Journal of Informatics and Interactive Technology
Volume 1, Nomor 3, Desember 2024
e-ISSN : 3048-1023

Diabetes Detection Optimisation with Hyperparameter Tuning in Random Forest Algorithm

Adrian Aji Septa[1], Amar Al Farizi [2], Anas Nur Khafid [3], Didi Prasetyo [4], Nur Cholis Romadhon [5], Fandy Setyo Utomo[6]
Informatics Study Program, Faculty of Computer Science, Amikom University, Purwokerto
email: [1]22sa11a080@mhs.amikompurwokerto.ac.id, [2]222sa11a083@mhs.amikompurwokerto.ac.id,
[3]22sa11a071@mhs.amikompurwokerto.ac.id, [4]22sa11a068@mhs.amikompurwokerto.ac.id,
[5]22sa11a017@mhs.amikompurwokerto.ac.id, [6]fandy_setyo_utomo@amikompurwokerto.ac.id

Article Info

Article history:

Received 17-06-2024
Revised 23-07-2024
Received 28-08-2024

Keywords:

Health
Random Forest
Optimization
Mean

ABSTRACT

Diabetes is a common disease suffered by many people, one of which is Diabetes Mellitus. This disease is caused by disorders in the pancreas that affect the body's metabolism due to the lack of production of the hormone insulin, the use of technology that is associated with diabetes is one step to be able to classify diabetes. This study aims to develop a diabetes classification model using the Random Forest algorithm. The methods used include dataset selection from the Pima Indians Diabetes Database, data pre-processing by replacing missing values using the mean, and data balancing using the SMOTE technique. The model was then trained and evaluated using confusion matrix to measure accuracy, precision, recall, and F1-score. The results showed that the Random Forest algorithm with grid search hyperparameters produced good performance with 79% accuracy, 76% precision, 83% recall, and 80% F1-score. The conclusion of this research is that the Random Forest algorithm is effective in classifying diabetes data and shows improved performance compared to other algorithms such as Logistic Regression. This model can be used for more accurate early detection of diabetes, thus helping in early treatment and reducing the number of disabilities and deaths due to diabetes.

Keywords: Health, Random Forest, Optimization , mean

Corresponding author:

Fandy Setyo Utomo
Informatics Study Program, Faculty of Computer Science, Amikom University, Purwokerto
Email: fandy_setyo_utomo@amikompurwokerto.ac.id

1. INTRODUCTION

Diabetes is a common disease suffered by many people, one of which is Diabetes Mellitus. This disease is caused by disorders in the pancreas that affect the body's metabolism due to the lack of production of the hormone insulin. Insulin itself is an important hormone that functions to convert sugar into energy used by the body. [1]

According to data from the International Diabetes Federation (IDF), worldwide, people with diabetes reached a high number in 2021, reaching 537 million sufferers. Later this figure is also expected to jump or will increase to reach 643 million sufferers in 2030, and will be expected to increase to 783 million sufferers in 2045. [2]

With the rapid development of technology, the benefits are felt, especially in the field of health, especially for diabetes. Innovation in the field of technology that is able to detect diabetes with an accurate process is needed at this time. This allows early treatment of diabetes, thereby reducing the number of sufferers, disability, and death [3]. By incorporating technology in the healthcare field, one of which is AI it can predict diabetes using a specific algorithm.The algorithm it self is an effective method, expressed in a limited set of steps. An algorithm is an instruction used to solve a problem that is organized logically and systematically [4]. In our research, we will involve the Random Forest algorithm. Random Forest is one type of classification algorithm that we choose, this algorithm consists of several decision trees. Each decision tree is formed identically and is formed based on the value of a sample random vector that is independently distributed across all trees [5] .In the process of using Random Forest, we also increase the parameters (hyperparameter tuning) so that the results of the accuracy increase more on the parameters, namely n_estimators, max_depth, min_samples_split, min_samples_leaf.

In a previous study entitled 'Early Detection of Diabetes Using Machine Learning with Logistic Regression Algorithm', the model evaluated using confusion matrix showed quite good results with a recall value of 77%, precision 75%, accuracy 77%, and F1-score 76% [6]. Pada penelitian berjudul In a study entitled 'Improving Random Forest Performance Through Feature Selection by PCA to Detect Early Stage Diabetes', the classification resistance using the Random Forest algorithm on the dataset obtained varying degrees of suitability. Before reducing the attributes, the accuracy obtained a percentage of 74.56% was obtained at a dataset ratio of 70:30, and the lowest was 72.55% at a ratio of 80:20. After attribute reduction, the lowest accuracy of 73.20% was recorded at 1 attribute reduction with a ratio of 80:20, while the highest accuracy of 77.63% was achieved at 2 attribute reduction [7]. Another study with the title 'A Comparison between Decision Tree and Random Forest in Determining the Risk Factors Associated with Type 2 Diabetes' has the results The prevalence rate of T2DM is 14% among these subjects. The decision tree model has an accuracy of 64.9%, sensitivity of 64.5%, specificity of 66.8%, and area under the ROC curve of 68.6%, while the random forest model has an accuracy of 71.1%, sensitivity of 71.3%, specificity of 69.9%, and area under the ROC curve of 77.3% [8]. In another study with the title 'Early Detection of Diabetes Using Random Forest Algorithm' also said that the results of early detection of diabetes were carried out using the Random Forest algorithm. This research shows the superiority of the Random Forest algorithm over the use of other algorithms in early detection of diabetes using the same data. In the model built, this study achieved an accuracy of 87%. This result shows an increase in performance from previous research. However, there is still room for further development. There are factors that need to be considered to improve the detection accuracy of diabetes, that can be done for future research, such as the application of data balancing methods, feature selection, building more complex models, and exploring larger data [9]. In the research that has been done, researchers classify using the random forest algorithm by replacing the missing value using np.nan which is filled with the mean value, and for unbalanced data we use the SMOTE technique and for model testing the author uses hyperparameter grid search.

