DOI: 10.20885/EKSAKTA.vol2.iss1.art5

Bidikmisi with capable conditions and vice versa. For the above reasons, the researchers chose to use the boosting and bagging algorithm to classify the Bidikmisi imbalance data. Two boosting algorithms are used, namely AdaBoost and SMOTE boosting, and one method, SMOTE bagging. Previously, random forest classification parameter optimization was used, which was then used as the base classifier for boosting and bagging models.

Table 1. Criteria of Bidikmisi Scholarship

Y	CC	AC	Criteria	Interpretation
1	0	0	False	Acceptance criteria is false if the grantee (Y=1) is followed with the category of a wealthy family
0	1	0	False	Acceptance criteria is false if the grantee (Y=0) is followed with the category of a poor family
1	1	1	True	Acceptance criteria is true if the grantee (Y=1) is followed with the category of a poor family
0	0	1	True	Acceptance criteria is true if the grantee (Y=0) is followed with the category of a wealthy family

Source: Suryaningtyas (2010)

Literature Review

Boosting Algorithm

Boosting is one of the ensemble methods that improve the performance of a learning algorithm by combining a collection of weak classifiers into a strong final classifier. The main idea of the boosting process is to select a set of training data (training examples) in a number of ways, which are then examined by a basic learner. This process can be accomplished by selecting training patterns that are expected to make the performance of the base classifier worse than even the performance of the base classifier on a regular basis. If this can be achieved, it is hoped that the base learner can create a new base classifier in the next iteration that is significantly different from its predecessor. This is because the base learner is expected to be a weak algorithm, but may also provide a base learner with the output of a classifier that does not make clear predictions [5]. The boosting algorithm used in this study is Adaptive Boosting M2, an extension algorithm from the original boosting algorithm.

Adaptive boosting M2 algorithm:

Input: $(\mathbf{x}_i, y_i),...,(\mathbf{x}_m, y_m)$ where $\mathbf{x}_i \in \mathbf{X}$ dan $y_i \in \mathbf{Y} = \{1,...,k\}$

Given: $B = \{(i, y) : i \in \{1,...,m\}, y \neq y_i\}$

Initial: $D_1(i, y) = 1/|B|$ for $(i, y) \in B$

for $t = 1, 2, ..., T$

Train weal learner use the distribution of D_t

Compute a weak hypothesis $h_t : X \times \mathrm{Y} \rightarrow [0,1]$ with pseudo-loss, Equation (1).

$$\varepsilon_t = \frac{1}{2} \sum_{(i,y) \in B} D_t(i, y)\left(1 - h_t(x_i, y_i) + h_t(x_i, y_i)\right) \quad (1)$$

if $e_t > 0{,}5$, then the learning process stops.

Set $\beta_t = \dfrac{\varepsilon_t}{(1-\varepsilon_t)}$ (2)

Update weight values

$$D_{t+1}(i, y) = \frac{D_t(i, y)}{Z_t} \times \beta_t^{\left(\frac{1}{2}\right)\left(1 + h_t(x_i, y_i) - h_t(x_i, y_i)\right)} \quad (3)$$

where Z_t is a normalization contact that makes $\sum_{i=1}^{m} D_{t+1}(i,y) = 1$

Output:

$$H(x) = \arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t} \right) h_t(x,y) \quad (4)$$

Bagging Algorithm

Bagging is a method of modifying the results of classification algorithms in machine learning. This method was introduced by [6] and stands for the word "bootstrap aggregating". In the classification with two possible classes, the classification algorithm forms the H: D classifier → {-1,1} as the basis for the training data. The bagging method determines the order of the *Ht* classifier, where t = 1, ..., T as a modification of the training data. The classifier is then combined into a combined classifier. The prediction results of the combined classifier are then given as combined weights of the individual classifier or referred to as the voting procedure. According to [6], bagging is an effective ensemble technique for unstable learning algorithms, where small changes in training data sets can make large changes in predictions for e.g. decision trees, neural networks, etc.

