Bulletin of Applied Mathematics and Mathematics Education
Volume 4, Number 2, October 2024, pp. 61-74
https://dx.doi.org/10.12928/bamme.v4i2.11104
e-ISSN 2776-1029, p-ISSN 2776-1002

Optimization of feature selection on semi-supervised data

Dian Eka Wijayanti[1], Sintia Afriyani[1*], Sugiyarto Surono[1], Deshinta Arrova Dewi[2]

[1]Universitas Ahmad Dahlan, Jalan Ahmad Yani, Tamanan, Banguntapan, Bantul, DIY 55711, Indonesia
[2]INTI International University, Persiaran Perdana BBN Putra Nilai, 71800 Nilai, Negeri Sembilan, Malaysia

*Corresponding e-mail: sintia2000015036@webmail.uad.ac.id

ARTICLE INFO

ABSTRACT

Article History
Received 10 July 2024
Revised 31 July 2024
Accepted 22 August 2024

Keywords
Classification
Feature selection
Particle Swarm Optimization
Semi-supervised data

How to cite this article:
Wiyayanti, D.E., Afriyani, S., Surono, S., Dewi, D. A. (2024). Optimization of feature selection on semi-supervised data. *Bulletin of Applied Mathematics and Mathematics Education*, *4*(2), 61-74.

This research explores feature selection optimization in semi-supervised text data by utilizing the technique of dividing data into training and testing sets and implementing pseudo labeling. Proportions of data division, namely 70:30, 80:20, and 90:10, were used as experiments, employing TF-IDF weighting and PSO feature selection. Pseudo labeling was applied by assigning positive, negative, and neutral labels to the training data to enrich information in the classification model during the testing phase. The research results indicate that the linear SVM model achieved the highest accuracy with a 90:10 data division proportion with a value of 0.9051, followed by Random Forest, which had an accuracy of 0.9254. Although RBF SVM and Poly SVM yielded good results, KNN showed lower performance. These findings emphasize the importance of feature selection strategies and the use of pseudo labeling to enhance the performance of classification models in semi-supervised text data, offering potential applications across various domains that rely on semi-supervised text analysis.

Introduction

At present, big data requires the use of mathematical techniques to handle increasingly complex problems and provide efficient solutions (Ning & You, 2019). Big data encompasses various formats and sources, ranging from text, images, and audio to sensor data, all of which require complex analysis to uncover useful patterns and insights (Adnan & Akbar, 2019). One type of data used in machine learning is semi-supervised data (Van Engelen & Hoos, 2020). Semi-supervised data involves a combination of labeled and unlabeled data to train models, allowing the system to gain a deeper and more accurate understanding of patterns within the data (Ouali et al., 2020).

In the digital era, data is rapidly expanding, especially textual data originating from various sources such as social media, websites, and other online platforms (Ghani et al., 2019). Sentiment analysis of textual data has become a primary focus of research because insights derived from user sentiments are highly valuable for businesses, marketing, and other decision-making processes (Birjali et al., 2021). However, textual data is often challenging to analyze directly due to its unstructured nature (Adnan & Akbar, 2019). Therefore, an efficient approach is required to transform textual data into numerical representations that can be utilized by machine learning algorithms (Jan et al., 2019).

Natural Language Processing (NLP) towards feature extraction is a crucial step that allows text to be converted into a numerical form understandable by machine learning models (Wang, Su, & Yu, 2020). NLP utilizes various feature extraction techniques, such as word embeddings, TF-IDF, and N-grams, to transform words and phrases in text into numerical representations (Ahuja et al., 2019). Feature extraction plays a central role in enabling NLP models to perform tasks such as text classification and sentiment analysis (Abdi et al., 2019). This paves the way for the development of effective and accurate artificial intelligence in understanding human language (Ahmed et al., 2020).

Feature selection in textual data is a crucial process in the development of effective machine learning models (Zebari et al., 2020). The right strategy should consider exploratory data analysis to understand word distribution, the prevalence of frequent and rare words, as well as the use of n-grams and weighting schemes such as TF-IDF (Sriram, 2020). Custom feature engineering and automatic feature selection techniques can also be employed to create an optimal feature set based on the goals of the respective NLP task (Anuradha et al., 2022). Careful feature selection plays a key role in enhancing the accuracy and performance of models in natural language processing (Mo et al., 2020).