2. METHOD

The dataset consisting of 768 data from Pima Indians will be processed using Random Forest Algorithm to predict diabetes as shown in Figure 1. The research flow is further explained in Figure 2.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
..	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

[768 rows x 9 columns]

Figure1 Diabetes Disease Dataset of pima Indians

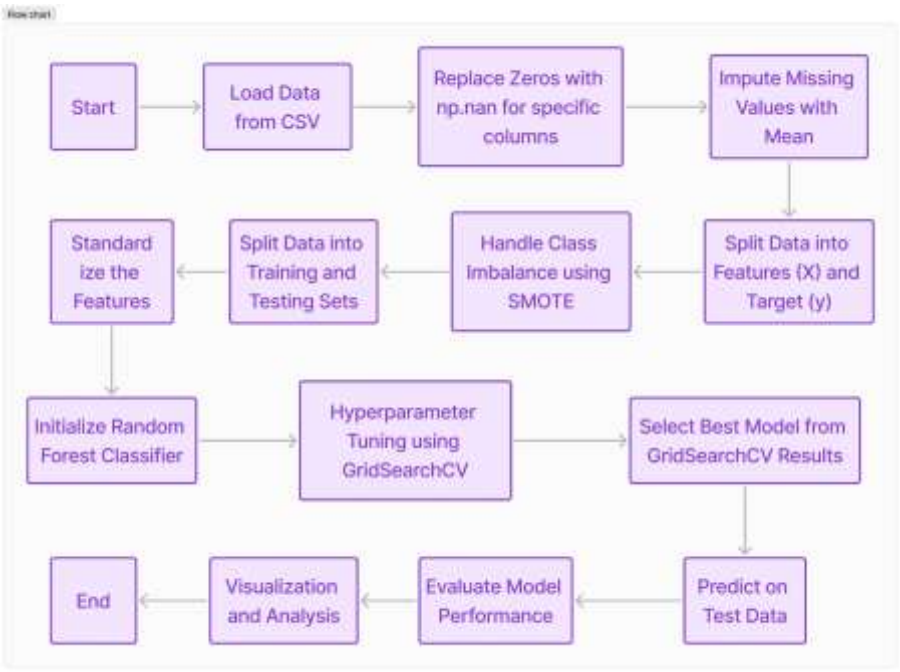

Figure 2 Research flowchart

The implementation of the Random Forest algorithm to predict data involves several steps, as seen in Figure 2. These steps start from dataset selection and model evaluation to data visualization. The use of Python scikit-learn library in the implementation of Random Forest simplifies the process of data manipulation, visualization, and analysis.

2.1 Defining the Dataset

The dataset used in this project was taken from the National Institute of Diabetes and Digestive and Kidney Diseases as part of the Pima Indians Diabetes Database. This dataset consists of data totaling 768 of all existing data

consisting of data on women aged 21 years and over. There are nine variables in this dataset, of which eight variables are independent variables, namely Pregnancies, Glucose, Blood Pressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction (DPF), and Age. And one more variable is the dependent variable, which is the Target or Outcome.

2.2 Pre Processing Data

In this pre-processing stage, data is examined to identify the presence of missing or incomplete data. Incomplete data will be filled using the average value of each existing variable so that the variable becomes complete. In addition, to overcome data imbalance, synthetic data generation is carried out using the SMOTE technique. The purpose of this technique is to maintain the proportion of the minority class to remain balanced with the majority class.

2.2.1 Data Cleaning

In the dataset used in this project, there are attributes that have incomplete values, so it is necessary to perform a data cleaning process. Data cleaning is a step to prepare the dataset by removing or filling empty values using the average value of each column. In this case, the blank value is identified as np.nan and then filled with the average value of each variable to ensure the variable is complete.

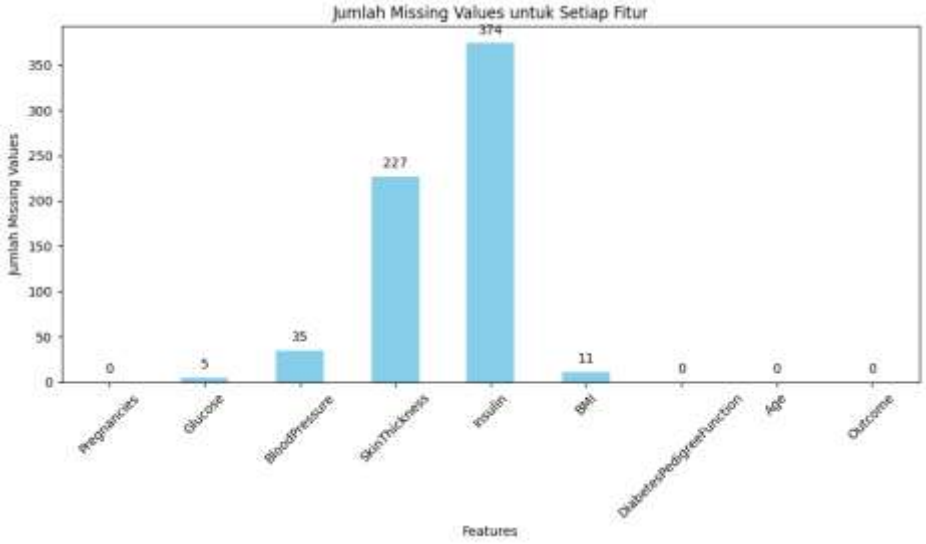

Figure 3 The initial state of the dataset before data cleaning

Figure 4 is the state of the data after cleaning by replacing empty values with np nan and filling np nan with the mean. From Figure 4, it can be explained that after cleaning the data, all missing value data is gone.