Synthetic minority Over-Sampling (SMOTE) Algorithm

SMOTE is one of the Chawla [7] methods that can handle the imbalanced dataset. The basic idea of SMOTE is to increase the number of samples in the minor class. It allows to match the main class by generating synthetic data based on the k nearest neighbor. In this case the nearest neighbor based on the Euclidean distance between data. Given a dataset with p variables $\mathbf{x}^T = \left[x_1, x_2, ..., x_p\right]$ and $\mathbf{z}^T = \left[z_1, z_2, ..., z_p\right]$ then the Euclidean distance *d(x,z)* generally as below:

$$d(\mathbf{x},\mathbf{z}) = \sqrt{(x_1 - z_1)^2 + (x_2 - z_2)^2 + ... + (x_p - z_p)^2} \quad (5)$$

Generate synthetic data using the following equation:

$$\mathbf{x}_{syn} = \mathbf{x}_i + (\mathbf{x}_{knn} - \mathbf{x}_i)\gamma \quad (6)$$

where:

x_{syn}: synthetic data
x_i: the *i*-th data from the minor class
x_{knn}: data from the minor class having the smallest distance to x_i
γ: a random number between 0 and 1.

SMOTE-Boosting Algorithm

This algorithm was proposed by [7] in 2002. SMOTE-boosting combines the SMOTE algorithm and the standard boosting procedure, by utilizing SMOTE to improve the prediction of minority classes. The purpose of combining the SMOTE and AdaBoost algorithms is to increase the value of the True Positive (TP) rate [7]. In the AdaBoost iteration procedure, the classifier component classification results are first put into probabilities form, which is later used to calculate the pseudo-loss.

SMOTE-boosting algorithm:

Input: $(\mathbf{x}_i, y_i), ..., (\mathbf{x}_m, y_m)$ where $\mathbf{x}_i \in \mathbf{X}$ dan $y_i \in \mathbf{Y} = \{1, ..., k\}$

Given: $B = \{(i,y) : i \in \{1, ..., m\}, y \neq y_i\}$

Initial: $D_1(i,y) = \frac{1}{|B|}$ for $(i,y) \in B$

for t = 1,...., T

Modify the distribution of D_t by creating N synthetic examples from minority class using the SMOTE algorithm.

Train weak learner using the distribution of D_t

Compute a weak hypothesis with pseudo-loss in Equation (7).

$$\varepsilon_t = \sum_{(i,y)\in B} D_t(i,y)(1 - h_t(x_i, y_i) + h_t(x_i, y_i)) \quad (7)$$

if $e_t > 0{,}5$, then the learning process stops.

Set $\beta_t = \frac{\varepsilon_t}{(1-\varepsilon_t)}$

Update weight value:

$$D_{t+1}(i,y) = \frac{D_t(i,y)}{Z_t} \times \beta_t^{\left(\frac{1}{2}\right)(1+h_t(x_i,y_i)-h_t(x_i,y_i))} \quad (8)$$

where Z_t is a normalization contact that makes $D_{t+1}(i,y) = 1$

Output:

$$H(x) = \arg\max_{y\in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t} \right) h_t(x,y) \quad (9)$$

SMOTE-Bagging Algorithm

SMOTE-bagging algorithms is a combination of SMOTE as mention earlier and bagging algorithms. SMOTE-bagging involves steps to generate synthetic data while creating subsets [8]. Based on SMOTE-bagging, each subset obtained by the bootstrap process is matched with SMOTE before the model is formed. Synthetic data is created based on two parameters, the amount of oversampling (N) from the minority class and k closest neighbors. Total oversampling is decided so that the number of major and minor classes is balanced.

