IMPLEMENTATION OF ADABOOST ALGORITHM ON C50 FOR IMPROVING THE PERFORMANCE OF LIVER DISEASE CLASSIFICATION

[1*]ANWAR SANUSI, [2]CHRYSTIA AJI PUTRA, [3]FAWWAZ ALI AKBAR

Study Program of Informatic, Faculty of Computer Science, Universitas Pembangunan Nasional “Veteran” Jawa Timur

Jl. Rungkut Madya No.1, Gn. Anyar, Kec. Gn. Anyar, Surabaya, East Java

e-mail: [1]anwarsanusisan@gmail.com, [2]ajiputra@upnjatim.ac.id, [3]fawwaz_ali.fik@upnjatim.ac.id

[*]Corresponding author

ABSTRACT

The liver is a vital organ that is important for humans because it plays a role in regulating hormone cycles, neutralizing toxins, and controlling the composition of blood. Liver disease is a common ailment worldwide. Often, this disease occurs without specific symptoms (asymptomatic). Therefore, liver disease is known as a "silent killer," and it is necessary to quickly and accurately diagnose and treat liver diseases. Data mining technology can be useful for rapidly detecting liver diseases from laboratory diagnosis results. One classification algorithm that can be used is the C50 algorithm. This algorithm is an improvement over the previous C45 algorithm, with several advantages such as efficient memory usage and more concise tree results. However, the C50 algorithm may experience overfitting on complex medical data, requiring the boosting process using AdaBoost. The AdaBoost algorithm can make the C50 algorithm more susceptible to overfitting. Another advantage of the AdaBoost algorithm is its ability to handle imbalanced datasets in terms of target labels. This research used 583 data from UCI machine learning with two target labels: liver disease and non-liver disease. The model was tested using k-fold cross-validation during the validation process. The research findings revealed that the C50 algorithm achieved higher accuracy after the boosting process. This is evident from the highest accuracy obtained in the testing, where the C50 algorithm boosted with AdaBoost achieved an accuracy of 75.16% compared to the C50 algorithm before boosting, which had an accuracy of 68.79%. There was a significant increase in accuracy by 6.37%. Furthermore, AdaBoost has proven to be effective in handling imbalanced datasets, as demonstrated by the 5.02% increase in the F1-score, from 80.02% to 85.04%. The C50 algorithm boosted with AdaBoost also outperformed the logistic regression algorithm boosted with AdaBoost in the previous study, with an accuracy improvement of 0.81%.

Keywords: *Data Mining, C50 Algorithm, AdaBoost Algorithm, Liver Disease Classification.*

1. INTRODUCTION

The liver, a vital organ in the human body, is located in the right upper abdomen, just below the diaphragm. Its functions include detoxification and neutralization of toxins, regulation of hormonal cycles, regulation of the composition of blood containing fats, sugars, proteins, and other substances. Additionally, the liver produces bile, which aids in the digestion process [1]. Liver disease is a common condition worldwide, which can be caused by factors such as alcohol consumption, obesity, and viral hepatitis infection. Liver disease disrupts the crucial functions of the liver in the body. Serious damage can occur when the liver does not function properly. Although the liver has the ability to regenerate and replace damaged cells, if the damage is severe and cells cannot regenerate or are lost, the liver will not be able to meet the body's needs [2]. Liver disease is often referred to as a silent killer because it often does not exhibit initial symptoms or is asymptomatic. This disease can have serious impacts on overall health, causing organ damage, and even death. Therefore, the diagnosis and treatment of liver disease must be prompt and accurate [3].

