Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th February 2025. Vol. 4. No. 2; e-ISSN: 2808-4519

Enhancing Election Staff Selection through Decision Tree-Based Classification

Rizal Rayyan Firdaus[1]*, Nana Suarna[2], Irfan Ali[3], Ahmad Rifai[4]

[1,2,3,4]STMIK IKMI Cirebon

rizalmeong48@gmail.com [1]*, nana.ikmi@gmail.com [2], irfanali0.0@gmail.com [3], a.rifaaii1408@gmail.com[4]

Abstract

The selection of competent election committee members is a critical aspect in ensuring the success of a fair and transparent election process. However, the subjective nature of this selection process necessitates a data-driven approach to optimize the selection of officials who meet the required competency criteria. This research aims to classify the competencies of prospective election committee members using the Decision Tree algorithm based on demographic data and technological attributes of the population. The study employs the Knowledge Discovery in Databases (KDD) methodology, which includes the stages of data selection, preprocessing, transformation, data mining, and evaluation. In this process, data collected through various attributes are processed to build a classification model. The Decision Tree algorithm is applied to extract patterns from the data, resulting in a decision tree that can classify individuals into different competency classes based on existing features. The research findings indicate that the Decision Tree algorithm effectively classifies respondents into several competency classes that represent varying levels of skills and interest in the election process. The model shows that Class 4 is the dominant class, indicating that most respondents have moderate competency in technological skills and interest in elections. Class 3 represents individuals with higher technological skills but moderate interest, while Classes 2 and 1 represent individuals with varying combinations of interest and skills. This study demonstrates that using the Decision Tree algorithm in the KDD process is highly effective in objectively classifying the competencies of prospective election committee members. By analyzing the interactions among relevant attributes, the model provides insights that can improve the accuracy of election official selection. This data-driven approach can be adapted to other contexts requiring competency classification, offering broader benefits for various criteria-based selection systems.

Keywords: *Decision Tree, Knowledge Discovery in Database, Machine Learning, Election Officer, Classification*

1. Introduction

Elections are fundamental in democratic systems, serving as a mechanism for citizens to choose leaders and influence governance.[1] The success of this process depends heavily on the reliability and efficiency of managing voter and election staff data. However, the increasing complexity and diversity of citizen data pose significant challenges for traditional data management methods. These methods often struggle to handle large volumes of unstructured and variable data, leading to inaccuracies in data processing and potential errors in the selection of election staff. Such errors not only undermine the quality of elections but also risk diminishing public trust in the democratic system.

In the digital era, advancements in information technology offer promising solutions to these challenges. One such technology is the Decision Tree algorithm, a powerful data mining technique widely recognized for its ability to classify and process data effectively. By leveraging the Decision Tree algorithm, it becomes possible to address the challenges associated with managing large datasets while improving the accuracy and efficiency of data-driven decision-making processes [2]. This algorithm holds particular potential in the context of elections, where its application can enhance the selection process for election staff by identifying key characteristics from complex datasets, ensuring that only qualified individuals are chosen.

This study focuses on exploring the implementation of the Decision Tree algorithm in managing citizen data to support the selection of election staff. The research is motivated by the need to overcome inefficiencies in traditional data processing methods, which are often time-consuming and prone to errors. By adopting a structured and systematic approach, this study aims to demonstrate how advanced classification techniques can be utilized to streamline data management, reduce inaccuracies, and enhance the overall quality of elections.

The anticipated outcomes of this study include significant contributions to the field of informatics by advancing methods for data classification and management. Furthermore, the research seeks to improve transparency and accountability in the electoral process, addressing public concerns about fairness and reliability. Beyond its application in elections, the findings from this study have the potential to inform broader efforts in data processing and governance, offering insights into how similar challenges can be addressed across various domains. Ultimately, this study aims to strengthen democratic processes through the integration of innovative technology, contributing to both academic knowledge and practical solutions for real-world problems.