SMOTE-bagging algorithm:
Let *D* the training dataset
for t = 1, 2,…, T, construct subset *Dt* from classes with the same number by executing the following:
Resample class *C* with replacement at 100%
For each class *i* (*1,2,…, C-1*):
Resample from original instances with replacement, where *b%* is multiple of 100%
Set $N = \left(N_C / N_i \right).(1-b\%).100$
Generate new instances by using SMOTE (*k, N)*
Train a classifier from D_t
Change percentage *b%*
Repeat step 2 and 3 until convergent

Testing on a new instance:
Generate outputs from each classifier
Return the class which gets the most votes from classifier *Ht*: *Dt* → *R*

$$H\left(d_i, c_j\right) = sign\left(\sum_{t=1}^{T} \alpha_t H_t\left(d_i, c_j\right) \right) \quad (10)$$

Performance Criteria for Classification

Confusion matrix contains information about the actual data class represented on the matrix row and the predictive data class on the column [9], it has actual and predictive data from the classification model, then presented using a cross-tabulation.

Table 2. Confussion Matrix

		Predictive	
		Positive Class	**Negative Class**
Real	Positive Class	True Positive (TP)	False Negative (FN)
	Negative Class	False Positive (FP)	True Negative (TN)

Area Under Curve (AUC) have been used to understand the performance of learning algorithms in minority classes. Then, to evaluate the overall method's performance, a geometric mean (G-mean) is employed. G-mean is a geometric mean of sensitivity and specificity. If all positive classes cannot be predicted, the G-mean will be zero so that a classification algorithm is expected to reach a high G-mean value [10]. Therefore, in this paper, to evaluate the overall method performance, a G-mean and AUC analysis can be used.

$$Specificity = \frac{\text{TN}}{(\text{TN}+\text{FP})} \times 100\% \tag{11}$$

$$Sensitivity = \frac{\text{TP}}{(\text{TP}+\text{FN})} \times 100\% \tag{12}$$

$$G\text{-}Mean = \sqrt{\text{Sensitivity} \times \text{Specificity}} \tag{13}$$

AUC obtained by calculating the value of true positive rate, which is the number of objects in positive classes that are correctly classified (TPR) and false positive rate, which is the number of objects in positive classes that are incorrectly classified (FPR).

$$\text{TPR} = \frac{\text{TP}}{(\text{TP}+\text{FN})} \tag{14}$$

$$\text{FPR} = 1 - \frac{\text{TN}}{(\text{TN}+\text{FP})} \tag{15}$$

$$\text{AUC} = \frac{1+\text{TPR}-\text{FPR}}{2} \tag{16}$$

Methodology

Source of Dataset

The data used is the Database of the Ministry of Research, Technology, and Higher Education through Bidikmisi channel. We used dataset from East Java province in 2013 – 2017.

Table 3. Data Description

Data	**Total Data (N)**	**Variable**	**Categories (Y)**	
Bidikmisi dataset	52631	11	1	Accepted = 50796
			0	Unaccepted = 1835

Reserch Design

In this research study, the experiments were used to perform on the Bidikmisi dataset summarize in Table 2, the dataset contains 52631 records and 11 variables. The variables can be classified as demographic attributes (such as occupation, education, housing ownership, area of residential land, area of a residential building, etc). The number of categories distributed into type, as we can see in **Fig. 1**. Classification analysis procedure using boosting and bagging algorithms is given in the research flowchart, as seen in **Fig. 2**.

E-ISSN: 2720-9326
P-ISSN: 2716-0459

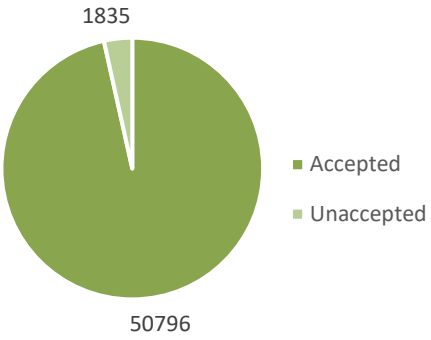

Figure 1. Data Characteristic Based on Student Status

Tools for Analysis

As we mentioned before, this study aims to determine which classification analysis had the best performance. By competing the ensemble methods which are AdaBoost, SMOTE-Boosting, SMOTE-Bagging. The methods is analyzed using R software and ebmc package.