In the field of healthcare, information technology plays a significant role in diagnosing specific diseases using classification algorithms. One commonly used algorithm is the C50 algorithm. This algorithm works by creating a decision tree based on the features present in the data and generating rules that can be used to classify the data [4]. However, in complex medical data, models are susceptible to overfitting, necessitating specific techniques to make the models more resilient to overfitting. One technique that can be used to address this issue is boosting using the adaboost algorithm. This algorithm is suitable for application in complex medical datasets because it can improve the accuracy of the model and make it more resistant to overfitting. One of the main advantages of AdaBoost is its ability to handle class imbalance issues in datasets. This is primarily achieved through the adjustment of sample weights performed by AdaBoost. By assigning higher weights to misclassified samples, AdaBoost gives more attention to difficult-to-classify classes, particularly the minority class.

In previous research, the use of AdaBoost to improve accuracy on imbalanced data has been explored. One study investigated the application of AdaBoost to boost the K-nearest neighbors (KNN) algorithm on an imbalanced dataset, resulting in improved accuracy compared to the KNN algorithm [5]. Another study focused on the classification of liver diseases using the C4.5 algorithm and AdaBoost. It demonstrated that the C4.5 AdaBoost method outperformed other algorithms in terms of accuracy, achieving a rate of 77.12% [6]. Additionally, in a study analyzing the performance of machine learning algorithms in liver disease prediction, the C4.5 algorithm achieved an accuracy of 70.29%, while the Naïve Bayes algorithm achieved an accuracy of 67.05% [7]. Moreover, a study on heart disease classification examined the effectiveness of the C4.5 algorithm with feature selection and subsequent application of the AdaBoost ensemble, which improved the accuracy by 10.33% in diagnosing heart disease [8]. Furthermore, there has been research using similar datasets and boosting algorithms, such as logistic regression boosted with AdaBoost, which achieved a high accuracy rate of 74.35% [9]. These previous findings will serve as valuable references for evaluating the performance of the C50 + AdaBoost algorithm in our study.

Therefore, based on the previous research conducted, the researchers are interested in conducting further research using the C50 algorithm, as it is an improved version of the C4.5 algorithm. Additionally, the researchers hope to achieve higher accuracy on unbalanced datasets by incorporating the boosting process using the AdaBoost method. The results of this research are expected to contribute to the development of classification techniques in complex medical data, particularly for liver disease, with good accuracy. This research is expected to contribute to the healthcare field by providing a classification model that can assist doctors in diagnosing liver diseases more quickly and effectively. Furthermore, this research is also expected to provide a better understanding of C5.0 in performing classification tasks on medical datasets.

2. RESEARCH METHODOLOGY

2.1 Data mining

Data mining is the process of discovering patterns, relationships, or valuable insights from large volumes of data. It involves extracting or mining knowledge from data by using various techniques, algorithms, and methods. The goal of data mining is to uncover hidden patterns, trends, and correlations that are not readily apparent and can provide meaningful information for decision-making, prediction, and optimization [10].

2.2 Decision tree

A decision tree is a supervised machine learning algorithm that is used for classification and regression tasks. It is a flowchart-like structure where each internal node represents a feature or attribute, each branch represents a decision rule, and each leaf node represents an outcome or class label. The tree is constructed by recursively partitioning the data based on the values of different features, aiming to create the most effective decision rules for predicting the target variable [11].

2.3 K-fold cross validation

The k-fold cross-validation method is a technique for testing the accuracy of a model built based on a specific dataset. In this method, the dataset is divided into two parts, namely the training data and the testing data. The dataset is then randomly divided into several partitions k times. Next, k iterations of the classification process are performed, where each testing process uses one partition as the testing data while the other partitions are used as the training data [12].

2.4 C50 algorithm

The C5.0 algorithm, also known as C5.0, is a decision tree-based machine learning algorithm used for classification tasks. It is an extension of the earlier C4.5 algorithm, developed by Ross Quinlan. C5.0 builds decision trees by recursively partitioning the data based on different features to create effective decision rules for predicting the target class.

