Journal of Theoretical and Applied Information Technology
15th March 2019. Vol.97. No 5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AUTOMATED COMPLAINTS CLASSIFICATION USING MODIFIED NAZIEF-ADRIANI STEMMING ALGORITHM AND NAIVE BAYES CLASSIFIER

[1]VANNIA FERDINA, [2]MARCEL BONAR KRISTANDA, [3]SENG HANSUN
Informatics Department, Universitas Multimedia Nusantara, Tangerang, Indonesia
E-mail: [1]niaferdina@gmail.com, [2]marcel.bonar@umn.ac.id, [3]hansun@umn.ac.id

ABSTRACT

Complaints provided by customers in the use of products or services is a feedback of the quality of products or services used by customers. In Universitas Multimedia Nusantara (UMN), students can deliver their complaints through an organization, i.e. Dewan Keluarga Besar Mahasiswa (DKBM) UMN. All students' complaints are manually classified into predefined categories by DKBM so that it can be delivered to related division. It costs a lot of time and human resources of DKBM UMN, and also caused misclassification of incoming complaints. In e-complaint system, a method that can be used to support efficient complaint processing is the use of automatic classification system because it can save both time and human resources. Naive Bayes Classifier (NBC) algorithm is one the algorithm that can be used to classify text automatically and for the preprocessing stage, modified Nazief-Adriani stemming algorithm is used. Based on the study conducted, it can be concluded that Naive Bayes Classifier algorithm with modified Nazief-Adriani stemming algorithm is able to do the classification well. This is indicated from the precision value of 91.86%, the recall value of 84.48%, and the f-1 score value of 86.29% for the ratio of training data and test data 90:10, and an average accuracy of 86%.
Keywords: *e-Complaint, Naive Bayes, Classifier Algorithm, Text Classification, Text Mining*

1. INTRODUCTION

Complaints provided by customers in the use of products or services is a feedback of the quality of products or services used by customers [1]. In university's terms, student's complaints about the services provided are important things to note because if it's not handled properly, it will lead to the higher students transfer [1]. In Universitas Multimedia Nusantara (UMN), students can deliver their complaints through an organization, i.e. Dewan Keluarga Besar Mahasiswa (DKBM) UMN [2]. All students' complaints are manually classified into predefined categories by DKBM so that it can be delivered to related division [2]. It cost a lot of time and human resources of DKBM UMN, and also caused misclassification of incoming complaints [2].

In the technological era, a lot of web based applications is developed and complaints management system is also implemented online [3]. In e-mail based complaints handling system, a method that can be used to support efficient complaint processing is the use of automatic classification system because it can save both time and human resources [4]. E-complaint user doesn't have to think the complaints' subject or where the complaints should be addressed and the task to categorizing complaints should be done automatically by the software [5].

Naive Bayes Classifier algorithm is one the algorithm that can be used to classify text automatically. Naive Bayes Classifier algorithm was often used as a standard of text classification because it's relatively fast and easy to implement [6]. Naive Bayes Classifier algorithm shows a very good performance and can optimally works despite the small number of training data [6-9]. The advantages of using the Naive Bayes Classifier algorithm (NBC) are it's easy to use, requires only one time scanning of training data, and requires only a small number of training data to estimate the parameters required in the classification [9]. The first stage of text classification is preprocessing and one of the preprocessing step is stemming process [10]. In this study, the stemming process is done by using Nazief-Adriani algorithm which can do stemming in Bahasa Indonesia with high precision [11]. In addition to the stemming rules based on Nazief-Adriani algorithm, there are some additional rules for addressing stemming failures in some types of affixes [12].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

So in this study, an automated complaints classification system in Bahasa Indonesia for University Students using Naïve Bayes Classifier with modified Nazief-Adriani stemming algorithm was built. The next section will give a brief explanation on some theoretical basis used in this study, followed by the research methodology in Section 3. The implementation results will be described in Section 4 together with some analysis using 10-fold cross validation and confusion matrix methods. Some conclusion remarks and further research will end the organization of this paper.

2. THEORETICAL BASIS

Some theoretical basis will be explained in this Section, such as text classification, modified Nazief-Adriani and Naïve Bayes classifier algorithms, class imbalance, confusion matrix, and k-fold cross validation.

2.1 Text Classification

In a modern culture, text is the most common form of formal information exchange [13]. There are a large number of text-based documents available in electronic form [14]. Data mining is a process of looking for patterns on the data, while text mining is related to the process of looking for patterns on the text [13]. The main purpose of text mining is to allow users to extract information from sources in the form of text and related to operations such as information retrieval, classification, and summarization [15]. Text classification is one example of text mining [16]. Text classification will automatically classify text documents based on predefined categories, such as sports, politics, or arts [15]. The words contained in a text is called feature and document is represented as a bag of words that ignores the order of words and its contextual effect [13]. The first stage of text classification is preprocessing [10]. Generally, the steps taken in preprocessing are as follows [15].

1. Tokenizing

Document is treated as a string and partitioned into token list. This method is used to process the content from text into single words [10]. At this stage, certain characters such as punctuations are also removed [17]. Before tokenizing is done, all letters on the text are converted into lowercase or uppercase [17]. There is also a filter process for words that doesn't start with letter [18].