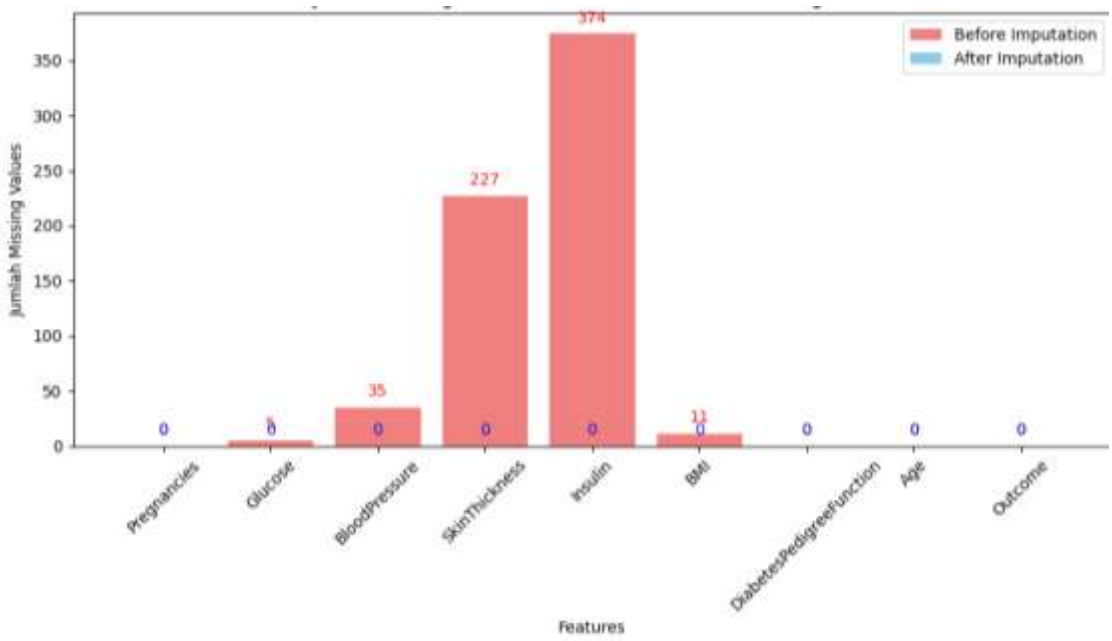

Figure 4 State of the dataset after data cleaning

2.2.2 Imbalance Data

The class imbalance problem is often encountered in various data sets in many fields, including software defect prediction, disease diagnosis, oil spill detection from satellite images, and online credit card fraud detection [10]. After cleaning the data, the next step is to check the outcome data, in which there is unbalanced data between the category variables o for negative, and 1 for positive. It appears that the positive category has a lower value, data balancing is needed using over sampling, one of which is using the SMOTE (Synthetic Minority Over-sampling Technique) method. SMOTE is one of the algorithms that handles such imbalances [11]. Therefore, the synthetic minority oversampling (SMOTE) algorithm uses the original data to synthesize new minority data that is different from the original data, in order to reduce the risk of overfitting. However, SMOTE is prone to overgeneralization and noise. To overcome this problem, the Safe-level-SMOTE oversampling technique was introduced, which determines the value of the safety level [12],[13]. The following figure 5 indicates that in the outcome variable the data is unbalanced or imbalance data, the contents of the outcome data are the outcomes of diabetics.

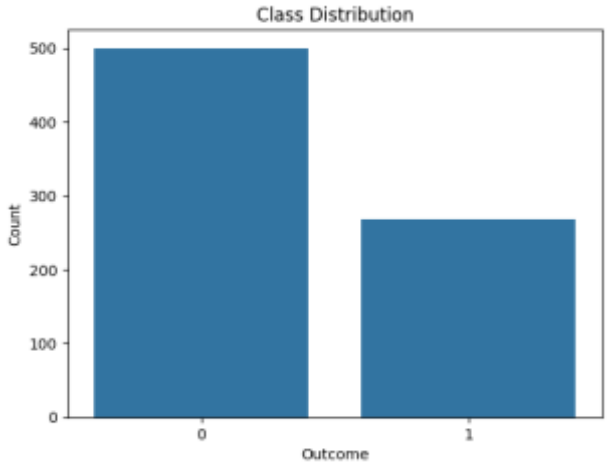

Figure 5 Outcome variables with imbalanced data

In Figure 6 below, the outcome variable before being done with the smote technique, the data is not balanced and after oversampling using smote, the data will be balanced as shown below.