The C5.0 algorithm uses an information gain or gain ratio measure to determine the best attribute to split the data at each internal node. It considers both categorical and continuous features and can handle missing values in the data. The algorithm continues splitting the data until it reaches leaf nodes where the majority class or the predicted class label is assigned. The process of constructing the tree model begins by finding the root node [13]. The calculation process starts by determining the entropy value of each attribute using the following formula:

$$Entropy(S) = \sum_{j=1}^{k} -p_j \log_2 p_j \quad (1)$$

with:

S	: Case set
k	: Number of classes in variable A
p_j	: The probability of Sj and S

After obtaining the entropy value of each attribute, the next step is to calculate the information gain of each attribute using the previously determined entropy values. The formula for calculating the information gain can be seen as follows:

$$Gain(S, A) = Entropy(S) - \sum_{i=1}^{m} \frac{|S_i|}{S} \times Entropy(S_i) \quad (2)$$

with:

S	: Case set
S_i	: Set of cases in category i
m	: Number of categories in variable A
$\|S_i\|$	: The absolute value of the set of cases in category i

After finding the entropy values of all attributes, the next step is to calculate the splitinfo value and gain ratio value. From each attribute, the attribute with the highest gain ratio will be selected as the root node in the model. Here are the formulas for calculating the split info value and gain ratio.

$$Split\ info(x) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \quad (3)$$

with:

$\|S_i\|$	: The absolute value of the set of cases in category i
$\|S\|$	: The absolute value of the set of cases in S

$$Gain\ Ratio = \frac{Gain(S,A)}{\sum_{i=1}^{m} Entropy(S_i)} \quad (4)$$

with:

$Gain(S, A)$	: Gain value of a variable
$\sum_{i=1}^{m} Entropy(S_i)$	: Total entropy of a specific attribute

2.5 Adaboost

The Adaboost (Adaptive Boosting) algorithm is one of the machine learning algorithms developed by Yoav Freund and Robert Schapire. Adaboost works by combining multiple simple machine learning models known as weak learners (such as shallow decision tree models) into a stronger predictive model. In each iteration, Adaboost gives higher weight to the data points that were difficult to predict by the previous models, aiming to enhance the focus on harder-to-predict data and reduce the focus on easily predictable data [14]. Here is the formula for calculating the adaboost algorithm:

$$F(x) = \sum_{t=1}^{T} \propto_t h_t(x) \quad (5)$$

with:

$F(x)$	: Strong Classifier
T	: Number of iterations
$\propto_t$	: Learning rate
$h_t(x)$	: Based classifier

2.6 Confusion matrix

The confusion matrix, also often referred to as an error matrix, is commonly used to measure the performance of a classification model based on error values. Essentially, the confusion matrix provides comparative information on the classification outcomes made by the classification model compared to the actual classification outcomes. The confusion matrix is presented in the form of a matrix table that illustrates the performance of the classification model on a known set of test data [15]. Here is the formula for calculating the confusion matrix algorithm:

$$accuracy = TP/(TP + FN) * 100\% \quad (6)$$
$$precision = TP/(TP + FP) * 100\% \quad (7)$$

$$recall = \frac{TP+TN}{TP+FP+TN+FN} * 100\% \quad (8)$$

$$f1 - score = 2 * \frac{recall*precision}{recall+precisio} \quad (9)$$

with:

TP	: True Positive
TN	: True Negative
FP	: False Positive
FN	: False Negative

2.7 Research Workflow

The workflow of this research is carried out sequentially, starting from data collection, algorithm implementation, and finally program evaluation. The detailed sequence of the research workflow can be seen in the following diagram (Figure 1).

2.8 Data collection

The first stage of this research is data collection. Data collection is the process of gathering relevant information or data for a specific purpose. During this stage, various activities are typically conducted to collect data, such as observations, interviews, questionnaires, literature review, field observations, or processing existing data. In this study, data collection utilized publicly available liver patient data from India, which is freely accessible through the UCI Machine Learning Repository accessed via the Kaggle website. This data consists of laboratory test results and includes 583 data points with 10 available attributes. This data set contains 416 liver patient records and 167 non-liver patient records. A dataset can be considered imbalanced if the result of calculating the majority class divided by the minority class approaches a value of 1.5-2 or exceeds it. In this dataset, the number of majority class instances is 416, while the number of minority class instances is 167. When dividing the majority class by the minority class, the result is more than 2. Therefore, this dataset can be considered imbalanced. The complete dataset can be accessed through the following link https://www.kaggle.com/datasets/ uciml/indian-liver-patient-records.

