Cost-Sensitive Boosting Algorithm for Classifying Underdeveloped Regions in Indonesia

B Suseno[1,*], **B Sartono**[1], **K A Notodiputro**[1]

[1] Department of Statistics, Faculty of Mathematics and Natural Sciences, IPB University, Indonesia

*Corresponding author's e-mail: bayu.suseno@apps.ipb.ac.id

Abstract. Imbalanced classes are indicated by having more instances of some classes than others. The cost-sensitive boosting algorithm is a modification of the AdaBoost algorithm, which aims to solve the problem of imbalanced classes. In this study, we evaluate the cost-sensitive Boosting algorithm AdaC2 using Indonesia's underdeveloped region's data. This study confirms that the cost-sensitive boosting algorithm (AdaC2) performs better in classifying the instances in the minority classes than standard classifiers algorithms.

1. Introduction

The classification method's objective is to classify objects on all class classifications accurately. Previous research often uses standard classifier methods such as decision trees, random forests, and boosting algorithms. These standard methods are often faced with the imbalanced classes problem listed as one of ten challenging problems in data mining research [25].

The imbalanced classes problem is characterized by the data distribution having more instances of some classes than others. In the case of bi-classes, this problem is indicated by one class having large instances while the other has a few [10]. The standard classifier methods generally need to improve on imbalanced class datasets that could have high overall accuracy; unfortunately, they give lower accuracy in the small class. In the problem of imbalanced classes, improvement in overall classification accuracy is not the most important thing because it is biased toward the majority class, while the accuracy in minority classes is not satisfying [10]. The improvement in accuracy in the minor class is often more important than the majority class, which significantly contributes to the overall accuracy. The researcher has found some methods, such as resampling techniques and cost-sensitive learning [25].

The first method, resampling techniques, is a pre-classification step that re-balanced the datasets using resampling techniques and then used a standard classifier to the data in the next step. The second method, cost-sensitive learning, such as cost-sensitive boosting, adds the cost matrix to the classifier algorithm, so the algorithm prioritizes classifying the minority class as having a higher cost than the majority class.

Boosting is one of the classification methods to improve the performance of weak learning algorithms. One of the algorithms developed is AdaBoost, which can adaptively to errors from weak hypotheses generated by WeakLearn [15]. The cost-sensitive boosting algorithm is a modification of the AdaBoost algorithm, which aims to solve the problem of imbalanced classes. The Cost-Sensitive Boosting algorithm has three different algorithms, namely the AdaC1, AdaC2, and AdaC3, depending on its cost factor to update the weight distribution [10]. Based on a previous study, the AdaC2 algorithm generally performs better than the AdaC1 and AdaC3 algorithms [10]. Numerous empirical studies

about the AdaC2 algorithm are using health sector dataset [1][3][4][10][11], image analysis [7][11], and machine learning repository UCI/keel dataset repository [2][6][8][21]. However, we have yet to find research that used the AdaC2 algorithm for socioeconomic datasets in their study. In this study, we are filling this gap by performing an empirical study of the AdaC2 algorithm to Indonesia's socioeconomic imbalanced classes datasets compared with standard methods as a benchmark. The objective is to classify the underdeveloped regions in this dataset with imbalanced class problems.

Developing underdeveloped regions in Indonesia is critical to improving society's welfare. According to the Presidential Regulation Document Number 63 of 2020, there were 62 underdeveloped districts from 514 districts, or about 12 percent of underdeveloped districts in Indonesia. These districts are distributed in many islands: 7 in Sumatra, 14 in Nusa Tenggara, 3 in Sulawesi, 8 in Maluku, and 30 in Papua island.

The remainder of the paper is organized as follows. Section 2 presents some literature reviews of cost-sensitive boosting, cost factors, and evaluation measures. Section 3 provides the analysis step in the empirical study using underdeveloped regions data in Indonesia. Section 4 analyzes and compares the weighting strategies of the AdaC2 algorithm. Finally, Section 5 highlights the conclusions and states some points for future research.

2. Literature Review

2.1. Cost-Sensitive Boosting

The AdaBoost algorithm was first introduced by Yoav Freund and Robert Schapire [15]. The AdaBoost algorithm creates several weak learners by adding weight to the training data and adjusting them for each round adaptively. The weight of misclassified instances will be increased while the correct ones will be decreased [24]. Cost-sensitive boosting algorithms are developed by introducing cost factors into the AdaBoost algorithm.

At the beginning round of the algorithm, all instances have the same weight. In the next round, the weight of misclassified instances in the previous round will be increased so that weak learners focus more on misclassified instances. This weak learner generates a weak hypothesis $h_t: X \to \{-1, +1\}$ concerning the distribution of D_t, in which $D_t(i)$ is the weight of instances i at round t. Suppose there are m instances $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X, y_i \in Y = \{-1, +1\}$

$$D_t(i) = \frac{1}{m} \text{ for } i = 1, \ldots, m \tag{1}$$

The Adaboost algorithm step is as follows:

- For t=1,... ,T:
  - Train weak learners by using distribution D_t
  - Getting weak hypothesis $h_t: X \to \{-1, +1\}$
  - Choose h_t that minimizes weighted errors $\epsilon_t = PR_{i\sim D_t}[h_t(x_i) \neq y_i]$
  - Choose $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
  - For $i = 1, \ldots, m$ update

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} \text{ if } h_t(x_i) = y_i \\ e^{\alpha_t} \text{ if } h_t(x_i) \neq y_i \end{cases} \tag{2}$$

$$D_{t+1}(i) = \frac{D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t} \tag{3}$$

By Z_t is the normalization factor

The final hypothesis is

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{4}$$

The Cost-Sensitive Boosting algorithm modifies AdaBoost by including the cost factor in the AdaBoost weighing update formula. Cost factors in the algorithm give higher weight to the misclassified instances in the minority class than the majority class, so the data distribution will be biased toward the minority class. Sun et al. [10] introduced three ways to modify the AdaBoost weighing update formula, namely AdaC1, AdaC2, and AdaC3, depending on the cost factor included in the weighting formula.

