Hepatitis Patient Classification using Random Forest Algorithms with Cost-Sensitive Method

Arifin Nugroho, Ricky Risnantoyo, Saifurrachman Chohan, Nuraeni Herlinawati, Sfenrianto

Abstract: Hepatitis is a common worldwide public health problem that attacks almost every population in various countries. Machine learning has been widely used to classify various diseases, including hepatitis. In this research, the Random Forest algorithm will be used along with the dataset of patients with hepatitis to classify whether the patient's condition will live or die. Missing value and imbalance class exists in this dataset. In that class, the sample of healthy and sick patients that often occurs in the disease dataset. We replace missing values using mean and median and to deal with this imbalance of class, we use cost-sensitive methods to put penalty in classification. A manual selection feature process is also carried out to look for features that can be removed while still maintaining the quality of accuracy and classification. The validation method used is 10-fold Cross-Validation and using Random Forest Algorithm with tuned parameter to find the best result in classifying the class. This research prioritizes classification results by considering the small amount of data and the imbalance of the class, so it can classify the class more successfully and accurate for hepatitis patients. The accuracy value obtained is 85.80%.

Keywords: machine learning, random forest, hepatitis, imbalance

I. INTRODUCTION

Along with the science and information technology development, machine learning arises as a new branch of science in the field of computers, which provides a new concept in the process of finding information in the health sector. Especially information prediction whether someone with a specific health condition has a chance to live or not.

Hepatitis is an inflammatory disease of the liver due to a viral infection that attacks and causes damage to the liver cells and functions. Hepatitis is one of the leading causes of liver cancer. Hepatitis can damage liver functions to neutralize poison and as one of the digestive systems of food in the body, which is very important for humans. Many studies conducted on hepatitis using the implementation of the machine learning algorithm.

One of them, in 2011, Bekir Karklik used backpropagation and naïve Bayes algorithm to obtain prediction information for hepatitis patients with an accuracy of 98% and 86% [4]. Following is a research table using data mining algorithms:

Table 1: Research on Hepatitis with Data mining Algorithms [3], [4], [5], [6]

Author	Year	Algorithm	Result (Accuracy)
Lale Ozyilmaz - Tulay Yildirim [3]	2003	• Multi-Layer Perceptron (MLP) • Radial Basis Function (RBF) • Conic Section Function Neural Network (CSFNN)	• MLP: 81,375% • RBF: 85% • CSFNN: 90%
Bekir Karlik [4]	2011	• Back Propagation • Naïve Bayes	• Naïve Bayes: 86% • Backpropagation: 98%
Varun Kumar Vijay Sharathi Gayatri Devi [5]	2012	• Support Vector Machine (SVM) • Support Vector Machine (SVM) with feature selection	• SVM: 79,33% • SVM with features selection: 83,12%
Kartikeyan-Thangaraju [6]	2013	• Random Forest	• Random Forest: 83%

From table 1, it can be seen that the accuracy of the random forest algorithm in previous studies is 83% [6]. In this study, it will be predicted whether a hepatitis patient has the possibility of survival in terms of age, sex, steroids, liver conditions, and other blood content conditions using the Random Forest algorithm with a cost-sensitive method. The use of cost-sensitive methods is expected to provide better classification results and accuracy.

II. THEORETICAL REVIEW

Random Forest is an ensemble method. The ensemble method is a way to improve the accuracy of classification methods by combining several classification methods. Random Forest begins with the essential data mining technique, the decision tree. In the decision tree, the input is entered at the top (root) then down to the bottom (leaf) to determine which data belongs to which class. Random forest is a classifier consisting of a collection of structured tree classifiers where each tree sends a vote for the most popular class in input x [7].

In other words, a random forest consists of a group of decision trees, where that group is used to classify data into a specific class.

Revised Manuscript Received on February 22, 2020.