2.9 Preprocessing data

The second stage of this research is data preprocessing. This stage aims to clean the data from noise before it is processed by the designated algorithms. The preprocessing in this study consists of two steps. Firstly, the data cleaning step, where the dataset obtained is cleaned or removed from data with missing values. As a result, the initial dataset of 583 entries is reduced to 479 entries. The second preprocessing step is feature selection, where the gender attribute is removed because, according to the statement of the medical expert [16], it does not affect liver disease. Therefore, removing this attribute is necessary to avoid bias in the model's accuracy.

2.10 Data splitting

In this stage, the data is now clean from noise and ready for processing. However, the data needs to be divided into two parts with different functions. The first part is the training data, which is used to build the model using the designated algorithm. The second part is the testing data, which is used to evaluate the model created using the training data from the previous process. Typically, in data splitting, the training data has a larger quantity compared to the testing data to provide more information for the model to learn from. Before performing the data split, the dataset will be shuffled first to enhance data generalization and avoid bias in training due to the data being sorted by specific features.

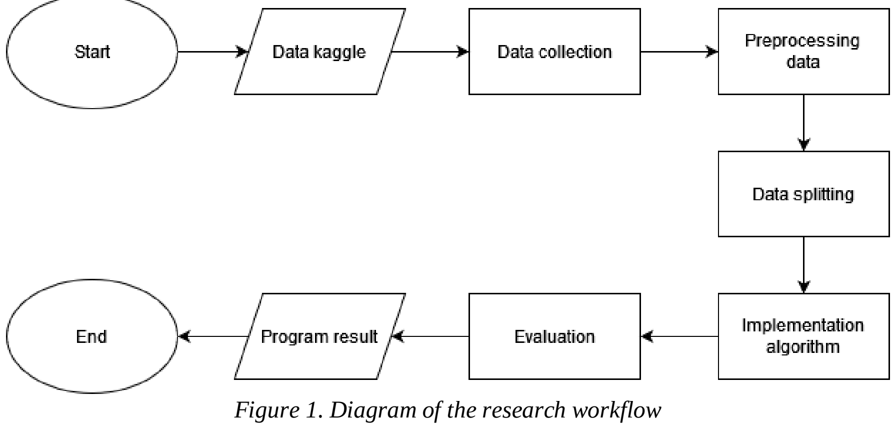

Figure 1. Diagram of the research workflow

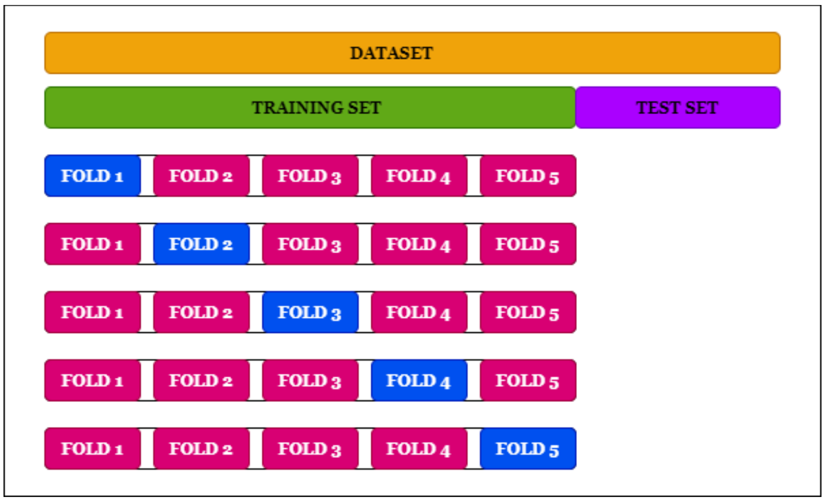

Figure 2. The Division of Training Data and Test Data in K-Fold Cross-Validation Scenario

In the first scenario, the data is divided in an 80:20 ratio, resulting in 463 instances for the training data and 116 instances for the testing data. In the second scenario, the data is divided in a 70:30 ratio, resulting in 405 instances for the training data and 174 instances for the testing data.