1.1. Decision Tree

A Decision Tree is a machine learning algorithm used for classification and regression tasks. It models decisions as a tree-like structure, where each internal node represents a test on a feature, each branch represents the outcome of the test, and each leaf node represents a final decision or output. The tree is constructed by recursively splitting the dataset based on features that maximize information gain (for classification) or minimize variance (for regression). Decision Trees are intuitive, easy to interpret, and can handle both numerical and categorical data, making them versatile for various applications.[2]

1.2. Knowledge Discovery Database

Knowledge Discovery in Databases (KDD) is the process of identifying valid, novel, potentially useful, and understandable patterns from large datasets. It involves multiple steps, starting with data selection, preprocessing, and transformation, followed by data mining to extract patterns, and finally, interpreting and evaluating the results. KDD serves as the foundation for extracting actionable insights and knowledge from raw data, often leveraging advanced techniques such as machine learning, statistics, and data visualization to uncover hidden relationships and trends. [3]

2. Research Method

2.1. Research Method

The methodology employed in this research is Knowledge Discovery in Databases (KDD). The workflow or sequence of steps utilized throughout the study is depicted in Figure 1 below, outlining the systematic approach taken for data analysis and interpretation**.**

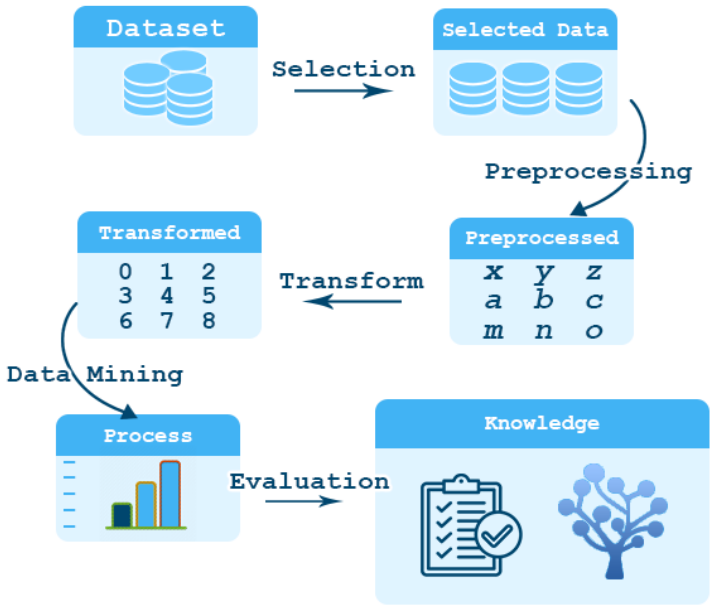

Fig. 1: Research Method

3. Result and Discussion

3.1. Selection

This stage aims to select the data that will be used in the classification process. The dataset used in this study consists of primary data collected through surveys and interviews with the population. As the author, I utilized a dataset containing questionnaires that address various aspects related to elections and technology. This data was stored in Microsoft Excel (XLSX) file format.

Table 1: Collected Data

NAME	SEX	BIRTH	AGE	AGE_Point	Q1_Komputer	Q2_Smartphone	Q3_Kepemiluan	Q4_Interest	AVG_POINT	LABEL
IRPAN XXXXXXX	M	1980-05-27	44	3	5	3	3	2	3.2	NO
PIPIT XXXXXXX	F	1986-07-16	38	4	5	4	3	2	3.6	KPPS-6/KPPS-7

NAURA XXXXXXX	F	2004-05-06	20	5	5	5	3	2	4.0	KPPS-4/KPPS-5
NAJWA XXXXXX	F	2014-10-01	10	0	1	1	1	1	0.8	NO
SUPYANI	M	1963-03-03	61	1	4	4	5	3	3.4	NO
SRI XXXX	F	1966-01-03	58	2	1	4	4	4	3.0	NO
DIAN XXXXXX	F	1992-03-16	32	4	5	5	4	2	4.0	KPPS-4/KPPS-5
MAHENDRA XXXXX	M	1983-06-21	41	3	2	4	4	2	3.0	NO

.

3.2. Preprocessing

Preprocessing is the initial step in processing raw data. This stage involves identifying the data, removing incomplete or duplicate entries, and reducing features by eliminating irrelevant attributes. This dataset is comprehensive and does not contain any missing values. Feature reduction or attribute elimination in this study was conducted using the Correlation Matrix operator.