Arifin Nugroho, Department of Computer Science - Postgraduate Programs STMIK Nusa Mandiri, Jakarta, Indonesia. Email: 14002306@nusamandiri.ac.id

Ricky Risnantoyo, Department of Computer Science - Postgraduate Programs STMIK Nusa Mandiri, Jakarta, Indonesia. Email: 14002301@nusamandiri.ac.id

Saifurrachman Chohan, Department of Computer Science - Postgraduate Programs STMIK Nusa Mandiri, Jakarta, Indonesia. Email: 14002305@nusamandiri.ac.id

* Correspondence Author

Nuraeni Herlinawati, Department of Computer Science - Postgraduate Programs STMIK Nusa Mandiri, Jakarta, Indonesia. Email: nuraeni.nhw@nusamandiri.ac.id

Sfenrianto, Department of Information Systems Management, BINUS Graduate Program – Master of Information Systems Management, Bina Nusantara University, Jakarta. Indonesia. Email: sfenrianto@binus.edu

Random forest is a supervised classification method. This method creates a forest with several trees. In general, the more trees in a forest, the stronger the forest can be seen. In the same case, the more trees, the higher the accuracy obtained. Decision Tree will use Information Gain and Gini Index to perform calculations in determining the root node and rule.

Random Forest also uses Information Gain and Gini Index to perform calculations in building trees, only Random Forest will build more than one tree. Each tree is constructed using a dataset with attributes taken randomly from the training data. In other words, each tree will depend on the value of an independent vector sample with the same distribution in each tree. During the classification process, each tree will vote for the most popular class [7].

In previous studies, the use of the Random Forest algorithm resulted in an accuracy value of 83% [6]. In the hepatitis patient dataset used in this study, there is an imbalance in the comparison between the two classes or labels. This happens because, in the data samples, we can find more healthy patients than sick patients, or in this case, there are more "Live" patients than "Die" patients.

Table 2: Class and the Amount

No.	Label/Class	Amount
1	live	123
2	die	32

Under these conditions, we try to do a balancing method so that the model can be better to be used in classifying classes. The method used is the Cost-Sensitive Learning method [2]. Based on the existing literature, this method seeks to reduce misclassification so that the classification model becomes better. This method is also used because the number of sample datasets is minimal.

The Cost-Sensitive Learning method can reduce the classification imbalance in the dataset by giving a penalty for misclassification. In this study, the penalty value given is in the range of 1-10 for class b(Die) so that the classifier is more attentive and careful in classifying class b(Die). The value of the balancing results is obtained after using ten experimental result tests.

III. PROPOSED MODEL

The dataset used in this study is hepatitis patient data, which consists of 155 patient data (instances) categorized in 20 attributes (features), including labels. This data is taken from UCI Machine Learning. The dataset used has 20 attributes (including labels), namely *Class, Age, Sex, Steroids, Antivirals, Fatigue, Malaise, Anorexia, Liver Big, Liver Firm, Spleen Palpable, Spiders, Ascites, Voices, Bilirubin, Alk Phosphate, Sgot, Albumin, Protime, and Histology*. Two classes in this dataset determine its value using *LIVE* and *DIE* parameters, which classify survival and die patients with hepatitis based on the patient's condition.

At the beginning of the experiment, preprocessing data will be carried out by replacing data with missing value parameters using the mean/median value, followed by handling the imbalance class using cost-sensitive methods. The manual feature selection process is also carried out to look for features that can be removed while maintaining the quality of accuracy and classification.

After the balancing and feature selection process, in this section, we conduct the classification of hepatitis patient data using the Random Forest Classification method with 10-fold Cross-Validation (split data).