The results of the study by Asri et al. (2023) indicate that Particle Swarm Optimization (PSO) outperforms other algorithms such as Ant Colony Optimization (ACO) and Genetic Algorithm (GA) in sentiment analysis on drug reviews. PSO is capable of generating a high-quality feature subset that enhances the accuracy of sentiment analysis models. In this research, PSO demonstrated the highest performance levels, with an average precision of 49.3%, recall of 73.6%, F-score of 59%, and accuracy of 57.2%.

Therefore, to enhance sentiment analysis performance, the semi-supervised data method becomes a relevant solution. This method leverages unlabeled data along with labeled data to improve precision, recall, F1-score, and accuracy in sentiment classification. Semi-supervised data, consisting of both labeled and unlabeled data, provides an opportunity to harness information contained in unlabeled data to enhance classification. Pseudo-labeling is one of the semi-supervised techniques used, involving assigning pseudo-labels to unlabeled data based on predictions from a classification model trained with labeled data. Thus, unlabeled data can be iteratively used to train the classification model, improving accuracy.

Method

This research will propose a new method for classifying semi-supervised data. The method will be tested using Google Colaboratory, employing Particle Swarm Optimization (PSO) for feature selection on semi-supervised data, alongside various classification algorithms. In the classification stage, the model will be evaluated using six model validation techniques, namely SVM-linear, SVM-RBF, SVM-Poly, Random Forest, and KNN, to obtain maximum accuracy values.

For data labeling, the pseudo-labeling technique will be utilized, where the training data is manually annotated, and the test data is labeled using the model trained on the training data. This approach is necessary because the data is in textual form and is aimed at sentiment analysis, with both labeled and unlabeled instances.

This design will process the training and testing data to evaluate the employed algorithmic methods. The process consists of five stages: data preprocessing, feature extraction using TF-IDF, feature selection with Particle Swarm Optimization (PSO), the semi-supervised process with pseudo-labeling, and model classification. The stages can be observed in Figure 1.

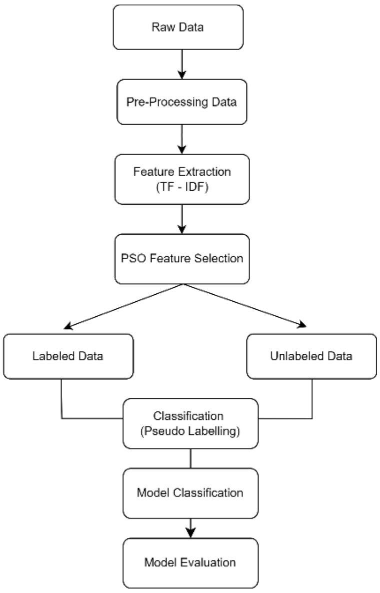


Figure 1. Stages of the proposed research model

Data collection

In this paper, the driver dataset encompasses aspects of sentiment related to the feelings and emotional states of drivers during their journeys, indicated through vehicle telematics, driver presence sensors, or direct inputs such as voice notes or surveys. Sentiment analysis involves a deeper understanding of driver satisfaction. The dataset is semi-supervised, with some data manually labeled to train the model and some unlabeled data to leverage semi-supervised learning techniques. Data collection involves monitoring devices inside the vehicle, providing a comprehensive overview of driver behavior and the environment. The analysis of the dataset is expected to provide insights into positive, negative, or neutral comments (Zepf et al., 2020).

Preprocessing data

This stage will involve sub processes including transform case, tokenization, filter token (by length), stop words, and stemming (porter stemming). The end result is a collection of cleaned or unique words (Chai, 2023). The data preprocessing stage is illustrated in the scheme shown in Figure 2.

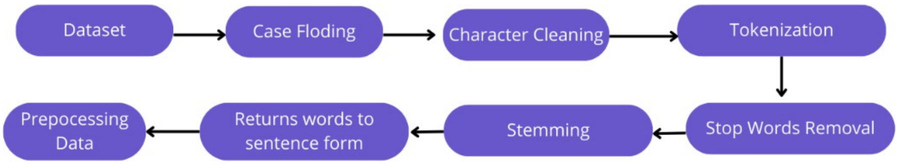


Figure 2. Data preprocessing stages

Feature extraction

After completing all preprocessing stages, the next step is to process the output and then extract features using the TF-IDF (Term Frequency-Inverse Document Frequency) method. This method calculates the weight of each word in the tweet data based on its frequency of occurrence in the document and in the document collection (Addiga & Bagui, 2022). These weights represent the importance of the word in both the document and the document collection. The formula for TF-IDF is as follows.

