OPEN ACCESS
ISSN 2356-5462
http://socj.telkomuniversity.ac.id/ijoict/

Intl. Journal on ICT
Vol. 10, No. 2, Dec 2024. pp. 161-169
doi: 10.21108/ijoict.v10i2.1007

Sentiment Analysis of Game Review using Random Forest

Mahendra Dwifebri Purbolaksono[1*], Utami Kusuma Dewi[2], Ryan Lingga Wicaksono[3], Adityo Permana Wibowo[4]

[1,2,3]*School of Computing, Telkom University*
Jalan Telekomunikasi no. 1, Bandung Indonesia

[4]*Faculty of Science & Technology, Yogyakarta University of Technology*
Jalan Siliwangi (North Ring Road), Jombor, Sleman, DI Yogyakarta, Indonesia

* mahendradp@telkomuniversity.ac.id

Abstract

Steam provides a platform for users to write reviews of games they have purchased, offering valuable feedback for developers. However, the large volume of reviews makes it difficult for developers to determine whether the sentiment is positive or negative. While Steam's rating system exists, it often fails to fully capture the sentiment expressed in the written reviews. To address this, sentiment analysis can help developers better understand user feedback. This study uses the Random Forest algorithm with TF-IDF feature extraction in Bigram and Trigram formats for sentiment classification. The findings show that using Bigram TF-IDF without Lemmatization in the preprocessing stage achieved the highest performance, with an average F1 score of 62%.

Keywords: Sentiment Analysis, Random Forest, TF-IDF, Bigram, Trigram, Steam Game Reviews

I. INTRODUCTION

THE COVID-19 pandemic and the enforcement of lockdown measures from 2020 to 2022 resulted in limited social activities and difficulty in finding entertainment outside the home. Authorities have restricted some entertainment venues to prevent widespread transmission. An alternative form of entertainment that does not require leaving the house is playing games [1]. Steam is a digital game distribution platform that provides a platform for both large (Triple-A) and small (Indie) game developers to showcase their games to a global audience [2]. Steam is a popular platform for game, allows users to provide feedback, but the sheer quantity of reviews makes it challenging for developers to determine overall sentiment, whether players like or dislike a game [3]. Therefore, sentiment analysis is a technique that helps identify positive or negative emotions in text, can be applied to automate this process and help developers better understand their community's feedback [4].

This research adopts a machine learning approach to sentiment analysis, specifically comparing inputs between Bigrams (two words) and Trigrams (three words) using various algorithms. Machine learning, particularly Supervised learning, is used to classify and predict sentiments, whether positive or negative, based on a training dataset. Previous research by Zuo et al. [5] indicates that most reviews collected from Steam show positive feedback, while a small portion shows negative feedback. In this study, classification is performed using the Random Forest method, chosen based on research by Hartmann et al. [6], which showed that Random Forest outperformed other methods such as ANN, KNN, NB, and SVM, achieving an accuracy of 75%. Additionally, research by Ahuja et al. [7] demonstrated that using TF-IDF for sentiment analysis preprocessing resulted in a 3%-4% improvement in results.

Received on 27 Sep 2024. Revised on 21 Nov 2024. Accepted and Published on 22 Dec 2024.

Building on previous research, this study focuses on using the Random Forest algorithm, a machine learning method, along with TF-IDF, a technique that converts words into numerical representations based on their importance, to classify the sentiment of game reviews. As the previous research [7] said TF-IDF can improve, so this study used TF-IDF as Extraction Features. By comparing different feature extraction methods such as Bigram and Trigram, this research aims to identify the most effective approach for improving sentiment classification accuracy, ultimately aiding game developers in enhancing their products based on user feedback. The dataset used is obtained from the Steam review page for the game Cyberpunk 2077 in English.

II. LITERATURE REVIEW

The research conducted by Zuo [5] compares two machine learning algorithms, decision tree, and Naive Bayes, for sentiment analysis of reviews on Steam. The results show that the decision tree outperforms Naive Bayes, achieving about 75% better accuracy. Febrianta [8], in their study, analyzed the sentiment of Steam reviews on local indie games. Out of 3497 analyzed reviews, 2442 received positive sentiment (69.8%), while 1055 received negative sentiment (30.2%). The topics frequently mentioned regarding aspects of the game reflect the feelings of the players, expressed through words like "bad", "good", "love", and "interesting". These reviews reflect the players' feelings while playing the game.

