Indian Institute of Technology Patna

CS571:AI and ML Lab

ASSIGNMENT-5: Decision Trees
Submission Date: 12^{h} May 2024

Baskar Natarajan - 2403res19(IITP001799)
Jyotisman Kar – 2403res35(IITP001751)
SEMESTER-1
MTECH AI & DSC
INDIAN INSTITUTE OF TECHNOLOGY PATNA.

1.Problem Description:..... 3
1. Decision Tree:..... 3
2. Components of Decision Trees: 3
3. Types of Decision Trees: 4
4. When to Use Decision Trees: 4
5. How to Use Decision Trees: 4
6. Advantages of Decision Trees: 5
7. Disadvantages of Decision Trees:..... 5
8. Real-Time Example: 5
9. GINI Index / Impurity:..... 5
10. Information Gain:..... 6
11. Pruning: 6
12. Ensemble Methods: 6
13. Model and Programming Steps: 7
    a. Importing Necessary Libraries: 7
    b. Loading the Dataset:..... 7
    c. Data Preprocessing: 7
    d. Splitting the Dataset: 7
    e. Model Building: 7
    f. Hyperparameter Tuning: 7
    a. Model Evaluation:..... 8
    b. Visualization: 8
    c. Documentation:..... 8
    d. Output:..... 8

1.Problem Description:

Problem Statement:

- Use Decision Trees to prepare a model on fraud data treating those who have taxable_income <= 30000 as "Risky" and others as "Good."

Data Description and Link:
https://www.dropbox.com/scl/fi/kd8z3309rk0yc5ugdnuqh/Fraud_check.csv?rlkey=ssj0u1w0s3ok09gb07e06swth&st=m1gmtrxs&dl=1

- **Undergrad**: A person is under-graduated or not
- **Marital.Status**: marital status of a person
- **Taxable.Income**: Taxable income is the amount of how much tax an individual owes to the government (not to use)
- **Work Experience**: Work experience of a person
- **Urban**: Whether that person belongs to an urban area or not

Implementation Details:

- **Assume**: taxable_income <= 30000 as “Risky=0” and others are “Good=1”
- Use the first 80% of data as a training set and the remaining 20% as a test set.
- Report accuracy on the test set

Documents to submit:

- Model code
- A detailed document describing results such as time taken for the execution, confusion matrix, and accuracy results

1. Decision Tree:

- A Decision Tree is a hierarchical structure that recursively splits data based on feature values, aiming to classify instances into predefined classes or predict continuous target variables.
- It is a supervised learning algorithm used for both classification and regression tasks.

2. Components of Decision Trees:

Nodes: Represent decision points based on feature values.

Edges: Connect nodes and represent the outcome of a decision.

Root Node: The topmost node in the tree, representing the initial decision point.

Internal Nodes: Decision points that lead to further splits.

Leaf Nodes: Terminal nodes representing the final class or predicted value.

Splitting Criteria: Measures used to determine the best feature and value to split the data.

Pruning: Technique to prevent overfitting by removing unnecessary branches.

3. Types of Decision Trees:

Classification Trees: Used for classification tasks where the target variable is categorical.

Regression Trees: Used for regression tasks where the target variable is continuous.

4. When to Use Decision Trees:

- Decision Trees are suitable for both classification and regression tasks.
- They are particularly useful when the data has a hierarchical structure and can be easily represented in a tree-like format.
- Decision Trees are interpretable and easy to understand, making them suitable for applications where model transparency is important.

5. How to Use Decision Trees:

Data Preparation: Preprocess the data by handling missing values, encoding categorical variables, and scaling features if necessary.

Model Training: Train the Decision Tree model using the prepared dataset.

Model Evaluation: Evaluate the model's performance using appropriate metrics such as accuracy, precision, recall, and F1-score for classification, or mean squared error for regression.

Model Tuning: Fine-tune hyperparameters using techniques like grid search and cross-validation to improve model performance.

Deployment: Deploy the trained model in production to make predictions on new data.