$$TF\ (t,d)\ =\ \frac{Number\ of\ occurrences\ of\ term\ \boldsymbol{t}\ in\ dokument\ \boldsymbol{d}}{Number\ total\ terms\ in\ document\ \boldsymbol{d}} \qquad (1)$$

$$+C_2\ x\ r_2 x(gbest_{ij} - x_{ij}(t)).$$

IDF is the inverse document frequency of word i, which is calculated using the following formula,

$$IDF\ (T, D)\ =\ log\ (\frac{Number\ of\ documents\ in\ corpus\ \boldsymbol{D}}{Number\ total\ terms\ in\ document\ \boldsymbol{t}}). \qquad (2)$$

So,

$$TF - IDF = TF(t, d)\ X\ IDF(T, D). \qquad (3)$$

PSO feature selection

The data, after feature extraction, undergoes feature selection using the Particle Swarm Optimization (PSO) method. This method is an optimization technique inspired by the behavior of flocks of birds or schools of fish. It aims to find the best features that can enhance classification performance by exploring the search space randomly and adjusting particle positions based on speed and fitness values (Sengupta, Basak, & Peters, 2018). In PSO, particles search and determine which tokens are the most suitable to be used as features. By selecting these tokens as the best features, the dimensionality of the document is reduced (Abualigah, Khader, & Hanandeh, 2018). However, the content contained in the document is preserved as the selected features highly represent the document. Figure 3 is the algorithm for Particle Swarm Optimisation (PSO).

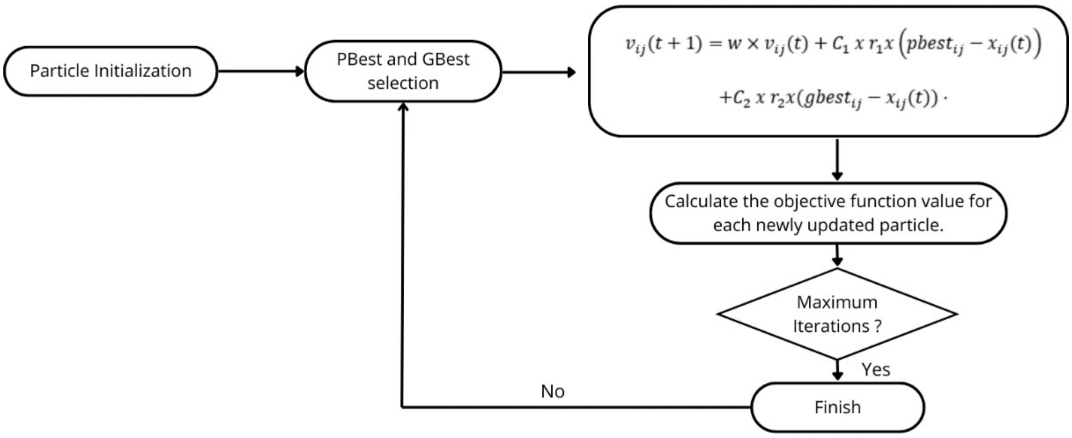

Particle Initialization
PBest and GBest selection
$v_{ij}(t+1) = w \times v_{ij}(t) + C_1\, x\, r_1 x \left(pbest_{ij} - x_{ij}(t)\right)$
$+C_2\, x\, r_2 x(gbest_{ij} - x_{ij}(t)) \cdot$
Calculate the objective function value for each newly updated particle.
Maximum Iterations ?
Yes
No
Finish

Figure 3. PSO Feature Selection Stages

Figure 3 illustrates that each particle in PSO has a position and velocity representing a solution to the objective function. Particle positions and velocities are randomly initialized at the beginning of the iteration. Subsequently, particle positions and velocities are iteratively updated using the following formulas,

$$v_{ij}(t+1) = w \times v_{ij}(t) + C_1\, x\, r_1 x\ (pbest_{ij} - x_{ij}(t)) \qquad (4)$$
$$+C_2\ x\ r_2 x(gbest_{ij} - x_{ij}(t)) \cdot$$

$$x_{ij}(t+1) =\ x_{ij}(t) +\ v_{ij}(t+1) \qquad (5)$$

where:

- $v_{ij}(t+1)$ is the velocity of particle i in dimension j iteration $t+1$;
- $v_{ij}(t)$ is the position of particle i in dimension j at iteration t;
- $pbest_{ij}$ is the best position of particle i in dimension j;
- $pbest_{ij}$ is the best position of particle i in dimension j so far;
- $gbest_{ij}$ is the best position among all particles in dimension j so far;

- w is the inertia factor controlling exploration and exploitation;
- $C_1\ x\ r_1$ and $C_2\ x\ r_2$ are the cognitive and social factors controlling the influence of Pbest and Gbest;
- $C_1\ x\ r_1$ and $C_2\ x\ r_2$ are random numbers between 0 and 1.