Result and Discussion

Preprocessing Bidikmisi Dataset

The steps involved in preprocessing data include combining data from various sources, cleaning data to remove noise, filling missing value and eliminating outliers. In this paper, we cleaned the missing value, therefore, there is no noise in the dataset. From **Table 4**, there was a missing value for each variable that will be removed from the study. Therefore, the total data used for research is 52631, after detecting the missing value and handling it by deleting the data.

Table 4. Missing Value in Bidikmisi Dataset

	N		%
	Valid	Missing	Missing
X1	53661	0	0
X2	53661	0	0
X3	53661	0	0
X4	53661	0	0
X5	53661	0	0
X6	52861	800	1,5
X7	52884	777	1,4
X8	53661	0	0
X9	53661	0	0
X10	53130	531	1
X11	53190	471	0,9

E-ISSN: 2720-9326
P-ISSN: 2716-0459

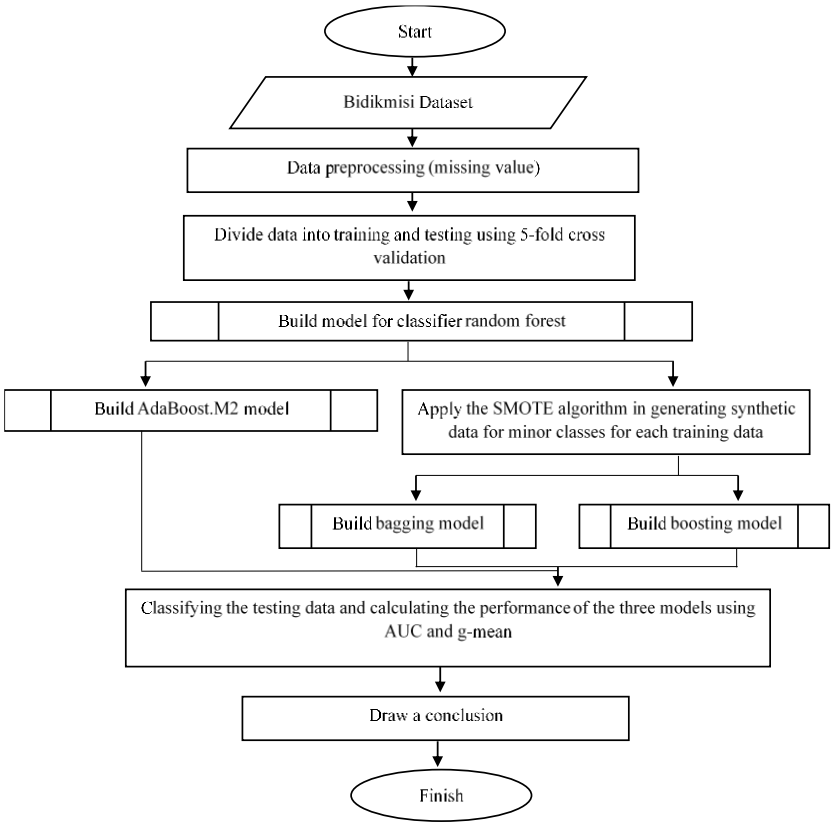

Figure 2. Flowchart Analysis on bidikmisi Dataset

Adaptive Boosting Classification Analysis

By dividing the dataset into training data and testing data, the model for AdaBoost was formed. This study used 5-fold cross-validation with a 20% partition for each fold. **Fig. 3** shown the performance criteria G-means and AUC for adaptive boosting algorithm, the value G-mean and AUC tend to reach a peak when the iteration is 50, with 50.256% and 9.027%.