Suppose there are m instances $(x_1, y_1, c_1), \dots, (x_m, y_m, c_m)$ with c_i as a cost factor, which is a non-negative real number of the domain R^+, so the weighting update formula with cost factor is

$$D_{t+1}(i) = \frac{D_t(i)\exp\left(-\alpha_t y_i c_i h_t(x_i)\right)}{Z_t} \rightarrow \text{AdaC1} \tag{5}$$

$$D_{t+1}(i) = \frac{c_i D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t} \rightarrow \text{AdaC2} \tag{6}$$

$$D_{t+1}(i) = \frac{c_i D_t(i)\exp\left(-\alpha_t y_i c_i h_t(x_i)\right)}{Z_t} \rightarrow \text{AdaC3} \tag{7}$$

The value of α_t in the AdaC2 algorithm can be calculated using the formula

$$\alpha_t = \frac{1}{2}\log\frac{\sum_{i,y_{i=h_t(x_i)}} c_i D^t(i)}{\sum_{i,y_{i\neq h_t(x_i)}} c_i D^t(i)} \tag{8}$$

2.2. Cost Factor

The cost factor is added in the cost-sensitive classifier to give a higher weight, sometimes called a cost penalty, to the misclassified instance in the minority class. Let $C(i,j)$ is the cost of false prediction of instance that is classified as class i actually, but classified as class j. So let the notation $C(+,-)$ it is the cost of misclassification of an instance in the minority class as a majority class and $C(-,+)$ is the cost of misclassification of an instance in the majority class as a minority class. In the case of imbalanced class problems, correct classifying instances in a minority class is more critical than in a majority class. The wrong classification in the minority classes could be much more harmful that was not expected by a researcher. So that the cost of misclassification instances of the minority class should be greater than the majority class notated by $C(+,-) > C(-,+)$, while correctly classifying an instance is given a value of 0 notated by $C(+,+) = C(-,-) = 0$. The algorithm uses these costs to minimize the cost misclassification [10].

The cost factor could be written in the matrix form as follows

$$C = \begin{bmatrix} 0 & C_{FN} \\ C_{FP} & 0 \end{bmatrix} \tag{9}$$

with notation $C(+,-) = C_{FP}$ and $C(-,+) = C_{FN}$ where $C_{FP} > 0$ is the cost of false-positive and $C_{FN} > 0$ is the cost of false negative. A cost of 0 is given to the correct classification, so $C_{TP} = C_{TN} = 0$ is the cost of true positive and true negative [13]. Values of C_{FP} and C_{FN} can be made different (asymmetry) so that the algorithm is more focused on classes with higher costs [21].

2.3. Single Decision Tree

Single decision tree segmenting the predictor space using splitting rules that can be summarized in a tree. This method uses recursive binary splitting to evaluate the Gini index as a measure of node purity.

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{10}$$

With G is the Gini index, and $\hat{p}_{mk}$ is the proportion of observations in the mth region from the kth class [28].

2.4. Random Forests

Breiman (2001) introduces Random forests [29] as an ensemble tree-based learning algorithm that averages predictions over many individual trees. The individual trees are built on bootstrap samples and a subset of random predictor variables. The algorithm from Random Forests is

- Set the number of trees to grow
- For each tree:
    a. Draw a bootstrap sample of size N from the training data
    b. Set the number of subset variables m. Randomly select a subset of m predictor variables from the total p for each node in each tree
    c. Grow the tree to maximum
    d. Use out-of-bag training data to estimate error
- Assign a class to new data as the majority vote among all the trees
- Estimate the classification accuracy

2.5. Underdeveloped Regions

Based on Indonesia's Government Regulation Number 78 of 2014 concerning the Development Acceleration of Underdeveloped Regions, the definition of underdeveloped regions is the districts whose areas and communities are less developed than others. A district is considered an underdeveloped area based on the criteria: (a) the economy of the community, (b) human resources, (c) infrastructure, (d) regional financial capabilities, (e) accessibility, and (f) regional characteristics.

2.6. Evaluation Measures

Evaluation measures are used to evaluate the classification method's performance. Overall accuracy is not appropriate for assessing the performance in the imbalanced classes cases since the minority classes have a lower influence on overall accuracy than the majority class [10]. Overall accuracy could not give information about accuracy in the minority class. This study uses evaluation measures recall, precision, and $F_{\beta=2}$ or F_2 to evaluate class classification methods performance. The F_2 measure is a harmonic average of precision and recall by giving a higher weight to recall than precision, or in other words, giving a higher weight to false negatives [20].

Table 1. Confusion Matrix

	True Positive	True Negative
Prediction as Positive	True Positives (TP)	False Positives (FP)
Prediction as Negative	False Negatives (FN)	True Negatives (TN)

$$\text{Recall(R)} = \text{TP}_{\text{rate}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

$$\text{Precision(P)} = \text{PP}_{\text{value}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{12}$$

$$\text{F}_2 = \frac{5 \times \text{Precision} \times \text{Recall}}{(4 \times \text{Precision}) + \text{Recall}} \tag{13}$$

TP = True Positive TN = True Negative
FP = False Positive FN = False Negative
PP = Positive Predictive

3. Data & Methodology

3.1. Data

This study uses socioeconomic datasets from Statistics Indonesia (BPS), the Ministry of Villages, Underdeveloped Regions Development and Transmigration (Kemendes PDTT), the Ministry of Investment (BKPM), and the Directorate General of the Fiscal Balance of the Ministry of Finance (DJPK). The number of instances in the datasets is 514 districts in Indonesia that are classified into two classes: the class of underdeveloped district, which counts 62 districts (12.06 percent) and the class of the developed district, which counts 452 districts (87.94 percent). These classifications are based on Indonesia's Presidential Regulation (Perpres) Number 63 2020. The number of predictor variables or features used in this research is 21. These variables are selected based on the previous research [25][26]. The list of variables used in this study can be seen in Table 2.