2. Stopwords removal

This step is the process of removing words that have no effect on a text [19]. For Indonesian language, examples of stopwords are the name of month, pronoun, conjunction, and others [18]. Preposition and conjunction in Indonesian language are also eliminated at this stage [20].

3. Stemming

Stemming process converts words into basic words and include language-dependent linguistic knowledge [21]. The purpose of the stemming process is to remove the affixes that exist on every word [22]. In this study, the stemming process is done by using Nazief-Adriani algorithm which can do stemming with high precision [11]. In addition to the stemming rules based on Nazief-Adriani algorithm, there are some additional rules for addressing stemming failures in some types of affixes [12].

After preprocessing, the text can be classified using several machine learning approaches, i.e. supervised and unsupervised learning [23]. This study is conducted using supervised learning approach because the category is predefined and sample document has been labeled with the appropriate category. One of the text classification algorithms that used the supervised learning approach is the Naive Bayes Classifier algorithm [24].

2.2 Modified Nazief-Adriani Stemming Algorithm

Nazief-Adriani algorithm is one of the popular stemming algorithms, especially for Bahasa Indonesia. Since the automated complaints classification system will be focused on the use of Bahasa Indonesia, we used this algorithm with some modification according to Asian, et al. [12]. It has some stages as follows [12].

1. Search for the word in dictionary. If the word is found, it's assumed that the word is a base word and the algorithm stops. If the word is not found, do step 2.
2. Remove the inflectional suffixes if any. Start from the inflectional particle ("-lah", "-kah", "-tah", and "-pun"), then the possessive pronoun ("-ku", "-mu", and "-nya"). Search for the word in dictionary. If the word is found, the algorithm stops. If the word is not found, do step 3.
3. Remove the derivation suffixes ("-an", "-i"). Search for the word in dictionary. If the word is found, the algorithm stops. If the word is not found, do step 3a.
   a. If the suffix "-an" is removed and a "-k" suffix is found, remove the suffix "-k". Then, do step 4. If the word is found, the algorithm stops. If the word is not found, do step 3b.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

b. Then removed suffix ("-i", "-an", or "-kan") is returned.

4. Remove the derivation prefix ("di-", "ke-", "se-", "me-", "be-", "pe-", "te-") with three times maximum iteration.
   a. Iteration stop if:
      - There is a combination of banned prefix and suffix as outlined in Table 1.

Table 1: Combination of prefix and suffix which is not allowed

Prefix	Suffix which is not Allowed
be-	i-
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan

      - The currently detected prefix is the same as the previously removed prefix.
      - Three prefixes have been removed.
   b. Identify the prefix's type and remove it. Prefix consists of two types as follows.
      - If the word's prefix is "di-", "ke-", or "se-", prefix can be directly removed from the word.
      - If the word's prefix is "me-", "be-", "pe-", or "te-", additional process is required to do the word beheading process. The rules of word beheading can be seen in Table 2.
   c. If the word is not found in the dictionary, repeat step 4. If the word is found, the algorithm stops.
   d. Do the recoding. This step is done based on the prefix type and can generate different words. Recoding rules can be seen in Table 2. Recoding is done by adding the recoding character at the beginning of the decapitated word. The recoding character is the lowercase letters after the hyphen ("-") or before the parenthesis if it refers to the list of word beheading rules [20].
5. If all the steps fail, the word inputs tested on this algorithm are considered as basic word.

In addition to the stemming rules based on Nazief-Adriani algorithm, there are some additional rules for addressing stemming failures in some types of affixes. The rules are outlined as follows [12].

1. Addition of particles "-pun" to the list of inflectional suffix.
2. If a word begins with "ber-" and has inflectional suffix "-lah", the prefix is removed before the suffix.
3. If a word begins with "ber-" and has derivation suffix "-an", the prefix is removed before the suffix.
4. If a word begins with "men-" and has derivation suffix "-i", the prefix is removed before the suffix.
5. If a word begins with "di-" and has derivation suffix "-i", the prefix is removed before the suffix.
6. If a word begins with "pe-" and has derivation suffix "-i", the prefix is removed before the suffix.
7. If a word begins with "ter-" and has derivation suffix "-i", the prefix is removed before the suffix.

After preprocessing, the text can be classified using several machine learning approaches, i.e. supervised and unsupervised learning [23]. On the supervised approach, category is predefined and category labeling is done to the sample document set or training document, while on than unsupervised approach, which is also called document clustering, the classification should be done entirely without reference to external information (categories are not defined first) [23]. This study is conducted using supervised learning approach because the category is predefined and sample document has been labeled with the appropriate category. One of the text classification algorithms that used the supervised learning approach is the Naive Bayes Classifier algorithm [24] that will be used in this study.