Pseudo labelling

The feature-selected data is then divided into two parts: the training data and the test data. The training data is the set that will be manually labeled as positive, negative, or neutral (Nahid et al., 2022). The test data is the set without labels, and labels will be assigned using the trained model from the training data. The training data is used to train the classification model, while the test data is used for automatic labeling using the Pseudo-labeling method (Lee, Gan, Tan, & Abdullah, 2019). The semi-supervised process is depicted in the scheme shown in Figure 4.

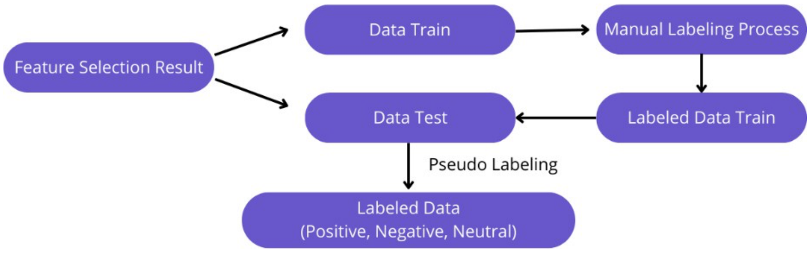


Figure 4. Semi-supervised process

Confusion matrix

The main stage of this research is classification, utilizing the Support Vector Machine (SVM) algorithm. In the classification of semi-supervised text data using Support Vector Machine (SVM) (Liu, Xu, & Li, 2018), the initial steps involve selecting an appropriate kernel, such as linear or radial basis function (RBF), to handle complex text structures (Calma, Reitmaier, & Sick, 2018). The labeled text data is then divided into training and testing sets. SVM trains the model by utilizing the training set, searching for the best hyperplane to separate the recognized classes by maximizing the margin between support vectors. Figure 5 illustrates the stage of the confusion matrix used to apply the previously built model.

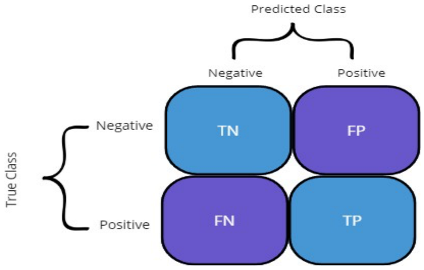


Figure 5. Confusion matrix stages

By employing semi-supervised techniques, SVM can optimally leverage unlabeled text data, enhancing the model's understanding of text variations and diversity. After training, SVM parameters can be adjusted, and the model is evaluated using the test set to measure its performance in classifying previously unseen text data.

Explanation:

- TP is True Positive (represents positive data predicted correctly)
- TN is True Negative (represents negative data predicted correctly)

- FP is False Positive (represents negative data predicted as positive)
- FN is False Negative (represents positive data predicted as negative)

To measure the accuracy of the model's performance, five criteria are used,

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$recall = \frac{TP}{TP + FN} \tag{7}$$

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$F1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{9}$$

The outcome of this stage is the values of precision, recall, and, of course, accuracy. These values will be compared to determine which model is the best. All validation results will yield the model and performance calculation results. Subsequently, the results will be presented in the form of a confusion matrix table, and simultaneously, the created model will be saved.

Results and discussion

The research yields outputs that can be analyzed to obtain valuable information. Here is an elaboration on the results of the conducted research.

Dataset

The data used is textual data obtained from Kaggle (https://www.kaggle.com/code/sasakitetsuya/semi-supervised-classification-on-a-text-dataset) related to driver comments. Table 1 presents the content of a collection of passenger comments regarding online drivers. The content of the dataset is as follows.

Table 1. Dataset

No.	Text
1.	Such an easy app to use
2.	The drivers and the services have been exceptional since ever
3.	All rides have been enjoyable.
4.	Driver very knew where I was
⋮	⋮
5896.	This app stinks too many interruptions and upgrades no good doesn't display whole album list wouldn't recommend.