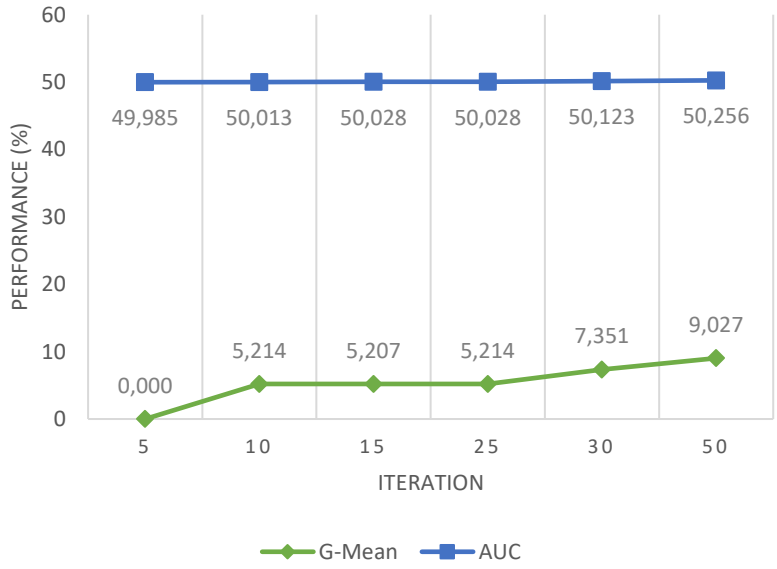

Figure 3. G-mean and AUC Value for Adaptive Boosting Algorithm

E-ISSN: 2720-9326
P-ISSN: 2716-0459

SMOTE_Boosting Classification Analysis

As we discussed before, the dataset divided into two sets of training data and testing data using 5-fold cross-validation. Then, create synthetic data to balance the class composition and minor classes using the SMOTE algorithm. Therefore, we conclude the G-mean and AUC values as shown in **Fig. 4**.

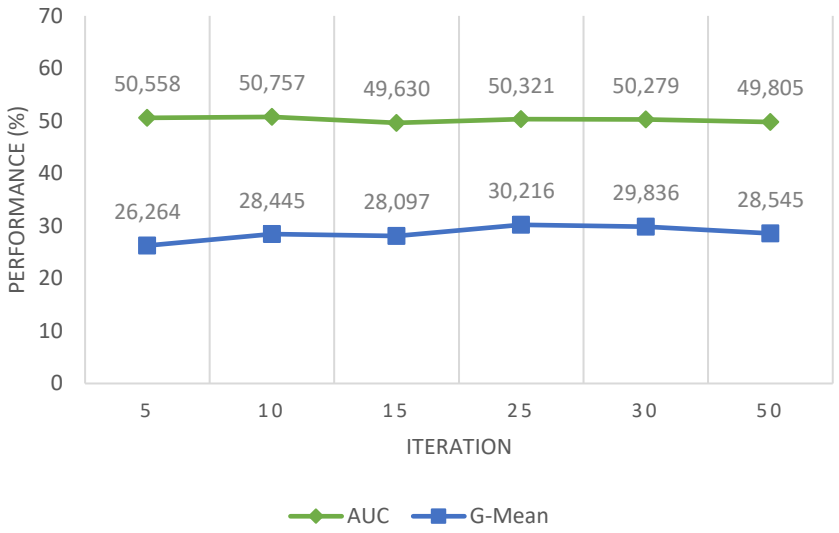

Figure 4. G-mean and AUC Value for SMOTE-Boosting Algorithm

Fig. 4 shows the method performance, G-mean and AUC values tend to be stable without a drastic increase. This value is bigger than the previous methods.

SMOTE-Bagging Classification Analysis

The initial step that must be done in the analysis of the bagging method has divided the data into training and testing dataset with 5-fold cross-validation. Then, bootstrapping is performed on the training data. Because this method combines bagging and SMOTE algorithm, after bootstrapping the data will be generated again using SMOTE. **Fig. 5** shows the classification performance for the Bidikmisi dataset using 5 iterations. The G-mean value reached a maximum at the 10th iteration, with a value of 33.129% and the maximum AUC value at the 50th iteration was 32.924%. This method slightly better than the previous methods based on the G-mean and AUC values.

