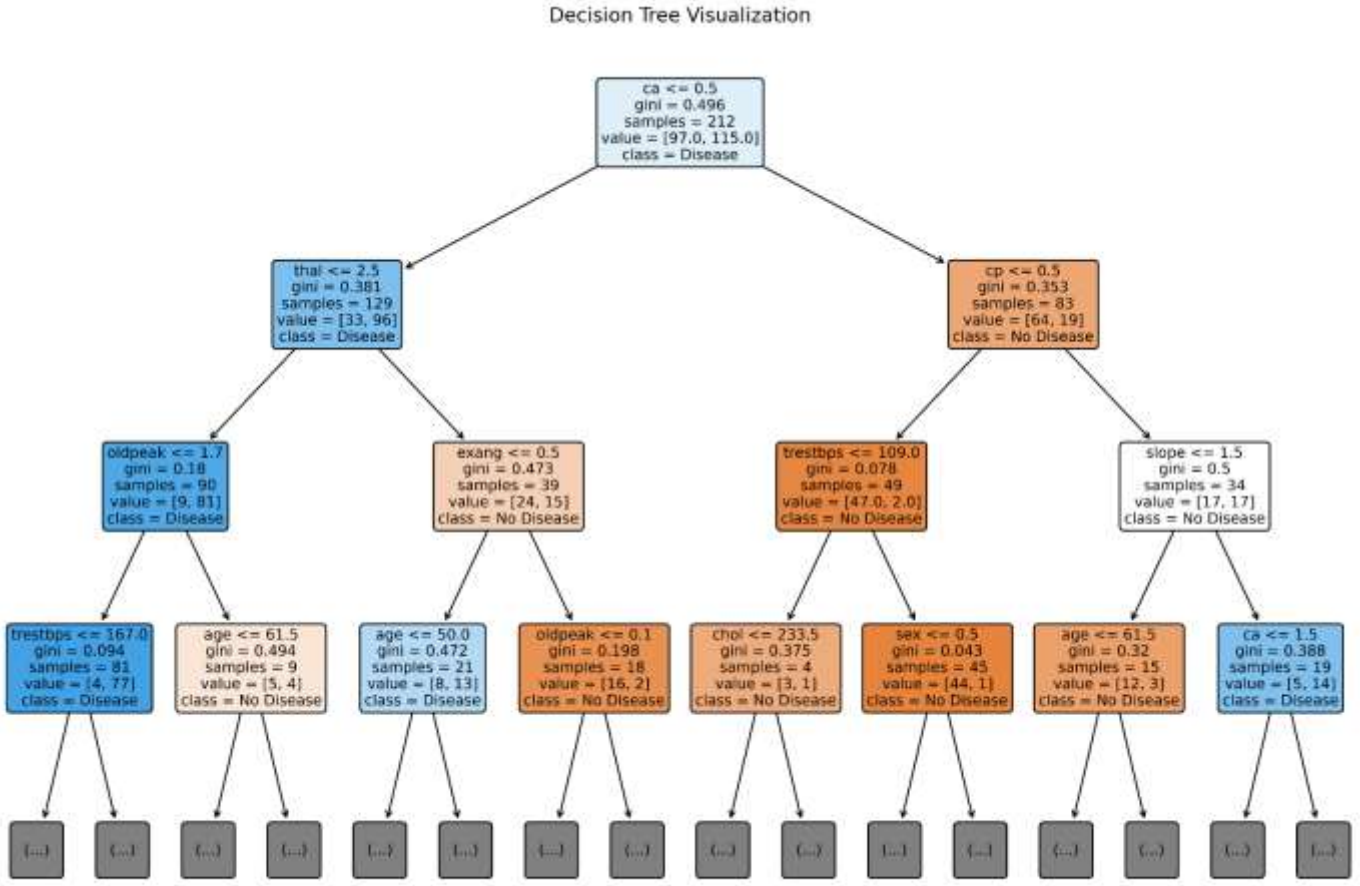


Reducing false negatives is particularly crucial in the medical context to avoid missing patients who need urgent medical attention. The feature importance plot shows the most important features on which the



model relies for its predictions, including ca (number of major vessels), thal (thalassemia status), cp (chest pain type), and old peak (ST depression with exercise). All these features play a significant role in deciding heart disease presence.

The decision tree graph shows the root node to be ca, emphasizing its role in prediction. Descending further down the tree, we can observe other features like thal and cp also reduce the classification process.



The confusion matrix plot visually displays the model's performance in classification. Methods like hyperparameter tuning and other algorithms can be used for further improvement in the model. The complete implementation, code, and plots are on the **GitHub** repository.

6.Advantages & Cons, and Comparison with Other ML Algorithms

6.1 Advantages

Decision Trees possess several advantages:

Simple to interpret and understand: Decision Trees provide a clear, pictorial representation of decision-making, making them highly interpretable even for non-technical individuals.

- Can work with categorical and numerical data: Most algorithms require special preprocessing, but Decision Trees work well with all types of data without any special preparation.

- Requires minimal data preprocessing: They do not require feature scaling or normalization, making the data preparation easier.

