JURNAL RISET INFORMATIKA **P-ISSN: 2656-1743 |E-ISSN: 2656-1735**
Vol. 4, No. 3. June 2022 DOI: https://doi.org/10.34288/jri.v4i3.173

Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

PERFORMANCE COMPARISON OF MUSHROOM TYPE CLASSIFICATION BASED ON MULTI-SCENARIO DATASET USING DECISION TREE C4.5 AND C5.0

Citra Mirna Wati-1*), Abd. Charis Fauzan-2, Harliana-3

Fakultas Ilmu Eksakta, Ilmu Komputer
Universitas Nahdlatul Ulama Blitar
https://unublitar.ac.id/
ctrmirna@gmail.com 1*), abdcharis@unublitar.ac.id 2, harliana@unublitar.ac.id 3
(*) Corresponding Author

Abstrak

Indonesia berada di daerah iklim tropis yang cukup mendukung untuk pertumbuhan jamur. Jamur dapat diklasifikasikan menjadi jamur beracun dan tidak beracun. Identifikasi jenis jamur penting untuk dilakukan karena jamur, terutama jamur beracun beresiko menimbulkan potensi yang berbahaya bagi manusia, seperti mengakibatkan sakit parah bahkan menyebabkan kematian. Tujuan penelitian ini adalah mengidentifikasi jenis jamur menggunakan pendekatan komputasi, yaitu Algoritma Decision Tree C4.5 dan C5.0. Kontribusi penelitian ini adalah penggunaan multi-scenario dataset serta komparasi performa algoritma decision tree C4.5 dan C5.0. Dataset yang digunakan adalah dataset klasifikasi jamur yang didapatkan dari kaggle.com tahapan metode dalam penelitian ini adalah studi literatur, pengumpulan data, serta pre-processing data yang didalamnya terdapat proses cleaning data dan proses partisi untuk multi-scenario dataset. Setelah itu, dilakukan implementasi Algoritma Decision Tree C4.5 dan C5.0 menggunakan library scikit-learn. Langkah terakhir adalah melakukan komparasi performa menggunakan confusion matrix. Hasil penelitian menunjukkan bahwa identifikasi jamur beracun menggunakan Algoritma Decision Tree C5.0 mendapatkan akurasi 97,05% untuk skenario 1, 97,00% untuk skenario 2, serta 97,11% untuk skenario 3. Sedangkan algoritma Decision Tre C4.5 menghasilkan akurasi sebesar 96,92% untuk skenario 1, 96,90% untuk skenario 2, serta 97,05% untuk skenario 3. Berdasarkan perbandingan performa hasil klasifikasi, disimpulkan bahwa algoritma Decision Tree C5.0 pada skenario 3 memiliki accuracy paling tinggi untuk identifikasi jamur beracun.

Kata kunci: Algoritma C4.5; Algoritma C5.0; Decision Tree; Jenis Jamur; Klasifikasi; Komparasi Performa Klasifikasi

Abstract

Indonesia has a tropical climate that supports mushroom growth. Mushroom classification into poisonous and non-poisonous mushrooms. Identification of the type of mushroom is vital because mushrooms, especially poisonous mushrooms, risk causing potential hazards to humans, such as causing serious illness and even death. This study aimed to identify the fungus type using a computational approach, namely the Decision Tree C4.5 and C5.0 Algorithms. This research contributes to using multi-scenario datasets and comparing the performance of the C4.5 and C5.0 decision tree algorithms. The dataset used is a fungal classification dataset obtained from kaggle.com. The method stages in this research are literature study, data collection, and data preprocessing, which includes a data cleaning process and a partitioning process for multi-scenario datasets. Afterwards, the Decision Tree Algorithms C4.5 and C5.0 were implemented using the sci-kit-learn library. The last step is to do a performance comparison using the confusion matrix. The results showed that identifying poisonous mushrooms using the Decision Tree C5.0 Algorithm obtained an accuracy of 97.05% for scenario 1, 97.00% for scenario 2, and 97.11% for scenario 3. At the same time, the Decision Tre C4.5 algorithm yielded an accuracy. by 96.92% for scenario 1, 96.90% for scenario 2, and 97.05% for scenario 3. Based on the comparison of the performance of the classification results, we conclude that the Decision Tree C5.0 algorithm in scenario 3 has the highest accuracy for fungal identification poisonous.

Keywords: C4.5 Algorithm; C5.0 Algorithm; Classification; Decision Trees; Type Of Mushroom; Classification Performance Comparison

P-ISSN: 2656-1743 | E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.173

JURNAL RISET INFORMATIKA
Vol. 4, No. 3 June 2022

Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

INTRODUCTION

Indonesia has a tropical climate area that is quite supportive of the growth of fungi (Fitriani et al., 2018). Fungi include the kingdom of fungi, which is one of the eukaryotic microorganisms (having a cell nucleus) that does not have chlorophyll, has spores as a means of dispersal, somatic structure of thallus in the form of a single cell (unicellular), and in the form of filaments or branched threads (multicellular) called hyphae (Mulyana, 2019). Classification of Fungi as plants or animals. There are fungal forms that can be seen directly (macroscopic), and pay attention to some use a microscope (microscopic) (Zubair & Rofiqul Muslikh, 2017). Such as the family of Agaricus and Lepiota (Ndifon, 2022), which can live and grow in various types of ecosystems, ranging from jungle forests, tourist areas, and around people's residences, with multiple shapes, different colours, and don't know much about the characteristics. So it is still difficult to identify (Lutfi Arisandi, 2019). For the classification of toxic and non-toxic mushrooms using the UCI dataset learning model, which provides specifications such as colour, odour, and shape of the fungus, identification of the type of fungus is essential because the fungus is at risk of causing potential harm to humans, such as causing severe illness and even causing death (Tank & Mumbai, 2021).

As a solution to these problems, the computer is one of the things needed. With the help of computers, the classification of poisonous mushrooms can be done in several ways, one of which is data mining. Data mining is a data process that uses artificial intelligence techniques to identify information related to large databases. Thus, one of the data mining techniques is a classification which is the basis of data analysis (Mardi, 2019). According to Bansar, Sharma & Goel, classification is a technique for determining group membership based on existing data (Kurniawan, 2018). To help classify the types of poisonous mushrooms in this study, I used the Python sci-kit-learn library with Decision Tree C4.5 and Decision Tree C5.0 Algorithms. The scikit-learn library is an open-source data analysis library for Machine Learning (ML) in python. Scikit-learn or Sklearn provides several machine learning algorithms and statistical modelling, including classification, regression, and clustering via the python interface. (Myrianthous, 2021).

Meanwhile, a Decision Tree is a flowchart structure with a tree, where each internal node indicates an attribute test. In each branch, it can represent test results and leaf nodes and represent classes or class distributions. In the scikit-learn decision tree, there are three nodes as a place for testing the attributes. The root node is the topmost node, the internal node that is between the leaf node and the end node. If a tree node reaches a predefined class level (i.e., one type of node), then the node is terminated.

To build a Decision Tree, determine the attributes tested on a node and then branch to other nodes. Splitting is finding characteristics for testing and branching (Vanfretti & Arava, 2020). The benefit of using a decision tree is that it can change the complex decision-making process into a simple one so that the form of data (tables) can become a decision tree model (Safii, 2018). However, the C4.5 and C5.0 algorithms must convert the categorical data numerically. It is because data mining in python cannot accept string input and can only be in numeric data.