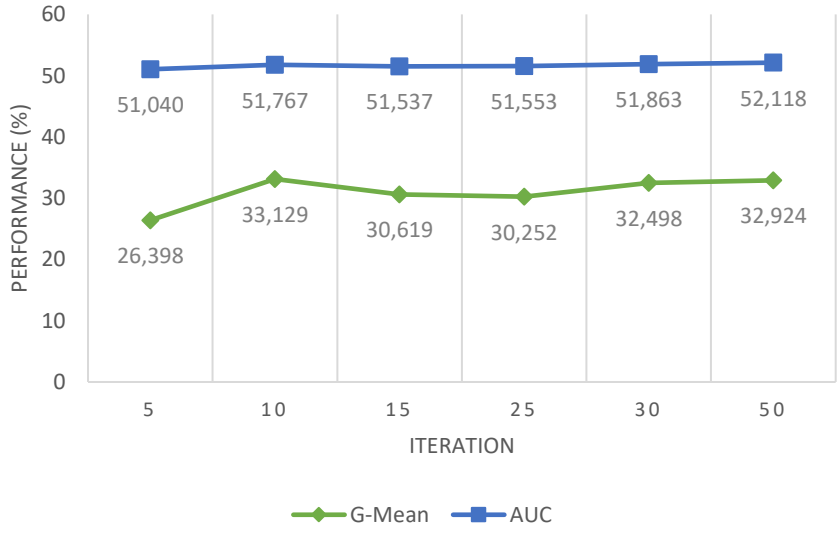

Figure 5. G-mean and AUC Value for SMOTE-Bagging Algorithm

Comparisson of Performance Classification Methods

After analyzing the Bidikmisi dataset using AdaBoost, SMOTE-Boosting, and SMOTE-Bagging, the comparison of the method performance of all optimum models is obtained. A comparison of several methods is measured using the performance of classification methods which include precision, recall, f-value, sensitivity, and specificity, g-mean and AUC which are the results of classification with the best parameters of each method. Comparison of the results of the classification is shown in Table 4.4, it can be seen that the performance of all methods shows that SMOTE-Boosting and SMOTE-Bagging have values that tend to be the same. The accuracy of the positive class classification made by the AdaBoost model, which is an average of the fifth fold classification of 8.8235% which means that on average only 8.8235% of observations in each fold of the Bidikmisi data have been classified correctly. If seen from the sensitivity and specificity values, AdaBoost can only classify 0.8174% of observations originating from unaccepted (minority) status as unacceptable classes but succeed in classifying 99.97% of observations originating from accepted (majority) status as classes received. The existence of imbalance cases in the data causes a low sensitivity value because the random forest separator function tends to classify observations into the majority class, so the classification of minority classes is only correctly classified as less than 1%. After balancing the data in both classes with SMOTE and boosting and bagging, better results were obtained. This is evidenced by the performance of the method with the g-mean value obtained using SMOTE-Boosting random forest and SMOTE-Bagging random forest is higher than AdaBoost.

Table 5. Comparisson of Performance Methods for Bisikmisi Dataset

Model	Average 5-fold							
	Accuracy	*Precision*	*Recall*	*F-Value*	*Sensitivity*	*Specificity*	*G-mean*	*AUC*
AdaBoost	0,9648	0.088235	0.008174	0.014963	0.008174	0.999705	0.090274	0.502561
SMOTE-Boosting	0,9074	0.041885	0.100817	0.056587	0.100817	0.937592	0.302161	0.50757
SMOTE-Bagging	0,9167	0.053549	0.119891	0.073504	0.119891	0.947239	0.331291	0.521178

Based on **Table 5.** it can also see that AdaBoost produces high accuracy and specificity values. This is because in its boosting process, AdaBoost managed to take advantage of the misclassification made by random forest in each boosting iterations, therefore it can improve the accuracy of classification, especially classification on the majority class. While SMOTE-Boosting and SMOTE-Bagging produce almost the same method of performance values on all criteria due to the process of balancing the distribution of training set classes, resulting in increased classification accuracy in the minority classes. The recall value provides information on how many minority classes are identified but may sacrifice precision by misclassifying the majority class. The F-Value combines precision and recall, to measure the goodness of the learning algorithm.