Table 2. List of variables used in the study

Types of variables	Name of variables	Types	Units	Data sources
Response/ Classes	1. Districts classification	Categorical	1 – Underdeveloped 0 – Developed	Indonesia's Presidential Regulation
Predictor variables/ Features	1. Gross Regional Domestic Product Per capita	Numerical	Thousand rupiah	BPS
	2. Gross Regional Domestic Product at a constant price basis 2010	Numerical	Million rupiah	BPS
	3. Regional Revenue	Numerical	Million rupiah	DJPK
	4. Regional Own-Source Revenue	Numerical	Million rupiah	DJPK
	5. Percentage of Poor People	Numerical	%	BPS
	6. Poverty Depth Index	Numerical	-	BPS
	7. Poverty Severity Index	Numerical	-	BPS
	8. Unemployment Rate	Numerical	%	BPS
	9. Labor Force Participation Rate	Numerical	%	BPS
	10. Poverty Lines	Numerical	Rupiah	BPS
	11. Human Development Index	Numerical	-	BPS
	12. Adjusted Per Capita Expenditure	Numerical	Thousand rupiah	BPS
	13. Life Expectancy at Birth	Numerical	Years	BPS
	14. Average Years of Schooling	Numerical	Years	BPS
	15. Expected Years of Schooling	Numerical	Years	BPS
	16. Gender Empowerment Index	Numerical	-	BPS
	17. Number of Poor People	Numerical	Thousands of peoples	BPS
	18. Number of Foreign Investment Projects	Numerical	Projects	BKPM
	19. Foreign Investment Value	Numerical	US$	BKPM
	20. Number of Domestic Investment Projects	Numerical	Project	BKPM
	21. Domestic Investment Investment Value	Numerical	Million rupiah	BKPM

Sources: Kemendes PDTT, BPS, DJPK, BKPM

3.2. Methodology

The methodology can be seen in Figure 1 with the explanation for each stage as follows:

1. In the beginning, we explored data to detect the problems in the dataset, such as missing data and imbalanced class problems.
2. Data analysis of the datasets is carried out in two ways. The first analysis uses all the instances, and the others use subset instances based on the region of their locations. The first region subset

includes the districts from Sumatra, Jawa, Bali, and Nusa Tenggara islands. The second region subset includes the Kalimantan, Sulawesi, Maluku, and Papua island districts.

3. We randomly partitioned the dataset into training and testing data with 70:30, 80:20, and 90:10 proportions in each class.
4. Bootstraping the training data as 500.
5. Evaluating the cost ratio used in the AdaC2 algorithm using evaluation measures recall, precision, and F_2. An evaluation of the cost ratio between false positives and false negatives is $C_{FP} : C_{FN} = [1{,}0: 0{,}1; 1{,}0: 0{,}2; 1{,}0: 0{,}3; 1{,}0: 0{,}4; 1{,}0: 0{,}5; 1{,}0: 0{,}6; 1{,}0: 0{,}7; 1{,}0: 0{,}8; 1{,}0: 0{,}9]$
6. Evaluating the performance of Single Decision Tree, Random Forest, AdaBoost, and AdaC2 using evaluation measures recall, precision, and F_2.
7. Choosing the best algorithm for classifying underdeveloped regions in Indonesia based on recall, precision, and F_2 evaluation measures.

The analysis of this study using software Microsoft Excel, R version 4.1. 3 and R Studio 2022.02.0 Build 443 by using package caret, readxl, rpart, randomForest, and IRIC [5].

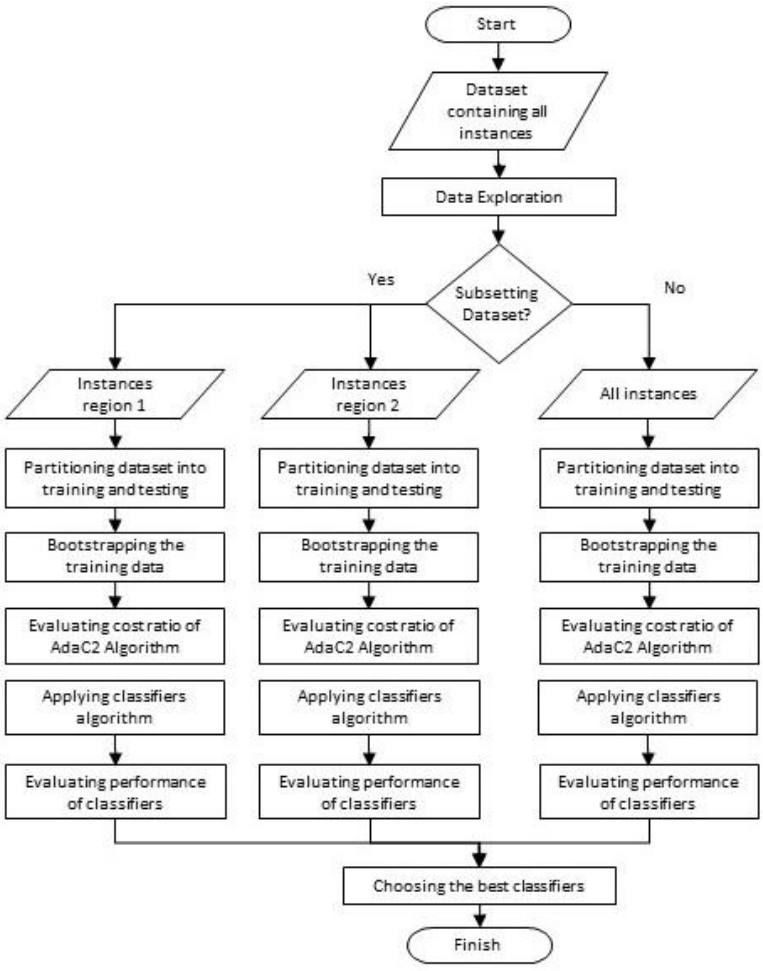

Figure 1. Methodology of the Study

4. Results and Discussion

4.1. Results

The complete dataset has moderately imbalanced classes, which are underdeveloped districts as a minority class, with a number of instances of 62 districts or 12.06 percent, and developed districts as a majority class, with a number of instances are 452 districts or 87.94 percent. However, after splitting the complete dataset into two regions, region, I have highly imbalanced classes, with 314 districts classified into 21 districts or 6.69 percent, as the underdeveloped class and 293 districts or 93.31 percent, as the developed region's class. Region II has low imbalanced classes, with 200 districts classified into 41 districts or 20.50 percent, as an underdeveloped class and 159 districts or 79.50 percent as a developed class. The number of instances and the proportion of each class can be seen in Table 3.