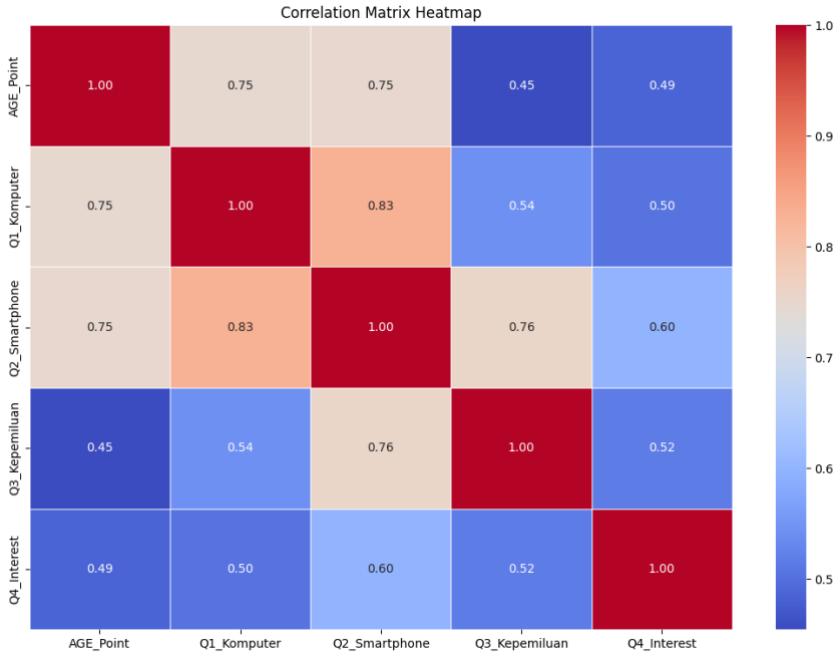

Fig. 2: Correlation Matrix

The figure above illustrates the results of the Correlation Matrix displayed using a Heatmap in Google Colab. The depth of the color in the matrix represents the strength of the correlation, with darker colors indicating stronger attribute relationships. Based on the Correlation Matrix analysis, the author removed several attributes to enhance model efficiency and accuracy. The attributes retained for use in the study are **AGE_Point**, **Q1_Komputer**, **Q2_Smartphone**, **Q3_Kepemiluan (Election)**, and **Q4_Interest**. The use of these attributes aims to simplify the model, reduce noise, and focus the analysis on features that are more relevant and impactful for prediction results.

After preprocessing, only six attributes were selected: **AGE_Point**, **Q1_Komputer**, **Q2_Smartphone**, **Q3_Kepemiluan(Election)**, **Q4_Interest**, and **LABEL**. These attributes were retained to streamline the model and focus on features essential for prediction accuracy. The results of preprocessing are summarized in the table below, which highlights the six selected attributes: AGE_Point, Q1_Komputer, Q2_Smartphone, Q3_Kepemiluan, Q4_Interest, and LABEL. These attributes were carefully chosen to streamline the analysis and enhance the efficiency and accuracy of the classification process.

Table 2: Result of Transformation

No	Age_Point	Q1_Komputer	Q2_Smartphone	Q3_Kepemiluan	Q4_Interest	LABEL
0	3	5	3	3	2	NO
1	4	5	4	3	2	KPPS-6/KPPS-7
2	5	5	5	3	2	KPPS-4/KPPS-5
3	0	1	1	1	1	NO
4	1	4	4	5	3	NO
5	2	1	4	4	4	NO
6	4	5	5	4	2	KPPS-4/KPPS-5

7	3	2	4	4	2	NO
8	2	1	2	4	2	NO
9	3	2	4	4	2	NO

3.3. Transformation

Transformation refers to the step of converting data from its initial format or representation into another format more suitable for testing during the data mining phase. This step is crucial in preprocessing, as its outcomes are directly utilized in the data mining process. In this study, the "LABEL" attribute in the dataset was transformed from its original textual representations ("KPPS-1," "KPPS-2/3," "KPPS-4/5," "KPPS-6/7," and "NO") into numerical values (0, 1, 2, 3, and 4).

Table 3: Result

No	Age_Point	Q1_Komputer	Q2_Smartphone	Q3_Kepemiluan	Q4_Interest	LABEL
0	3	5	3	3	2	4
1	4	5	4	3	2	3
2	5	5	5	3	2	2
3	0	1	1	1	1	4
4	1	4	4	5	3	4
5	2	1	4	4	4	4
6	4	5	5	4	2	2
7	3	2	4	4	2	4
8	2	1	2	4	2	4
9	3	2	4	4	2	4

3.4. Data Mining

In this stage, the research focuses on model development using the Decision Tree method. The implementation involves detailed steps, including determining optimal parameters for the Decision Tree, such as the node selection criteria and the maximum tree depth. Below is the application of the data mining process performance on Google Colab using the Decision Tree method.