Tan [9], in their research, compared five machine learning models for large text classification tasks like sentiment analysis. Of all the models tested, SVM performed the best, achieving around 91% accuracy. Eberhard [10] categorized game reviews from the Steam platform into Unhelpful, Helpful, and Top Review categories. Helpful reviews are generally longer, use complex language, and are more critical of the product. However, meaningless reviews, comedic reviews, and off-topic reviews were also found. Khomsah [11] studied sentiment analysis using the Random Forest method with Word2Vec feature extraction using the SkipGram model. This research used around 31947 comments on YouTube related to the 2019 presidential election. The tested models achieved an average accuracy of around 90.1% to 91%, but there was a slight decrease in test accuracy, although not significant. Epoch and window size in Skip-Gram affect the model's accuracy level.

Another study [12] focuses on analyzing sentiments expressed on Twitter regarding the upcoming 2024 presidential election in Indonesia. With campaigns intensifying both offline and on social media platforms like Twitter, Facebook, and YouTube, public opinions are varied and plentiful. The research categorizes sentiments into positive, negative, and neutral, visualizing tweet content corresponding to each category. Utilizing a tree-based classification method, specifically a decision tree, the study achieves an impressive accuracy of 99.3%. This accuracy outperforms regression-based methods by 2.5%, highlighting the decision tree's efficacy in classifying and understanding public sentiment dynamics crucial for assessing voter sentiments and campaign strategies leading up to the election.

In paper [13], the study focuses on sentiment analysis applied to Twitter data in Indonesia, particularly concerning public figures. Sentiment analysis involves analyzing opinions expressed in tweets and categorizing them as positive or negative. Twitter is highlighted as a significant platform for discussions on various topics, including public figures, which often trend. The methodology involves text processing and classification using a backpropagation neural network. The research tested this approach on 69 datasets, achieving an accuracy of 62.3%. Out of these, 43 classifications were correct. The study aims to understand public sentiment towards public figures through computational analysis of social media data, emphasizing the application of machine learning techniques for sentiment classification.

III. RESEARCH METHOD

In this section, we will outline the structure of the system being built. The steps applied in sentiment analysis of game reviews involve feature extraction using TF-IDF and application to the classification model using the Random Forest algorithm. The system schema design is depicted in a flow diagram as shown in Fig. 1.

A. Dataset

At this stage, dataset Retrieval is carried out by crawling data from the Steam website, amounting to 1000 reviews. Afterward, manual labeling is conducted, resulting in 592 positives and 408 negatives [14].

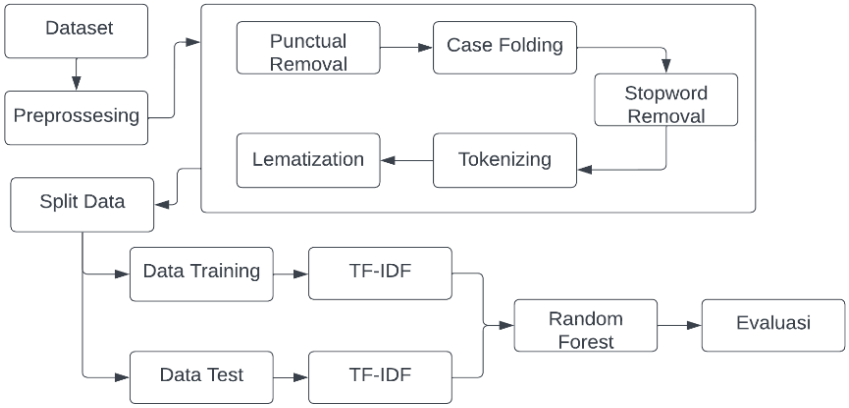

Fig. 1. Project Framework

B. Preprocessing

At this stage, preprocessing is carried out, or the process of transforming the dataset into data that is easily processed by machine learning, as follows:

1) Punctual Removal: In punctual removal, punctuation, numbers, HTML tags, and special characters other than words are removed from the dataset, such as commas and periods as shown in Table I.