Table 3. Number of Instances and Proportion

Types of Analysis	Classes		Number of Instances (Districts)	Proportion (%)
A. Uses overall instances	Underdeveloped		62	12.06
	Developed		452	87.94
B. Uses instances by region	Region I	Underdeveloped	21	6.69
		Developed	293	93.31
	Region II	Underdeveloped	41	20.50
		Developed	159	79.50

4.1.1. Analysis using All Instances. Initially, the complete dataset was partitioned into training and testing data with defined proportions. Then the training data was bootstrapped, which produced 500 bootstrapped data. The cost performance evaluation is performed on each bootstrapped data to find the best cost to be used in the AdaC2 algorithm. The evaluation showed that the performance of the AdaC2 algorithm increases as the value of the cost ratio C_{FP}: C_{FN} decreases, but at a particular value of cost ratios, the performance becomes stagnant and starts to decrease. The recall average value was decreasing as the cost ratio decreased. However, the precision average value increased when the cost ratio C_{FP}: C_{FN} decreased. The best cost ratio was chosen based on the F_2 value, which is at a cost ratio 1:0.4. The evaluation result can be seen in Figure 2.

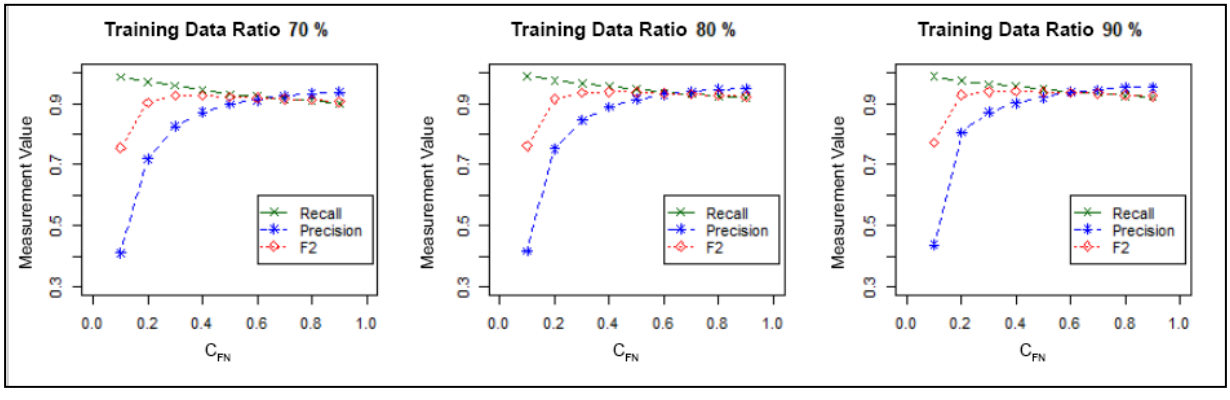

Figure 2. F_2, Recall, and Precision Average Value of Overall Data Analysis by Cost Ratio

In the next step, we used classifiers to classify instances and evaluate their performance. In the three scenario proportion of the training and testing data with moderate imbalance problem, the AdaC2 algorithm outperformed significantly compared with the other classifiers based on recall and F_2 measures. The random forest algorithm has a precision average value that is higher and more significant

in any proportion of training data used. Based on F_2, it can be concluded that the AdaC2 algorithm outperforms the other algorithms because it better classifies the instances in minority classes, as shown by the higher recall and F_2 average value in Table 4.

Table 4. Evaluation Measures Value of Classifiers on Complete Data Analysis

Training: Testing	Algorithm Name	Recall	Precision	F_2
70 : 30	SD Tree	0.822 ± 0.011	0.867 ± 0.008	0.826 ± 0.009
	RF	0.898 ± 0.007	**0.953 ± 0.005**	0.907 ± 0.006
	AdaBoost	0.896 ± 0.007	0.943 ± 0.005	0.903 ± 0.006
	AdaC2 1: 0.4	***0.945 ± 0.005**	0.872 ± 0.007	***0.928 ± 0.005**
80 : 20	SD Tree	0.840 ± 0.010	0.876 ± 0.009	0.843 ± 0.009
	RF	0.915 ± 0.008	**0.959 ± 0.005**	0.921 ± 0.006
	AdaBoost	0.913 ± 0.008	0.957 ± 0.005	0.919 ± 0.007
	AdaC2 1: 0.4	***0.957 ± 0.006**	0.890 ± 0.008	***0.940 ± 0.005**
90 : 10	SD Tree	0.842 ± 0.014	0.887 ± 0.011	0.843 ± 0.012
	RF	0.919 ± 0.010	**0.962 ± 0.007**	0.924 ± 0.009
	AdaBoost	0.919 ± 0.010	0.958 ± 0.007	0.924 ± 0.009
	AdaC2 1: 0.4	***0.957 ± 0.008**	0.902 ± 0.010	***0.941 ± 0.007**

Note: *Significant at 95% confidence level

4.1.2. Analysis using Instances by Region. The complete dataset was split into two datasets, called Region I and Region II. The Region I dataset has a higher imbalanced class proportion than the complete dataset. However, the Region II dataset has a lower imbalanced class proportion than the complete dataset. We first analyzed the Region I dataset. The dataset was partitioned into training and testing datasets. The training dataset was bootstrapped 500 times, which produced 500 bootstrapped region I training data. The cost performance evaluation of the AdaC2 was performed to find the best cost ratio. The evaluation showed that the performance of the AdaC2 algorithm increases as the value of the cost ratio C_{FP}: C_{FN} decreases, but at a particular value of cost ratios, the performance becomes stagnant and starts to decrease. The recall average value was decreasing as the cost ratio decreased. However, the precision average value increased when the cost ratio C_{FP}: C_{FN} decreased. The best cost ratio based on the F_2 evaluation measures is 1:0.3, as shown in Figure 3.