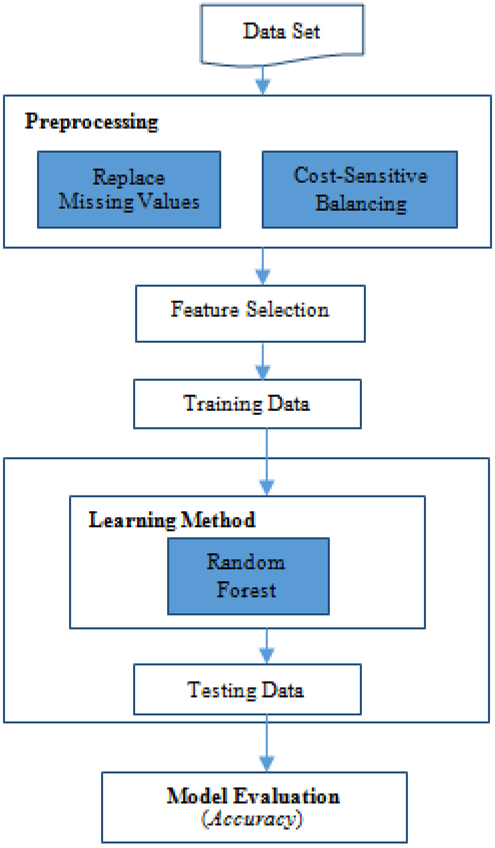


Fig 1. Proposed Model

IV. EXPERIMENT RESULTS AND ANALYSIS

A. Replace Missing Values

Predictive models require us to look at data before we start creating models. Table 3 show the missing values in the data set.

Table 3: Hepatitis Dataset Missing Values

NO	ATTRIBUTES	MISSING VALUE
1	Class	0
2	Age	0
3	Sex	0
4	Steroid	1
5	Antivirals	0
6	Fatigue	1
7	Malaise	1
8	Anorexia	1
9	Liver Big	10
10	Liver Firm	11
11	Spleen Palpable	5
12	Spiders	5
13	Ascites	5
14	Varices	5
15	Bilirubin	6
16	Alk Phosphate	29
17	Sgot	4
18	Albumin	16
19	Protime	67
20	Histology	0

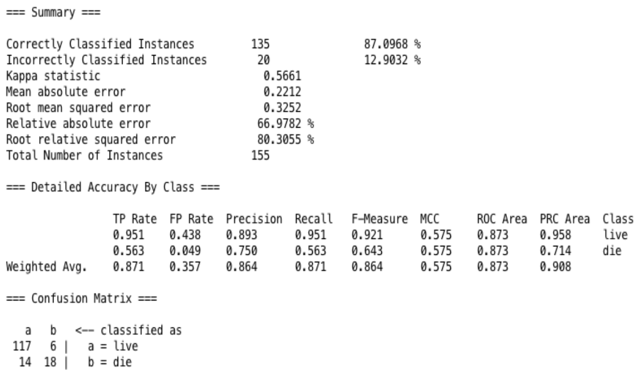
In machine learning, looking at data means exploring, cleaning, and visualizing data through graphics and plots. This is known as the Exploratory Data Analysis. In this research, pre-processing data is carried out by replacing detected missing values.

By using the Replace Missing Value method with the mean and median, missing data can be replaced by the mean and median values for each existing feature. This method is rather simple and gives a relatively small error rate, especially with a small dataset based on comparison [1].

B. Class Balancing

In this hepatitis patient dataset, the comparison between the two classes or labels is an imbalance. This happens because there are more healthy patient data samples than sick patient data samples, or in this case, more "Live" patients than "Die" patients.

Before doing the balancing process, we first try to see how the results are obtained using a dataset without the balancing process. Through this stage, we can get an understanding of the effect of this imbalance on the classification results. Following are the results of the classification using the Random Forest 10-fold Cross Validation method without balancing and feature selection:



```
=== Summary ===

Correctly Classified Instances         135               87.0968 %
Incorrectly Classified Instances        20               12.9032 %
Kappa statistic                          0.5661
Mean absolute error                      0.2212
Root mean squared error                  0.3252
Relative absolute error                 66.9782 %
Root relative squared error             80.3055 %
Total Number of Instances              155

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.951    0.438    0.893      0.951    0.921      0.575    0.873     0.958     live
                 0.563    0.049    0.750      0.563    0.643      0.575    0.873     0.714     die
Weighted Avg.    0.871    0.357    0.864      0.871    0.864      0.575    0.873     0.908

=== Confusion Matrix ===

   a   b   <-- classified as
 117   6 |   a = live
  14  18 |   b = die
```

Fig 2. Classification Results Without Balancing

Based on Figure 2 the results obtained an accuracy of 87%. This value can be considered quite reasonable when compared to the results of previous literature, which is 83%. However, in the Confusion Matrix section, it can be seen that there are still many improper classifications of Live and Die patients. Especially in the classification of Die patients. Misclassification is almost 50% of the total data. Given the Die patient sample is less than the Live patient sample.