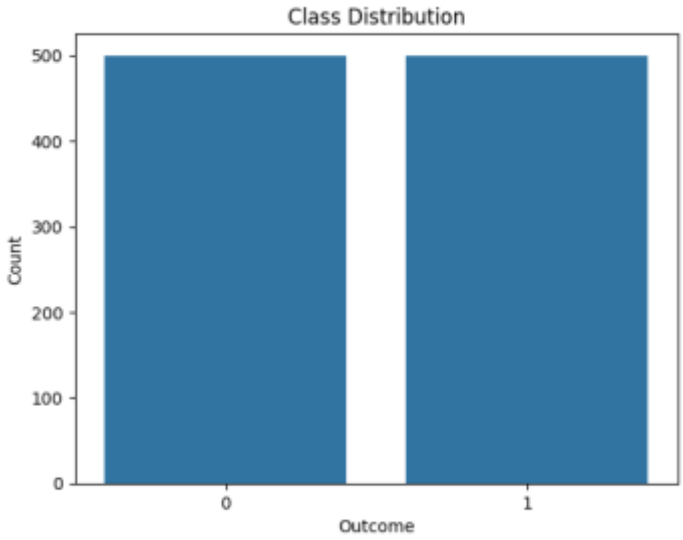

Figure 6 Outcome variables after oversampling using smote

2.3 Classification Using Random Forest

Before the existence of random forests or the development of random forests, there were other studies on methods that produce multiple classifications and combine the results [14].

Random Forests have been used in various research fields involving regression, survival analysis, and classification [15]. Random Forest is a classification algorithm in the form of a large number of decision trees. Each decision tree stands independently with unspecified but identically distributed sample values, and is applied uniformly to all trees in the ensemble [11]. Random Forest is a method in supervised learning developed by Leo Breiman. This algorithm is well-known for its accuracy in prediction, its ability to handle a large number of input variables without overfitting, and its ability to reduce the correlation between decision trees, which is a hallmark of ensemble methods [16]. In addition, Random Forest also has a lower error rate for diabetes datasets than other classification algorithms, and has also been proven effective in classifying cases [13]. The following is an overview of the framework of the random forest algorithm.

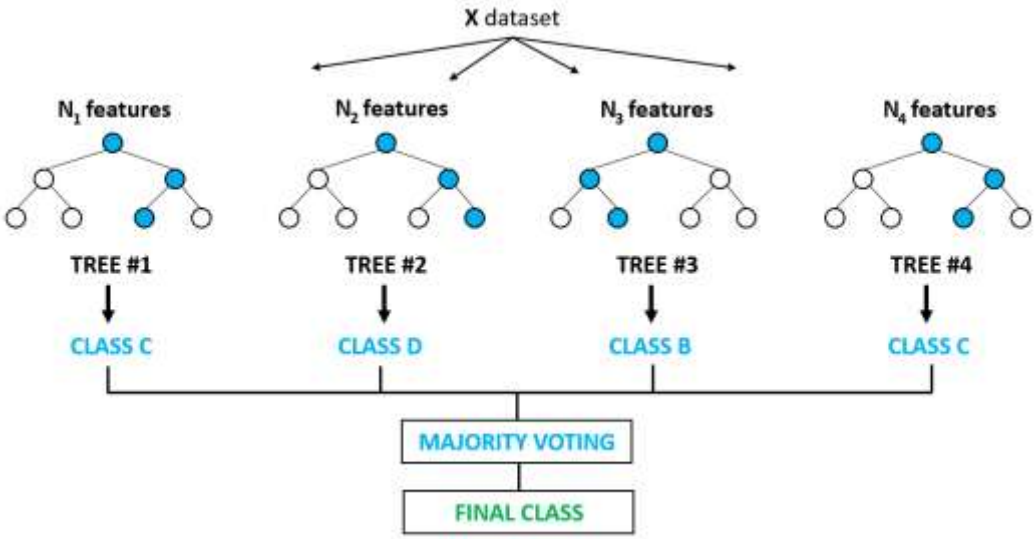

Figure 7 How the random forest algorithm works
Source: Kaggle.com

From Figure 7, it can be explained that the Random Forest Algorithm operates by building a large number of decision trees from randomly selected subsets of data (Random Feature Selection). Each decision tree provides a prediction for the new data, and the final prediction result is determined through Majority Voting. Overall, Random Forest combines the predictions from each tree and determines the final result based on the majority vote of those predictions [17]. One method to improve the performance of training datasets is to perform parameter tuning (hypertuning) or experimentation with repeated k values. The goal is to find the optimal k value that produces the best performing model. This process involves dividing the dataset into two parts: training data and testing data. On the training data, the model is trained using various k values to find the value that yields optimal performance. The best k value is then used to train the model on the testing data [18].

2.4 Random Forest Basic Equation

The first step in determining a decision tree is to calculate the entropy and information gain values. Entropy is used to measure the impurity of attributes, while information gain measures how much information is gained when splitting nodes. The calculation of this greatly affects the main nodes and the splitting nodes, and continues until the calculation results reach zero. From this step, the following are the equations for calculating entropy and information gain values [19].

$$Entropy(s) = \sum_{i=1}^{n} -p_i log_2 p_1 \quad (1)$$

Decription:
A : Attributes
S : Dataset
| S_i | : Number of Samples for the i-th value
| S | : The total number of data

$$Information\ Gain(A) = Entropy(s) = \sum_{i=1}^{n} \frac{S_1}{S} x\ Entropy(S_1) \quad (2)$$

Description:
S : Dataset
n : Number of Classes
p_i : Probability of Ith class in output S

2.5 Evaluation Model

Evaluation of classification models through confusion matrix is needed to measure the performance and results of the model on test data. Confusion matrix is a calculation tool used to analyze how well a classification model identifies data based on existing classes [20]. Confusion matrix displays the classification performance by showing the correctness or incorrectness of the prediction on the object [14]. In the confusion matrix, there are important concepts such as True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). True Positive (TP) refers to the amount of data that is correctly predicted as positive by the model, corresponding to the actual class. True Negative (TN) is the amount of data that is correctly predicted as negative by the model, also corresponding to the actual actual class. False Positive (FP) occurs when the model incorrectly predicts data that should be negative as positive. Meanwhile, False Negative (FN) occurs when the model incorrectly predicts data that should be positive as negative. The confusion matrix provides a clear picture of the distribution of the classification model's predictions against the actual class of the data, thus helping in evaluating the performance and accuracy of the model [3] which can be seen in Figure 8.