The problem to be addressed is the classification of textual data using semi-supervised techniques with the highest possible accuracy. The dataset used is quite substantial, comprising 5,896 documents.

Preprocessing data

The stage that follows document collection is preprocessing. It includes the following steps:

(1) *Cleaning Text;* This stage is executed with a package to perform more complex text deletion operations using regular expressions (regex) and strings for removal, such as removing punctuation marks. In the case of token removal, there is no specific mathematical formula used; therefore, Python operations are employed, as explained in Table 2.

10.12928/bamme.v4i2.11104

Table 2. Data cleaning stage

Stage	Syntax	Function
Special Character Cleaning	• re.sub(r'[^\w\s]', text)	Omit special characters, such as punctuation marks, symbols, or non- alphanumeric characters.
Cleaning Stopwords	• tokens : word_tokenize(text) • stop_words : set(stopwords.words('indonesian')) • filtered_tokens : [word for word in tokens if word.lower() not in stop_words]	Removes common stop words that don't provide much information. Using the nltk library.

(2) Tokenization; Tokenization is used to break down text into smaller units called tokens. In this research, a word-based tokenization method is employed, considering punctuation marks. This allows the separation of words based on spaces or punctuation marks as word separators, such as spaces, periods, commas, and others. This approach results in tokens representing individual words in the text. Tokenization is depicted as in the following mathematical equation.

$$T = (t_1, t_2, \ldots, t_n) \qquad (10)$$

Description:

- T is set of tokens t
- t_i is the i token in the collection T
- n is total number of tokens

Stemming

The stemming process can help improve effectiveness in natural language processing. It works by transforming tokens into their base word forms. Its main purpose is to reduce different words that have the same root into the same form. By using the same base word, the machine can understand that these words are closely related semantically. In the analysis results, the number of words in the document decreases after stemming is applied. This helps reduce the complexity of the text data. The reduction in the number of words after stemming can decrease the dimensionality of the data and enhance processing efficiency.

Stop words removal

In this stage, a dictionary-based method is used for removing stop words, consisting of more than 100 common stop words in both English and Indonesian. The data processed in the previous stage will be analyzed by eliminating stop words listed in the dictionary in Table 2, as illustrated in the following mathematical equation.

$$RemoveStopWords(T, S) \qquad (11)$$

Explanation:

- T is set of tokens
- S is list of stop words
- T is set of tokens after removal

The removal of stop words also helps to enhance the focus on more important, cleaner, concise, and topic-relevant keywords, thereby improving the performance of subsequent text analysis methods. Table 3 shows the results of data processing, including text cleaning, tokenization, stemming, and stop word removal. The results of text preprocessing are as follows.

Table 3. Text pre-processing

No.	Text	Pre-processing Text
1.	Such an easy app to use	easy app use
2.	The drivers and the services have been exceptional since ever	driver service except since ever
3.	All rides have been enjoyable.	ride enjoy
4.	Driver very knew where I was	driver knew
⋮	⋮	⋮
5896.	This app stinks too many interruptions and upgrades no good doesn't display whole album list wouldn't recommend	app stink many interrupt upgrade good display whole album list.

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF (Term Frequency-Inverse Document Frequency) is a method used to assess the importance of a word in a document within a collection of documents or a text corpus. This method is valuable in natural language processing, information retrieval, and text pattern mining. Table 4 displays the results of TF-IDF weighting, indicating the significance levels of keywords in each document, providing a more profound understanding of the essence and differences between texts.

Table 4. TF-IDF weighting result

No.	Pre-processing Text	App	Driver	Ride	...	Stink	Upgrade	Zyada
1.	easi app use	0.409832	0.0	0.0	...	0.0	0.0	0.0
2.	driver servic	0.0	0.407279	0.0	...	0.0	0.0	0.0
3.	ride enjoy	0.0	0.0	0.727181	...	0.0	0.0	0.0
4.	driver knew	0.0	0.558512	0.0	...	0.0	0.0	0.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
5896.	app stink mani interrupt upgrad good display whole album list	0.120880	0.0	0.0	...	0.40805	0.301312	0.0

TF-IDF weighting offers valuable insights to identify the focus and main theme of each document in the dataset, facilitating a better understanding of the crucial elements that emerge in their respective contexts.