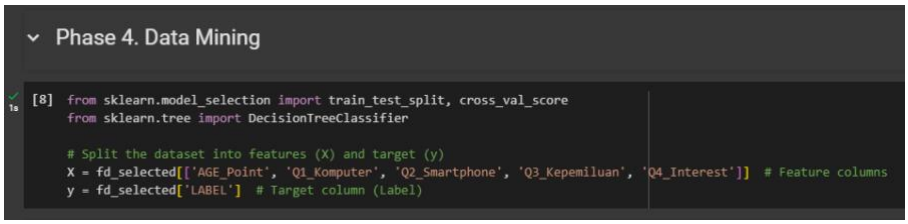

Fig. 3: Split the dataset into features and target

The figure above explains that the code in step 8 utilizes the Scikit-learn library to build a classification model using the Decision Tree algorithm. The dataset is divided into feature variables (X), including columns such as AGE_Point, Q1_Komputer, Q2_Smartphone, Q3_Kepemiluan, and Q4_Interest, and the target variable (y), represented by the LABEL column, which serves as the prediction target. These features characterize candidates by attributes like age points, technological capabilities, and task interest. The code prepares the dataset for further processing by splitting it into features and target variables, essential for training the Decision Tree classification model.

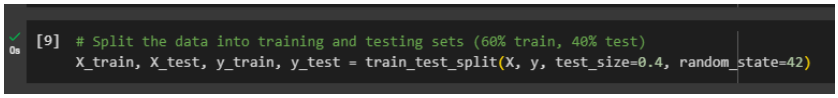

Fig. 4: Split the data into training and testing sets

The figure above illustrates the process of splitting the dataset into training and testing sets using Scikit-learn's train_test_split function. A proportion of 60% of the data is allocated for training and 40% for testing, specified by the parameter *test_size=0.4*. The feature data (X) and target data (y) are divided into four subsets: *X_train, X_test, y_train,* and *y_test*. The training data is used to train the model, while the testing data evaluates its performance. The use of *random_state=42* ensures consistent data splitting across runs.

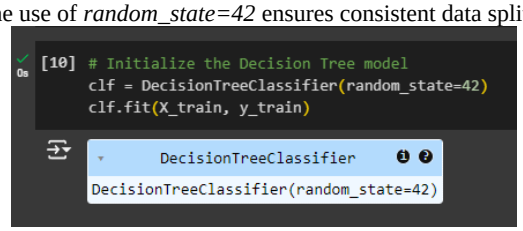

Fig. 5: Initialize the Decision Tree Model

The figure above describes the initialization of a Decision Tree model using Scikit-learn's "*DecisionTreeClassifier*" with the parameter "*random_state=42"*, ensuring consistent results across runs. The model is then trained on the training data *("X_train and y_train"*) using the *".fit()"* method. This training process enables the model to learn the relationships between the features *"(X_train)"* and the target labels *(y_train*), allowing it to predict target labels based on new input features.

```
# Validate using cross-validation using 10-fold
cv_scores = cross_val_score(clf, X_train, y_train, cv=10)

# Print cross-validation scores
print("Cross-validation scores for each fold:", cv_scores)
print(f"Average cross-validation accuracy: {cv_scores.mean() * 100:.2f}%")
```

Fig. 6: Cross Validation

The code in the figure above performs 10-fold cross-validation to assess the performance of the Decision Tree model using the "*cross_val_score"* function. The training data *"(X_train and y_train)"* is split into 10 subsets (folds), with the model being trained and tested on each fold iteratively. The accuracy scores for each fold are stored in "*cv_scores*". Additionally, the code outputs the accuracy for each fold and calculates the average accuracy, providing an overall measure of the model's ability to generalize to new data.

```
Cross-validation scores for each fold: [0.9375     0.90625    0.84375    0.875      0.875      0.90625
 0.84375    0.9375     0.9375     0.93548387]
Average cross-validation accuracy: 89.98%
```