2.3 Naïve Bayes Classifier

Naive Bayes Classifier is a classification algorithm based on probability and Bayesian theorem with the assumption that each variable is independent of each other [25]. Naive Bayes Classifier algorithm is one of the most efficient and effective algorithm in machine learning and data mining caused by its independent assumptions [8]. Naive Bayes Classifier algorithm shows a very good classification performance and also can work well despite the very strong dependencies between features [7, 8]. Testing of the Naive Bayes Classifier algorithm yields a higher average of f-measure compared to the K-Nearest Neighbor algorithm in text classification [26]. While using naive design and over-simplified assumptions, Naive Bayes Classifier algorithm works reasonably

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

well in a complex, real world situations [9].

There are two models of Naive Bayes Classifier that is often used [7]. The first one, namely Multi-variate Bernoulli, represents the document as a vector of binary attributes indicating whether or not a word is present in the document [7]. The second model, the Multinomial model, represents the document as a set of word occurrences and the number of occurrences is taken into account [7]. Multinomial model is generally better than Multi-variate Bernoulli model and it can reduce 27% error compared to the Multi-variate Bernoulli model [7]. The Multinomial model of Naive Bayes Classifier algorithm has two assumptions; the bag of words assumption which states that the position of the word in the document has no effect and the conditional independence assumption which states the probability of each feature is independent [27].

Naive Bayes Classifier algorithm takes two stages in the process of text classification, the training stage and the classification stage [25]. In the training stage, the process of analyzing the sample documents and determining the prior probability for each category based on sample documents is conducted [25]. The probability of vocabulary raised in the class in the sample document (likelihood probability) is calculated using Eq. (1) [28].

$$P(W_t \mid C = k) = \frac{\sum_{i=1}^{N_k} X_{it}}{\sum_{s=1}^{|V|} \sum_{i=1}^{N_k} X_{is}} \quad (1)$$

The value of X_{it} is the number of occurrences of vocabulary W_t in the sample document D_i in the class k. The value of N_k is a total amount of sample documents in the class k. The value of $\sum_{i=1}^{N_k} X_{it}$ is a total number of occurrences of vocabulary W_t in the sample document D_i in the class k, while the value of $\sum_{s=1}^{|V|} \sum_{i=1}^{N_k} X_{is}$ is a total number of occurrences of vocabulary W_s in the sample document D_i in the class k, calculated for each word W_s in V. Probability calculation with Eq. (1) raises a problem when there is one vocabulary that does not appear at all in the sample document, which makes the probability of 0 [28]. Just because a word does not appear in the training data document, it does not mean that it will not appear on other documents in the same category [28]. Therefore, Laplace's law of succession or add one smoothing is applied, adding a value of 1 to the number of occurrences of each word. Probability measurement with add one smoothing is done by substituting Eq. (1) with Eq. (2) [28].

$$P(W_t \mid C = k) = \frac{1 + \sum_{i=1}^{N_k} X_{it}}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{N_k} X_{is}} \quad (2)$$

The denominator of Eq. (2) is added with the total vocabulary in the sample document ($|V|$) to ensure a normalized probability after the numerator added with value 1 [28]. In the classification stage, the posterior probability calculation for each class of the test document, i.e. the probability of classes raised in the document, is done using Eq. (3) [28].

$$P(C \mid D) \propto P(C) \prod_{h=1}^{len\,(D)} P(U_h \mid C) \quad (3)$$

The value of $P(C)$ is the prior probability obtained from the total amount of sample documents in the class C (N_k) compared to the total amount of sample documents (N) in the training stage. The value of $P(C)$ is calculated with Eq. (4) [28].

$$P(C = k) = \frac{N_k}{N} \quad (4)$$

The value of $P(U_h \mid C)$ in the Eq. (3) is a probability value of word U_h in the class C which is the h^{th} word of the test document D based on the value obtained from the calculation using Eq. (2) at the training stage [28]. Class determination can then be done by finding the highest posterior probability value (maximum a posteriori) [27]. If the value of $P(C = k \mid D)$ is greater than the value of $P(C = j \mid D)$, it can be concluded that document D is classified in the class k [28].

2.4 Class Imbalance

Most of the real word data is unbalanced in the context of the proportion of data available for each class or category [29]. In classification, data is called unbalanced if there is a class or category that has a relatively smaller amount of data compared to other classes [30]. A class that has a relatively larger amount of data is called majority and a class that has a relatively smaller amount of data compared to other classes is called minority [30]. In classification with unbalanced data distribution, data from minority class would be more likely to be misclassified than data from the majority class and the classification algorithm would tend to bias toward the majority class [30-33].

2.5 Confusion Matrix

Text classification testing with Naive

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Bayes Classifier algorithm is done by measuring precision, recall, and F-1 score [7,34]. Confusion matrix is a table that states the amount of test data that is correctly classified and the amount of test data incorrectly classified [25]. Precision is a measure that estimates the probability of a correct prediction of a positive class [35]. The calculation of precision of each class can be done using Eq. (5) [34].

$$Precision = \frac{True\ Positives}{True\ Positives + Fal \quad Positives} \quad (5)$$

Recall is the proportion of all predictions of the positive class which is correctly predicted as positive class [35]. The calculation of recall of each class can be done using Eq. (6) [34].

$$Recall = \frac{True\ Positives}{True\ Positives + \quad Negatives} \quad (6)$$

The value of F-1 score is a relation between the positive label of the data and the label given by the classifier [36]. The calculation of F-1 score of each class can be done using Eq. (7) [34].

$$F-1\ score = \frac{2 \times Precision \times Recall}{Precision + R} \quad (7)$$

There are two methods that can be used to obtain a single result from all the calculations of each class, i.e. macro-averaging and micro-averaging [24]. Macro-averaging simply averages the count of each class, while micro-averaging collects the confusion matrix of each class called pooled table and performs calculations based on the pooled table [24]. Macro-averaging can take into account the effectiveness of a class that has little number of test data [24]. Therefore, measurements using the macro-averaging method are used in this study.