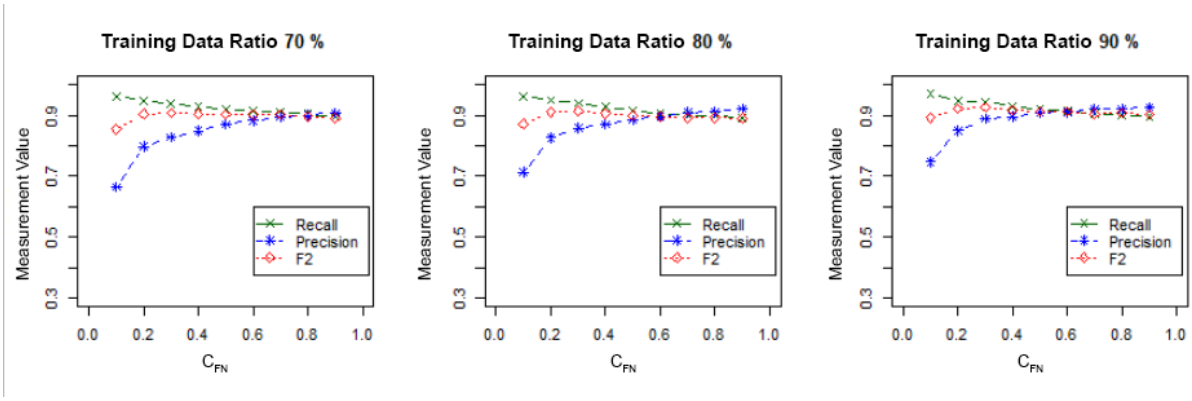

Figure 3. F_2, Recall, and Precision Average Value of Region I Data Analysis by Cost Ratio

In the next step, classifiers were implemented and evaluated based on the evaluation measures value. The AdaC2 algorithm outperformed the other classifiers based on recall and F_2 average values. However, the random forest has the best precision value than other classifiers. Based on the results of this evaluation, the AdaC2 algorithm can provide better performance than the other classifiers based on recall and F_2 average value that can be seen in Table 5.

Table 5. Evaluation Measures Value of Classifiers on Region I Data Analysis

Training: Testing	Cost Ratio	Recall	Precision	F_2
70 : 30	SD Tree	0.738 ± 0.018	0.845 ± 0.015	0.740 ± 0.015
	RF	0.862 ± 0.014	***0.961** ± **0.007**	0.874 ± 0.012
	AdaBoost	0.874 ± 0.013	0.930 ± 0.010	0.879 ± 0.011
	AdaC2 1: 0.3	***0.940** ± **0.010**	0.829 ± 0.013	***0.908** ± **0.009**
80 : 20	SD Tree	0.746 ± 0.021	0.855 ± 0.016	0.753 ± 0.017
	RF	0.895 ± 0.014	***0.953** ± **0.009**	0.899 ± 0.012
	AdaBoost	0.891 ± 0.015	0.935 ± 0.010	0.891 ± 0.013
	AdaC2 1: 0.3	***0.942** ± **0.011**	0.858 ± 0.014	**0.914** ± **0.010**
90 : 10	SD Tree	0.739 ± 0.029	0.864 ± 0.020	0.806 ± 0.019
	RF	0.900 ± 0.018	***0.966** ± **0.010**	0.908 ± 0.015
	AdaBoost	0.921 ± 0.016	***0.966** ± **0.010**	0.922 ± 0.014
	AdaC2 1: 0.3	**0.945** ± **0.015**	0.889 ± 0.016	**0.927** ± **0.012**

Note: *Significant at 95% confidence level

In the Region II dataset analysis, the dataset was partitioned into training and testing datasets. The training dataset was bootstrapped 500 times, producing 500 Region II datasets. The cost evaluation was performed to find the best cost ratio for AdaC2. The evaluation showed that the performance of the AdaC2 algorithm increases as the value of the cost ratio C_{FP}: C_{FN} decreases, but at a particular value of cost ratios, the performance becomes stagnant and starts to decrease. The recall average value was decreasing as the cost ratio decreased. However, the precision average value increased when the cost ratio C_{FP}: C_{FN} decreased. The best cost ratio for the Region II dataset based on the F_2 evaluation measures is 1:0.5, which can be seen in Figure 4.

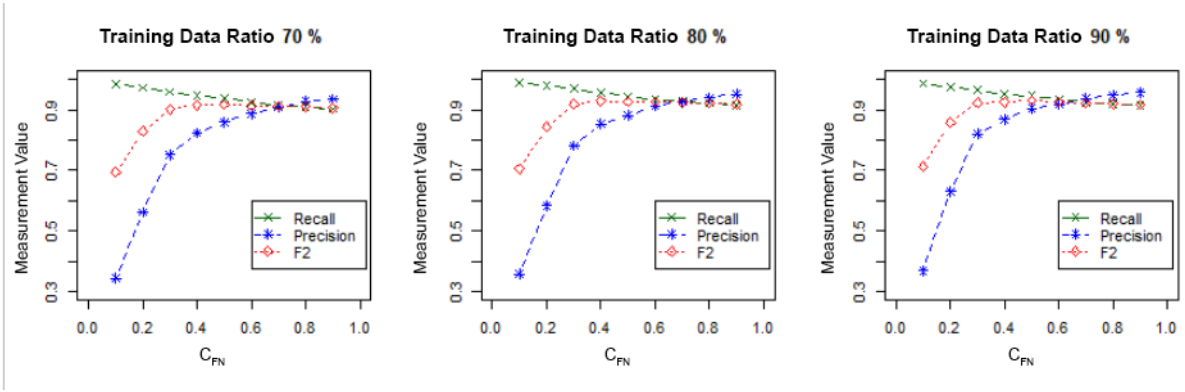

Figure 4. F_2, Recall, and Precision Average Value of Region II Data Analysis by Cost Ratio

In the following step analysis, we performed implementation and evaluation of the classifiers. The AdaC2 significantly outperformed the other classifiers in the three scenarios of training and testing data proportion based on recall and F_2 average value. However, the random forest has the best precision average value compared with the other classifiers. Based on the results of this evaluation, it can be concluded that the AdaC2 algorithm can perform better than the other classifiers, shown by a better average value on recall and F_2, which can be seen in Table 6.