Table 4: Default Penalty Classification

Sample Default	Cost Matrix Penalty		Confusion Matrix		Acc
	Live	Die	Live	Die	
	0	1	117	6	87,0968%
	1	0	14	18	

Knowing the result using unbalanced data, we try to do a balancing method so that the model can be better to be used in classifying classes. The method used is the Cost-Sensitive Learning method. Based on the existing literature [2], this method seeks to reduce misclassification so that the classification model becomes better. This method is also used because the number of sample datasets is minimal.

The Cost-Sensitive Learning method can reduce the classification imbalance in the dataset by giving a penalty for misclassification. In table 4 we use default classification penalties.

Knowing the initial result, next, in this experiment, we will give a penalty. The penalty value given is in the range of 1-10 for class b(Die) so that the classifier is more attentive and careful in classifying class b(Die). The value of the balancing results is obtained after using ten experimental result tests. Following are the results of balancing experiments in providing penalties:

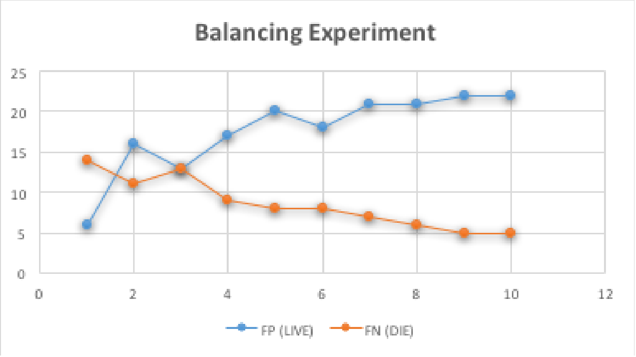


Fig 3. Balancing Experiment

Based on Figure 3, the data which is considered to be appropriate is by giving a penalty of 6 when a misclassification in class b (Die) occurred. These results are obtained by taking into account False Positive (FP) and False Negative (FN) values. Also, we consider the accuracy and pay attention to patterns. The greater than 6, the misclassification in Live classes will be even more significant.

Table 5: Penalty Classification Sample

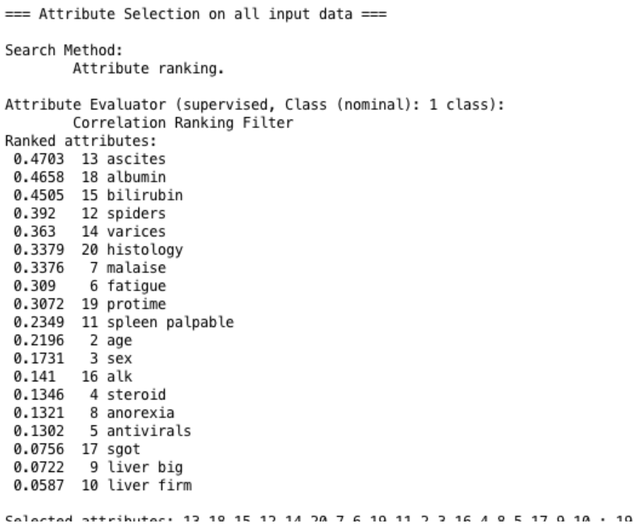
Sample e	Cost Matrix Penalty		Confussion Matrix		Acc
	Live	Die	Live	Die	
	0	1	105	18	83,2258%
	6	0	8	24	

C. Features Selection

The manual feature selection method is conducted by eliminating features one by one while taking into account the results of the classification. As a guide, we also use the Correlation Attribute Evaluation method, as well as for comparisons. Figure 4 are the results of the Correlation Attribute Evaluation.