Fig. 7: Cross Validation Result

The cross-validation results in the figure indicate that the Decision Tree model performs exceptionally well, with accuracy scores ranging from 84.38% to 93.75% across folds, and an overall average accuracy of 89.98%. These results highlight the model's consistent performance on different subsets of data, with minimal variation between folds. The high average accuracy and low variance suggest that the model is stable and reliable for making predictions on unseen data. Based on a previous study titled "Penerapan Data Mining Menggunakan Algoritma Decision Tree C4.5 Untuk Memprediksi Mahasiswa Drop Out Di Universitas Wiraraja", this research achieved an accuracy of 81.82%. Comparatively, the current study demonstrates better accuracy, highlighting the effectiveness of its approach in leveraging Decision Tree algorithms for classification tasks[4]. This study also surpasses previous research based on Analisis Cuaca di Australia Menggunakan Algoritma J48 Decision Tree, which achieved an accuracy of around 84.19%. The comparison highlights the superior effectiveness of the methodology and parameter optimization in the current study for enhancing classification accuracy.[5].

3.5. Interpretation and Evaluation

The purpose of model interpretation is to understand how the model makes predictions, such as identifying important features and the decision rules applied. This is particularly useful for transparent models like Decision Trees. Evaluation, on the other hand, measures model performance using metrics like accuracy, precision, recall, F1-score, and cross-validation. These evaluations ensure that the model performs well not only on training data but also generalizes effectively to unseen data, providing insights into its reliability for practical applications.

Phase 5. Evaluation & Interpretation

```
[13] from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
     import seaborn as sns
     import matplotlib.pyplot as plt
     import numpy as np

     # Make predictions on the test set
     y_pred = clf.predict(X_test)

     # Evaluate the model
     print("Classification Report:")
     print(classification_report(y_test, y_pred,  zero_division=0))
```

Fig. 8 Prediction on the test set and evaluate model

The process in the figure above is used to evaluate the performance of the Decision Tree model on the test data (X_test). First, the model makes predictions on the test data using the ".predict()" method, and the results are stored in the variable y_pred. Evaluation is then performed by generating a classification report with Scikit-learn's "classification_report function", providing metrics such as precision, recall, F1-score, and support for each class. The parameter "zero_division=0" prevents errors from divisions by zero in cases where a class has no predicted values. And the result of the process above can be seen below.

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

           0       1.00      1.00      1.00         2
           1       0.90      0.90      0.90        10
           2       0.92      0.92      0.92        12
           3       0.84      0.87      0.86        31
           4       0.98      0.97      0.98       158