Table 6. Evaluation Measures Value of Classifiers on Region II Data Analysis

Training: Testing	Cost Ratio	Recall	Precision	F_2
70 : 30	SD Tree	0.779 ± 0.012	0.898 ± 0.010	0.793 ± 0.010
	RF	0.890 ± 0.009	***0.967** ± **0.005**	0.902 ± 0.007
	AdaBoost	0.896 ± 0.008	0.950 ± 0.006	0.904 ± 0.007
	AdaC2 1: 0.5	***0.938** ± **0.007**	0.861 ± 0.009	***0.918** ± **0.006**
80 : 20	SD Tree	0.779 ± 0.015	0.902 ± 0.011	0.792 ± 0.013
	RF	0.895 ± 0.010	***0.970** ± **0.005**	0.906 ± 0.009
	AdaBoost	0.903 ± 0.010	0.960 ± 0.006	0.911 ± 0.008
	AdaC2 1: 0.4	***0.958** ± **0.007**	0.853 ± 0.011	***0.930** ± **0.006**
90 : 10	SD Tree	0.781 ± 0.019	0.910 ± 0.013	0.795 ± 0.016
	RF	0.900 ± 0.013	***0.974** ± **0.007**	0.908 ± 0.011
	AdaBoost	0.906 ± 0.013	0.964 ± 0.008	0.911 ± 0.011
	AdaC2 1: 0.5	***0.947** ± **0.010**	0.905 ± 0.012	***0.932** ± **0.009**

Note: *Significant at 95% confidence level

5. Conclusion and Future Works

In conclusion, our empirical study using socioeconomic data in Indonesia confirms that the cost-sensitive boosting algorithm (AdaC2) performs better in the imbalanced classes condition compared with the other standard classifiers such as Single Decision Tree, Random Forest, and AdaBoost. The cost performance evaluation of the AdaC2 algorithm needs to be tuned to get the best performance of the algorithm. The cost ratio evaluation shows that as the cost ratio increases, the recall average value increases while the precision average value decreases and vice versa. It needs to find the best balance between recall and precision in F_2 evaluation measures. As a suggestion for further development, the AdaC2 algorithm could be compared with the other imbalance data algorithm methods. The AdaC2 algorithm could be improved for classifying ordinal or multiclass response variables.

Acknowledgment.
The authors thank the reviewers for their comments and BPS-Statistics Indonesia for funding this research.

Appendices

Appendix 1 Cost Ratio Evaluation of AdaC2 Algorithm on Complete Dataset Analysis

Training : Testing	Cost Ratio	Recall	Precision	F2
70 : 30	1: 0.1	0.989 ± 0.002	0.411 ± 0.011	0.756 ± 0.007
	1: 0.2	0.973 ± 0.004	0.720 ± 0.010	0.903 ± 0.004
	1: 0.3	0.960 ± 0.005	0.825 ± 0.008	0.927 ± 0.004
	1: 0.4	**0.945 ± 0.005**	**0.872 ± 0.007**	**0.928 ± 0.005**
	1: 0.5	0.931 ± 0.006	0.899 ± 0.007	0.923 ± 0.005
	1: 0.6	0.923 ± 0.006	0.912 ± 0.006	0.919 ± 0.005
	1: 0.7	0.916 ± 0.007	0.925 ± 0.006	0.916 ± 0.005
	1: 0.8	0.910 ± 0.007	0.933 ± 0.005	0.913 ± 0.006
	1: 0.9	0.901 ± 0.007	0.938 ± 0.005	0.906 ± 0.006
80 : 20	1: 0.1	0.990 ± 0.003	0.417 ± 0.010	0.762 ± 0.007
	1: 0.2	0.978 ± 0.004	0.752 ± 0.011	0.917 ± 0.005
	1: 0.3	0.965 ± 0.005	0.847 ± 0.009	0.936 ± 0.005
	1: 0.4	**0.957 ± 0.006**	**0.890 ± 0.008**	**0.940 ± 0.005**
	1: 0.5	0.947 ± 0.006	0.914 ± 0.007	0.938 ± 0.005
	1: 0.6	0.937 ± 0.007	0.931 ± 0.007	0.933 ± 0.006

Training : Testing	Cost Ratio	Recall	Precision	F2
90 : 10	1: 0.7	0.932 ± 0.007	0.939 ± 0.006	0.931 ± 0.006
	1: 0.8	0.927 ± 0.007	0.947 ± 0.006	0.928 ± 0.006
	1: 0.9	0.920 ± 0.008	0.950 ± 0.006	0.923 ± 0.006
	1: 0.1	0.989 ± 0.004	0.438 ± 0.012	0.773 ± 0.008
	1: 0.2	0.975 ± 0.006	0.806 ± 0.013	0.929 ± 0.006
	1: 0.3	0.965 ± 0.007	0.872 ± 0.011	0.940 ± 0.006
	1: 0.4	**0.957 ± 0.008**	**0.902 ± 0.010**	**0.941 ± 0.007**
	1: 0.5	0.947 ± 0.009	0.920 ± 0.009	0.938 ± 0.007
	1: 0.6	0.940 ± 0.009	0.938 ± 0.008	0.936 ± 0.007
	1: 0.7	0.936 ± 0.009	0.946 ± 0.008	0.935 ± 0.008
	1: 0.8	0.926 ± 0.010	0.954 ± 0.007	0.928 ± 0.009
	1: 0.9	0.923 ± 0.010	0.954 ± 0.007	0.925 ± 0.009