Figure 4 results provide information about the Rank of the Feature, which contributes to the classification results from Range 0-1. The closer the value to 1, the better. While a value closer to 0 means that the specific feature does not contribute much.

Before conducting the experiment, we conducted a classification using all features as a comparison with the results of the feature selection to be obtained. Following is the classification results using full features with 10-fold Cross-Validation and class balancing. Table 6 is the result of full feature classification.



```
=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 1 class):
        Correlation Ranking Filter
Ranked attributes:
 0.4703  13 ascites
 0.4658  18 albumin
 0.4505  15 bilirubin
 0.392   12 spiders
 0.363   14 varices
 0.3379  20 histology
 0.3376   7 malaise
 0.309    6 fatigue
 0.3072  19 protime
 0.2349  11 spleen palpable
 0.2196   2 age
 0.1731   3 sex
 0.141   16 alk
 0.1346   4 steroid
 0.1321   8 anorexia
 0.1302   5 antivirals
 0.0756  17 sgot
 0.0722   9 liver big
 0.0587  10 liver firm

Selected attributes: 13,18,15,12,14,20,7,6,19,11,2,3,16,4,8,5,17,9,10 : 19
```

Fig 4. Correlation Attribute Evaluation

Figure 4 show the rank of the attribute. The highest is *acisities* and the lowest is *liver firm*. Table 6 shows the results classification using all feature.

Table 6: Full Features Classification Results

Features	Live	Die	Acc
Full Features	105	18	83,2258
	8	24	

After table 6 result, we try to eliminate each feature (19 features) one by one. Figure 5 shows the results.

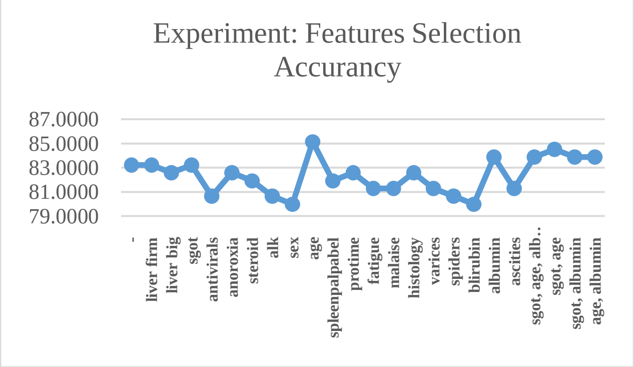


Fig 5. Feature Selection Against Accuracy Percentage

From figure 5, three features can be removed, even two of them provide improvements to the classification results.

Table 7: Removable Features

Without	Live	Die	Acc
sgot	105	18	83,2258
	8	24	
age	106	17	85,1613
	6	26	
albumin	105	18	83,8710
	7	25	

From table 7, the Sgot feature can be removed and does not reduce the classification results and their accuracy. While Age and Albumin can be removed and also provide improvements. Next, we try to combine those three to see the results. Table 8 shows the result:

Table 8: Combination of Removable Features

Without	Live	Die	Acc
sgot, age, albumin	106	17	83,871
	8	24	
sgot, age	106	17	84,5161
	7	25	
sgot, albumin	106	17	83,871
	8	24	
age, albumin	104	19	83,871
	7	25	

From the experimental results in Table 8, it was found that the age feature can be removed/not used and also can improve the classification results and accuracy than using all the features. While removing the albumin and sgot features give some improvement although not as good as by just removing the age feature. When we remove features using a combination of the three features, the results are not better than when the experiment conducted by just removing the age feature. Taking this result into consideration, the age feature will not be used in the classification process.