Please note that Algorithm C5.0 is an extension of Algorithm C4.5, which is also an extension of ID3, and Algorithm C5.0 is a classification algorithm suitable for large data sets. The C5.0 algorithm is better than C4.5 in speed, memory, and efficiency. In the manual calculation of the C5.0 algorithm, the attribute selection uses the gain ratio. The gain ratio to select test attributes for each node in the tree. The working steps of tree creation in the C5.0 Algorithm are similar to tree creation in the C4.5 algorithm. The similarities include the calculation of entropy and gain. If the C4.5 algorithm stops until the gain calculation, then the C5.0 Algorithm will continue by calculating the gain ratio using the existing information gain and entropy. (Pratiwi et al., 2020). In contrast to automatic calculations with the python library, scikit-learn with this library is very helpful in data analysis.

Based on previous research, building a decision tree using the C4.5 algorithm for online stress stability assessment got an accuracy of 92.04% with 91.72% attributes. The process includes three steps: sample acquisition, attribute selection, and DT construction. First, perform a P-V curve analysis to generate samples for DT. Participation factor analysis and assistive algorithm to select attributes for the DT. The C4.5 algorithm was finally applied to construct the DT. A case study of the practical power of the system shows that DT can extract operation from offline stress stability analysis results and help system operators assess voltage stability status in real-time (Meng et al., 2020). The second previous study classified

P-ISSN: 2656-1743 |E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.XXX
Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

districts/cities with high and low public health development indexes using the Decision Tree C5.0 algorithm. The results showed that the model's accuracy increased with 60 iterations, with a relatively small error in the 10th iteration. The final accuracy, sensitivity, and specificity obtained were 97.09%, 96.72%, and 97.62%, respectively (Fajri et al., 2021). The difference between the author's research and previous studies is a different dataset. Because this study compares multi-scenario datasets, the accuracy results are not much different. The datasets used are also categorical data converted into numerical data in the python library sci-kit-learn.

The dataset in this study is divided into multi-scenarios to find the best performance of the C4.5 and C5.0 algorithms. The survey conducted in this research compared classification performance between Decision trees C4.5 and C5.0 based on the division of multi-scenario datasets. Thus, each performance evaluation on the C4.5 and C5.0 algorithms found the best performance on the results of each test scenario.

RESEARCH METHODS

This research will implement in a python programming language to obtain a description of the best algorithm results using the scikit-learn python library.

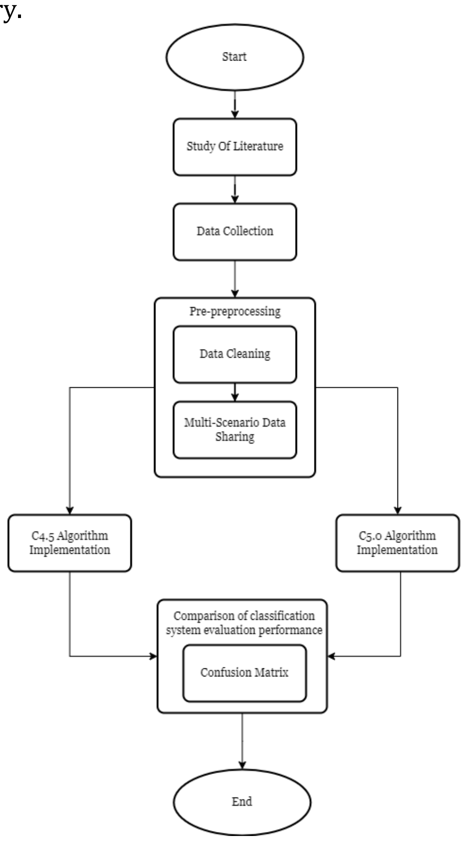

Start
Study Of Literature
Data Collection
Pre-preprocessing
Data Cleaning
Multi-Scenario Data Sharing
C4.5 Algorithm Implementation
C5.0 Algorithm Implementation
Comparison of classification system evaluation performance
Confusion Matrix
End

Figure 1. Research Flowchart

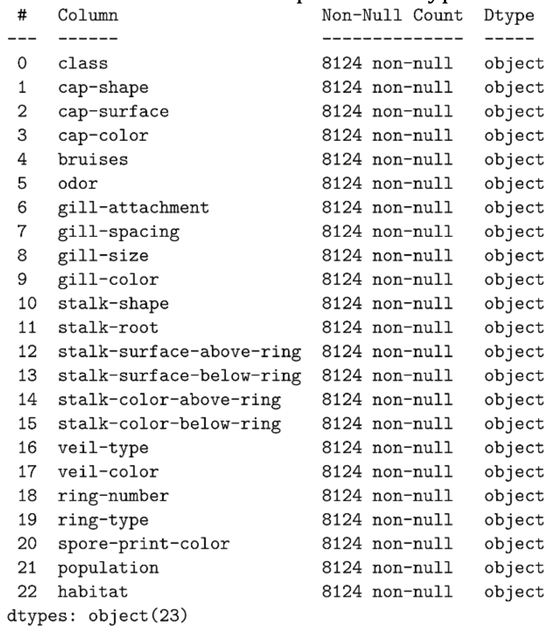
In Figure 1, there are several stages of the research process, namely literature study, data collection, and data preprocessing, including data cleaning and partitioning processes for multi-scenario datasets. In addition, there is an implementation process for the Decision Tree Algorithms C4.5 and C5.0 with the python scikit-learn library, as well as a comparison of classification performance using a confusion matrix.

Study of literature

The literature study to find several references from journals, e-books, and the internet, starting from searching the Python Scikit-Learn Library to help analyze research data on the C4.5 algorithm, C5.0 algorithm, Confusion Matrix evaluation, and search for mushroom classification datasets.

Data collection

Data collection is a step to produce data ready for data modelling. The dataset obtained is categorical, where this data is at the source https://www.kaggle.com/datasets/uciml/mushroom-classification and used to calculate the confusion matrix in the scikit-learn library, making decision trees for Algorithms C4.5 and C5. 0, this research data has a dataset in the form of a complete raw CSV along with 8124 data, 22 attributes, and classification of edible and poisonous types.

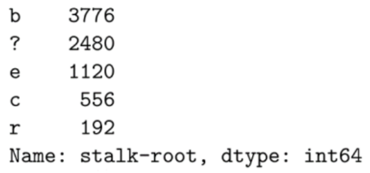


```
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   class                     8124 non-null   object
 1   cap-shape                 8124 non-null   object
 2   cap-surface               8124 non-null   object
 3   cap-color                 8124 non-null   object
 4   bruises                   8124 non-null   object
 5   odor                      8124 non-null   object
 6   gill-attachment           8124 non-null   object
 7   gill-spacing              8124 non-null   object
 8   gill-size                 8124 non-null   object
 9   gill-color                8124 non-null   object
 10  stalk-shape               8124 non-null   object
 11  stalk-root                8124 non-null   object
 12  stalk-surface-above-ring  8124 non-null   object
 13  stalk-surface-below-ring  8124 non-null   object
 14  stalk-color-above-ring    8124 non-null   object
 15  stalk-color-below-ring    8124 non-null   object
 16  veil-type                 8124 non-null   object
 17  veil-color                8124 non-null   object
 18  ring-number               8124 non-null   object
 19  ring-type                 8124 non-null   object
 20  spore-print-color         8124 non-null   object
 21  population                8124 non-null   object
 22  habitat                   8124 non-null   object
dtypes: object(23)
```

Figure 2. Mushroom Classification Data

Figure 2, Attribute data before processing is Cap Shape, Cap Surface, Cap Color, Bruises, Odor, Gill

Attachment, Gill Spacing, Gill Size, Gill Color attributes, Stalk Shape, stalk root, Stalk Surface Above Ring, Stalk Surface Below Ring, Stalk Color Above Ring, Stalk Color Belo Ring, Veil Type, Veil Color, Ring Number, Ring Type, Spore Print Color, Population, and Habitat.