Appendix 2 Cost Ratio Evaluation of AdaC2 Algorithm on Region I Dataset Analysis

Training : Testing	Cost Ratio	Recall	Precision	F2
70 : 30	1: 0.1	0.962 ± 0.008	0.665 ± 0.020	0.856 ± 0.009
	1: 0.2	0.950 ± 0.009	0.797 ± 0.014	0.905 ± 0.008
	1: 0.3	**0.940 ± 0.010**	**0.829 ± 0.013**	**0.908 ± 0.009**
	1: 0.4	0.930 ± 0.010	0.849 ± 0.012	0.906 ± 0.009
	1: 0.5	0.920 ± 0.011	0.871 ± 0.012	0.903 ± 0.010
	1: 0.6	0.915 ± 0.011	0.883 ± 0.011	0.902 ± 0.010
	1: 0.7	0.913 ± 0.011	0.894 ± 0.011	0.903 ± 0.010
	1: 0.8	0.904 ± 0.012	0.901 ± 0.011	0.897 ± 0.011
	1: 0.9	0.894 ± 0.012	0.909 ± 0.010	0.891 ± 0.011
80 : 20	1: 0.1	0.963 ± 0.008	0.714 ± 0.021	0.873 ± 0.010
	1: 0.2	0.951 ± 0.010	0.827 ± 0.016	0.911 ± 0.009
	1: 0.3	**0.942 ± 0.011**	**0.858 ± 0.014**	**0.914 ± 0.010**
	1: 0.4	0.929 ± 0.012	0.871 ± 0.013	0.907 ± 0.010
	1: 0.5	0.917 ± 0.013	0.885 ± 0.013	0.901 ± 0.011
	1: 0.6	0.907 ± 0.013	0.896 ± 0.013	0.896 ± 0.012
	1: 0.7	0.901 ± 0.014	0.910 ± 0.012	0.894 ± 0.012
	1: 0.8	0.899 ± 0.014	0.912 ± 0.012	0.893 ± 0.012
	1: 0.9	0.893 ± 0.014	0.922 ± 0.011	0.891 ± 0.012
90 : 10	1: 0.1	0.970 ± 0.011	0.748 ± 0.022	0.892 ± 0.011
	1: 0.2	0.948 ± 0.015	0.851 ± 0.018	0.922 ± 0.011
	1: 0.3	**0.945 ± 0.015**	**0.889 ± 0.016**	**0.927 ± 0.012**
	1: 0.4	0.930 ± 0.016	0.895 ± 0.016	0.918 ± 0.013
	1: 0.5	0.921 ± 0.017	0.908 ± 0.015	0.913 ± 0.014
	1: 0.6	0.915 ± 0.018	0.912 ± 0.015	0.913 ± 0.014
	1: 0.7	0.905 ± 0.019	0.920 ± 0.014	0.905 ± 0.015
	1: 0.8	0.903 ± 0.019	0.921 ± 0.015	0.908 ± 0.015
	1: 0.9	0.897 ± 0.019	0.927 ± 0.014	0.904 ± 0.015

Appendix 3 Cost Ratio Evaluation of AdaC2 Algorithm on Region II Dataset Analysis

Training : Testing	Cost Ratio	Recall	Precision	F2
70 : 30	1: 0.1	0.987 0.004	0.343 0.012	0.694 0.008
	1: 0.2	0.975 0.005	0.562 0.015	0.831 0.007
	1: 0.3	0.960 0.006	0.752 0.012	0.902 0.005
	1: 0.4	0.949 0.006	0.824 0.009	0.917 0.005
	1: 0.5	**0.938 0.007**	**0.861 0.009**	**0.918 0.006**
	1: 0.6	0.926 0.007	0.889 0.009	0.915 0.006
	1: 0.7	0.917 0.008	0.908 0.008	0.913 0.006
	1: 0.8	0.910 0.008	0.928 0.007	0.911 0.007
	1: 0.9	0.903 0.008	0.936 0.007	0.907 0.007
80 : 20	1: 0.1	0.991 0.004	0.357 0.013	0.705 0.008

Training : Testing	Cost Ratio	Recall		Precision		F2	
	1: 0.2	0.980	0.005	0.584	0.016	0.844	0.007
	1: 0.3	0.971	0.005	0.781	0.012	0.919	0.005
	1: 0.4	**0.958**	**0.007**	**0.853**	**0.011**	**0.930**	**0.006**
	1: 0.5	0.945	0.007	0.882	0.010	0.927	0.006
	1: 0.6	0.936	0.008	0.913	0.009	0.927	0.007
	1: 0.7	0.929	0.008	0.928	0.008	0.925	0.007
	1: 0.8	0.923	0.009	0.940	0.007	0.923	0.007
	1: 0.9	0.915	0.009	0.952	0.007	0.919	0.008
	1: 0.1	0.988	0.005	0.368	0.013	0.713	0.009
	1: 0.2	0.977	0.007	0.631	0.018	0.859	0.008
	1: 0.3	0.967	0.008	0.821	0.015	0.924	0.007
	1: 0.4	0.952	0.010	0.869	0.013	0.926	0.009
90 : 10	**1: 0.5**	**0.947**	**0.010**	**0.905**	**0.012**	**0.932**	**0.009**
	1: 0.6	0.937	0.011	0.918	0.011	0.927	0.010
	1: 0.7	0.927	0.012	0.939	0.010	0.923	0.010
	1: 0.8	0.921	0.012	0.949	0.009	0.920	0.011
	1: 0.9	0.916	0.013	0.958	0.009	0.918	0.011

References

[1] Gan, D., Shen, J., An, B., Xu, M., Liu, N., Integrating TANBN with cost-sensitive classification algorithm for imbalanced data in medical diagnosis, Computers & Industrial Engineering (2019), doi: https://doi.org/10.1016/j.cie.2019.106266

[2] Xinmin Tao, Qing Li, Wenjie Guo, Chao Ren, Chenxi Li, Rui Liu, Junrong Zou, Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification, Information Sciences (2019), doi: https://doi.org/10.1016/j.ins.2019.02.062

[3] Lee, Wonji & Jun, Chi-Hyuck & Lee, Jong-Seok. (2016). Instance categorization by support vector machines to adjust weights in AdaBoost for imbalanced data classification. Information Sciences. 381. 10.1016/j.ins.2016.11.014.