Feature selection with Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is an optimization algorithm inspired by the collective behavior of birds searching for food. PSO can be used to find the optimal solution to an objective function by adjusting the position and velocity of a group of particles representing potential solutions.

10.12928/bamme.v4i2.11104

The objective function used in PSO can vary depending on the optimization goal. In the context of feature selection, the objective function can be accuracy of classification, classification error, or a combination of both. Table 5 displays the results of feature selection using the Particle Swarm Optimization (PSO) method, showing the Term Frequency-Inverse Document Frequency (TF-IDF) values for various keywords identified as important features in a document or text corpus. Based on Equations (1) and (2), the results are obtained as follows.

Table 5. PSO feature selection result

No	Text feature selection	Value of TF-IDF
1.	aaichi	0.4149941237872556
2.	aap	0.8898267339828092
3.	ab	0.5829472684191462
4.	abandon	0.9945317878849859
5.	abosult	0.2970729013669
⋮	⋮	⋮
3526.	zyada	0.28219767279584956.

Each row in the table represents one keyword that has undergone the selection process using the PSO algorithm. For example, the word "aap" has a TF-IDF value of 0.8898, indicating its high significance in the context of the analyzed document or corpus. On the other hand, the word "absolute" has a lower TF-IDF value, namely 0.2971, possibly indicating lower relevance or less significant frequency in the text. This table provides insights into keywords considered important in text analysis based on the PSO method and their TF-IDF values.

Pseudo labelling

This research examines the impact of using pseudo labeling, considering the results of feature selection on the performance of machine learning models. Pseudo labeling is a method that utilizes a pre-trained model to label unlabeled data, integrated with the results of feature selection to enhance the accuracy and efficiency of the model. In Table 6 the feature selection process is conducted to identify the most informative or relevant feature subset from the dataset One of the data train-test splits is 70:30, and here are the results.

Table 6. Pseudo labeling result

No.	Text Pre-processing	Predict Label
1.	Thank pubg mobil creator team keep go	Positive
2.	New updat regard chead show onlin redund	Neutral
3.	Even make playlist shuffle skip songs	Negative
⋮	⋮	⋮
1769.	Like music hear music like friend dueg pandem	Positive

The experimental results indicate that the combination of pseudo-labeling and feature selection can provide a significant improvement in the predictive performance of the model. This result highlights the optimization of the model through the integration of pseudo-labeling with feature selection techniques, which can be applied in various contexts and enhance our understanding of strategies to improve machine learning performance.

Confusion matrix

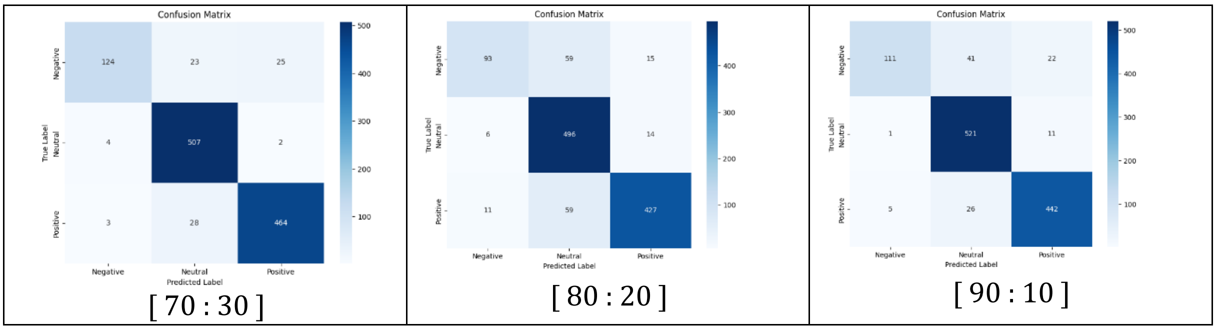
Analyzing the confusion matrix in the classification results provides a comprehensive overview of the model's performance and accuracy evaluation. The confusion matrix shows how well the model can correctly predict positive and negative classes and how often classification errors occur. By analyzing the values in Equations (1), (2), (3), and (4), we can calculate evaluation metrics such as accuracy, precision, recall, and F1-score, providing deep insights into the classification model's performance and aiding in identifying areas for improvement.

Table 7 presents the performance evaluation results of several classification models used for a specific task with variations in the proportion of training and testing data. Here are the evaluation results of the classification model's performance.