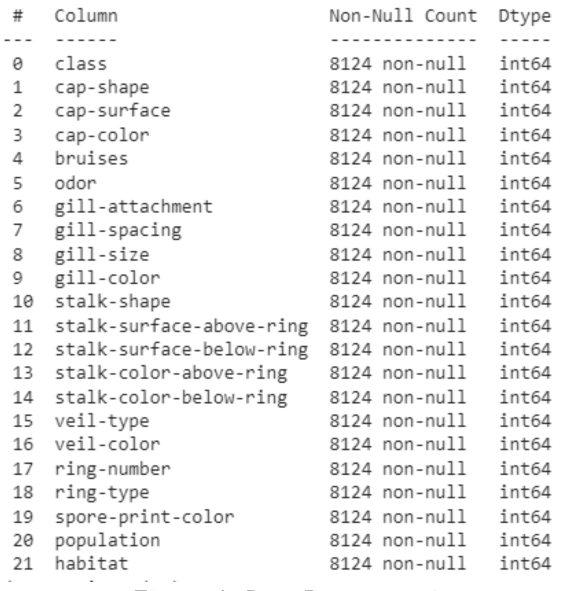
```
b     3776
?     2480
e     1120
c      556
r      192
Name: stalk-root, dtype: int64
```

Figure 3. Instance Attribute Missing Value

Figure 3, After the data is collected, it has an instance of the stalk-root attribute with a missing value of 2,480.

Pre-Preprocessing and Multi-Scenario Data

Preprocessing data is used to make it easier to manage, and then the data is cleaned in function not to select null and missing value attributes. After collecting the stalk-root attribute dataset, which has a missing value of 2,480 instances, before the stalk root attribute data is collected, clean up the data by removing the stalk-root attribute. In this preprocessing for data cleaning, it is possible to delete rows or columns with missing values. In this study, the authors delete the column part, namely the stalk-root attribute. Python programming has a scikit-learn library called the LabelEncoder module to analyze data. In the scikit-learn library, this module assigns class labels to categorical data, namely e (edible) and p (poisonous), where the class labels will help create decision trees automatically.



```
#   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
0   class                     8124 non-null   int64
1   cap-shape                 8124 non-null   int64
2   cap-surface               8124 non-null   int64
3   cap-color                 8124 non-null   int64
4   bruises                   8124 non-null   int64
5   odor                      8124 non-null   int64
6   gill-attachment           8124 non-null   int64
7   gill-spacing              8124 non-null   int64
8   gill-size                 8124 non-null   int64
9   gill-color                8124 non-null   int64
10  stalk-shape               8124 non-null   int64
11  stalk-surface-above-ring  8124 non-null   int64
12  stalk-surface-below-ring  8124 non-null   int64
13  stalk-color-above-ring    8124 non-null   int64
14  stalk-color-below-ring    8124 non-null   int64
15  veil-type                 8124 non-null   int64
16  veil-color                8124 non-null   int64
17  ring-number               8124 non-null   int64
18  ring-type                 8124 non-null   int64
19  spore-print-color         8124 non-null   int64
20  population                8124 non-null   int64
21  habitat                   8124 non-null   int64
```

Figure 4 . Data Preprocessing

Figure 4, Previously, there were 22 attributes in data collection because there were missing values, so they had to be preprocessed to become 21 attributes. More specifically, before and after preprocessing, the attribute results are not the same because the stalk-root attribute disappears after processing.

In the dataset processing, there is a training data separation scikit-learn function named train_test_split() which helps estimate the performance of data mining algorithms implemented in the C4.5 and C5.0 algorithms. Function Divides the data set into train and test sets to evaluate how well the data mining performs so that there are no errors when evaluating the data (Isita, 2022). After that, partition the data into three scenarios, namely scenario 1= 70% training, 30% testing, scenario 2 = 75% training, 25% testing, then scenario 3 = 80% training, 20% testing helps compare the best performance of the C4.5 algorithm and C5.0 so that using three scenarios instead of using only one comparison, did not find the best performance if only 21 scenarios 1.

Implementation Of C4.5 and C5.0 Algorithms

The data analysis technique used is scikit-learn using python programming, where sentiment analysis compares the methods, namely Decision Tree Algorithms C4.5 and C5.0. The steps used in its implementation are to use the C4.5 and C5.0 algorithm implementation process with the library:

1. from import sklearn. Tree DecisionTreeClassifier helps classify each algorithm C4.5 and C5.0.
2. From sklearn. Tree import export_graphviz helps create an algorithm tree where the tree will later be in the form of a numeric tree.

System Evaluation (confusion Matrix)

Confusion Matrix is a method commonly used to test data testing accuracy, recall, precision, and error rate as a benchmark (Wardhani & Octaviano, 2020). The testing of this research was carried out by:

1. Calculate accuracy score, classification report, f1 score, precision score, and recall score.
2. Python library from sklearn.metrics import confusion_matrix,accuracy_score, classification_report, f1_score, precision_score, recall_score

Table 1. Confusion Matrix

Actual	Prediction	
Classification	Edible (e)	Poisonous (p)
Edible (e)	TP	FN
Poisonous (p)	FP	TN

P-ISSN: 2656-1743 |E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.XXX
Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

In Table 1, there is information to test 3 scenarios of data testing, namely:
TP = number of non-toxic, edible mushroom conditions (e) that predicts to be correct as an edible class (e)
TN = number of poisonous conditions (p), which predicts correctly as a poison class (p)
FP = number of poisonous mushroom conditions (p) which that predicts to be wrong as an edible class (e)
FN = number of edible mushroom conditions (e), which that predicts to be wrong as a poisonous class (p).

From each positive and negative value, accuracy, precision, recall, f1-score, and specificity calculations can be done to get the results of testing on the C4.5 and C5.0 algorithms. Equations 6, 7, and 8 are the percentages of the total data set identified or assessed.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%. \quad (6)$$

$$\text{Precision} = \frac{TP}{TP+FP} \times 100\%. \quad (7)$$

$$\text{Recall} = \frac{TP}{TP+FN} \times 100\% \quad (8)$$

$$\text{Specificity} = \frac{TN}{TN+FP} \times 100\% \quad (9)$$

$$\text{F1 Score} = 2\,\frac{Precision \times recall}{Precision + recall} \times 100\% \quad (10)$$

RESULTS AND DISCUSSION

Multi-Scenario Dataset Results

Scenario 1 Table 2, The data used to compare the Decision Tree C4.5 and C5.0 algorithms become 70% training data and 30% testing data.

Table 2. Scenario 1

	C4.5	C5.0	Total
Training	70%	70%	5.687
Testing	30%	30%	2.437

Scenario 2 Table 3, The data used to compare the Decision Tree algorithms C4.5 and C5.0 become 75% training data and 25% testing data.

Table 3. Scenario 2

	C4.5	C5.0	Total
Training	75%	75%	6.093
Testing	25%	25%	2.031

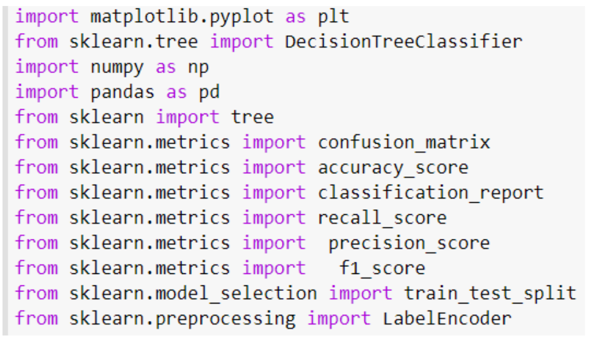
Scenario 3 Table 4, The data used to compare the Decision Tree algorithms C4.5 and C5.0 become 80% training data and 20% testing data.