2.6 K-fold Cross Validation

Testing the accuracy of text classification can be done using *k*-fold cross validation [37, 38]. Cross validation is a statistical method for evaluating learning algorithm by dividing data into two segments, one segment is used for the training stage and one segment is used to validate the model [39]. The general form of cross validation is *k*-fold cross validation [39]. In *k*-fold cross-validation, the data are divided into *k* subsection with relatively equal amounts of data between subsections [40]. The training and testing process is performed on *k* number of iterations and on each iteration, different subsection is used for the testing process, while the other *k*-1 subsections are used for the training process [39]. The final evaluation is the average accuracy result of each validation step *k* [38]. The advantages of the *k*-fold cross validation method are that in this method, the way data placed does not affect because each data will appear once in the test data and appear as much as *k*-1 times in the training data [41]. Compared to other *k* values, 10-fold cross validation is the value of *k* accepted as the most reliable method because it can provide accurate error estimation of a model of various algorithms and applications [37, 38].

3. RESEARCH METHODOLOGY

3.1 Data Source

The source of complaints data used in this study is UMN students' complaints data obtained from Dewan Keluarga Besar Mahasiswa (DKBM) UMN. The data obtained consists of 500 data and are divided into five predefined categories. There categories are Akademik (Academic), Kegiatan (Activities), Fasilitas (Facility), BEM (Students' Organization), and Lainnya (Other). There are 150 complaints data for Akademik category, 63 complaints data for Kegiatan category, 215 complaints data for Fasilitas category, 50 complaints data for BEM category, and 22 complaints data for Lainnya category.

3.2 Methodology

1. Literature Study

Literature study is used to study theories related to automatic text classification, including text classification, class imbalance, Naive Bayes Classifier algorithm, confusion matrix, and k-fold cross validation. The theories of the literature study are used as the basis of research.

2. Application Design

The design process of the application's flow that is used to test the Naive Bayes Classifier algorithm is performed. This stage also conducted the data collection needed in testing, including complaints data, basic word of Indonesian language data, and stopwords of Indonesian language data. The basic word of Indonesian language is obtained from Bahtera dictionary [42] and the stopwords of Indonesian language data is obtained from a study titled "A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia" [43]. The steps performed for the training process is described in Figure 1.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