Fig. 6 presents a boxplot of G-mean values generated in each model. The G-mean value generated using the SMOTE-Bagging shown by the yellow color in the figure tends to be slightly higher compared to SMOTE-Boosting, the value ranges from 26% to 33%. The blue box is the SMOTE-Boosting, shown that the variation of the G-mean tends to be smaller than the other two algorithms, the value ranges from 26% to 30%.

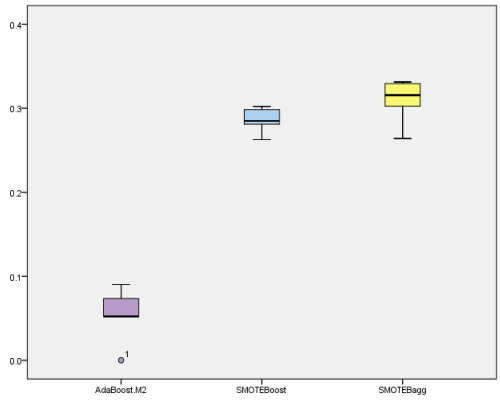

Figure 6. Boxplot Performance of G-mean

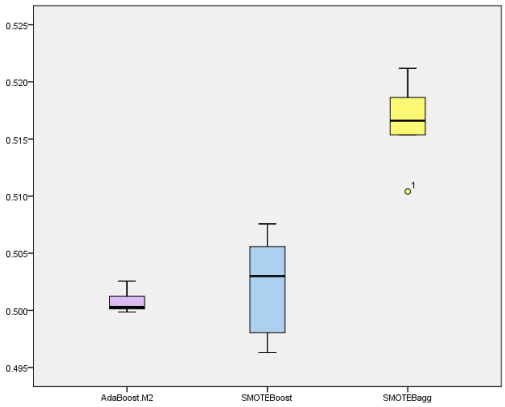

Figure 7. Boxplot Performance of AUC

Next, **Fig. 7** presents a boxplot of the AUC value generated in each model. AUC values generated using the SMOTE-Bagging indicated by the yellow color in the image, tend to be higher than in other methods. The variations range from 51% to 52%.

Bidikmisi Applicant Classification Results

Bidikmisi Scholarship is a government scholarship for prospective students who are economically disadvantaged and have good academic potential. Because this scholarship is for poor students, the main requirement for applying for a scholarship is if the income per capita of parents/guardians is divided by the number of family members in the amount of Rp. 750,000.00 every month. Before proceeding to the next discussion, there are several steps undertaken to obtain the values presented in **Table 6**.

1. Take the variable Y as the response variable
2. Select variables "father's income", "mother's income", and "family dependant"
3. Create a new variable by counting the amount of "father's income" and "maternal income" divided by "the number of family dependant", name it with "Code category (CC)".
4. Coding the variable "CC" with the following criteria:
   0 = if CC > Rp. 750,000 per head in the family, included in the condition of a wealthy family
   1 = if CC < Rp. 750,000 per head in the family, fall into the condition of poor families
5. Match the response variable (Y) to the CC in Step 4 to the AC (Acceptance Condition) with the Bidikmisi acceptance classification from previous methods (see table 1).

Table 6. Example of The Identification of Bidikmisi Data classification Condition

Object	Parents Income (rupiah)	Income per Capita (rupiah)	Actual	Prediction	Acceptance Condition
1	625,000	156,250	1	1	True
2	1,875,000	937,500	1	0	False
3	875,000	875,000	1	0	False
4	1,375,000	1,375,000	1	0	False
5	875,000	175,000	1	1	True
6	1,875,000	625,000	1	1	True
7	2,750,000	687,500	1	1	True
8	875,000	218,750	1	1	True
9	1,000,000	500,000	1	1	True
10	1,250,000	178,571	1	1	True

Based on Bidikmisi requirements, students are entitled to get a scholarship if the maximum income per capita is Rp. 750,000.00. The results of the "prediction" status of scholarship acceptance indicate that there are 3 students who should not be entitled to get a scholarship, the "actual" results indicate that the ten students turned out to have received scholarship status. In total, there were 10,762 students who get a status "incorrectly classified" condition as shown in **Fig. 8**.