TABLE I
PUNCTUAL REMOVAL

Before	After
Even with all the issues that I have encountered during my sessions with Cyberpunk I thoroughly enjoy playing this game. Yes some of the complaints you may have heard are warranted, but the bugs/crashes have been few and far in between from my point of view. Hopefully the problems are solved so that everyone can get experience this game the way it was intended.	*Even with all the issues that I have encountered during my sessions with Cyberpunk I thoroughly enjoy playing this game Yes some of the complaints you may have heard are warranted but the bugscrashes have been few and far in between from my point of view Hopefully the problems are solved so that everyone can get experience this game the way it was intended.*

2) Case Folding: Case folding involves converting uppercase letters to lowercase within the text of words in the dataset for consistency in the normalization of text where all letters are lowercase, for example, 'Even' becomes 'even' as shown in Table II.

TABLE II
CASE FOLDING

Before	After
Even with all the issues that I have encountered during my sessions with Cyberpunk I thoroughly enjoy playing this game Yes some of the complaints you may have heard are warranted but the bugscrashes have been few and far in between from my point of view Hopefully the problems are solved so that everyone can get experience this game the way it was intended	*even with all the issues that i have encountered during my sessions with cyberpunk i thoroughly enjoy playing this game yes some of the complaints you may have heard are warranted but the bugscrashes have been few and far in between from my point of view hopefully the problems are solved so that everyone can get experience this game the way it was intended*

3) Stopword Removal: Stopword removal is eliminating irrelevant conjunctions that commonly appear in the text of review datasets and do not carry significant meaning, such as the word 'the' as shown in Table III.

TABLE III
STOPWORD REMOVAL

Before	After
even with all the issues that i have encountered during my sessions with cyberpunk i thoroughly enjoy playing this game yes some of the complaints you may have heard are warranted but the bugscrashes have been few and far in between from my point of view hopefully the problems are solved so that everyone can get experience this game the way it was intended	*even issues encountered sessions cyberpunk thoroughly enjoy playing game yes complaints may heard warranted bugscrashes far point view hopefully problems solved everyone get experience game way intended*

4) Tokenization: Tokenization involves splitting a string (sentence) into separate segments of words (tokens) to facilitate further text analysis. For example, "great game" becomes ['great', 'game'] as shown in Table IV.

TABLE IV
TOKENIZATION

Before	After
even issues encountered sessions cyberpunk thoroughly enjoy playing game yes complaints may heard warranted bugscrashes far point view hopefully problems solved everyone get experience game way intended	*['even', 'issues', 'encountered', 'sessions', 'cyberpunk', 'thoroughly', 'enjoy', 'playing', 'game', 'yes', 'complaints', 'may', 'heard', 'warranted', 'bugscrashes', 'far', 'point', 'view', 'hopefully', 'problems', 'solved', 'everyone', 'get', 'experience', 'game', 'way', 'intended']*

5) Lemmatization: Lemmatization involves transforming a word into its base form by understanding the context of the word and recognizing and grouping it from its dictionary form. For example, 'requiring' becomes 'require' as shown in Table V.

TABLE V
LEMMATIZATION

Before	After
['even', 'issues', 'encountered', 'sessions', 'cyberpunk', 'thoroughly', 'enjoy', 'playing','game','yes','complaints','may','heard','warranted', 'bugscrashes', 'far', 'point', 'view', 'hopefully', 'problems', 'solved', 'everyone', 'get', 'experience', 'game', 'way', 'intended']	*['even', 'issue', 'encountered', 'session', 'cyberpunk', 'thoroughly', 'enjoy', 'playing', 'game','yes','complaint','may','heard','warranted', 'bugscrashes', 'far', 'point', 'view', 'hopefully', 'problem', 'solved', 'everyone', 'get', 'experience', 'game', 'way', 'intended']*

C. Split Data

At this stage, the dataset will be divided into two sets: the training sata and the test data. The training data will use 80%, comprising 800 reviews of Steam games, while the test data will use 20%, comprising 200 reviews of Steam games.

D. Feature Extraction (TF-IDF)

At this stage, both the Training Data and the Test Data will undergo feature extraction using the TF-IDF method to measure the weight of each word. Term Frequency-Inverse Document Frequency (TF-IDF) is an algorithm for calculating how often a word appears while considering how many files contain that word. Hence, IDF will weigh how much the word is found in many files, and if it's found too often, it will be considered less important. The formula 1 is used for the calculation process [15]:

$$TFIDF(d,t) = TF(d,t) \,.\log \frac{N}{df(t)} \tag{1}$$

In the context of text mining and information retrieval, TF-IDF (Term Frequency-Inverse Document Frequency) plays a crucial role in determining the importance of a term within a document relative to a

collection of documents. TFIDF(d,t) represents the TF-IDF weight assigned to the term t in document d. TF(d,t) refers to the frequency of the term t in document d, indicating how often the term t appears within that specific document. N represents the total number of documents in the collection, providing the context for the calculation. Df(t) denotes the document frequency of term t, which signifies how many documents in the collection contain term t. In this study, TF-IDF will provide Bigram and Trigram instead of Unigram. The differences between those models are shown in Fig. 2 below.