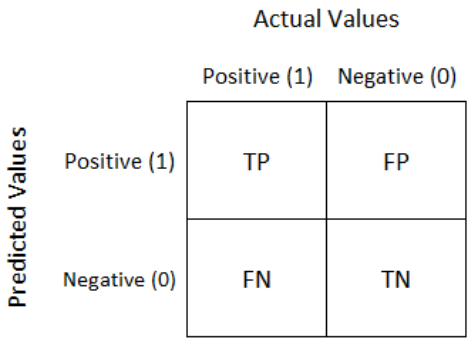

Figure 8 confusion matrix

From Figure 8 it can be explained that the confusion matrix can be used to evaluate the performance of the classification model by calculating accuracy. Accuracy is the percentage of test data that is correctly classified by the classification model that has been built[3].

3. RESULTS AND DISCUSSION

3. 1 Handling Missing Value

This study uses the Pima Indian dataset which has 768 data, all of which are women aged 21 years and over and consists of nine variables with details of eight independent variables, namely Pregnancies, Glucose, Blood Pressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction / DPF, and Age; and one dependent variable, namely Target / Outcome can be seen in Figure 1. Furthermore, at the preprocessing stage in the variables 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI' there is still a value of 0 and which will be replaced with np nan, then np nan will be filled with the mean or average as handling missing values. Furthermore, there is data balancing to handle unbalanced data using the SMOTE technique which can be seen in explanation 2.2.2.

Furthermore, there is a process of handling unbalanced data with a total data of 500 non-diabetics and 268 diabetics or more precisely 65.1% for non-diabetics and 34.9% for diabetics.

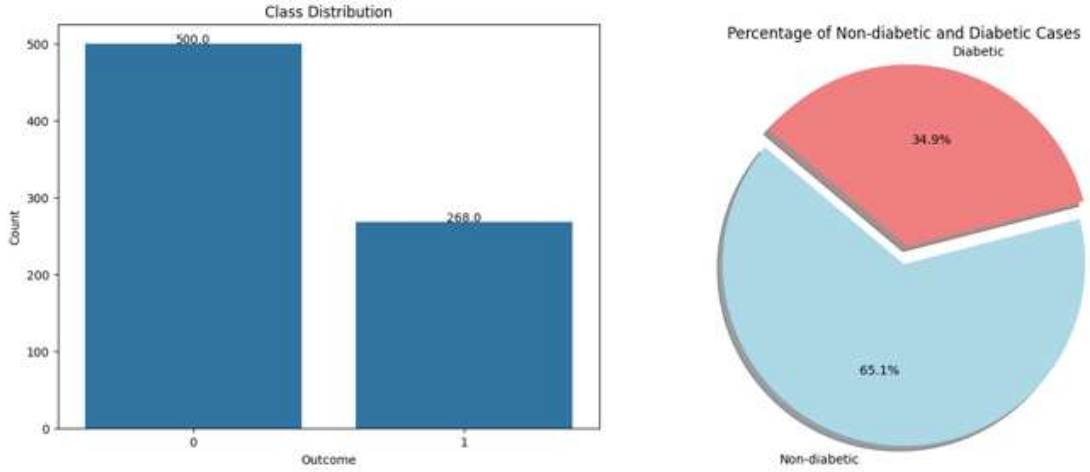

Figure 9 Number of balanced data on outcome variables

After applying the SMOTE technique, the originally unbalanced data has been transformed into balanced, as shown in Figure 6. Next, there is an exploratory analysis of the data, which can be seen in Figure 10. The figure shows the correlation coefficients between the variables, where each cell in the grid reflects the correlation between the two variables corresponding to the intersecting rows and columns. In addition, the distribution plot for each variable is also shown in Figure 11.