Table 4. Scenario 3

	C4.5	C5.0	Total
Training	80%	80%	6.499
Testing	20%	20%	1.625

Python Library Implementation

This scikit-learn library module is to help with data processing or data training for data mining needs. One thing this research needs is importing data for its implementation.

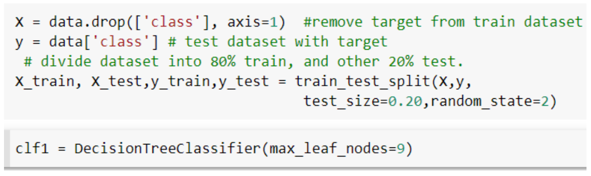


```
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
import numpy as np
import pandas as pd
from sklearn import tree
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import recall_score
from sklearn.metrics import  precision_score
from sklearn.metrics import   f1_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

Figure 5. Python Library

In Figure 5, some of the modules used for this research study are:

1. Matplotlib is a Python library for creating static and interactive visualizations. Matplotlib is useful for confusion matrix graphs.
2. Numpy and pandas are useful for preprocessing datasets.
3. LabelEncoder helps convert data sets to numeric data(Latifah et al., 2019).
4. Python's Skleran library, sklearn.metrics, is used for the confusion matrix library and automatically calculates the accuracy score, recall score, f1_score, specificity, and precision score.
5. LabelEncoder sklearn.model_selection is helpful as a data partition for training and testing.
6. Sklearn. A tree is a classification algorithm C4.5 and C5.0 using a decision tree.



```
X = data.drop(['class'], axis=1)  #remove target from train dataset
y = data['class'] # test dataset with target
 # divide dataset into 80% train, and other 20% test.
X_train, X_test,y_train,y_test = train_test_split(X,y,
                                    test_size=0.20,random_state=2)
```

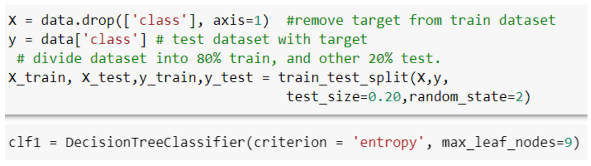
```
clf1 = DecisionTreeClassifier(max_leaf_nodes=9)
```

Figure 6. Implementation of Decision Tree Algorithm C4.5

In Figure 6, In the process of the imported dataset, the following steps are:

1. Data. drop helps remove missing value data
2. Split data is helpful for data partitioning.
3. Second, input the Decision Tree C4.5 Algorithm equation with the Scikit-learn library, DecisionTreeClassifier. DecisionTreeClassifier

is helpful for model classification in the form of decision trees.



```
X = data.drop(['class'], axis=1)  #remove target from train dataset
y = data['class'] # test dataset with target
 # divide dataset into 80% train, and other 20% test.
X_train, X_test,y_train,y_test = train_test_split(X,y,
                                        test_size=0.20,random_state=2)
```

```
clf1 = DecisionTreeClassifier(criterion = 'entropy', max_leaf_nodes=9)
```

Figure 7. Implementation of Decision Tree Algorithm C5.0

In Figure 7. In the process of the imported dataset, the following steps are:

1. Data. drop helps remove missing value data
2. Split data is helpful for data partitioning.
3. Second, input the Decision Tree C5.0 Algorithm equation with the Scikit-learn library, DecisionTreeClassifier. DecisionTreeClassifier is helpful for model classification in the form of decision trees. The difference is that the C5.0 Decision Tree Algorithm quickly establishes criteria. = "entropy" in DecisionTreeClassifier because that is the first process when setting Decision Tree Algorithm C5.0

Confusion Matrix Results in Scenario 2 Algorithm C4.5

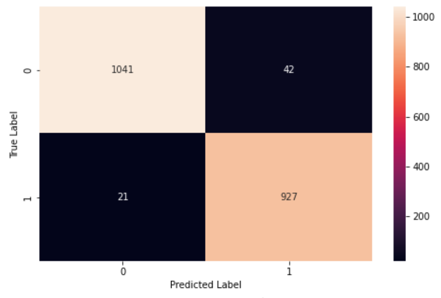
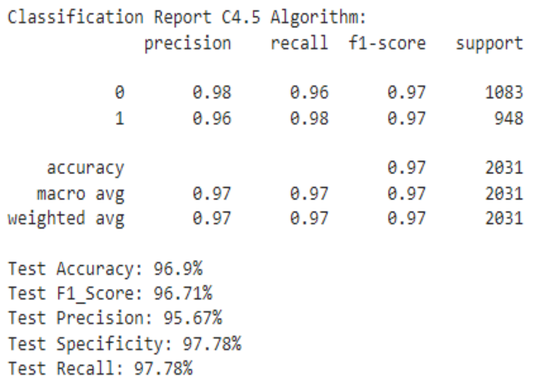


Figure 8. Graph of Scenario 2

In Figure 8, Graph of Confusion Matrix scenario 1 decision tree algorithm C4.5 gets variable results in the form of TP = 1.041, FN = 42, FP = 21, and TN = 927.



```
Classification Report C4.5 Algorithm:
              precision    recall  f1-score   support

           0       0.98      0.96      0.97      1083
           1       0.96      0.98      0.97       948

    accuracy                           0.97      2031
   macro avg       0.97      0.97      0.97      2031
weighted avg       0.97      0.97      0.97      2031

Test Accuracy: 96.9%
Test F1_Score: 96.71%
Test Precision: 95.67%
Test Specificity: 97.78%
Test Recall: 97.78%
```

Figure 9. Performance Metrics Scenario 2

In Figure 9, Confusion Matrix scenario 2 of the decision tree algorithm C5.0 shows that the test data produces 96.9% accuracy, 97.78% recall, 95.67% precision, F1-score 96.71%, and Specificity 97, 78%

Confusion Matrix Results in Scenario 3 Algorithm C4.5

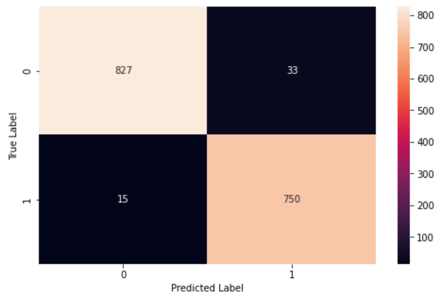
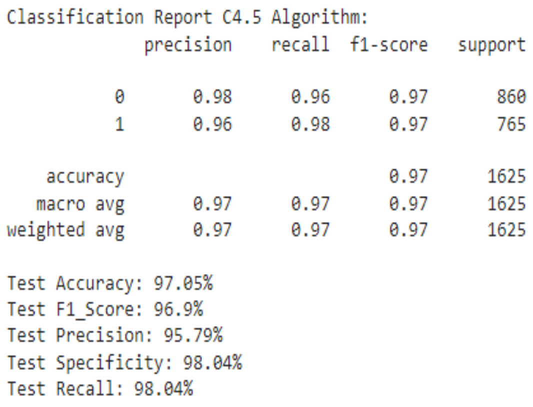


Figure 10. Scenario Graphics 3

In Figure 10, the confusion matrix graph with scenario three partition data, the test requires test data to measure the performance of the Decision Tree Algorithm C4.5. The variables in the confusion matrix are TP = 827, FN = 33, FP = 15, and TN = 750