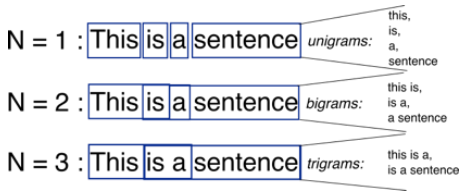

Fig. 2. N-Gram Model

E. *Classification (Random Forest)*

At this stage, classification of the data is performed. In this research, the method used for classification is the Random Forest Classifier. Random forest is a classification method consisting of an ensemble of decision trees. [16] The model as we can see in Fig. 3.

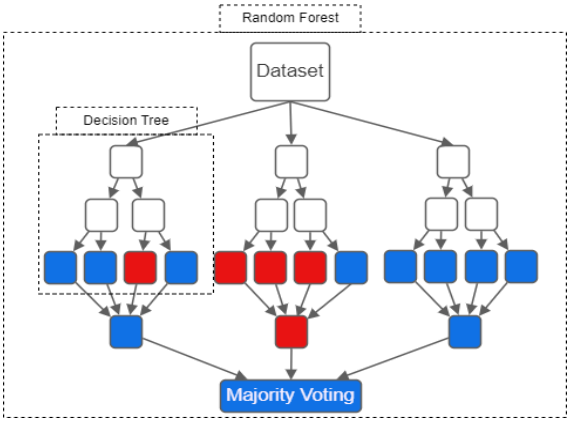

Fig. 3. Random Forest Model

In random forest classification, an ensemble of decision trees is created by training on the sample data from each decision tree. The predictions obtained from the branching are then calculated for the probability of the feature being incorrectly predicted using the Gini Index. The smaller the Gini value from the prediction result, the greater the likelihood of that prediction being selected [17]. Pi is the proportion of the total population's share held by the i-th group or individual. P+ represents the probability or proportion associated with a positive outcome or event. P- represents the probability or proportion associated with a negative outcome or event.

$$Gini\ Index = 1 - \sum_{i=1}^{n} (Pi)^2 = 1 - [(P_+)^2 + (P_-)^2] \quad (2)$$

F. *Evaluation*

At this stage, accuracy measurement is conducted using a Confusion Matrix on the classified data. The Confusion Matrix is used to conclude the classification performance from the results of the test data [17]. The following formula is used for calculating the Confusion Matrix in Table VI [18]:

TABLE VI
CONFUSION MATRIX

Confusion Matrix	Positive	Negative
Positive	*True Positive*	False Negative
Negative	*False Positive*	False Negative

The table obtained from the confusion matrix represents the outcomes divided into True Positive, True Negative, False Positive, and False Negative. From these outcomes, formulas are used to calculate Precision, Recall, Accuracy, and F1-Score [19]. Precision is the percentage of true positive values from positive data (Equation 3). Recall is the percentage of true positive values from the predicted results (Equation 4). F1-Score is the average of precision and recall values.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

$$F1 - Score = \frac{TP}{TP + FN} \tag{5}$$

IV. RESULTS AND DISCUSSION

At this stage, we conduct 3 testing scenarios. In the first scenario, we compare the results obtained from using Lemmatization and without Lemmatization in preprocessing. In the second scenario, we compare the results of TF-IDF using Bigram and Trigram. In the third scenario, we compare the results of using and without Hyperparameters in a Random Forest.

1) Scenario 1: Comparison between Lemmatization and Without Lemmatization: In this test scenario, the results were obtained by comparing Lemmatization and Without Lemmatization using TF-IDF Bigram, and Random Forest without Hyperparameters.

TABLE VII
RESULT OF SCENARIO 1

Confusion Matrix	Avg Precision	Avg Recall	Avg F1-Score
Lemmatization	71%	60%	57%
Without Lemmatization	66%	57%	53%

From the Table VII results obtained, the scenario that uses Lemmatization achieves better results compared to the scenario without Lemmatization. This is because the average score is obtained due to the words that have been transformed into their base form through the process of Lemmatization according to their semantic grouping, making it easier for machine learning to understand compared to not using Lemmatization, which involves many forms of words learned by the machine.