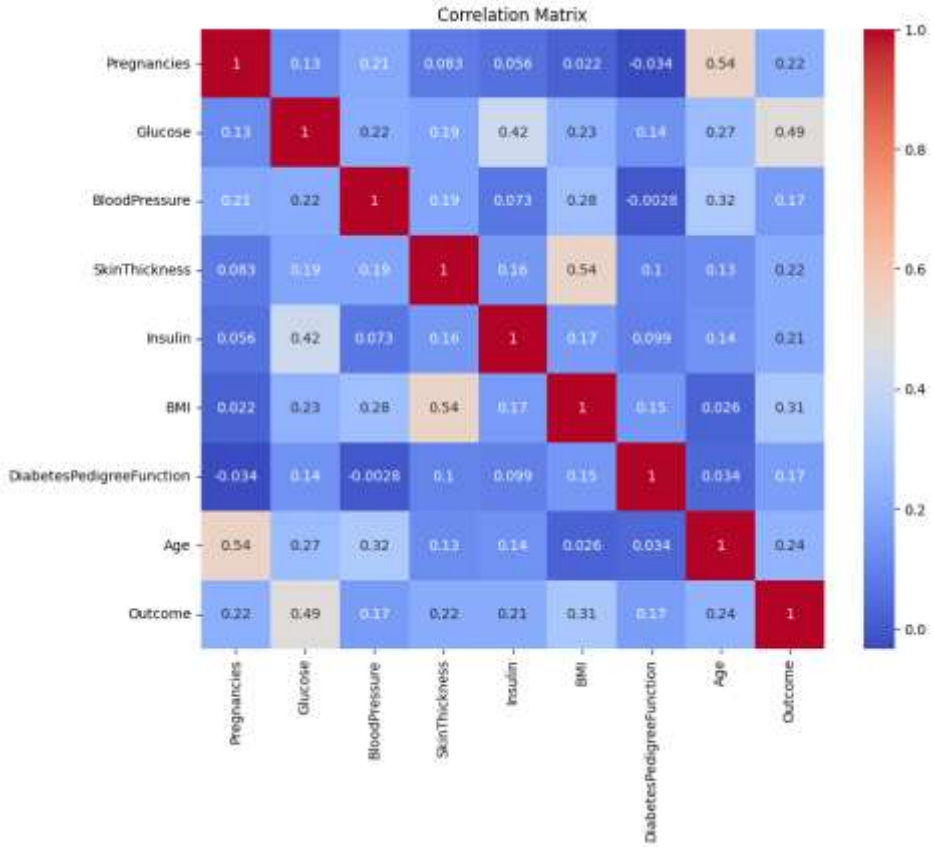

Figure 10 Correlation Matrix

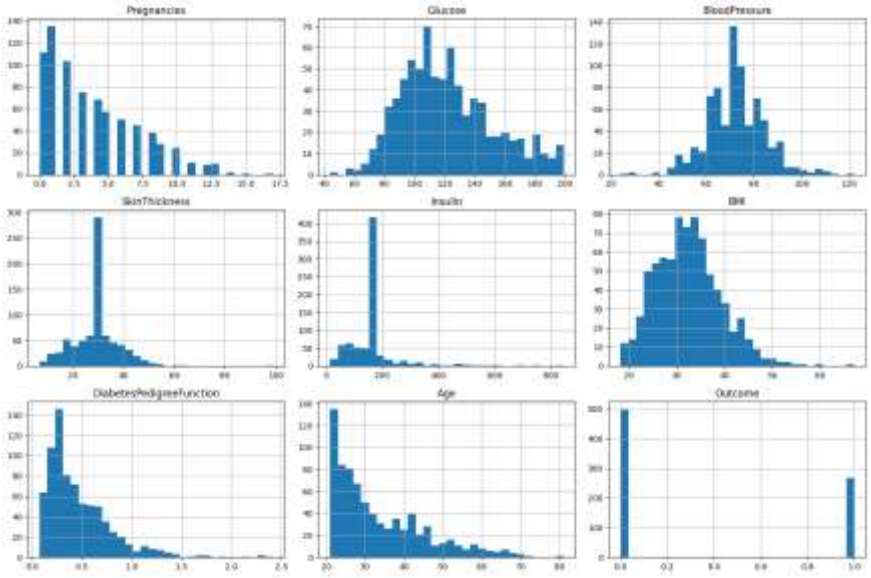

Figure 11 Distribution plots of each variable

Next, there is the pre-processing stage of dividing the training data and testing data, which is 70:30, meaning 70% for testing data and 30% for test data. The next step involves the feature standardization process, which aims to rescale the data so that it has a mean = 0 (centered) and standard deviation = 1, as described in Medium. After that, the Random Forest algorithm was chosen as the classification algorithm, and hyperparameter tuning was performed using the Grid Search method, which can be seen in Figure 12.

```
# Initialize Random Forest Classifier
rf_classifier = RandomForestClassifier(random_state=42)

# Hyperparameter tuning using GridSearchCV with more detailed parameters
param_grid_rf = {
    'n_estimators': [100, 200, 300, 400],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10, 15],
    'min_samples_leaf': [1, 2, 4, 6],
    'max_features': ['sqrt', 'log2']  # Removed 'auto'
}

grid_search_rf = GridSearchCV(rf_classifier, param_grid_rf, cv=5)
grid_search_rf.fit(X_train, y_train)
```

Figure 12 hyperparameters for random forest

In Figure 12, we used the parameter 'n_estimators': [100, 200, 300, 400], 'max_depth': [None, 10, 20, 30], 'min_samples_split': [2, 5, 10, 15], 'min_samples_leaf': [1, 2, 4, 6], and 'max_features': ['sqrt', 'log2']. These parameters aim to improve the model performance of the Random Forest algorithm.

3.2 Model Evalution Result

From the hyperparameter experiment seen in Figure 13, the values of Accuracy, Precision, Recal and F1 Score as seen in Figure 13 are calculated on average from 5 trials.