[4] A. de Haro-García, G. Cerruela-García, N. García-Pedrajas, Ensembles of Feature Selectors for dealing with Class-Imbalanced Datasets: A proposal and comparative study, Information Sciences (2020), doi: https://doi.org/10.1016/j.ins.2020.05.077

[5] B. Zhu, Z. Gao, Z. Junkai, K. L. M. V. Seppe, IRIC: An R library for binary imbalanced classification, Softwares, 10.100341, 2019, https://doi.org/10.1016/j.softx.2019.100341

[6] Bing Zhu, Bart Baesens, Seppe K.L.M. vanden Broucke, An empirical comparison of techniques for the class imbalance problem in churn prediction, Information Sciences (2017), doi: 10.1016/j.ins.2017.04.015

[7] Jun-Feng GE, Yu-Pin LUO, A Comprehensive Study for Asymmetric AdaBoost and Its Application in Object Detection, Acta Automatica Sinica, Volume 35, Issue 11, 2009, Pages 1403-1409, ISSN 1874-1029, https://doi.org/10.1016/S1874-1029(08)60115-9.

[8] López, Victoria & Fernández, Alberto & García, Salvador & Palade, Vasile & Herrera, Francisco. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information Sciences. 250. 113–141. 10.1016/j.ins.2013.07.007.

[9] Antonelli, Michela & Ducange, Pietro & Marcelloni, Francesco. (2014). An experimental study on evolutionary fuzzy classifiers designed for managing imbalanced datasets. Neurocomputing. 146. 125–136. 10.1016/j.neucom.2014.04.070.

[10] Y. Sun, M. S. Kamel, A. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, The Journal of the Pattern Recognition Society, 40, 3358–3378, 2007, DOI:https://doi.org/10.1016/j.patcog.2007.04.009

[11] Bartosz Krawczyk, Mikel Galar, Lukasz Jelen, 'Francisco Herrera, Evolutionary Undersampling Boosting for Imbalanced Classification of Breast Cancer Malignancy, Applied Soft Computing Journal (2015), http://dx.doi.org/10.1016/j.asoc.2015.08.060

[12] E. Alfaro, M. Gamez, and N. Garcia, Ensemble Classification Methods with Applications in R, John Wiley & Sons Ltd, New York, 2019.
[13] C. Elkan, The Foundations of Cost-Sensitive Learning, Proceedings of the Seventeenth International Conference on Artificial Intelligence, Seattle, 4-10 August 2001.
[14] A. Fernández, S. García, M. Galar, R.C. Prati, B. Krawczyk, F. Herrera, Learning from Imbalanced Data Sets, Springer Nature, Switzerland, 2018.
[15] Y. Freund, R. E. Schapire, A Decision-Theoretic Generalization of Online Learning and an Application to Boosting, Journal of Computer and System Sciences, 55 (1) 119-139, 1999, https://doi.org/10.1006/jcss.1997.1504.
[16] T. Hastie, R. Tibshirani, J. Friedman. The Elements of Statistical Learning Second Edition, Springer Science Business Media, New York, 2017.
[17] G. James, D. Witten, T. Hastie, R. Tibshirani, An Introduction to Statistical Learning with Applications in R Second Edition, Springer Science+Business Media, New York, 2021.
[18] M. Kubat, R. Holte, S. Matwin, Machine learning for the detection of oil spills in satellite radar images, Mach. Learn. 30, 1998, pp. 195–215.
[19] C. Ling and V. Sheng, Victor, Cost-Sensitive Learning and the Class Imbalance Problem, Encyclopedia of Machine Learning, 2010.
[20] A. Maratea, A. Petrosino, M. Manzo, Adjusted F-measure and kernel scaling for imbalanced data learning, Information Sciences, 257, 331–341, 2014, 10.1016/j.ins.2013.04.016.
[21] N. Nikolaou, N. Edakunni, M. Kull, et al., Cost-sensitive boosting algorithms: Do we really need them? Mach Learn 104, 359–384, 2016, https://doi.org/10.1007/s10994-016-5572-x
[22] R. E. Schapire, Y. Freund, Boosting Foundations and Algorithms, The MIT Press, London, 2012.
[23] Y. Sun, A. Wong, Y. Wang, Inference Parameters of Cost-Sensitive Boosting Algorithms, 3587, 21-30, 2005, DOI:10.1007/11510888_3
[24] Wang, Ruihu. (2012). AdaBoost for Feature Selection, Classification and Its Relation with SVM, A Review. Physics Procedia. 25. 800-807. 10.1016/j.phpro.2012.03.160.
[25] Yang, Q., & Wu, X. (2006). 10 Challenging Problems in Data Mining Research. Int. J. Inf. Technol. Decis. Mak., 5, 597-604.
[26] Lailiyah, A.N., Priyarsono, D.S., Hutagaol, M. P.. (2022). Dinamika dan Faktor-Faktor yang Memengaruhi Kemiskinan di Kawasan Barat Indonesia (KBI) dan Kawasan Timur Indonesia (KTI). IPB University.
[27] Firdaus, M.. (2013). Ketimpangan Pembangunan Antar Wilayah Di Indonesia: Fakta Dan Strategi Inisiatif. IPB University.
[28] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. (2013). An introduction to statistical learning: with applications in R. New York: Springer.
[29] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324