2) Scenario 2: Comparison between TF-IDF Bigram and Trigram: In this test scenario, the results obtained from comparing TF-IDF Bigram and Trigram using Lemmatization preprocessing, and Random Forest without Hyperparameters.

From the results obtained in Table VII, using Bigram and Trigram, the average scores for Precision, Recall, and F1-Score are better in Bigram compared to Trigram. This is because Trigram has more features, around 28,910 features, which directly influences Bigram, which only has about 24,453 features. However, the abundance of features can also lead to overfitting, which may deteriorate the performance of the model. The

accuracy for bigram token is high because the dataset contains many similar terms in these token types, resulting in relatively high TF-IDF values [21].

TABLE VIII
RESULT OF SCENARIO 2

Confusion Matrix	Avg Precision	Avg Recall	Avg F1-Score
Bigram	65%	57%	53%
Trigram	46%	50%	38%

3) Scenario 3: Comparison between Random Forest with Hyperparameters and without Hyperparameters: In this test scenario, the results were obtained by comparing hyperparameters in a Random Forest using Lemmatization preprocessing and TF-IDF Bigram. The parameter used is only Max_depth, which represents the maximum depth of the trees to be created, with the default value being none [20].

TABLE IX
RESULT OF SCENARIO 3

Confusion Matrix	Avg Precision	Avg Recall	Avg F1-Score
Without Hyper Parameter	71%	60%	57%
660	69%	58%	55%
665	**71%**	**63%**	**62%**
670	64%	56%	51%
675	72%	59%	56%
680	68%	60%	59%

From the results obtained in Table IX, it is found that using hyperparameters, particularly with Max_depth = 665, yields the most optimal result. The best result achieves an F1-score of 62%, which is better than the results obtained by other tuning values.

4) Discussion: Based on the testing scenarios conducted in machine learning for sentiment analysis on data obtained from the Steam website, particularly the reviews of the game Cyberpunk 2077 in English, after undergoing preprocessing (punctual removal, case folding, stopword removal, tokenizing, without Lemmatization), utilizing the TF-IDF Bigram method, and using Hyperparameters with Max_depth = 665, an average F1-Score of around 62% was achieved. This F1-Score value is better than the average F1-Score results obtained from other methods such as without Lemmatization, TF-IDF Trigram, and without Hyperparameters Tunning.

The reason why the performance results of the model using Lemmatization, TF-IDF Bigram, and Hyperparameters are better is that the number of word features learned by the machine learning model is not too excessive, thereby avoiding overfitting. Additionally, words that have been transformed and grouped into their base forms according to their meanings enable the machine to recognize the dataset used in this testing. The tuning of Hyperparameters also plays a crucial role in optimizing the performance of the machine learning model.

This study can be utilized by game developers to accurately assess comments. Negative comments can serve as critiques to improve game quality, while positive comments can inspire developers to create even better games in the future.

V. CONCLUSION

This research can create a good model to be applied on platforms for end-users or developers. However, we see several aspects that can be improved in the structure of this model. From the error checking conducted by the machine learning model, the errors obtained include meaningful words such as 'great' and 'good' in negative-labeled reviews causing the model to predict them as positive. While 'bad' and 'bug' in positive reviews lead to negative predictions. Classification errors also occur when reviews are too long or short, and some words lack

meaning for prediction, such as 'yaaaaah'. The machine learning model created still struggles to predict reviews that are sarcastic or have meanings contrary to the given labels.

Suggestions for further research include trying other Hyperparameters that may improve the results, increasing the number of datasets tested to add dataset variety, and possibly using feature selection methods to achieve even better results. This study can be utilized by game developers to accurately assess comments. Negative comments can serve as critiques to improve game quality, while positive comments can inspire developers to create even better games in the future.

VI. DATA AND COMPUTER PROGRAM AVAILABILITY

The data and program used in this paper can be accessed on the Telkom University Dataverse at the following site: https://dataverse.telkomuniversity.ac.id/dataset.xhtml?persistentId=doi:10.34820/FK2/9SH7GB.

VII. REFERENCES

[1] Kriz, W. C. (2020). Gaming in the Time of COVID-19. Simulation & Gaming, 51(4), 403-410.