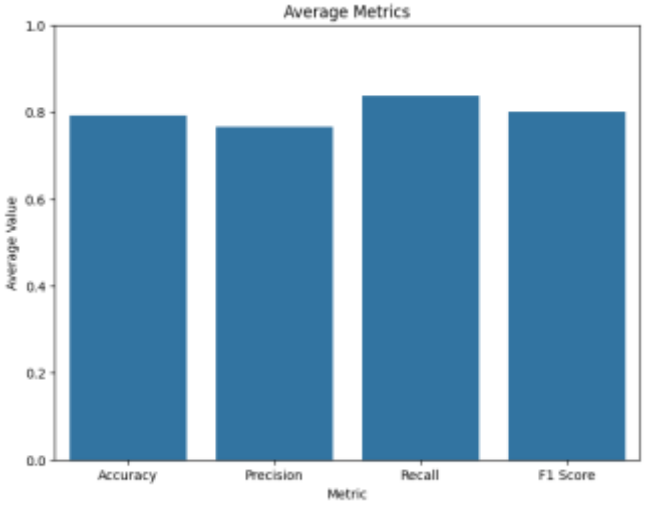

Figure 13 Average model evaluation results

From Figure 13, the results of evaluating the classification model with several performance metrics are obtained. The model accuracy reaches 0.79 or 79%, indicating that 79% of the total predictions made by the model are correct. Precision has a value of 0.768 or 76.8%, which means that 76.8% of all positive predictions made by the model are truly positive. The model has a recall of 0.834 or 83.4%, which indicates that the model successfully identified 83.4% of all true positive cases in the data. The model's F1 Score is 0.8 or 80%, which is the harmonic mean of precision and recall, providing a balance between the two metrics. Overall, these metrics show that the model is good in terms of accuracy, precision, recall, and F1 Score, with balanced values across these metrics. After obtaining the model evaluation results, the confusion matrix was calculated which can be seen in Figure 14 and a snapshot of the results of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) can be seen in Figure 15.

```
# Compute confusion matrix
conf_matrix_rf_tuned = confusion_matrix(y_test, y_pred_rf_tuned)
TN, FP, FN, TP = conf_matrix_rf_tuned.ravel()

# Print the results
print(f"True Positives (TP): {TP}")
print(f"True Negatives (TN): {TN}")
print(f"False Positives (FP): {FP}")
print(f"False Negatives (FN): {FN}")
print(f"Total Correct Predictions: {TP + TN}")
print(f"Total Incorrect Predictions: {FP + FN}")
```

Figure 14 Confusion matrix code snippet

```
True Positives (TP): 72
True Negatives (TN): 101
False Positives (FP): 50
False Negatives (FN): 8
Total Correct Predictions: 173
Total Incorrect Predictions: 58
```

Figure 15 confusion matrix results

From Figure 16, it can be explained that the evaluation results of a classification model, which includes 72 True Positives (TP) and 101 True Negatives (TN), means that the model successfully correctly identified 72 positive cases and 101 negative cases. However, the model also generated 50 False Positives (FP) and 8 False Negatives (FN), i.e. 50 negative cases incorrectly identified as positive and 8 positive cases incorrectly identified as negative. Overall, the model made 173 correct predictions and 58 incorrect predictions. From this information can be seen in Figure 17 for a comparison of the results of the prediction class and the actual class.

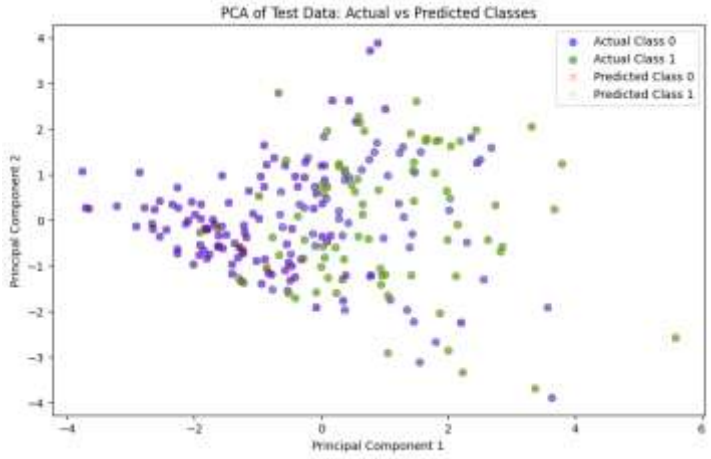

Figure 16 comparison of confusion matrix data

4. CONCLUSION

Missing value is a common phenomenon in data, which can occur due to various reasons such as tool malfunction, inaccurate calculation, unrecorded data, and various other technical issues [21].

In the data pre-processing stage, replacing the value 0 with np.nan and filling with the average value to handle missing values, as well as applying the hyperparameter grid search technique with the parameters 'n_estimators': [100, 200, 300, 400], 'max_depth': [None, 10, 20, 30], 'min_samples_split': [2, 5, 10, 15], 'min_samples_leaf': [1, 2, 4, 6], 'max_features': ['sqrt', 'log2'], successfully improve model performance Based on the research conducted, it can be concluded that the Random Forest Algorithm with grid search hyperparameters is able to classify diabetes data from Pima Indians with good performance. The model achieved 79% accuracy, 76% precision, 83% recall, and 80% F1 Score. The previous basic model results with an average accuracy of 77%, precision of 75%, recall of 77%, and F1-score of 76%; this shows that there is an increase. Compared to the previous basic model using the Logistic Regression algorithm, which has an average accuracy of 77%, precision 75%, recall 77%, and F1-score 76%. For further research for this case, we can use stratified as a data sampling handler and experiment on hyper parameter tuning.

REFERENCE

[1] D. Gunawan, Muhammad Ichsan Sugiarto and I. Mardianto, "JEPIN (Jurnal Edukasi dan Penelitian Informatika) Peningkatan Kinerja Akurasi Prediksi Penyakit Diabetes Mellitus Menggunakan Metode Grid Seacrh pada Algoritma Logistic Regression," *J. Edukasi dan Penelit. Inform.*, vol. 6, no. 3, pp. 280–284, 2020.