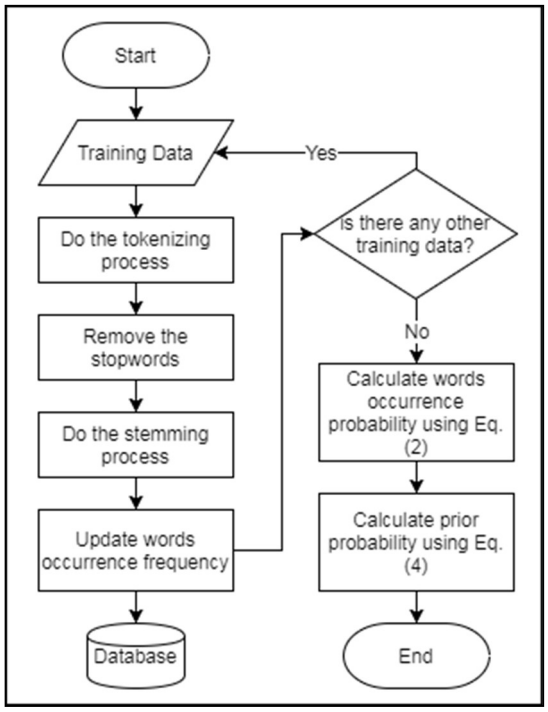

Figure 1: Flowchart of training process

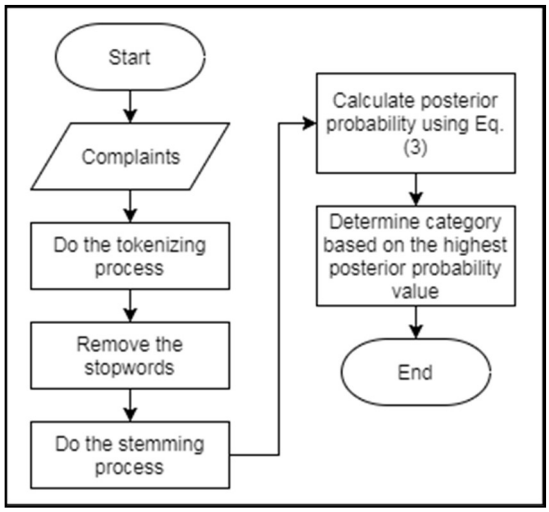

Figure 2: Flowchart of Testing Process

Each training data will go through the tokenizing process first to convert the whole string into a token list. After that, the stopwords contained in the token list is being removed. Each word in the token list is then being stemmed to obtain its basic form. After the stemming process, update the words occurrence frequency in the vocabulary for its category. If there is any other training data, repeat those processes. If there isn't any other training data, calculate the words occurrence probability for each word in each category using Eq. (2). After that, calculate prior probability for each category using Eq. (4). Meanwhile, the steps performed for the testing process is described in Figure 2. Complaint data that are included in the test data set will also go through the tokenizing process to convert the complaint string into a token list. After that, stopwords is being removed from the token list. Each word in the token list is then being stemmed to obtain its basic form. Calculation of the posterior probability is performed using Eq. (3) for each category. Finally, the complaint's category is determined based on the highest posterior probability value obtained from the previous calculation using Eq. (3).

3. Application Development

The development of application that is used to test the Naive Bayes Classifier algorithm is performed. The application is developed using PHP programming language version 5.6.8 and MySQL Database version 5.5.32.

4. Naive Bayes Classifier Algorithm-Training Phase

The Naive Bayes Classifier algorithm training is performed using complaints data obtained from DKBM UMN. Complaints data are divided into two segments, training data and test data. For testing with confusion matrix, training data and test data are divided into seven scheme of comparison ratio of training data and test data [44]. The first step of the training stage is preprocessing of the training data. Complaints data that become training data will go through the preprocessing phase, including tokenizing, stopword removal, and stemming. The next step is the training itself. The training phase of Naive Bayes Classifier algorithm begins by calculating the likelihood probability, i.e. the probability of each word being raised in each category using Eq. (2). Next, calculate the prior probability of each category using Eq. (4). The resulting probability value is then stored into the database and will be used in the classification stage of the text.

5. Naive Bayes Classifier Algorithm-Testing Phase

The Naive Bayes Classifier algorithm testing is performed using confusion matrix to calculate precision, recall, f-1 score value. The average accuracy value is measured using 10-fold cross validation method. Complaints data that become the test data will go through preprocessing stage, including tokenization, stopword removal, and stemming as done to the training data. The next step is the classification process. Classification is performed for all complaints that are included in the test data. For each complaints data, the calculation of posterior probability for each class is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

performed using Eq. (3). After the value of posterior probability for each category is calculated, the category of complaint is determined by finding the category with the highest posterior probability value (maximum a posteriori).

a. Testing with 10-fold Cross Validation

For the testing with 10-fold cross validation, the complaint data obtained from DKBM UMN will be divided into 10 partitions. The training and testing process for 10-fold cross validation is performed for 10 iterations. On each iteration, 9 partitions will be used as training data and 1 partition will be used as test data and the partitions used are different on each iteration. For each iteration, classification accuracy is calculated and for the final evaluation, the average classification accuracy with Naive Bayes Classifier algorithm is obtained.

b. Evaluation with Confusion Matrix

After all complaints that are included in the test data is categorized using Naive Bayes Classifier algorithm, the calculation of confusion matrix data is performed for each category by comparing the actual category with the predicted category obtained from the Naive Bayes Classifier algorithm. The confusion matrix is then used to calculate the value of precision, recall, and F-1 score with Eq. (5), Eq. (6), and Eq. (7) sequentially with the macro-averaging technique.

4. RESULTS AND DISCUSSION

4.1 Implementation Result

Figure 3 shows the implementation result of the main page after the user login into the system.

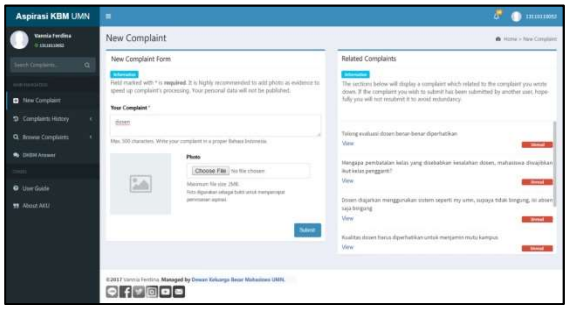

Figure 3: Home page of e-complaint system

Using this system, user can input and submit a new complaint via a form and then the Naïve Bayes Classifier algorithm will be run to classify the complaints based on prior probabilities calculated. Figure 4 shows the complaints detail page where the complaints' category is obtained from the Naïve Bayes Classifier result.

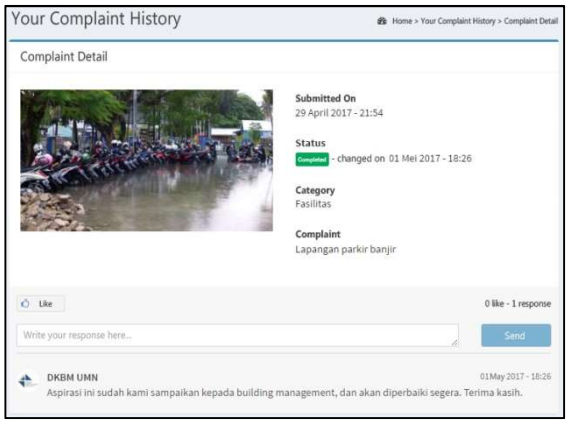

Figure 4: Complaints detail page

4.2 10-fold Cross Validation Result

Testing with 10-fold cross validation is performed using the same training data and test data in the test with confusion matrix. The training data is then divided into 10 partitions with the even distribution of data amounts of each category on each partition. The recapitulation result of accuracy calculation is shown in Figure 5.