```
Classification Report C4.5 Algorithm:
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       860
           1       0.96      0.98      0.97       765

    accuracy                           0.97      1625
   macro avg       0.97      0.97      0.97      1625
weighted avg       0.97      0.97      0.97      1625

Test Accuracy: 97.05%
Test F1_Score: 96.9%
Test Precision: 95.79%
Test Specificity: 98.04%
Test Recall: 98.04%
```

Figure 11. Performance Metrics Scenario 3

In Figure 12. Confusion Matrix scenario 3 of the decision tree algorithm C5.0 shows that the test data produces 97.05% accuracy, 98.04% recall, 95.79% precision, F1-score 96.9%, and 98 Specificity, 04%

Confusion Matrix Results in Scenario 1 Algorithm C5.0

P-ISSN: 2656-1743 |E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.XXX
Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

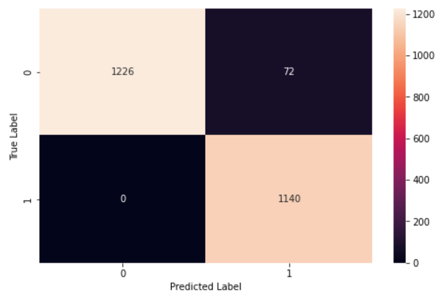
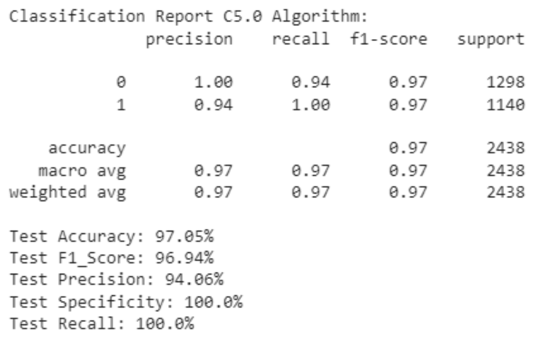


Figure 12. Graph of Confusion Matrix Scenario 1

In Figure 12. Visualization of the Confusion Matrix scenario 1 decision tree algorithm C5.0 there are variable results in the form of TP = 1.226, FN = 72, FP = 36, and TN = 1.104



```
Classification Report C5.0 Algorithm:
              precision    recall  f1-score   support

           0       1.00      0.94      0.97      1298
           1       0.94      1.00      0.97      1140

    accuracy                           0.97      2438
   macro avg       0.97      0.97      0.97      2438
weighted avg       0.97      0.97      0.97      2438

Test Accuracy: 97.05%
Test F1_Score: 96.94%
Test Precision: 94.06%
Test Specificity: 100.0%
Test Recall: 100.0%
```

Figure 13. Performance Metrics Scenario 1

In Figure 13. The confusion matrix scenario 1 of the decision tree algorithm C5.0 shows that the test data produces 97.05% accuracy, 100.0% recall, 94.06% precision, F1-score 96.94%, and Specificity 100, 0%

Confusion Matrix Results in Scenario 2 Algorithm C5.0

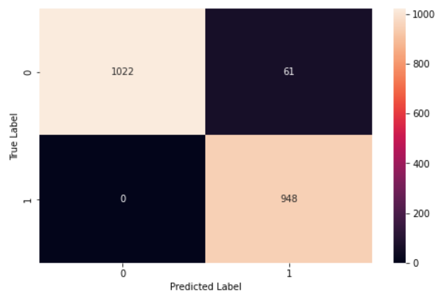
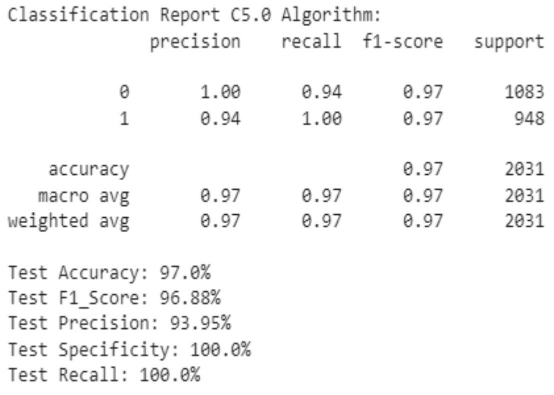


Figure 14. Graph of Confusion Matrix Scenario 2

In Figure 14. Graph of Confusion Matrix scenario 2 decision tree algorithm C5.0 there are variable results in the form of TP = 1.022 , FN = 61, FP = 0 , and TN = 948



```
Classification Report C5.0 Algorithm:
              precision    recall  f1-score   support

           0       1.00      0.94      0.97      1083
           1       0.94      1.00      0.97       948

    accuracy                           0.97      2031
   macro avg       0.97      0.97      0.97      2031
weighted avg       0.97      0.97      0.97      2031

Test Accuracy: 97.0%
Test F1_Score: 96.88%
Test Precision: 93.95%
Test Specificity: 100.0%
Test Recall: 100.0%
```

Figure 15. Performance Metrics Scenario 2

In Figure 15. The confusion matrix scenario 1 of the decision tree algorithm C5.0 shows that the test data produces 97.0% accuracy, 100.0% recall, 93.95% precision, F1-score 96.88%, and Specificity 100, 0%.

Confusion Matrix Results in Scenario 3 Algorithm C5.0

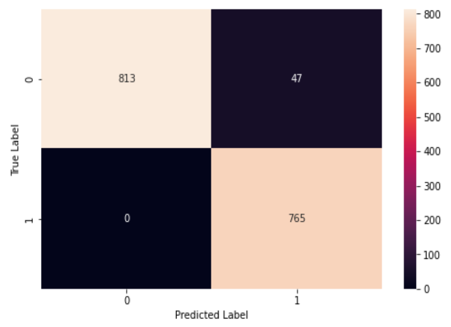
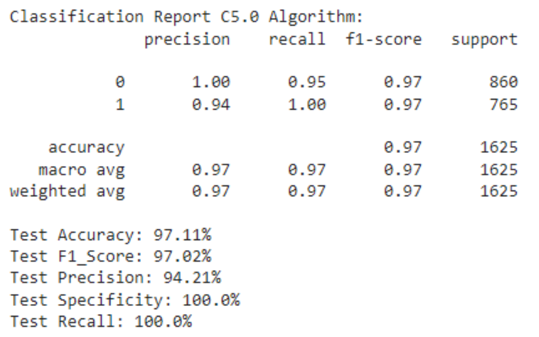


Figure 16. Graph of Confusion Matrix 3

In Figure 16. Graph of Confusion Matrix scenario 3 C5.0 algorithm, there are variable results in the form of TP = 813, FN = 47, FP = 0 , and TN = 765.