2.11 Implementation of the Algorithm

This stage involves implementing the selected algorithm into a programming code to build a model from the previously divided data. The algorithms used in this research are C50 and adaboost. The program is created using the Python programming language. In the implementation of the C50 algorithm boosted with Adaboost, a total of 70 weak classifiers will be used. This is done to find the best accuracy from the iterations performed. The implemented program will be used on the testing data that was divided earlier.

2.12 Evaluation

The evaluation stage is the testing phase of the previously developed system. The results of this system testing will determine the effectiveness of the previously built model using a confusion matrix. The testing conducted in this stage includes comparing the accuracy and f1-score of the C50 algorithm before the boosting process with the accuracy and f1-core of the C50 algorithm after the boosting process using adaboost. In the testing phase, the evaluation of accuracy level and F1-score will utilize k-fold cross-validation with a value of k=5. The following is the division of training data and test data for each k iteration (Figure 2). The accuracy and F1-score results obtained in each iteration k will be combined by calculating their average value as the final evaluation score of the model.

3. RESULT AND DISCUSSIONS

In this study, the testing process is conducted alternately for each scenario. The accuracy measurement is performed by applying the C50 algorithm without the boosting process, followed by applying the C50 algorithm with the boosting process. This is done to assess the improvement in accuracy achieved after applying the boosting process to the C50 model. Each experiment iteration of the developed model will be evaluated using the confusion matrix method. The evaluation results will provide accuracy, precision, recall, and f1-score values. The following are the evaluation results for each conducted experimental scenario.

3.1 Testing scenario with 80:20 dataset split

In this experiment series, the training data used to build the decision tree model consists of 463 instances. The model that has been created will be evaluated using 116 test instances. In this test, the accuracy of the C50 algorithm will be compared before and after the boosting process using adaboost. The boosting process will be performed for 70 iterations. The following is a summary table of the results from the evaluation of the C50 model in the first scenario (Table 1).

Table 1. The results from the evaluation of the C50 model in the first scenario

No	K	Accuracy	Precision	Recall	F1-score
1	1	67,24%	74,22%	84,7%	79,12%
2	2	67,24%	71,87%	86,25%	78,4%
3	3	73,27%	83,51%	82,6%	83,06%
4	4	71,55%	82,41%	81,52%	81,96%
5	5	64,65%	69,6%	87,65%	77,59%
Rata-rata		68,79%	76,32%	84,54%	80,02%

Table 2. The results from the evaluation of the C50 + Adaboost model in the first scenario

No	K	Accuracy	Precision	Recall	F1-score
1	1	73.27 %	73,68%	98,82%	84,42%
2	2	74,13%	73,14%	98,75%	84,04%
3	3	78,44%	81,09%	93,47%	87,3%
4	4	79,31%	83,33%	92,39%	87,62%
5	5	70,68%	70,53%	97,53%	81,86%
Rata-rata		75,16%	76,35%	96,39%	85,04%

The Table 1 provides the results of the performance of the C50 model for each iteration k. The final results from the table can be obtained by calculating their average values. The accuracy performance of the model is found to be 68.79% with an F1-score of 80.02%. Next, the C50 model will be boosted using Adaboost to enhance its performance. The following is a summary table of the results from the evaluation of the C50 + Adaboost model in the first scenario.