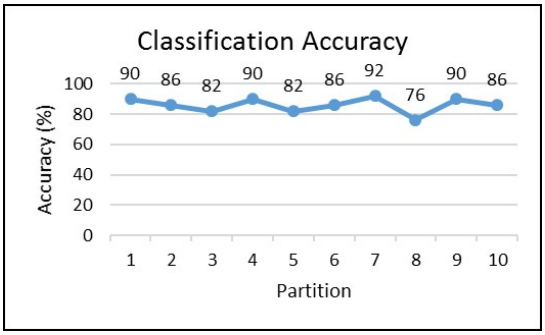

Figure 5: Classification accuracy measurement

From the calculation of the accuracy of each partition, the average accuracy of complaints classification using Naive Bayes Classifier algorithm is 86%.

4.3 Confusion Matrix Result

Testing with confusion matrix is done by dividing the complaint data obtained from DKBM UMN into two segments, training data and test data. The comparison of the amount of training data and test data is built using seven scheme of ratio. For each ratio scheme, the training and testing stage are performed which then generates a confusion matrix. Confusion matrix of each category is used to calculate the value of precision, recall, and F-1 score. To obtain a single result from precision,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

recall, and F-1 score of each ratio, macro-averaging technique is used. The result of the calculation of precision, recall, and F-1 score for each ratio scheme are shown in Table 3.

Table 3: Recapitulation of precision, recall, and f-1 score calculation

Ratio	Precision	Recall	F-1 Score
30 : 70	0.599933	0.649229	0.579374
40 : 60	0.652439	0.67303	0.619729
50 : 50	0.691772	0.687042	0.642424
60 : 40	0.637719	0.674155	0.649167
70 : 30	0.688863	0.681567	0.677605
80 : 20	0.759192	0.744037	0.739698
90 : 10	0.91864	0.844762	0.862893

From the calculations in Table 3, the highest value of precision, recall, and F-1 score were obtained at the 90:10 ratio of training data for each category. The value of precision, recall, and F-1 score is generally increasing from 30:70 to 90:10. At the 60:40 ratio scheme, there is a decrease in precision and recall value compared to precision and recall value at the 50:50 ratio scheme. The result of the analysis of a decrease in precision and recall value is the presence of class imbalance in the confusion matrix of Lainnya category. At the 50:50 ratio of Lainnya category, there is one complaint data which is correctly classified, while at the 60:40 ratio of Lainnya category, there is no complaints data which is correctly classified.

5. CONCLUSION

Based on the study conducted, it can be concluded that Naive Bayes Classifier algorithm with modified Nazief-Adriani stemming algorithm is able to do the classification well. This is indicated from the precision value of 91.86%, the recall value of 84.48%, and the F-1 score value of 86.29% for the ratio of training data and test data 90:10, and an average accuracy of 86%. The class imbalance occurs with Lainnya category as a minority class. Testing with confusion matrix shows that Naive Bayes Classifier classification result affected with the class imbalance, which tends to classify data from the minority class as a part of the majority class data. This results to a higher false negative value that affects the value of precision, recall, and F-1 score measurement. The future research can be conducted to study about how the class imbalance affects the classifier performance and how to prevent those effects.

REFERENCES:

[1] S. Indriyani and S. Mardiana, "Pengaruh Penanganan Keluhan (Complaint Handling) Terhadap Kepercayaan dan Komitmen Mahasiswa Pada Perguruan Tinggi Swasta di Bandar Lampung," *Jurnal Bisnis Darmajaya*, Vol.2, No.1, 2016, pp. 1-13.

[2] I. Angela, Universitas Multimedia Nusantara, Tangerang, Personal communication, 2017.

[3] R. Razali and J. Jaafar, "Complaint Handling Theoretical Framework," *Proceedings of 2012 International Conference on Computer and Information Science (ICCIS)*, 1, 2012, pp. 382-385.

[4] K. Coussement and D. Van den Poel, "Improving Customer Complaint Management by Automatic Email Classification using Linguistic Style Features as Predictors," *Decision Support Systems*, Vol.44, No.4, 2008, pp. 870-882.

[5] A.D. Zaugg, *Online Complaint Management at Swisscom – A Case Study.* [Online]. Available from: http://boris.unibe.ch/58051/1/AB193.pdf [Accessed 05/03/2017].

[6] J.D. Rennie, L. Shih, J. Teevan, and D.R. Karger, "Tackling The Poor Assumptions of Naive Bayes Text Classifiers," *Proceedings of International Conference on Machine Learning (ICML)*, 3, 2003, pp. 616-623.