```
Classification Report C5.0 Algorithm:
              precision    recall  f1-score   support

           0       1.00      0.95      0.97       860
           1       0.94      1.00      0.97       765

    accuracy                           0.97      1625
   macro avg       0.97      0.97      0.97      1625
weighted avg       0.97      0.97      0.97      1625

Test Accuracy: 97.11%
Test F1_Score: 97.02%
Test Precision: 94.21%
Test Specificity: 100.0%
Test Recall: 100.0%
```

Figure 17. Performance Metrics Scenario 3

In Figure 17. Confusion Matrix scenario 3 of the decision tree algorithm C5.0 shows that the test data

Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

produces 97.11% accuracy, 100.0% recall, 94, 21% precision, F1-score 97.02%, and Specificity 100, 0%.

Python Library Decision Tree Classification

Use Several conditions in selecting the decision tree model, namely the model with the most number of rules. The rules of the comparative algorithm in this study have created the most rule decision trees and the highest accuracy model accuracy. Testing on the decision tree algorithm by dividing the attributes on each node, so the use of splitting testing and training is to create a decision tree taken from the training dataset. (Patel & Prajapati, 2018).

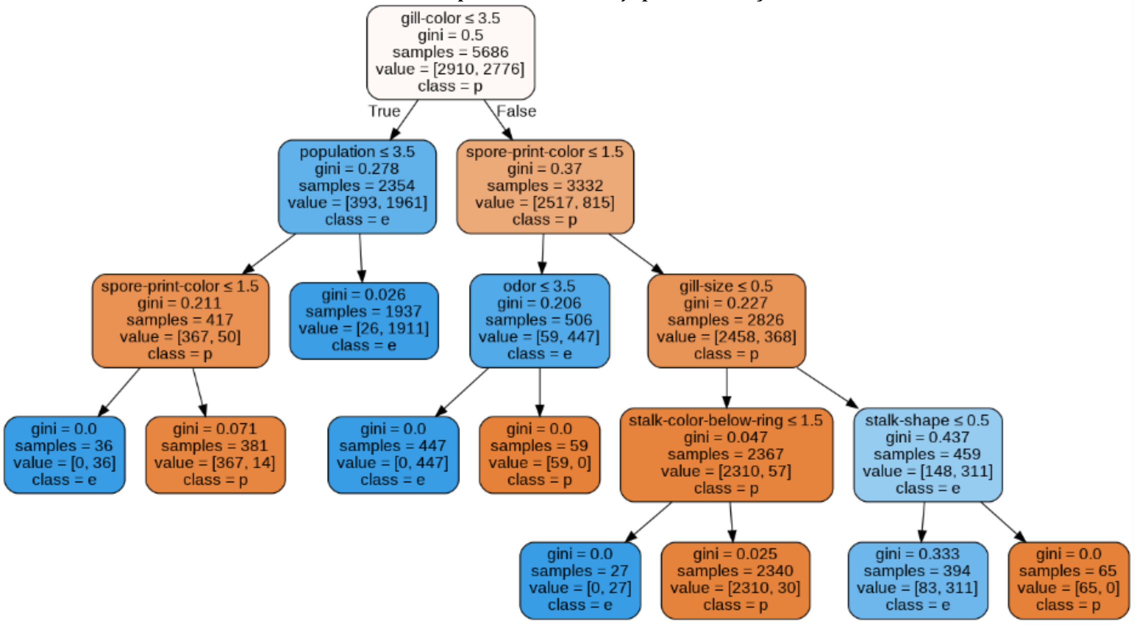


Figure 18. Decision Tree Results Algorithm C4.5 Scenario 1

In figure 18, the Graphviz tool produces a tree depth of 4, where the root is at 0, and the two lower leaves are at 4th depth. The minimum sample used at each node is 29. At the 1st depth, in the spore-print-colour poisonous (p) branch, the Gini index is 0.37, indicating that eight samples have the same class, namely deadly, and the rest are on edible branches (e) or non-toxic. In a decision tree, nodes are usually written in the form "Are the types of mushrooms poisonous (poisonous) and edible (e)," While in the decision tree model obtained by scikit-learn, categorical is considered numerical so form a knot "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 3.5". In the preprocessing process, using a LabelEncoder, Convert categorical or discrete data to numeric and process like continuous data in a decision tree. Label Encoder converts absolute into ordered numbers, which is inappropriate, especially if the data is not ordinal. The decision tree evaluation results show an accuracy value of 96.92% in the Decision Tree algorithm C4.5 Scenario 1. Training data and testing data are encoded simultaneously. The decision tree will have difficulty predicting the data inputted separately and is still in the form of categorical data.

P-ISSN: 2656-1743 |E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.XXX

Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

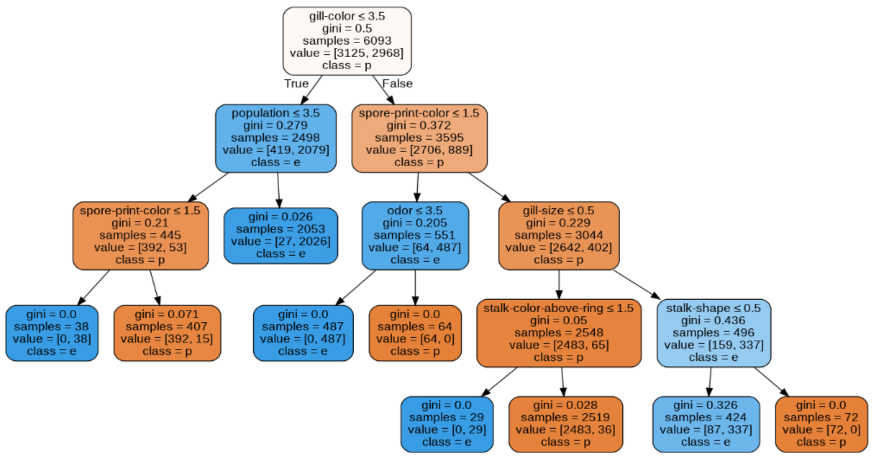


Figure 19. Decision Tree Results Algorithm C4.5 Scenario 2

In figure 19, the Graphviz tool generates a tree depth of 4, where the root is at depth 0, and the two lower leaves are at depth 4. The minimum sample used at each node is 29. At depth 1, the spore-coloured toxic branch (p) shows that the Gini index is 0.372, which indicates that eight samples have the same class, which is toxic, and the rest are at the edible branch (e) or non-toxic.

In a decision tree, nodes are usually written in the form "Are the types of mushrooms poisonous (poisonous) and edible (e)," While in the decision tree model obtained by scikit-learn, categorical is considered numerical so that form a knot "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 3.5". The results of the decision tree evaluation show an accuracy value of 96.90% on the Decision Tree algorithm C4.5 Scenario 2

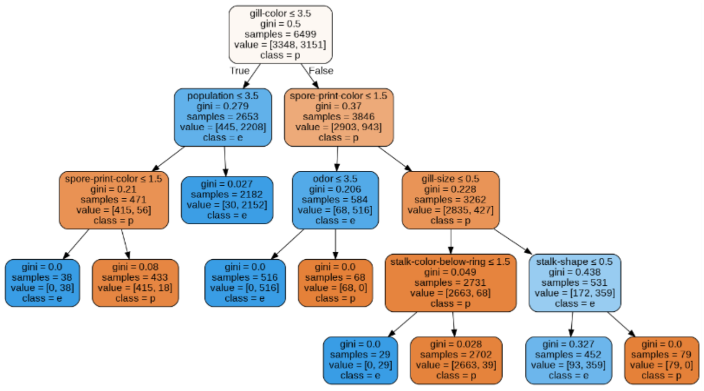


Figure 20. Decision Tree Results Algorithm C4.5 Scenario 3

Figure 20, from the Graphviz tool, produces a tree depth of 4, where the root is at depth 0, and the two lower leaves are at depth 4. The minimum sample used at each node is 29. At the 1st depth, in the spore-print-colour poisonous (p) branch, the Gini index is 0.37, indicating that eight samples have the same class, namely toxic, and the rest are on edible branches (e) or non-toxic. In a decision tree, nodes are usually written in the form "Are the types of mushrooms poisonous (poisonous) and edible (e)," While in the decision tree model obtained by scikit-learn, categorical is considered numerical so that form a knot "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 3.5". The results of the decision tree evaluation show an accuracy value of 97.05% on the Decision Tree algorithm C4.5 Scenario 3

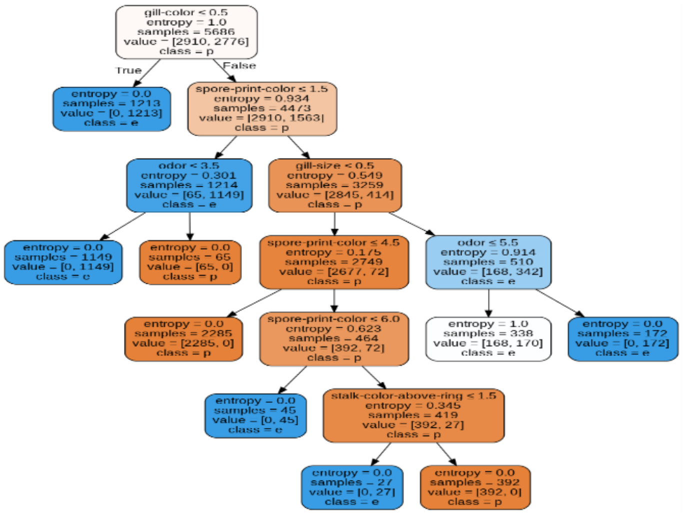


Figure 21. Decision Tree Results Algorithm C5.0 Scenario 1