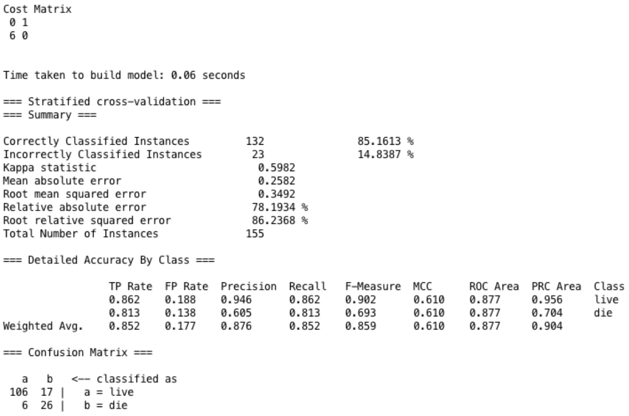
D. Random Forest Classification

After performing the balancing and feature selection process, in this section, we conducted a classification of hepatitis patients data using the Random Forest classification method with 10-fold Cross-Validation (split data), balancing using FN(Die) penalty with a weight value of 6 and without using age feature. There are three highly influential parameters in the classification process using Random Forest in Weka, namely:

Table 9: Random Forest Parameter

Nama Parameter	Alias	Fungsi	Nilai Default
maxDepth	-depth	Seting batas kedalaman Tree (branch)	0(unlimited)
numFeature	-K	Setting jumlah feature yang akan dirandom	0
numIteration	-I	Setting banyaknya Tree	100

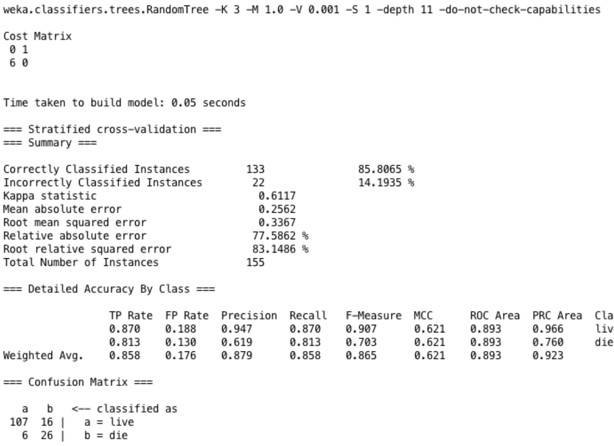
Using the above parameters, we tested one by one to see the results of the classification of tunning parameters with a total number of tests of 42 times. Experiments begin by setting individual parameters to determine the effect and the results. However, before tunning the parameters, we tried to classify with the default parameters to see the results as a comparison. Following are the results with parameters at default values:



```
Cost Matrix
 0 1
 6 0


Time taken to build model: 0.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         132               85.1613 %
Incorrectly Classified Instances        23               14.8387 %
Kappa statistic                          0.5982
Mean absolute error                      0.2582
Root mean squared error                  0.3492
Relative absolute error                 78.1934 %
Root relative squared error             86.2368 %
Total Number of Instances              155

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.862    0.188    0.946      0.862    0.902      0.610    0.877     0.956     live
                 0.813    0.138    0.605      0.813    0.693      0.610    0.877     0.704     die
Weighted Avg.    0.852    0.177    0.876      0.852    0.859      0.610    0.877     0.904

=== Confusion Matrix ===

   a   b   <-- classified as
 106  17 |   a = live
   6  26 |   b = die
```

Fig 6. Random Forest Results with Default Parameters

After obtaining the classification results with the default parameters, we try to combine the Random Forest parameters. The result, compared to using the default parameter, the accuracy value is better when the *maxDeph*> 10 (in the case of this experiment is 11), *numFeature* = 3, and *numeration* = 130.



```
weka.classifiers.trees.RandomTree -K 3 -M 1.0 -V 0.001 -S 1 -depth 11 -do-not-check-capabilities

Cost Matrix
 0 1
 6 0


Time taken to build model: 0.05 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         133               85.8065 %
Incorrectly Classified Instances        22               14.1935 %
Kappa statistic                          0.6117
Mean absolute error                      0.2562
Root mean squared error                  0.3367
Relative absolute error                 77.5862 %
Root relative squared error             83.1486 %
Total Number of Instances              155

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.870    0.188    0.947      0.870    0.907      0.621    0.893     0.966     live
                 0.813    0.130    0.619      0.813    0.703      0.621    0.893     0.760     die
Weighted Avg.    0.858    0.176    0.879      0.858    0.865      0.621    0.893     0.923

=== Confusion Matrix ===

   a   b   <-- classified as
 107  16 |   a = live
   6  26 |   b = die
```

Fig 7. Experimental Final Result

Figure 7 the results of experiments conducted after filling in missing values using mean/median values, balancing with a six weighted penalty for FN(Die), and feature selection by eliminating age feature and tuning in Random Forest algorithm parameters.