[7] A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Classification," *AAAI-98 Workshop on Learning for Text Categorization*, 752, 1998 pp. 41-48.

[8] H. Zhang, "The Optimality of Naive Bayes," *AA*, Vol.1, No.2, 2004, p. 3.

[9] B.K. Bhardwaj and S. Pal, "Data Mining: A Prediction for Performance Improvement Using Classification," *International Journal of Computer Science and Information Security*, Vol.9, No.4, 2011, pp. 136-140.

[10] S. Vijayarani, J. Ilamathi, and Nithya, "Preprocessing Techniques for Text Mining – An Overview," *International Journal of Computer Science & Communication Network*, Vol.5, No.1, 2015, pp. 7-16.

[11] L. Agusta, "Perbandingan Algoritma Stemming Porter Dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia," *Proceedings of Konferensi Nasional Sistem dan Informatika*, 2009, pp. 196-201.

[12] J. Asian, H.E. Williams, and S.M.M. Tahaghoghi, "Stemming Indonesian," *Proceedings of the Twenty-eighth Australasian*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Conference on Computer Science, 38, 2005, pp. 307-314.

[13] I.H. Witten, *Text Mining*, Department of Computer Science, University of Waikato, New Zealand, 2004.

[14] S.M. Kamruzzaman, F. Haider, and A.R. Hasan, *Text Classification Using Data Mining.* [Online]. Available from: https://pdfs.semanticscholar.org/7edf/fc15cde8dbf434b4f11bbae4fa27900ef401.pdf [Accessed 16/02/2017].

[15] V. Korde and C.N. Mahender, "Text Classification and Classifiers: A Survey," *International Journal of Artificial Intelligence & Applications (IJAIA)*, Vol.3, No.2, 2012, pp. 85.

[16] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys (CSUR)*, Vol.34, No.1, 2001, pp. 1-47.

[17] A.T.J. Harjanta, "Preprocessing Text Untuk Meminimalisir Kata yang Tidak Berarti Dalam Proses Text Mining," *Jurnal Informatika Upgris*, Vol.1, No.1, 2015.

[18] Y. Ganisaputra and R. Tan, "Pembuatan Aplikasi Datamining Facebook dan Twitter Dengan Naive Bayes Classifier," *Jurnal Informatika*, Vol.9, No.2, 2013, pp. 173-188.

[19] A.F. Hidayatullah and S.N. Azhari, "Analisis Sentimen dan Klasifikasi Kategori Terhadap Tokoh Publik Pada Twitter," *Proceedings of Seminar Nasional Informatika (SEMNASIF)*, Vol.1, No.1, 2015.

[20] A. Firdaus, Ernawati, and A. Vatresia, "Aplikasi Pendeteksi Kemiripan Pada Dokumen Teks Menggunakan Algoritma Nazief dan Adriani dan Metode Cosine Similarity," *Jurnal Teknologi Informasi*, Vol.10, No.1, 2014.

[21] D. Sharma and S. Jain, "Evaluation of Stemming and Stop Word Techniques on Text Classification Problem," *International Journal of Scientific Research in Computer Science Engineering*, Vol.3, No.2, 2015, pp. 1-4.

[22] B. Kurniawan, S. Effendi, and O.S. Sitompul, "Klasifikasi Konten Berita Dengan Metode Text Mining," *Jurnal Dunia Teknologi Informasi*, Vol.1, No.1, 2012, pp. 14-19.

[23] D. Tsarev, M. Petrovskiy, and I. Mashechkin, "Supervised and Unsupervised Text Classification via Generic Summarization," *International Journal of Computer Information System and Industrial Management Applications*, Vol.5, 2013, pp. 509-515.

[24] D.C. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*, Cambridge: Cambridge University Press, 2009.

[25] A. Indriani, "Klasifikasi Data Forum Dengan Menggunakan Metode Naive Bayes Classification," *Proceedings of Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*, Vol.1, No.1, 2014.

[26] R. Anggono, A.A. Suryani, and A.P. Kurniati, *Analisis Perbandingan Metode K-Nearest Neighbor dan Naive Bayes Classifier Dalam Klasifikasi Teks*. Bachelor Thesis, Department of Computer Science, Universitas Telkom, Bandung, 2009.

[27] R.A. Aziz, M.S. Mubarok, and Adiwijaya. "Klasifikasi Topik pada Lirik Lagu dengan Metode Multinomial Naive Bayes," *Proceedings of Indonesia Symposium on Computing*, 2016, pp. 139-148.

[28] H. Shimodaira, "Text Classification Using Naive Bayes," *Learning and Data Note,* Vol.7, 2014, pp.1-9.

[29] Y.M. Huang, C.M. Hung, and H.C. Jiau, "Evaluation of Neural Networks and Data Mining Methods on a Credit Assessment Task for Class Imbalance," *Nonlinear Analysis: Real World Applications*, Vol.7, No.4, 2006, pp. 720-747.

[30] T.R. Hoens and N.V. Chawla, "Imbalanced Datasets: From Sampling to Classifier," *Imbalanced Learning: Foundations, Algorithms and Applications*, Wiley, 2013, pp.43-59.