The Table 2 provides the results of the performance of the C50 + Adaboost model at each iteration k. It can be observed that the performance of this model improved compared to the previous model before the boosting process. The accuracy of the model improved to 75.16% with an F1-score of 85.04%. Therefore, it can be concluded that the boosting process has been successful in enhancing the model's performance, with an accuracy improvement of 6.37% and an F1-score improvement of 5.02%.

3.2 Testing scenario with 70:30 dataset split

In this series of experiments, the training data used to build the decision tree model consists of 405 instances. The model created will be evaluated using 174 instances of test data. In this test, the accuracy of the C50 algorithm before and after the boosting process using adaboost will be compared. The boosting process will be performed for 70 iterations. The following is a summary table of the results from the evaluation of the C50 model in the second scenario (Table 3).

The Table 3 presents the results of the performance of the C50 model in each iteration k. The values from the five iterations were obtained, and the average of the five values was calculated as the final value of the model. The accuracy performance of the model was found to be 68.68% with an F1-score of 79.69%. Next, the C50 model will be boosted using Adaboost to enhance its performance. The following is a summary table of the results from the evaluation of the C50 + Adaboost model in the second scenario.

The Table 4 provides the results of the performance of the C50 + Adaboost model in each iteration k. It is found that the performance of this model has improved compared to the previous model before the boosting process. The accuracy of the model improved to 73.9% with an F1-score of 84.18%. Therefore, it can be concluded that the boosting process has proven to enhance the model's performance with an accuracy improvement of 5.22% and an F1-score improvement of 4.49%.

Table 3. The results of the second scenario test

No	K	Accuracy	Precision	Recall	F1-score
1	1	66,66%	76%	77,23%	76,61%
2	2	66,09%	73,75%	82,53%	77,9%
3	3	71,26%	80,71%	83,08%	81,88%
4	4	74,71%	77,27%	92,96%	84,39%
5	5	64,7%	69,7%	87,7%	77,67%
Average		68,68%	75,48%	84,7%	79,69%

Table 4. The results of the second scenario test

No	K	Accuracy	Precision	Recall	F1-score
1	1	72,41%	74,5%	92,68%	82,6%
2	2	73,56%	75,31%	94,44%	83,8%
3	3	76,43%	78,44%	96,32%	86,46%
4	4	76,43%	76,04%	99,21%	86,1%
5	5	70,68%	71,6%	95,86%	81,97%
Average		73,9%	75,17%	95,7%	84,18%

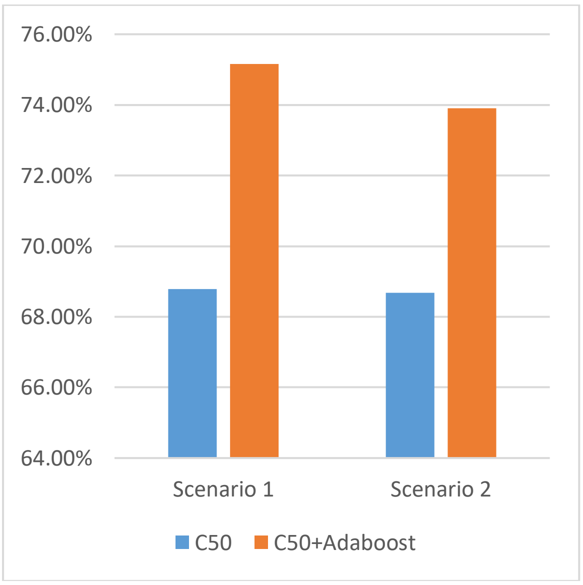

Figure 3. Accuracy comparison of C50 algorithm with C50 + adaboost algorithm

3.3 Testing result

The accuracy results from the first and second scenarios show a similarity, where the boosted model using adaboost algorithm has better accuracy. The accuracy improvement in the first scenario is 6.37%, while in the second experimental scenario, the accuracy improvement is 5.22%. The accuracy obtained using the C50+Adaboost algorithm has also been proven to be better when compared to the logistic regression algorithm, with an accuracy improvement of 0.81% [9]. These improvements demonstrate that adaboost is effective in enhancing the accuracy of the C50 model in handling complex medical datasets with multiple attributes. The graph in Figure 3 presents a comparison of the accuracy improvements from each experimental scenario.