6. Advantages of Decision Trees:

- Easy to understand and interpret.
- Can handle both numerical and categorical data.
- Non-parametric, meaning they do not make assumptions about the distribution of data.
- Robust to outliers and missing values.
- Can capture non-linear relationships between features and target variables.

7. Disadvantages of Decision Trees:

- Prone to overfitting, especially on complex datasets with many features.
- Can be sensitive to small variations in the data.
- Lack of robustness, meaning small changes in the data can lead to significantly different trees.
- May not generalize well to unseen data, especially when the tree is deep.

8. Real-Time Example:

- Suppose you are building a credit risk assessment system for a bank.
- Decision Trees can be used to classify loan applicants into "Low Risk," "Medium Risk," and "High Risk" categories based on features such as credit score, income, debt-to-income ratio, etc.
- The bank can then use these classifications to make decisions on whether to approve or deny a loan application.

9. GINI Index / Impurity:

- GINI index is a measure of impurity or randomness in a dataset.
- In the context of Decision Trees, GINI index is used to evaluate the homogeneity of a node.
- A GINI index of 0 indicates perfect homogeneity, meaning all the samples in the node belong to the same class.
- A GINI index of 0.5 indicates maximum impurity, meaning the samples are evenly distributed across all classes.
- Mathematically, for a node kk with KK classes, the GINI index is calculated as:

$$GINI(k) = 1 - \sum_{i=1}^{K} p(i)^2$$

- p(i) is the probability of an instance being classified as class i in node k.

10. Information Gain:

- Information gain is a measure used to decide the order of feature splitting in Decision Trees.
- It quantifies the reduction in entropy or increase in homogeneity achieved by splitting a dataset based on a particular feature.
- Entropy measures the randomness or impurity in a dataset. The lower the entropy, the more homogeneous the dataset is.
- Information gain is calculated as the difference between the entropy of the parent node and the weighted average of the entropies of the child nodes after the split.
- Mathematically, the information gain IGIG for a split on feature AA is given by:

$$IG(A) = H(parent) - \sum_j \frac{N(j)}{N} H(child_j)$$

- H denotes entropy, $N(j)$is the number of instances in the child node j, N is the total number of instances in the parent node, and $childj$ represents each child node.

11. Pruning:

- Pruning is a technique used to prevent overfitting in Decision Trees.
- It involves removing parts of the tree that do not provide significant predictive power on the validation set.
- Pre-pruning involves stopping the tree-building process early by setting constraints on tree depth, minimum samples per leaf, or minimum samples per split.
- Post-pruning involves building the full tree first and then removing or collapsing nodes that are least useful based on statistical significance tests or cross-validation.

12. Ensemble Methods:

- Ensemble methods combine multiple Decision Trees to improve predictive performance.
- Random Forest is an ensemble method that builds multiple Decision Trees using random subsets of the training data and features.

- Gradient Boosting Machines (GBM) sequentially train Decision Trees, with each tree correcting the errors of the previous one, to improve prediction accuracy.

13. Model and Programming Steps:

a. Importing Necessary Libraries:

- Imports essential libraries for data manipulation, model building, evaluation, visualization, and time tracking.

b. Loading the Dataset:

- Loads the dataset "Fraud_check.csv" using pandas.

c. Data Preprocessing:

- Dataset is loaded and preprocessed:
- A new column "Income_Category" is created based on "Taxable.Income".
- Encodes categorical variables ("Undergrad", "Marital.Status", "Urban") into numerical format using LabelEncoder.

d. Splitting the Dataset:

- Splits the dataset into training and testing sets using train_test_split from scikit-learn.
- Uses 80% of the data for training and 20% for testing, with a random state for reproducibility.

e. Model Building:

- DecisionTreeClassifier is initialized.
- Hyperparameter Tuning:
- GridSearchCV is used to find the best hyperparameters via cross-validation.

f. Hyperparameter Tuning:

- Defines a parameter grid containing different hyperparameters for the DecisionTreeClassifier:
- max_depth: Maximum depth of the tree.