    accuracy                           0.95       213
   macro avg       0.93      0.93      0.93       213
weighted avg       0.95      0.95      0.95       213
```

Fig. 9: Classification Report

The classification report results show that the Decision Tree model performs excellently in classifying data across various classes. The model achieves an overall accuracy of 95% with 213 samples. Class 0 shows perfect predictions with precision, recall, and F1-score of 1.00. Classes 1, 2, and 3 perform well with F1-scores of 0.90, 0.92, and 0.86, respectively. Class 4, with the highest sample count (158), excels with an F1-score of 0.98. The macro and weighted averages of 0.93 and 0.95 confirm solid model performance, with minimal bias towards larger classes.

The Confusion Matrix generated by the Decision Tree model for the classification applied to the test (train set) can be seen in the image below.

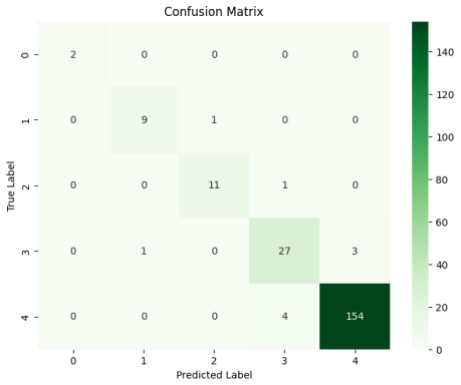

Fig. 10: Confusion Matrix Decision Tree

Based on the confusion matrix shown in the image above, the following can be observed:

1. *Accuracy*

   $True\ Positive\ (TP) = 2 + 9 + 11 + 27 + 154 = 203$

   $\text{Total Data} = 203 + 0 + 2 + 1 + 7 = 213$

   Thus

$$Accuracy = \frac{True\ Positive}{Total\ Data} = \frac{203}{213} = 0.95$$

2. For Class 0 (True Label 0), the model correctly predicted 2 samples as Class 0, with no misclassification into other classes. This indicates perfect classification performance for this specific class.

   TP = 2 FP = 0 FN = 0

$$Precision = \frac{TP}{TP + FP} = \frac{2}{2} = 1.0$$

$$Recall = \frac{TP}{TP + FN} = \frac{2}{2} = 1.0$$

$$F1 - \text{Score} = 2 \times \frac{Precission\ \times\ Recall}{Precission + Recall}$$

$$F1 - \text{Score} = 2 \times \frac{1.0\ \times\ 1.0}{1.0 + 1.0} = \ 1.0$$

3. For Class 1 (True Label 1), the model correctly predicted 9 samples as Class 1, but made 1 misclassification, where a sample from Class 1 was predicted as Class 2. This indicates a small error in the model's ability to distinguish between Class 1 and Class 2.

   TP = 9 FP = 1 FN = 1

$$Precision = \frac{TP}{TP + FP} = \frac{9}{9 + 1} = 0.9$$

$$Recall = \frac{TP}{TP + FN} = \frac{9}{9 + 1} = 0.9$$

$$F1 - \text{Score} = 2 \times \frac{Precission \times Recall}{Precission + Recall}$$

$$F1 - \text{Score} = 2 \times \frac{0.9 \times 0.9}{0.9 + 0.9} = 0.9$$

4. For Class 2 (True Label 2), the model correctly predicted 11 samples as Class 2, but made 1 misclassification where a sample from Class 2 was predicted as Class 3. This indicates a minor error in distinguishing between Class 2 and Class 3.

TP = 11 FP = 1 FN = 1

$$Precision = \frac{TP}{TP + FP} = \frac{11}{11 + 1} = 0.9167$$

$$Recall = \frac{TP}{TP + FN} = \frac{11}{11 + 1} = 0.9167$$

$$F1 - \text{Score} = 2 \times \frac{Precission \times Recall}{Precission + Recall}$$

$$F1 - \text{Score} = 2 \times \frac{0.9167 \times 0.9167}{0.9167 + 0.9167} = 0.9167$$

5. For Class 3 (True Label 3), the model correctly predicted 27 samples as Class 3, with some misclassifications: 1 sample was predicted as Class 1, and 3 samples as Class 4.

TP = 27 FP = 5 FN = 4

$$Precision = \frac{TP}{TP + FP} = \frac{27}{27 + 5} = 0.843$$

$$Recall = \frac{TP}{TP + FN} = \frac{27}{27 + 4} = 0.871$$

$$F1 - \text{Score} = 2 \times \frac{Precission \times Recall}{Precission + Recall}$$

$$F1 - \text{Score} = 2 \times \frac{0.843 \times 0.871}{0.843 + 0.871} = 0.856$$

6. For Class 4 (True Label 4), the model correctly predicted 154 samples as Class 4, with 4 misclassifications where Class 4 samples were predicted as Class 3.

TP = 154 FP = 4 FN = 4

$$Precision = \frac{TP}{TP + FP} = \frac{154}{154 + 4} = 0.975$$

$$Recall = \frac{TP}{TP + FN} = \frac{154}{154 + 4} = 0.975$$

$$F1 - \text{Score} = 2 \times \frac{Precission \times Recall}{Precission + Recall}$$