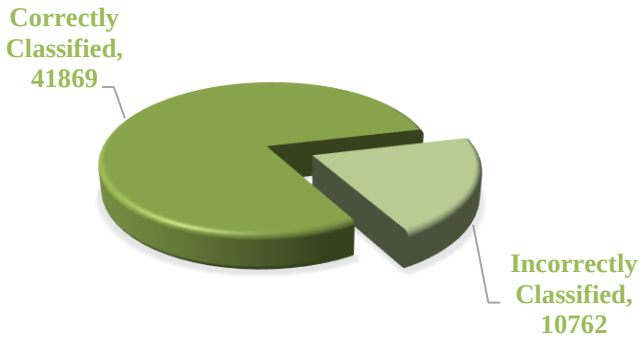

Figure 8. Identification of Bidikmisi Data Classification Condition

E-ISSN: 2720-9326
P-ISSN: 2716-0459

To find out how good the methods used in this research, the prediction results for each method are explained in the following **Table 7**.

Table 7. Identification of Classification Methods

Method	Iteration	∑ Incorrectly Classified	∑ Correctly Classified	% Correctly Classified
AdaBoost	5	356	10170	96,618
	10	379	10147	96,399
	15	375	10151	96,437
	25	376	10150	96,427
	30	383	10143	96,361
	50	387	10139	96,323
SMOTE-Boosting	5	984	9542	90,652
	10	1083	9443	89,711
	15	1280	9247	87,839
	25	1281	9246	87,830
	30	1259	9268	88,039
	50	1279	9247	87,840
SMOTE-Bagging	5	880	9464	91,639
	10	1192	9334	88,676
	15	1052	9474	90,006
	25	1020	9506	90,309
	30	1123	9403	89,331
	50	1122	9404	89,341

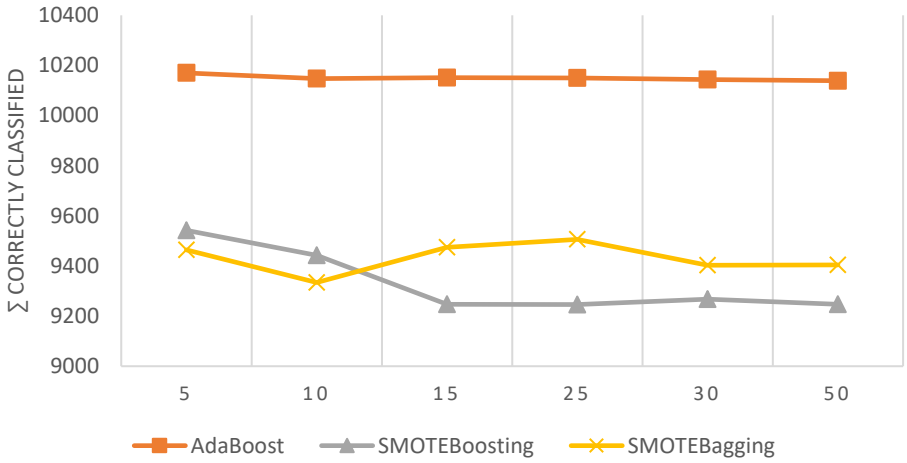

Figure 9. Identification of Classification Methods

Conclusions

Data mining often used to uncover hidden patterns within a large amount of data. Then, these hidden patterns can be potentially be used to predict future behavior. In relation to Educational Data Mining (EDM) which has been discussed at the beginning, the data mining algorithms were performed to find a prediction for Bidikmisi dataset. It was previously known that the data class has imbalance conditions, therefore, three ensemble algorithms were performed, i.e. Adaptive Boosting, SMOTE-Boosting, and SMOTE-Bagging. All models have been evaluated using stratified 5-fold cross-validation, therefore, the performance criteria for each method are examined.

a. The results of the imbalance class showed that the SMOTE-Bagging and SMOTE-Boosting algorithm has better accuracy than the AdaBoost algorithm. It could be said that both methods were quiete successful in taking advantage the SMOTE algorithm combine with boosting and bagging algorithm.