Table 7. Classification model performance evaluation results

Classification Model	Share proportion	Precision	Recall	F1-score	Accuracy
Linier SVM	70:30	0.8988	0.887	0.8852	0.8873
RBF SVM		0.8692	0.8449	0.8388	0.8449
Poly SVM		0.7599	0.6627	0.6276	0.6627
Random Forest		0.8923	0.8864	0.8850	0.8864
KNN		0.7077	0.5814	0.5322	0.5814
Linier SVM	80:20	0.8760	0.8619	0.8580	0.8619
RBF SVM		0.8613	0.8407	0.8350	0.8407
Poly SVM		0.7632	0.6839	0.6487	0.6839
Random Forest		0.8750	0.8712	0.8685	0.8712
KNN		0.6804	0.6119	0.5752	0.6119
Linier SVM	90:10	0.9110	0.951	0.9009	0.9051
RBF SVM		0.8828	0.8644	0.8551	0.8644
Poly SVM		0.7853	0.7085	0.6774	0.7085
Random Forest		0.9295	0.9254	0.9225	0.9254
KNN		0.7193	0.5881	0.5459	0.5881.

Five types of evaluated models include Linear SVM, RBF SVM, Polynomial SVM, Random Forest, and K-Nearest Neighbors (KNN). The proportions of training and testing data are divided into 70:30, 80:20, and 90:10. Model evaluation is performed using standard classification metrics, namely precision, recall, F1-score, and accuracy. From these results, it can be concluded that Linear SVM with a proportion of 90:10 provides the highest performance with a precision of 0.9110, recall of 0.951, F1-score of 0.9009, and accuracy of 0.9051. The Confusion matrix results are shown in Figure 6.



[70 : 30]

[80 : 20]

[90 : 10]

Figure 6. Confusion matrix

Figure 6 shows the results of the confusion matrix in evaluating the classification model's performance, considering four main metrics: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). In the context of positive, negative, and neutral labels, TP represents the number of data correctly classified as positive predictions, while TN indicates data correctly classified as negative. FP and FN represent errors in positive and negative classifications. For neutral labels, TP and TN can reflect the model's performance in identifying data as neutral. The confusion matrix helps provide a more detailed understanding of the model's ability to differentiate between different classes.

Conclusion

Based on the accuracy results for three different data train-test split proportions (70:30, 80:20, 90:10), several conclusions can be drawn regarding the performance of the classification models used. Figure 7 displays the accuracy results of the classification model use.

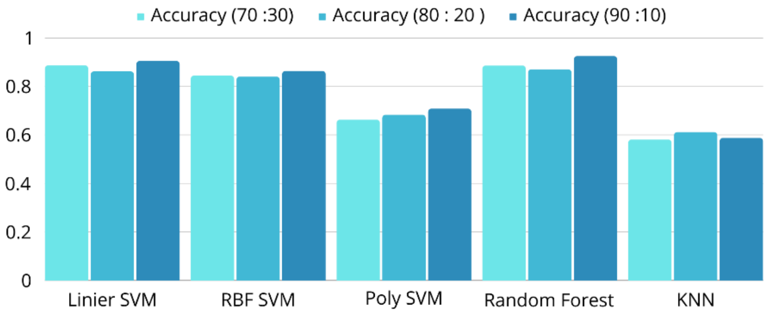


Figure 7. Model classification accuracy result chart

Based on Figure 7, Linear SVM shows good consistency across all three proportions, with the highest accuracy at the 90:10 data split being 0.9051. Random Forest also demonstrates strong performance, particularly at the 90:10 data split with an accuracy of 0.9254. On the other hand, the Polynomial SVM model exhibits lower accuracy compared to other models, especially at the 70:30 proportion, which is 0.6627. The present findings could enrich the previous studies in the classification model performance (Abdi et al., 2019; Mo et al., 2020; Asri et al., 2023). However, it's essential to consider other factors such as precision, recall, and F1-score to gain a more comprehensive understanding of the model's performance in classifying data. Additionally, it's crucial to note that model selection should be tailored to the data characteristics and the specific objectives of the classification task.

Acknowledgement

This study was funded by from Universitas Ahmad Dahlan the fiscal year 2023 with the contract number: PD- 104/SP3/LPPM-UAD/XII/2023.

References

Abdi, A., Shamsuddin, S. M., Hasan, S., & Piran, J. (2019). Deep learning-based sentiment classification of evaluative text based on multi-feature fusion. *Information Processing & Management, 56*(4), 1245-1259.