$$F1 - \text{Score} = 2 \times \frac{0.975 \times 0.975}{0.975 + 0.975} = 0.975$$

Based on the analysis of the confusion matrix for the Decision Tree model, it shows a train accuracy of 89,98% and a test accuracy of 95%. The confusion matrix indicates a total of 203 correct prediction out of 213 total data. Class 0 shows the best performance with perfect Precision and Recall 100% indicating that the model classifies all instances of this class correctly without errors. Classes 1 and 2 have similar Precision and Recall 90 % and 91,7%, demonstrating good performance but with some minor misclassifications. Class 3 has lower performance with Precision and Recall of 87,1%, suggesting more frequent misclassifications. Class 4, despite having the largest dataset, still performs well with Precision of 95% and Recall of 97,5%, accurately classifying most instances. These metrics highlight the model's strengths and weaknesses in handling class imbalances.

After the confusion matrix, we can visualize the decision tree and can be seen below.

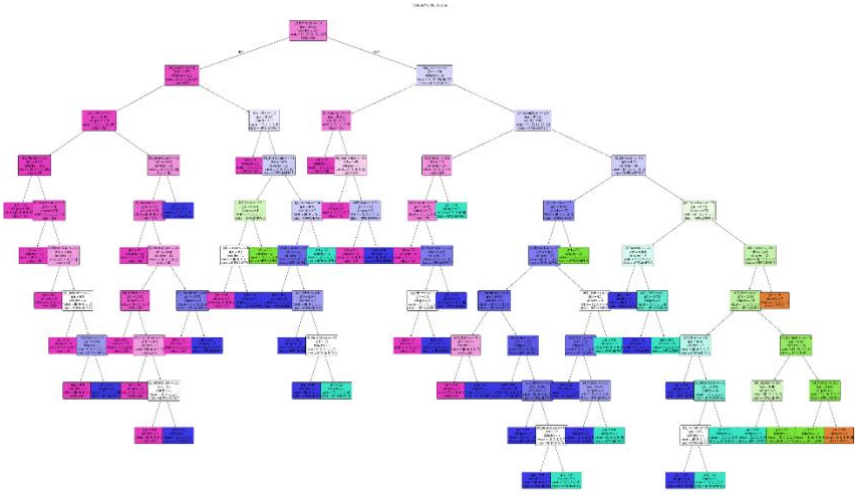

Fig. 11: Decision Tree

Based on the decision tree shown above, we can analyze and observe the results of the decision tree. The following conclusions can be drawn:

1. If Q1_Komputer ≤ 3.50 and Q4_Interest ≤ 2.50, AGE_Point ≤ 4.50, then classify as Class 4.
2. If Q1_Komputer ≤ 3.50, Q4_Interest ≤ 2.50, AGE_Point > 4.50, then classify as Class 3.
3. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point ≤ 1.50, then classify as Class 4.
4. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point > 1.50, then classify as Class 4.
5. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point > 1.50, Q2_Smartphone ≤ 4.50, then classify as Class 4.
6. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point > 1.50, Q2_Smartphone > 4.50, then classify as Class 3.
7. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point > 1.50, Q2_Smartphone > 4.50, Q3_Kepemiluan ≤ 4.50, then classify as Class 3.
8. If Q1_Komputer ≤ 3.50, Q4_Interest > 2.50, AGE_Point > 1.50, Q2_Smartphone > 4.50, Q3_Kepemiluan > 4.50, then classify as Class 2.
9. If Q1_Komputer ≥ 3.50, AGE_Point ≤ 2.50, then classify as Class 4.
10. If Q1_Komputer > 3.50, AGE_Point > 2.50, then classify as Class 4.
11. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan ≤ 2.50, then classify as Class 4.
12. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan > 2.50, then classify as Class 3.
13. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan > 2.50, Q4_Interest ≤ 2.50, then classify as Class 2.
14. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan > 2.50, Q4_Interest > 2.50, then classify as Class 1.
15. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan > 2.50, Q4_Interest > 4.00, then classify as Class 1.
16. If Q1_Komputer > 3.50, AGE_Point > 2.50, Q3_Kepemiluan > 2.50, Q4_Interest > 3.50, then classify as Class 0.

Reference [6] found that using Decision Tree for medicine supply achieved an accuracy of 80% also from reference [7] achieved 94,27% and 98% accuracy that was count manually.

4. Conclusion

The test results indicate that the Decision Tree algorithm achieved an accuracy of 95.00%, surpassing the average recorded in a literature review of 16 related journals. Based on existing studies, the Decision Tree (C4.5 and C5.0) algorithms have proven effective and accurate in various classification and prediction tasks, such as scholarship selection, predicting inactive students, and assessing credit risk. The main strength of this algorithm lies in its ability to handle both numerical and categorical data simultaneously and produce transparent and easily interpretable results. This makes it particularly useful for decision-making contexts requiring detailed explanations, such as credit