[2] Lin, D., Bezemer, C. P., & Hassan, A. E. (2018). An empirical study of early access games on the Steam platform. Empirical Software Engineering, 23(2), 771-799.

[3] Huang, J. (2018). What can we recommend to game players?-Implementing a system of analyzing game reviews (Master's thesis).

[4] Ahmad, M., Aftab, S., Muhammad, S. S., & Ahmad, S. (2017). Machine learning techniques for sentiment analysis: A review. Int. J. Multidiscip. Sci. Eng, 8(3), 27.

[5] Zuo, Z. (2018). Sentiment analysis of steam review datasets using naive bayes and decision tree classifier.

[6] Hartmann, Jochen, et al. "Comparing automated text classification methods." International Journal of Research in Marketing 36.1 (2019): 20-38.

[7] Ahuja, R., Chug, A., Kohli, S., Gupta, S., & Ahuja, P. (2019). The impact of features extraction on the sentiment analysis. Procedia Computer Science, 152, 341-348.

[8] Febrianta, M. Y., Widiyanesti, S., & Ramadhan, S. R. (2021). Analisis Ulasan Indie Video Game Lokal Pada Steam Menggunakan Sentiment Analysis Dengan Algoritma Naive Bayes Classifier Dan Lda-based Topic Modeling. eProceedings of Management, 8(4).

[9] Tan, J. Y., Chow, A. S. K., & Tan, C. W. (2021, October). Sentiment Analysis on Game Reviews: a Comparative study of Machine Learning Approaches. In International Conference on Digital Transformation and Applications (ICDXA) (Vol. 25, p. 26).

[10] Eberhard, L., Kasper, P., Koncar, P., & Gütl, C. (2018, October). Investigating helpfulness of video game reviews on the steam platform. In 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS) (pp. 43-50). IEEE.

[11] Khomsah, S. (2021). Sentiment analysis on YouTube comments using word2vec and random forest. Telematika: Jurnal Informatika dan Teknologi Informasi, 18(1), 61-72.

[12] Ramadhan, N. G., Wibowo, M., Mohd Rosely, N. F. L., & Quix, C. (2022). Opinion mining indonesian presidential election on twitter data based on decision tree method. JURNAL INFOTEL, 14(4), 243-248. https://doi.org/10.20895/infotel.v14i4.832

[13] Safruddin, A., Hermawan, A., & Wibowo, A. P. (2020). Implementation of Backpropagation Neural Network in Sentiment Analysis on Twitter To Public Figures. Compiler, 9(2), 101-108.

[14] Purbolaksono, M. D. (2024). Steam Game Review Dataset. Telkom University Dataverse. https://doi.org/10.34820/FK2/9SH7GB

[15] Hakim, A. A., Erwin, A., Eng, K. I., Galinium, M., & Muliady, W. (2014, October). Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach. In 2014 6th international conference on information technology and electrical engineering (ICITEE) (pp. 1-4). IEEE.

[16] Putri, N. F., Al Faraby, S., & Dwifebri, M. (2021). Analisis Sentimen Pada Produk Kecantikan Dari Ulasan Female Daily Menggunakan Information Gain Dan Svm Classifier. eProceedings of Engineering, 8(5).

[17] Kulkarni, V. Y., & Sinha, P. K. (2012, July). Pruning of random forest classifiers: A survey and future directions. In 2012 International Conference on Data Science & Engineering (ICDSE) (pp. 64-68). IEEE.

[18] Yang, BS., Di, X. & Han, T. (2018). Random forests classifier for machine fault diagnosis. Journal Mechanical Science Technology 22, 1716–1725. https://doi.org/10.1007/s12206-008-0603-6

[19] Ting, K.M. (2011). Confusion Matrix. In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_157.

[20] Daviran, M., Maghsoudi, A., Ghezelbash, R., & Pradhan, B. (2021). A new strategy for spatial predictive mapping of mineral prospectivity: Automated hyperparameter tuning of random forest approach. Computers & Geosciences, 148, 104688.

[21] Widiyaningtyas, T., Zaeni, I. A. E., & Al Farisi, R. (2019, October). Sentiment Analysis Of Hotel Review Using N-Gram And Naive Bayes Methods. In *2019 Fourth International Conference on Informatics and Computing (ICIC)* (pp. 1-5). IEEE.