Abualigah, L. M., Khader, A. T., & Hanandeh, E. S. (2018). A new feature selection method to improve the document clustering using particle swarm optimization algorithm. *Journal of Computational Science, 25*, 456-466.

Adnan, K., & Akbar, R. (2019). An analytical study of information extraction from unstructured and multidimensional big data. *Journal of Big Data, 6*(1), 1-38.

Addiga, A., & Bagui, S. (2022). Sentiment analysis on Twitter data using term frequency- inverse document frequency. *Journal of Computer and Communications, 10*(8), 117-128.

Ahmed, Z., Mohamed, K., Zeeshan, S., & Dong, X. (2020). Artificial intelligence with multi-functional machine learning platform development for better healthcare and precision medicine. *Database, 2020*, baaa010.

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.

Anuradha, T., Tigadi, A., Ravikumar, M., Nalajala, P., Hemavathi, S., & Dash, M. (2022). Feature extraction and representation learning via deep neural network. In *Computer Networks, Big Data and IoT: Proceedings of ICCBI 2021* (pp. 551-564). Springer Nature Singapore.

Asri, A. M., Ahmad, S. R., & Yusop, N. M. M. (2023). Feature selection using particle swarm optimization for sentiment analysis of drug reviews. *International Journal of Advanced Computer Science and Applications, 14*(5).

Birjali, M., Kasri, M., & Beni-Hssane, A. (2021). A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems, 226*, 107134.

Calma, A., Reitmaier, T., & Sick, B. (2018). Semi-supervised active learning for support vector machines: A novel approach that exploits structure information in data. *Information Sciences, 456*, 13-33.

Chai, C. P. (2023). Comparison of text preprocessing methods. *Natural Language Engineering, 29*(3), 509-553.

Ghani, N. A., Hamid, S., Hashem, I. A. T., & Ahmed, E. (2019). Social media big data analytics: A survey. *Computers in Human Behavior, 101*, 417-428.

Jan, B., Farman, H., Khan, M., Imran, M., Islam, I. U., Ahmad, A. & Jeon, G. (2019). Deep learning in big data analytics: A comparative study. *Computers & Electrical Engineering, 75*, 275- 287.

Lee, V. L. S., Gan, K. H., Tan, T. P., & Abdullah, R. (2019). Semi-supervised learning for sentiment classification using small number of labeled data. *Procedia Computer Science, 161*, 577- 584.

Liu, Y., Xu, Z., & Li, C. (2018). Online semi-supervised support vector machine. *Information Sciences, 439*, 125-141.

Mo, Y., Zhao, D., Du, J., Syal, M., Aziz, A., & Li, H. (2020). Automated staff assignment for building maintenance using natural language processing. *Automation in Construction, 113*, 103150.

Nahid, A. A., Sikder, N., Abid, M. H., Toma, R. N., Talin, I. A., & Lasker, E. A. (2022). Home occupancy classification using machine learning techniques along with feature selection. *International Journal of Engineering and Manufacturing, 12*(3), 38.

Ning, C., & You, F. (2019). Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming. *Computers & Chemical Engineering, 125*, 434-448.

Ouali, Y., Hudelot, C., & Tami, M. (2020). An overview of deep semi-supervised learning. *arXiv preprint*. arXiv:2006.05278.

10.12928/bamme.v4i2.11104

Sengupta, S., Basak, S., & Peters, R. A. (2018). Particle Swarm Optimization: A survey of historical and recent developments with hybridization perspectives. *Machine Learning and Knowledge Extraction, 1*(1), 157-191.

Sriram, S. (2020). An evaluation of text representation techniques for fake news detection using: TF-IDF, word embeddings, sentence embeddings with linear support vector machine. *Dissertation*. TU Dublin.

Van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning, 109*(2), 373-440.

Wang, D., Su, J., & Yu, H. (2020). Feature extraction and analysis of natural language processing for deep learning English language. *IEEE Access, 8*, 46335-46345.

Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D., & Saeed, J. (2020). A comprehensive review of dimensionality reduction technsiques for feature selection and feature extraction. *Journal of Applied Science and Technology Trends, 1*(2), 56-70.

Zepf, S., Hernandez, J., Schmitt, A., Minker, W., & Picard, R. W. (2020). Driver emotion recognition for intelligent vehicles: A survey. *ACM Computing Surveys (CSUR), 53*(3), 1-30.

This page is intentionally left blank.

10.12928/bamme.v4i2.11104