Table 10: Final Classification Result

Method	Live	Die	Acc
Cost-Sensitive Random forest Classification	107	16	85,8065
	6	26	

Table 10 show the classification and accuracy value obtained. Accuracy is 85.80% better than the previous studies using the same machine learning dataset and algorithm [6], and the results of the classification of Live and Die patients are also better than without tuning parameters and balancing.

V. CONCLUSION

After going through several experimental processes, the imbalance classification can be reduced by the cost-sensitive method by giving the right penalty value for the imbalance class. The Feature selection by selecting feature used or not and tuning parameter process in the Random Forest method by adjusting the parameter can improve the classification results and their accuracy. The research can be continued using other balancing methods that can provide better class balancing results by applying other machine learning algorithms.

REFERENCES

1. Grzymala-Busse, J.W., & Hu, M. (2000). A Comparison of Several Approaches to Missing Attribute Values in Data Mining. Rough Sets and Current Trends in Computing. DOI:10.1007/3-540-45554-X_46
2. N. Thai-Nghe, Z. Gantner and L. Schmidt-Thieme, Cost-sensitive learning methods for imbalanced data, The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, 2010, pp. 1-8. doi: 10.1109/IJCNN.2010.5596486
3. Ozyilmaz, L., & Yildirim, T. (2003). Artificial neural networks for diagnosis of hepatitis disease. Proceedings of the International Joint Conference on Neural Networks, 2003., 1, 586-589 vol.1. DOI:10.1109/IJCNN.2003.1223422
4. Bekir Karlik. (2011). Hepatitis Disease Diagnosis Using Backpropagation and the Naive Bayes Classifiers. Journal of Science and Technology vol. 1 no. 1
5. Varun Kumar, Vijay Sharathi, Gayatri Devi. (2012). Hepatitis Prediction Model based on Data Mining Algorithm and Optimal Feature Selection to Improve Predictive Accuracy. International Journal of Computer Applications (0975 – 8887) Volume 51– No.19, August 2012
6. Karthikeyan, T., & Thangaraju, P. (2013). Analysis of Classification Algorithms Applied to Hepatitis Patients. DOI:10.5120/10157-5032
7. Han, Jiawei. (2012). Data Mining Concepts and Techniques Third Edition. USA:Elsevier

AUTHORS PROFILE

Arifin Nugroho is a Student of Master of Computer Science - Postgraduate Programs, STMIK Nusa Mandiri, Jakarta, Indonesia. Research interest in Data Mining, Information System, and Computer Science.

Ricky Risnantoyo is a Student of Master of Computer Science - Postgraduate Programs, STMIK Nusa Mandiri, Jakarta, Indonesia. Research interest in Data Mining, Information System, and Computer Science.

Saifurrachman Chohan is a Student of Master of Computer Science - Postgraduate Programs, STMIK Nusa Mandiri, Jakarta, Indonesia. Research interest in Data Mining, Information System, and Computer Science.

Nuraeni Herlinawati is a Student of Master of Computer Science - Postgraduate Programs, STMIK Nusa Mandiri, Jakarta, Indonesia. Research interest in Data Mining, Information System, and Computer Science.

Dr. Sfenrianto, S, Kom, M. Kom is a Faculty Member of the Information Systems Management Department, BINUS Graduate Program – Master of Information Systems Management, Bina Nusantara University, Jakarta, Indonesia. With lecturing subject: Digital Business and E-Commerce Management. Research interest in Digital Business, e-Commerce, business intelligence, E-Learning and Information System.