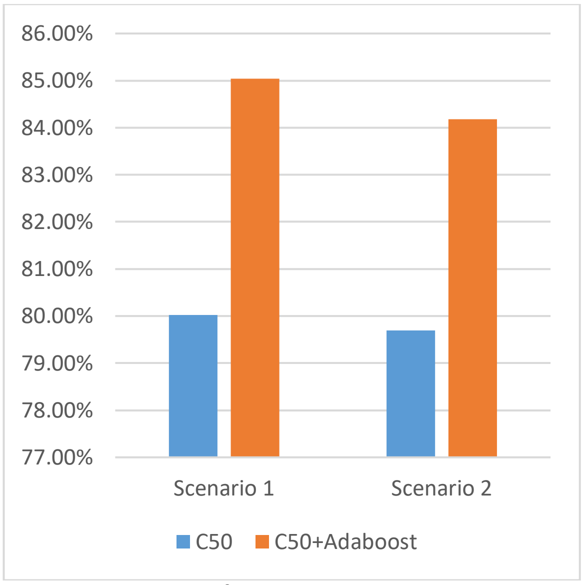

Figure 4. F1-score comparison of C50 algorithm with C50 + adaboost algorithm

Not only accuracy, but Adaboost has also been proven to handle imbalanced datasets. This is evident from the test results where the F1-score improved in models that were boosted with Adaboost in all experimental scenarios. The f1-score improvement in the first scenario is 5.02%, while in the second experimental scenario, the f1-score improvement is 4.49%. These improvements demonstrate that adaboost has been proven effective in handling imbalanced medical data. The graph in Figure 4 presents a comparison of the f1-score improvements from each experimental scenario.

4. CONCLUSION

Based on the results of the research and testing conducted using the C50 and AdaBoost algorithms, the researchers concluded that the C50 algorithm, after the boosting process, achieved better accuracy. This is evidenced by the highest accuracy obtained in the testing, where the C50 algorithm boosted with AdaBoost achieved an accuracy of 75.16% compared to the C50 algorithm before the boosting process, which had an accuracy of 68.79%. The accuracy improvement obtained is quite good with an increase of 6.37%. Adaboost has also been proven to handle imbalanced datasets, as evidenced by the increase in the F1-score by 5.02%, from 80.02% to 85.04%. The accuracy results obtained from the C50 algorithm boosted with AdaBoost were also better than the logistic regression algorithm boosted with AdaBoost on the same dataset in the previous study, with an increase in accuracy of 0.81%. However, this increase in accuracy, when compared to the logistic regression + Adaboost algorithm, is still considered small. Therefore, the addition of dataset handling techniques such as SMOTE is needed to balance the number of majority and minority classes. The expectation is that the accuracy and F1-score values can achieve better improvement.

REFERENCES

[1] I. Setiawati, A. Permana, A. Hermawan, (2019). "*Implementasi Decision Tree Untuk Mendiagnosis Penyakit Liver*," Journal of Information System Management (JOISM), Vol. 1 No. 1, pp.13-17.

[2] B.V. Haekal, E. Patimah, D.S. Prasvita, (2021). Klasifikasi Penyakit Liver dengan Menggunakan Decision Tree, *in Proceedings of Seminar Ilmiah Nasional Online Mahasiswa Ilmu Komputer dan Aplikasinya*, Vol. 2, No. 1, pp.655-659.

[3] E. Pusporani, S. Qomariyah, and I. Irhamah, (2019). "Klasifikasi Pasien Penderita Penyakit Liver dengan Pendekatan Machine Learning," *INFERENSI*, vol. 2, no. 1, pp.25-32.