[31] R. Longadge, S.S. Dongre, and L. Malik, "Class Imbalance Problem in Data Mining: Review," *International Journal of Computer Science and Network (IJCSN)*, Vol.2, No.1, 2013.

[32] R. Blagus and L. Lusa, "SMOTE for High-Dimensional Class-Imbalanced Data," *BMC Bioinformatics*, Vol.14, No.106, 2013, pp. 1-16.

[33] N.V. Chawla, N. Japkowicz, and A. Kotcz, Editorial: Special Issue on Learning from Imbalanced Data Sets. *ACM Sigkdd Explorations Newsletter*, Vol.6, No.1, 2004, pp.1-6.

[34] V.V. Asch, *Macro- and Micro-averaged Evaluation Measures,* 2013.

[35] E. Costa, A.C. Lorena, A.C.P.L.F. Carvalho, and A.A. Freitas, "A Review of Performance Evaluation Measures for Hierarchical Classifiers," *AAAI-2007 Workshop*, 2007, pp. 1-6.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[36] M. Sokolova and G. Lapalme, "A Systematic Analysis of Performance Measures for Classification Tasks," *Information Processing and Management: An International Journal*, Vol.45, No.4, 2009, pp. 427-437.

[37] M. Koppel, S. Argamon, and A.R. Shimoni, "Automatically Categorizing Written Text by Author Gender," *Literary and Linguistic Computing*, Vol.7, No.4, 2002, pp. 401-412.

[38] S. Karimi, J. Yin, and J. Baum, "Evaluation Methods for Statistically Dependent Text," *Computational Linguistics*, Vol.41, No.3, 2015, pp. 539-548.

[39] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation," *Encyclopedia of Database Systems*, 2009, pp. 532-538.

[40] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, Vol.14, No.2, 1995, pp. 1137-1145.

[41] K. Polat and S. Güneş, "Classification of Epileptiform EEG Using a Hybrid System Based on Decision Tree Classifier and Fast Fourier Transform," *Applied Mathematics and Computation*, Vol.187, No.2, 2007, pp.1017-1026.

[42] R. Liyantanto, *Kata Dasar Bahasa Indonesia*. [Online]. Available from: https://liyantanto.wordpress.com/2010/12/06/kata-dasar-bahasa-indonesia/ [Accessed 03/03/2017].

[43] F.Z. Tala, *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia,* Report, Institute for Logic, Language and Computation, Universiteit van Amsterdam, The Netherlands, 2003.

[44] A. Hamzah, "Klasifikasi Teks dengan Naive Bayes Classifier (NBC) Untuk Pengelompokan Teks Berita dan Abstract Akademis," *Proceedings of Seminar Nasional Aplikasi Sains & Teknologi (SNAST)*, Periode III, 2012.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Rules of word beheading according to Firdaus, et al. [20]

Rules No.	Word format	Beheading
1	berV…	ber-V … | be-rV
2	berCAP…	ber-CAP… (C != 'r' & P != 'er')
3	berCAerV…	ber-CaerV… (C != 'r')
4	belajar	bel-ajar
5	berC1erC2…	be-C1erC2… (C1 != 'r' | 'l')
6	terV…	ter-V… | te-rV…
7	terCerV…	ter-CerV (C != 'r')
8	terCP…	ter-CP… (C!='r' & P !='er')
9	teC1erC2…	te-C1erC2… (C1 != 'r')
10	me{l|r|w|y}V…	me – {l|r|w|y} V…
11	mem{b|f|V}…	mem-{b|f|v}…
12	Mempe	mem-pe…
13	mem{rV|V}…	me-m{rV|V}… | me-p{rV|V}
14	men{c|d|j|s|z}…	men-{c|d|js|z}…
15	menV…	me-nV… | me-tV
16	meng{g|h|q|k}…	meng-{g|h|q|k}…
17	mengV…	meng-V… | meng-kV…| mengV-... if V='e'
18	menyV…	meny-sV…
19	mempA…	mem-pA… (A != 'e')
20	pe{w|y}V…	pe-{w|y}V…
21	perV…	per-V… | pe-rV…
22	perCAP…	per-CAP… (C != 'r' & P != 'er')
23	perCAerV…	per-CAerV… (C != 'r')
24	pem{b|f|V}…	pem-{b|f|V}…
25	pem{rV|V}…	pe-m{rV|V}… | pe-p{rV|V}…
26	pen{c|d|j|z}…	pen-{c|d|j|z}…
27	penV…	pe-nV… | pe-tV…
28	pengC…	peng-C…
29	pengV…	peng-V… | peng-kV… | pengV-... if V='e'
30	penyV…	peny-sV…
31	pelV…	pe-lV…, except "pelajar" will be "ajar"
32	peCerV…	per-erV … (C!= {r|w|y|l|m|n})
33	peCP	pe-CP… (C!={r|w|y|l|m|n} & P!= 'er')
34	terC1erC2…	ter-C1erC2... (C1!= 'r')
35	peC1erC2…	pe-C1erC2... (C1!={r|w|y|l|m|n})

Note:
C: consonant letter
V: vocal letter
A: consonant or vocal letter
P: word fragment, such as 'er'