[2] I. D. Federation, "Data Diabetes in Indonesia (2021)," IDF. [Online]. Available: https://idf.org/our-network/regions-and-members/western-pacific/members/indonesia/

[3] Gde Agung Brahmana Suryanegara, Adiwijaya, and Mahendra Dwifebri Purbolaksono, "Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 5, no. 1, pp. 114–122, 2021, doi: 10.29207/resti.v5i1.2880.

[4] G. G. Maulana, "Pembelajaran Dasar Algoritma Dan Pemrograman Menggunakan El-Goritma Berbasis Web," *J. Tek. Mesin*, vol. 6, no. 2, p. 8, 2017, doi: 10.22441/jtm.v6i2.1183.

[5] Z. Jin, J. Shang, Q. Zhu, C. Ling, W. Xie, and B. Qiang, "RFRSF: Employee Turnover Prediction Based on Random Forests and Survival Analysis," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12343 LNCS, pp. 503–515, 2020, doi: 10.1007/978-3-030-62008-0_35.

[6] Erlin, Yulvia Nora Marlim, Junadhi, Laili Suryati, and Nova Agustina, "Deteksi Dini Penyakit Diabetes Menggunakan Machine Learning dengan Algoritma Logistic Regression," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 11, no. 2, pp. 88–96, 2022, doi: 10.22146/jnteti.v11i2.3586.

[7] Z. Susanti, P. Sirait, and E. S. Panjaitan, "Peningkatan Kinerja Random Forest Melalui Seleksi Fitur Secara Pca Untuk Mendeteksi Penyakit Diabetes Tahap Awal.," *Sains dan Teknol.*, vol. 4, no. 3, pp. 51–56, 2023.

[8] H. Esmaily, M. Tayefi, H. Doosti, M. Ghayour-Mobarhan, H. Nezami, and A. Amirabadizadeh, "A comparison between decision tree and random forest in determining the risk factors associated with type 2 diabetes," *J. Res. Health Sci.*, vol. 18, no. 2, 2018.

[9] C. N. Noviyanti and A. Alamsyah, "Early Detection of Diabetes Using Random Forest Algorithm," *J. Inf. Syst. Explor. Res.*, vol. 2, no. 1, pp. 41–48, 2024, doi: 10.52465/joiser.v2i1.245.

[10] A. Ali and S. M. Shamsuddin, "Proceedings of the Third International Conference on Computing, Mathematics and Statistics (iCMS2017),"

Proc. Third Int. Conf. Comput. Math. Stat., pp. 19–30, 2019, doi: 10.1007/978-981-13-7279-7.
[11] A. R. Ramdan, N. Widyasono, and H. Mubarok, "Darknet, Malware, KNN, Forensik Prediksi Jaringan TOR dan VPN menggunakan Algoritma K-Nearest Neighbour pada Trafik Darknet," *J. Sist. Cerdas*, vol. 5, no. 1, pp. 21–35, 2022, doi: 10.37396/jsc.v5i1.167.
[12] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "New imbalanced bearing fault diagnosis method based on Sample-characteristic Oversampling TechniquE (SCOTE) and multi-class LS-SVM," *Appl. Soft Comput.*, vol. 101, p. 107043, 2021, doi: 10.1016/j.asoc.2020.107043.
[13] Asniar, N. U. Maulidevi, and K. Surendro, "SMOTE-LOF for noise identification in imbalanced data classification," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3413–3423, 2022, doi: 10.1016/j.jksuci.2021.01.014.
[14] P. A. Hartanto, "PENERAPAN ALGORITMA DECISION TREE UNTUK SELEKSI PENERIMA BEASISWA (STUDI KASUS: SMPN 1 SOREANG)," *J. Compr. Sci.*, vol. Vol 2, no. 7, pp. 1294–1302, 2023.
[15] A. M. Prasad, L. R. Iverson, and A. Liaw, "Newer classification and regression tree techniques: Bagging and random forests for ecological prediction," *Ecosystems*, vol. 9, no. 2, pp. 181–199, 2006, doi: 10.1007/s10021-005-0054-1.
[16] T. N. Nuklianggraita, A. Adiwijaya, and A. Aditsania, "On the Feature Selection of Microarray Data for Cancer Detection based on Random Forest Classifier," *J. Infotel*, vol. 12, no. 3, pp. 89–96, 2020, doi: 10.20895/infotel.v12i3.485.
[17] J. Point, "Random Forest Algorithm," javapoint. [Online]. Available: https://www.javatpoint.com/machine-learning-random-forest-algorithm#:~:text=Instead of relying on one,prevents the problem of overfitting.
[18] T. Cai, "Breast Cancer Diagnosis Using Imbalanced Learning and Ensemble Method," *Appl. Comput. Math.*, vol. 7, no. 3, p. 146, 2018, doi: 10.11648/j.acm.20180703.20.
[19] F. Diba, "Analisis Random Forest Menggunakan Principal Component Analysis Pada Data Berdimensi Tinggi," *Indones. J. Comput. Sci.*, vol. 12, no. 4, pp. 2152–2160, 2023, doi: 10.33022/ijcs.v12i4.3329.
[20] Suyatno, *Machine Learning tingkat dasar dan tingkat lanjut*. Bandung: INFORMATIKA, 2018. [Online]. Available: https://digilib.amikompurwokerto.ac.id/index.php?p=show_detail&id=11989
[21] S. Nikfalazar, C. H. Yeh, S. Bedingfield, and H. A. Khorshidi, "Missing data imputation using decision trees and fuzzy clustering with iterative learning," *Knowl. Inf. Syst.*, vol. 62, no. 6, pp. 2419–2437, 2020, doi: 10.1007/s10115-019-01427-1.