In figure 21, the results generated from the Graphviz tool and the entropy criterion for the C5.0 algorithm have a tree depth of 4, where the root is at the 0th depth, and the bottom two leaves are at the 4th depth. The minimum sample used at each node is 27. At the 1st depth, in the spore-print-colour poisonous (p) branch, it is seen that the entropy is 0.934, which indicates that eight samples have the same class, which is poisonous, and the rest are in the branch. edible (e) or non-toxic.

In a decision tree, nodes are usually written in the form "Are the types of mushrooms poisonous (poisonous) and edible (e)". While in the decision tree model obtained by scikit-learn, categorical is considered numerical so form a knot "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 0.0" because the entropy criterion is 0.0, then it is not branched. The results of the decision tree evaluation show an accuracy value of 97.05% on the Decision Tree algorithm C5.0 Scenario 1

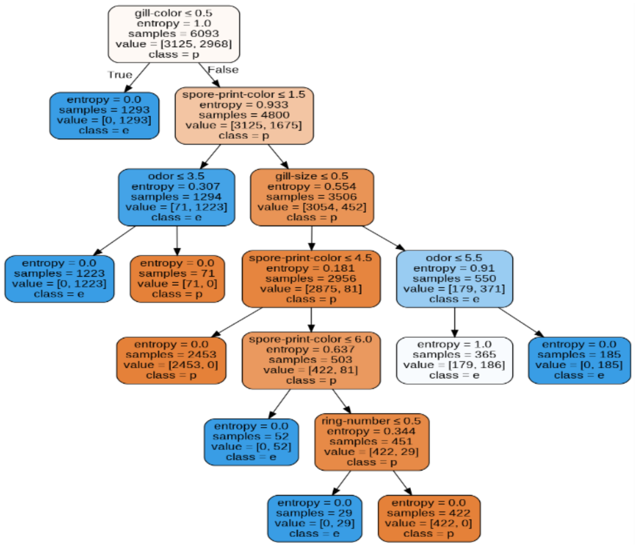


Figure 22. Decision Tree Results Algorithm C5.0 Scenario 2

Figure 22 shows the results generated from the Graphviz tool and the entropy criterion for the C5.0 algorithm with a tree depth of 4, where the root is at the 0th depth, and the bottom two leaves are at the 4th depth. The minimum sample used at each node is 29. At the 1st depth, in the spore-print-colour poisonous (p) branch, it is seen that the entropy is 0.933, which indicates that eight samples have the same class, which is poisonous, and the rest are in the branch. edible (e) or non-toxic.

In a decision tree, nodes are usually written in the form "Are the types of mushrooms

P-ISSN: 2656-1743 |E-ISSN: 2656-1735
DOI: https://doi.org/10.34288/jri.v4i3.XXX

Accredited rank 3 (SINTA 3), excerpts from the decision of the Minister of RISTEK-BRIN No. 200/M/KPT/2020

poisonous (poisonous) and edible (e)". While in the decision tree model obtained by scikit-learn, categorical is considered numerical so form a knot "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 0.0" because the entropy criterion is 0.0 then it is not branched. The decision tree evaluation results show an accuracy value of 97.00% on the Decision Tree algorithm C5.0 Scenario 2

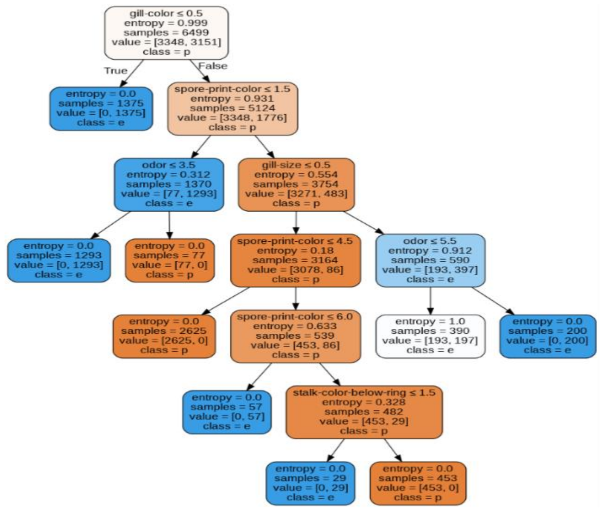


Figure 23. Decision Tree Results Algorithm C5.0 Scenario 3

In Figure 23, the results generated from the Graphviz tool and the entropy criterion for the C5.0 algorithm have a tree depth of 4, where the root is at the 0th depth, and the bottom two leaves are at the 4th depth. The minimum sample used at each node is 29. At the 1st depth, in the spore-print-color poisonous (p) branch, it is seen that the entropy is 0.931, which indicates that eight samples have the same class, which is poisonous, and the rest are in the branch. edible (e) or non-toxic. In a categorical decision tree, nodes are usually written in the form "Are the types of mushrooms poisonous (poisonous) and edible (e)". While in the decision tree model obtained by scikit-learn, categorical is considered numeric so that the vertices are of the form "what type of poisonous mushroom is 1.5" and "whether the type of mushroom is non-toxic 0.0" because the entropy criterion is 0.0 then it is not branched. The decision tree evaluation results show an accuracy value of 97.11% in the Decision Tree algorithm C5.0 Scenario 3

Comparison Results

This stage has resulted in a comparison process of research accuracy, namely classification using the Decision Tree Algorithms C4.5 and C5.0.

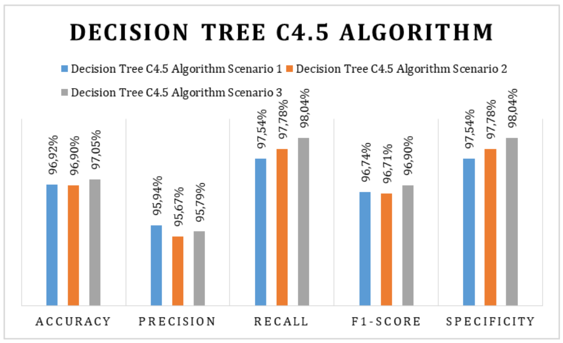


Figure 24. Decision Tree Graph C4.5 Algorithm

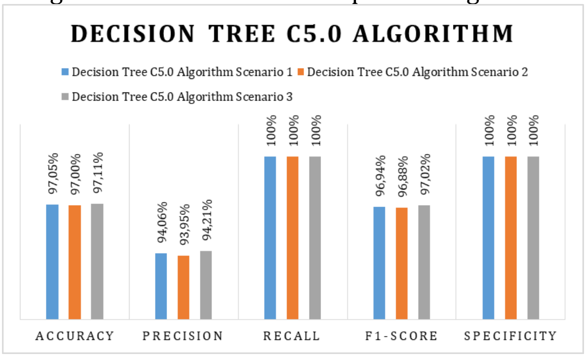


Figure 25. Decision Tree Graph C4.5 Algorithm

In Figure 24, the Decision Tree Graph of the C4.5 Algorithm gets an accuracy of 96.92% for scenario 1, 96.90% for scenario 2, and 97.05% for scenario 3. Precision 95.94% for scenario 1.95, 94% for scenario 2, 95.79% for scenario 3. Recall 97.54% for scenario 1, 97.78% for scenario 2, 95.79% for scenario 3. F1-Score 96.74% for scenario 1, 96.71% for scenario 2, 96.90% for scenario 3. Specificity

97.54% for scenario 1, 97.78% for scenario 2, 98.04% for scenario 3.

Figure 25, Decision Tree Graph C4.5 Algorithm gets 97.05% accuracy for scenario 1, 97.00% for scenario 2, and 97.11% for scenario 3. Precision for scenario 1 94.06%, 93.95% for scenario 2, 94.21% for scenario 3. Recall 100% for scenario 1, 100% for scenario 2, 100% for scenario 3. F1-Score 96.94% for scenario 1, 96.88% for scenario 2, 97.02% for scenario 3. 100% specificity for scenario 1, 100% for scenario 2, 100% for scenario 3.

CONCLUSIONS AND SUGGESTIONS

Conclusion