- min_samples_split: Minimum number of samples required to split an internal node.
- min_samples_leaf: Minimum number of samples required to be at a leaf node.
- Utilizes GridSearchCV to perform an exhaustive search over the specified parameter values, optimizing model performance with 5-fold cross-validation.

a. Model Evaluation:

- The best model is evaluated on the test set.
- Accuracy, confusion matrix, and classification report are printed.

b. Visualization:

- Confusion matrix heatmap and Decision Tree visualization are plotted.

c. Documentation:

- Results including best parameters, accuracy, confusion matrix, classification report, and Decision Tree statistics are saved to "results_document.txt".
- This program streamlines the process of building and evaluating a Decision Tree classifier for fraud detection, providing insights into model performance and visualizations for better understanding.
- Plotted.

d. Output:

Best Parameters: {'max_depth': 5, 'min_samples_leaf': 4, 'min_samples_split': 2}

Best Accuracy: 0.7812499999999999

Accuracy on Test Set: 0.7583333333333333

Confusion Matrix: [[91 3] [26 0]]

Decision Tree Statistics:

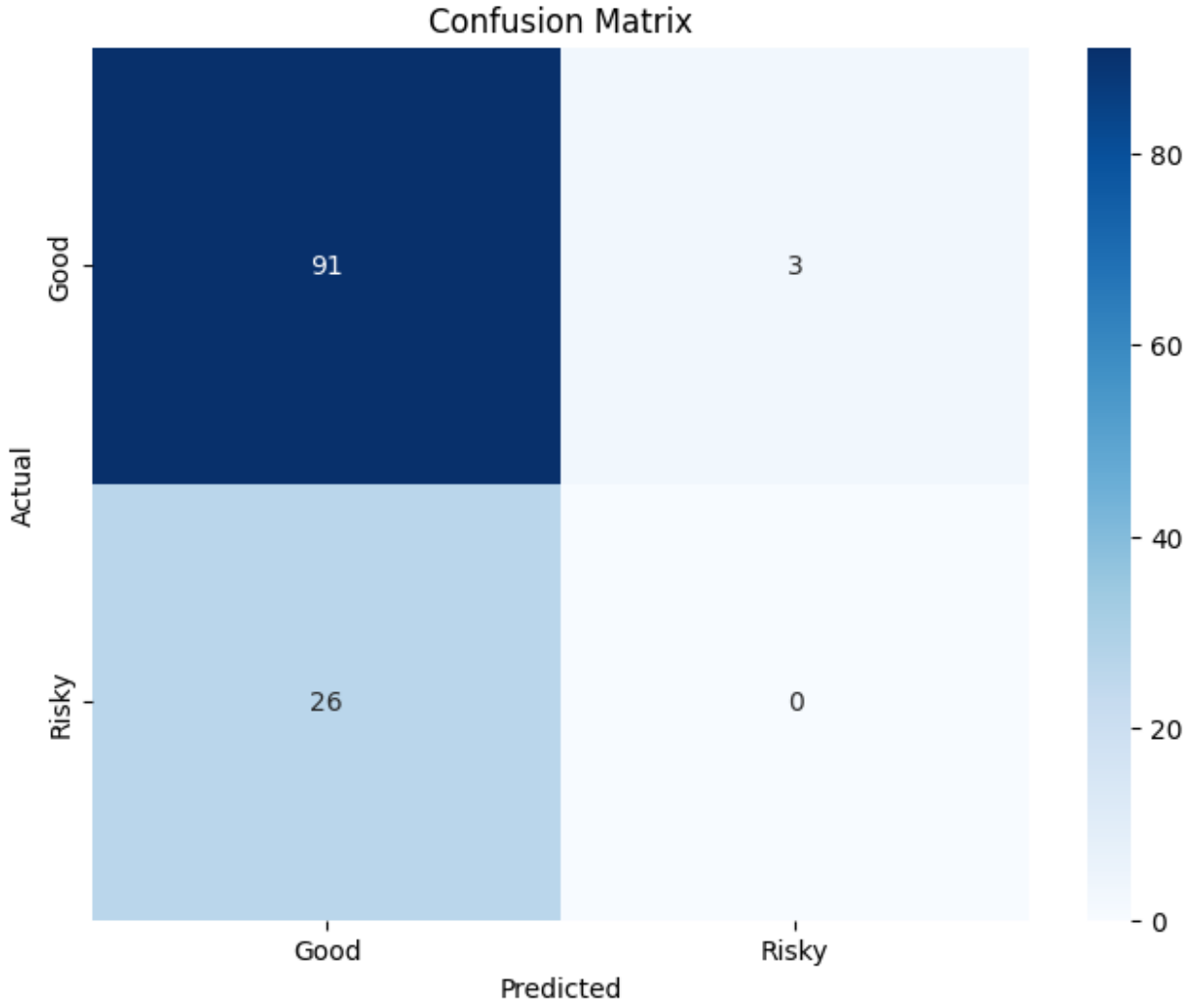
Number of nodes: 39

Depth of tree: 5

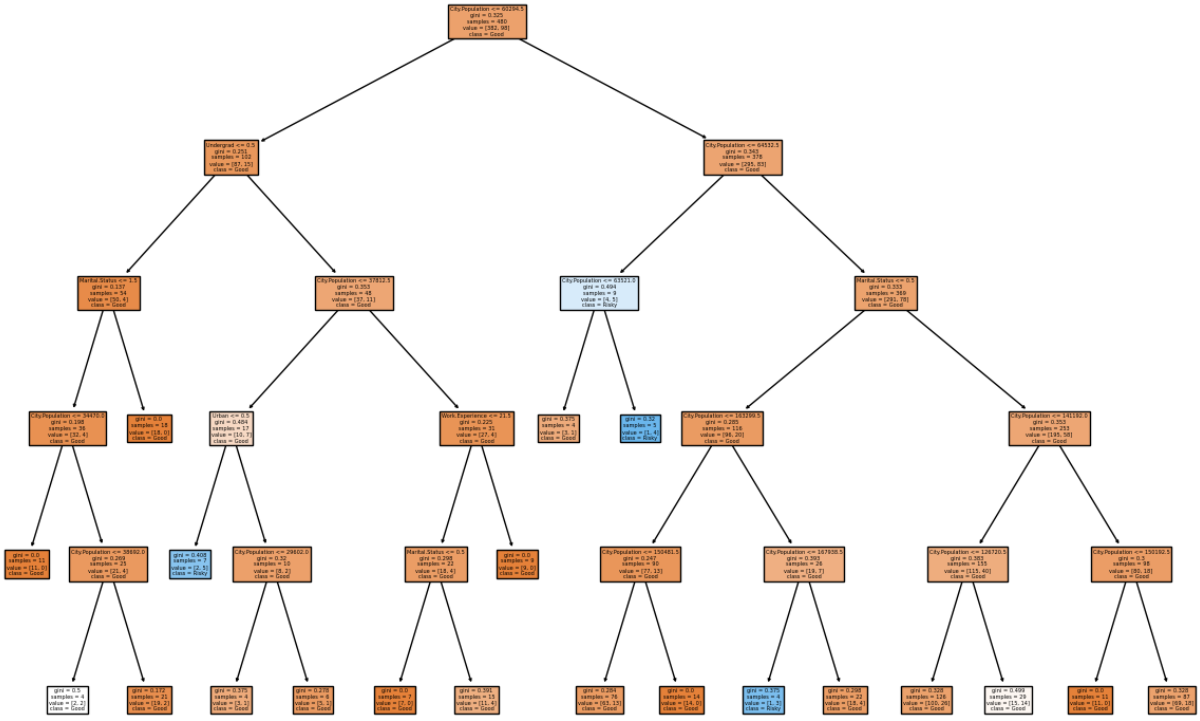
```
Classification Report:
              precision    recall  f1-score   support

        Good       0.78      0.97      0.86        94
       Risky       0.00      0.00      0.00        26

    accuracy                           0.76       120
   macro avg       0.39      0.48      0.43       120
weighted avg       0.61      0.76      0.68       120
```



Decision Tree



Time taken for execution: 1.8353145122528076 seconds