- Friendly to missing values: Decision Trees accommodate missing values using surrogate splits, which makes them more usable in practical scenarios.

- Computationally efficient when working with small data sets: They are fast to train on small data sets compared to other complex models like neural networks.

6.2 Weaknesses

Despite their strengths, Decision Trees also possess certain weaknesses:

- Prone to overfitting: Unless regularized, Decision Trees have the tendency to form extremely complex models that learn noise rather than real patterns from data.

- Unstable: Small variations in training data can lead to very different tree structures, hence they are less reliable for certain applications.

- Biased towards dominant classes: Decision Trees tend to be biased towards the majority class when the dataset is imbalanced unless additional techniques, e.g., class weighting, are applied.

- Not as good at very complicated relationships: Unlike deep learning models, Decision Trees struggle with very nonlinear patterns.

6.3 Comparison with Other Algorithms

- Random Forest vs. Decision Trees: While Decision Trees are individual models, Random Forest is a combination of multiple Decision Trees to minimize accuracy and avoid overfitting. Random Forest increases robustness and stability since it averages out the result from many trees; hence, Random Forest is always a go-to for the vast majority of practical applications.

- Decision Trees vs. Support Vector Machines (SVM): SVM is more suitable for high-dimensional spaces and can handle complex decision boundaries with ease. Decision Trees are easier to interpret and faster to train, especially when dealing with large datasets.

- Decision Trees vs. Neural Networks: While Neural Networks have the ability to learn complex data patterns and also perform well when dealing with extensive datasets, they need huge computing power and, in many respects, are referred to as "black boxes" since their internal workings make decision-making more opaque than Decision Trees.

---

7. Applications

Decision Trees have wide-ranging applications in various industries due to their interpretability and decision-making capability.

- Healthcare: Decision Trees are employed to classify diseases based on symptoms and medical history. For example, they can anticipate the risk of heart disease from patient attributes such as age, cholesterol, and blood pressure.

- Finance: Banks and financial institutions utilize Decision Trees in credit risk scoring. Based on variables such as income, credit history, and loan amount, they determine the likelihood of repaying or defaulting on the loan and increase lending quality.

- Marketing: Decision Trees are utilized by companies to segment customers, allowing them to classify customers based on buying behaviour, demographics, or interests. This allows for targeted marketing and personalized recommendations, increasing customer engagement.

- Recommendation Systems: Decision Trees are used by online shopping and streaming websites to personalize user experiences. They recommend products, movies, or services based on browsing history, purchase behaviour, and user interests, all of which are examined to personalize recommendations for individual users.

Due to their versatility, Decision Trees are a central element in data-driven decision-making, providing actionable insights in numerous industries.

8.How to Enhance Accuracy

8.1 Feature Engineering

Choosing the appropriate features is essential to enhance Decision Tree accuracy. Elimination of irrelevant or strongly correlated features prevents overfitting. Adequate encoding of categorical variables, for example, one-hot encoding or label encoding, ensures that the model learns in the correct way.

8.2 Hyperparameter Tuning

Tree parameters' optimization improves performance:

- **Max Depth:** Restricts tree growth to avoid very complex structures that fit noise.

- **Min Samples per Split:** Prevents over-splits by requiring a minimum sample, improving generalization.

8.3 Ensemble Methods

Gathering multiple Decision Trees enhances accuracy and stability:

- **Bagging:** Trains multiple trees on different subsets of data and aggregates predictions to reduce variance.

- **Boosting:** Enhances poor trees in a step-by-step manner by focusing on misclassified instances, increasing predictive power.

- **Random Forest:** An ensemble of Decision Trees that reduces overfitting and increases robustness and accuracy.

Employing these techniques ensures enhanced generalization and better model performance.

9.Conclusion

Decision Trees are an elementary machine learning algorithm because they are simple, interpretable, and versatile. They work through recursively partitioning data based on feature values in order to create a tree-based model that has the ability to make quick decisions. Their functionality in making classifications and regression enables them to become a priceless treasure in various areas.

One of the greatest strengths of Decision Trees is that they are easy to understand, even for non-specialists. They need little preprocessing of data and can handle both numerical and categorical data. They do have some weaknesses, though, such as overfitting, particularly when the tree becomes too deep. Also, small changes in the dataset result in extremely different tree structures, making them less stable than certain

other machine learning models. These issues can be addressed through hyperparameter tuning, feature selection, and ensemble methods such as Random Forest and Boosting, which significantly enhance performance. Ensemble methods with multiple Decision Trees enhance accuracy, reduce variance, and increase generalization.

Though limited, Decision Trees are still an essential tool in machine learning and are extensively applied in areas like healthcare, finance, and marketing. Their ability to interpret and perform well guarantees that they will remain an invaluable asset in data-driven decision-making.

---

10. References

1. Quinlan, J. R. (1986). "Induction of Decision Trees." Machine Learning.
2. Breiman, L. (2001). "Random Forests." Machine Learning.
3. Hastie, T., Tibshirani, R., & Friedman, J. (2009). "The Elements of Statistical Learning." Springer.
4. Scikit-learn documentation: https://scikit-learn.org/stable/modules/tree.html
5. Open ML dataset repository: https://www.openml.org/