feasibility analysis or student dropout prediction [8] [9][10]. However, despite its high accuracy, the Decision Tree algorithm has weaknesses, including its susceptibility to overfitting if not carefully managed and the need for sufficient high-quality data. Additionally, while Decision Tree is computationally faster compared to algorithms like Support Vector Machines (SVM) or K-Nearest Neighbors (KNN), the choice of the appropriate algorithm must align with the data characteristics and analytical goals. In the context of this research on election officer selection, using the Decision Tree algorithm shows great potential for building an accurate and efficient model, particularly when the data includes diverse variables requiring clear interpretations [7] .Overall, despite challenges in data management and mitigating overfitting, the successful application of Decision Tree across various fields demonstrates its value in improving prediction accuracy and facilitating data-driven decision-making [4] [11]. A related study using a different algorithm for population data classification by [1] found that testing with a Confusion Matrix revealed that the KNN algorithm (K=3) achieved 93.75% accuracy, with high precision and recall values of 93.67% and 93.38%, respectively.

References

[1] A. Zulfan Najib, S. Achmadi, and K. Auliasari, "Sistem Klasifikasi Data Penduduk Untuk Menentukan Tempat Pemungutan Suara (Tps) Dengan Metode K-Nearest Neighbor (Knn) Berbasis Website," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 2, pp. 1323–1330, 2024, doi: 10.36040/jati.v8i2.9130.

[2] I. D. Mienye and N. Jere, "A Survey of Decision Trees: Concepts, Algorithms, and Applications," *IEEE Access*, vol. 12, no. June, pp. 86716–86727, 2024, doi: 10.1109/ACCESS.2024.3416838.

[3] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Mag.*, vol. 17, no. 3, pp. 37–53, 1996.

[4] I. Iddrus and D. W. Sari, "Penerapan Data Mining Menggunakan Algoritma Decision Tree C4.5 Untuk Memprediksi Mahasiswa Drop Out Di Universitas Wiraraja," *J. Adv. Res. Inform.*, vol. 1, no. 02, pp. 1–7, 2023, doi: 10.24929/jars.v1i02.2684.

[5] F. F. Kusuma, "Penerapan Data Mining Untuk Akurasi Analisis Cuaca di Australia Menggunakan Algoritma J48 Decision Tree," *J. Comput. Sci. Inf. Syst. J-Cosys*, vol. 3, no. 2, pp. 65–68, 2023, doi: 10.53514/jco.v3i2.396.

[6] Dwita Elisa Sinaga, Agus Perdana Windarto, and Rizki Alfadillah Nasution, "Analisis Data Mining Algoritma Decision Tree Pada Prediksi Persediaan Obat (Studi Kasus : Apotek Franch Farma)," *KLIK Kaji. Ilm. Inform. dan Komput.*, vol. 2, no. 4, pp. 123–131, 2022, doi: 10.30865/klik.v2i4.328.

[7] I. Rahmianti, "Analisis Kelayakan Pemberian Kredit Koperasi Dengan Metode Data Mining Decision Tree," *J. Inform. dan Rekayasa Elektron.*, vol. 5, no. 2, pp. 153–161, 2022, doi: 10.36595/jire.v5i2.663.

[8] K. Khotimah, "Teknik Data Mining menggunakan Algoritma Decision Tree (C4.5) untuk Prediksi Seleksi Beasiswa Jalur KIP pada Universitas Muhammadiyah Kotabumi," *J. SIMADA (Sistem Inf. dan Manaj. Basis Data)*, vol. 4, no. 2, pp. 145–152, 2022, doi: 10.30873/simada.v4i2.3064.

[9] N. Y. L. Gaol, "Prediksi Mahasiswa Berpotensi Non Aktif Menggunakan Data Mining dalam Decision Tree dan Algoritma C4.5," *J. Inf. Teknol.*, vol. 2, pp. 23–29, 2020, doi: 10.37034/jidt.v2i1.22.

[10] A. Yudistira and M. Nurkhamid, "Penggunaan Data Mining Dalam Hit Rate Importasi Jalur Merah Dengan Model Decision Tree," *J. Perspekt. Bea Dan Cukai*, vol. 5, no. 2, pp. 187–202, 2021, doi: 10.31092/jpbc.v5i2.1297.

[11] K. A. Putri, D. Febriawan, and F. N. Hasan, "Implementation of Data Mining to Predict Student Study Period with Decision Tree Algorithm (C4.5)," *J. Sisfokom (Sistem Inf. dan Komputer)*, vol. 13, no. 1, pp. 31–39, 2024, doi: 10.32736/sisfokom.v13i1.1943.