[4] A. C. Wijaya, N. A. Hasibuan, and P. Ramadhani, (2018). "*Implementasi Algoritma C5.0 Dalam Klasifikasi Pendapatan Masyarakat (Studi Kasus: Kelurahan Mesjid Kecamatan Medan Kota)*," Jurnal Informasi dan Teknologi Ilmiah (INTI), vol. 5, no. 3, pp.262-268.

[5] N. Novianti, M. Zarlis, and P. Sihombing, (2022). "*Penerapan Algoritma Adaboost Untuk Peningkatan Kinerja Klasifikasi Data Mining Pada Imbalance Dataset Diabetes*," Jurnal Media Informatika Budidarma, vol. 6, no. 2, pp. 1200-1206, doi: 10.30865/mib.v6i2.4017.

[6] W. Wahyudi, (2021)."*Implementasi Data Mining Untuk Klasifikasi Penyakit Liver Dengan C4.5 Adaboost*," Jurnal Ilmiah Teknik Informatika dan Komunikasi, Vol. 1, No.3. doi: https://doi.org/10.55606/juitik.v1i3.120.

[7] M. Nurkholifah, J. Jasmarizal, Y. Umar, and R. Rahmaddeni, (2023). *"Analisa Performa Algoritma Machine Learning Dalam Prediksi Penyakit Liver*," Jurnal Indonesia : Manajemen Informatika dan Komunikasi, vol. 4, no. 1, pp. 164–172, doi: 10.35870/jimik.v4i1.149.

[8] M. Qois Syafi, A. Alamsyah, (2022). "*Increasing Accuracy of Heart Disease Classification on C4.5 Algorithm Based on Information Gain Ratio and Particle Swarm Optimization Using Adaboost Ensemble*," Journal of Advances in Information Systems and Technology, vol. 4, no. 1.

[9] K. Idris and S. Bhoite, (2019). "Applications of Machine Learning for Prediction of Liver Disease,". [Online]. Available: https://towardsdatascience.com/understanding-

[10] R. R. Putra and C. Wadisman, (2018). "*Implementasi Data Mining Pemilihan Pelanggan Potensial Menggunakan Algoritma K Means*," INTECOMS: Journal of Information Technology and Computer Science, vol. 1, no. 1, pp. 72–77, doi: 10.31539/intecoms.v1i1.141.

[11] P. B. N. Setio, D. R. S. Saputro, and B. Winarno, (2020). "Klasifikasi dengan Pohon Keputusan Berbasis Algoritme C4.5," vol. 3, pp. 64–71, [Online]. Available: https://journal.unnes.ac.id/sju/index.php/prisma/

[12] C. Anam and H. B. Santoso, (2018). "*Perbandingan Kinerja Algoritma C4.5 dan Naive Bayes untuk Klasifikasi Penerima Beasiswa*," Energy, vol. 8, no. 1, pp.13-19.

[13] M. Fajri, I. T. Utami, and Muh. Maruf, (2022). "*Comparison of C4.5 and C5.0 Algorithm Classification Tree Models for Analysis of Factors Affecting Auction*," Indonesian Journal of Statistics and Its Applications, vol. 6, no. 1, pp. 13–22, doi: 10.29244/ijsa.v6i1p13-22.

[14] Annah and Hasriani, (2019). "Klasifikasi Warga Penerima Bantuan Stimulan Perumahan Swadaya Menggunakan Metode ADABOOST," *in Proceedins of Seminar Ilmiah Sistem Informasi Dan Teknologi Informasi*, vol. 8, no.1, pp.169-180.

[15] D. Normawati and S. A. Prayogi, (2021). "*Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitte*r," J-SAKTI (Jurnal Sains Komputer dan Informatika), vol. 5, no. 2, pp.697-711.

[16] M. Abdar, M. Zomorodi-Moghadam, R. Das, and I. H. Ting, (2017). "*Performance analysis of classification algorithms on early detection of liver disease*," Expert Systems with Applications, vol. 67, pp. 239–251, doi: 10.1016/j.eswa.2016.08.065.