The author's research shows that the difference between the author's research and previous research is a different dataset because this study compares a multi-scenario dataset. Hence, the accuracy results are not much different and display categorical data converted into numeric data in the python library scikit-learn. Identifying poisonous mushrooms using the Decision Tree C5.0 Algorithm gets an accuracy of 97.05% for scenario 1, 97.00% for scenario 2, and 97.11% for scenario 3. At the same time, the Decision Tre C4.5 algorithm produces an accuracy of 96, 92% for scenario 1, 96.90% for scenario 2, and 97.05% for scenario 3. Comparing the classification results' performance, the Decision Tree C5.0 algorithm in scenario 3 has the highest accuracy for identifying poisonous mushrooms.

Suggestion

The weakness in this study is that the author only found some references to the Decision Tree C5.0 Algorithm so that further research can compare other algorithms.

REFERENCES

Fajri, M., Ramadhan, A., & Hidayatullah, R. (2021). Implementation of boosting on the C5 . 0 algorithm in the health development index data. *Journal of Science and Technology*, *10*(1), 12–15. https://doi.org/10.22487/25411969.2021.v10.i1.15466

Fitriani, A., Krisnawati, Y., Anorda, M. O. R., & Lanjarini, K. (2018). Jenis-Jenis Dan Potensi Jamur Makroskopis Yang Terdapat Di Pt Perkebunan Hasil Musi Lestari Dan PT. Djuanda Sawit Kabupaten Musi Rawas. *Jurnal Biosilampari: Jurnal Biologi*, *1*(1), 21–28. https://doi.org/10.31540/biosilampari.v1i1.49

Isita. (2022). *How to split a Dataset into Train and Test Sets using python*. Geeks for Geeks. https://www.geeksforgeeks.org/how-to-split-a-dataset-into-train-and-test-sets-using-python/

Kurniawan, Y. I. (2018). Perbandingan Algoritma Naive Bayes dan C.45 dalam Klasifikasi Data Mining. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(4), 455. https://doi.org/10.25126/jtiik.201854803

Latifah, R., Setia, E., & Edi, P. (2019). Model Decision Tree untuk Prediksi Jadwal Kerja menggunakan Scikit-Learn. *SEMNASTEK*, *October*, 1–6.

Lutfi Arisandi, M. (2019). Perbandingan Metode Naive Bayes Dan ID3 Untuk Klasifikasi Jamur Beracun Dan Tidak Beracun. *Simki-Techsain*, *3*(April), 33–35.

Mardi, Y. (2019). Data Mining : Klasifikasi Menggunakan Algoritma C4 . 5 Data mining merupakan bagian dari tahapan proses Knowledge Discovery in Database (KDD) . Jurnal Edik Informatika. *Jurnal Edik Informatika*, *2*.

Meng, X., Zhang, P., Xu, Y., & Xie, H. (2020). Construction of decision tree based on C4.5 algorithm for online voltage stability assessment. *Electrical Power and Energy Systems*, *118*(December 2019), 1–8. https://doi.org/10.1016/j.ijepes.2019.105793

Mulyana, Y. (2019). The Detection of Pathogenic Fungi on Prayer Rugs of The Mosques at Jatinangor Campus of Universitas Padjadjaran. *Journal of Medicine and Health*, *2*(3), 806–817.

Myrianthous, G. (2021). *Is There Any Difference Between Scikit-Learn and Sklearn*. Towards Data Science.

Ndifon, E. M. (2022). Journal of Asia-Paci fi c Biodiversity Systematic appraisal of macrofungi (Basidiomycotina : Ascomycotina) biodiversity of Southern Africa : Uses , distribution , checklists. *Journal of Asia-Pacific Biodiversity*, *15*(1), 80–85. https://doi.org/10.1016/j.japb.2021.09.008

Pratiwi, R., Hayati, M. N., & Prangga, S. (2020). Perbandingan Klasifikasi Algoritma C5.0 Dengan Classification And Regression Tree (Studi Kasus : Data Sosial Kepala Keluarga Masyarakat Desa Teluk Baru Kecamatan Muara Ancalong Tahun 2019). *Jurnal Ilmu Matematika Dan Terapan*, *14*(2), 267–278. https://ojs3.unpatti.ac.id/index.php/barekeng/article/view/1835

Safii, M. (2018). Implementasi Data Mining Dengan Metode Pohon Keputusan Algoritma ID3

Untuk Menentukan Status Mahasiswa. *Jurnal Mantik Penusa*, *2*(1), 82–86. https://e-jurnal.pelitanusantara.ac.id/index.php/mantik/article/view/353

Tank, K., & Mumbai, N. (2021). A Comparative Study on Mushroom Classification using Supervised Machine Learning Algorithms. *International Journal of Trend in Scientific Research and Development (IJTSRD)*, *5*(5), 716–723.

Vanfretti, L., & Arava, V. S. N. (2020). Decision tree-based classification of multiple operating conditions for power system voltage stability assessment. *Electrical Power and Energy Systems*, *123*(April), 106251. https://doi.org/10.1016/j.ijepes.2020.106251

Wardhani, L. K., & Octaviano, D. (2020). Perbandingan Seleksi Fitur Term Frequency & Tri-Gram Character Menggunakan Algoritma Naïve Bayes Classifier (Nbc) Pada Tweet Hashtag # 2019gantipresiden. *Jurnal Kajian Ilmu Dan Teknologi*, *9*(1), 103–114.

Zubair, A., & Rofiqul Muslikh, A. (2017). Identifikasi jamur menggunakan metode k-nearest neighbor dengan ekstraksi ciri morfologi. *Fakultas Teknologi Informasi*, 965–972.