SHM PUBLISHER
SURYA HIJAU MANFAAT

Journal of Student Research Exploration

https://shmpublisher.com/index.php/josre
p-ISSN 2964-1691 | e-ISSN 2964-8246

Email spam detection: a comparison of svm and naive bayes using bayesian optimization and grid search parameters

Dzakky Budiman[1], Zayyan[2], Ainun Mardiana[3], Alfira Aulia Mahrani[4]
[1,2,3] Department of Informatics Engineering, Universitas Negeri Semarang, Indonesia
[4]Department of Informatics Engineering, Universitas Ahmad Dahlan, Indonesia

Article Info

Article history:

Received November 28, 2024
Revised January 30, 2024
Accepted January 31, 2024

Keywords:

Support vector machine
Naïve bayes
Grid search
Bayesian
Spam email

ABSTRACT

Spam emails are still a big problem, crowding out inboxes and annoying email users everywhere. SVM and Naive Bayes are frequently used algorithms that have demonstrated excellent performance in performing text classification, including spam detection. The purpose of this study is to evaluate the overall performance of SVM and Naive Bayes in the context of detecting spam emails using default parameters. This research utilizes Bayesian Optimization and Grid Search Parameters for both SVM and Naive Bayes models to help maximize the performance of the constructed models. This study uses a spam email dataset that has 2 sample groups, namely spam and ham. Of the three parameter selection methods that have been tested on the SVM Algorithm, Bayesian Optimization is a parameter tuning method that has the most satisfying results in accuracy, precision, recall, and f1 scores respectively with values of 98.5642%, 99.4048%, 89.

1. Introduction

Spam emails are still a big problem, crowding out inboxes and annoying email users everywhere [1]. To maintain the integrity and security of email communications, spam must be identified and successfully filtered. Spam emails are unsolicited, often irrelevant or malicious messages sent in bulk to a large number of recipients without their consent. These emails can contain various

[1] ***Corresponding Author:***
Dzakky Budiman,
Department of Informatics Engineering,
Universitas Negeri Semarang,
Sekaran, Gunungpati, Kota Semarang, Indonesia
Email: dzakkybudiman19@student.unnes.ac.id
DOI: https://doi.org/10.52465/josre.v2i1.260

forms of unwanted content, such as advertisements, scams, phishing attempts, malware, or other fraudulent activities.

The effects of spam emails can be significant and far-reaching. Firstly, they inundate email inboxes, making it difficult for users to find and attend to legitimate messages. This overcrowding can lead to decreased productivity as users spend time sorting through and deleting spam. In addition to being a nuisance to individual users, spam emails can also have broader societal impacts. They can contribute to the spread of misinformation, facilitate cybercrime, and undermine trust in online communication systems. Therefore, effective detection and filtering mechanisms are essential to mitigate the negative effects of spam and ensure the integrity and security of email communications. Therefore, spam email detection is an important issue to research.

SVM and Naive Bayes are frequently used algorithms that have demonstrated excellent performance in performing text classification [2], [3], including spam detection [4]. Despite the existing research on Naive Bayes and SVM classifiers for spam email detection, a thorough analysis of their performance and the effect of hyperparameter tuning still needs to be carried out [5]. Previous studies [6], [7] have not directly evaluated the performance of the Naive Bayes and SVM classifiers under various hyperparameter settings, specifically with Bayesian optimization and grid search methods, in email spam detection. This study compares the classifiers Naive Bayes and SVM for detecting spam emails in a systematic way, considering the impact of hyperparameter tuning methods such as Bayesian optimization and Grid Search. We hope to improve classifier performance and find the best parameter settings for each algorithm by experimenting with these different configurations through optimization.

Support Vector Machine (SVM) is a powerful and adaptable supervised learning technique that seeks to identify the best decision boundaries for classifying data points [4]. This is done by constructing hyperplanes that maximize the distance between various classes in a high-dimensional feature space. Several fields, including natural language processing and text categorization, have made effective use of SVM because of its proficiency in complex relationships and high-dimensional data [8]. In SVM, each email is represented as a point in a high-dimensional feature space, with each dimension representing a different feature [9]. A feature might, for example, indicate how often a certain word occurs in an email or whether a certain trait exists. The goal of SVM is to identify hyperplanes with as wide a distance between spam and non-spam emails as possible [10]. Margin is the distance between the closest data point of each class and the hyperplane. This margin is what SVM wants to increase because the bigger the margin, the more SVM has high generalization performance. Support vectors are data points that are closest to the margins or those on the margins [11]. This support vector is very important in determining decision boundaries and assigning new, unknown e-mails to a class. By placing new unseen emails on the right side of the decision boundary depending on their feature representation, SVM can classify them after the hyperplane has been assigned. Spam class is assigned to emails on one side of the hyperplane, and non-spam is assigned to emails on the other side.

Naive Bayes (NB) is a classification algorithm based on Bayes' theorem [12]. Because it assumes that each data feature or characteristic is independent, this algorithm is considered "naive". Naive Bayes determines the prior probability of each class, or the probability that each class will appear in the data set. To calculate this, this algorithm divides the total number of instances by the number of instances in each class. The probability of each feature appearing in each class is calculated for each feature. This is done by counting the number of instances in each class where the feature has a certain value and dividing that number by the total number of instances in that class. Naive Bayes will determine the posterior probability of each class given a new unlabelled case. Posterior probability is the possibility of a class based on the observed features. To get the posterior probability, this algorithm will multiply the class prior probability with the probability of each feature in that class [13]. Every class experiences this. The new instance must then be assigned to the class with the highest posterior probability. The speed and simplicity of the Naive Bayes algorithm are its main advantages [14]. When the independence assumption is held consistently, this algorithm can handle very large volumes of data and work effectively. However, this assumption may not hold for complex interactions between traits, which can be a drawback in some situations.

Bayesian optimization is a technique used to find the best set of hyperparameters for machine learning models [15]. The goal of Bayesian optimization is to search through a space of possible hyperparameter values and identify the combination that produces the best model performance on a particular task, such as classification [15]. Accuracy, precision, and recall are examples of assessment metrics that can be used to measure the resulting performance [16]. When an initial set of hyperparameter values is selected, the model is trained and assessed using these values, which is known as Bayesian optimization. Statistical models known as alternate models, such as the Gaussian Process, are built on performance to roughly represent the relationship between hyperparameters and evaluation metrics. Bayesian optimization makes a whole new set of hyperparameter recommendations based on the surrogate model, taking into account the uncertainty of the surrogate model and the best performance [17]. The replacement model is modified as new entries are made, and this process is repeated until the perfect set of hyperparameters is identified. Bayesian optimization is able to intelligently explore the hyperparameter space and concentrate on promising regions by leveraging the knowledge gathered from previous evaluations, which ultimately leads to an ideal set of hyperparameters [18].

Grid search is a methodical approach used in machine learning to adjust model hyperparameters [19]. This parameter significantly affects the performance and behaviour of the model. A grid-like structure is created in a grid search by specifying a predefined set of values for each hyperparameter. All possible combinations of hyperparameter values are shown in this grid. The performance of the model is then assessed using every possible combination of hyperparameters by the grid search algorithm as it repeatedly searches through this grid. The model is trained and scored for each combination using a selected performance measure, such as an f1 score. The optimal hyperparameter for the model is the one whose value produces the best performance metric. Grid search ensures that all possible combinations of hyperparameters are investigated, thereby enabling the best set of hyperparameters for a particular model to be found [20]. This eliminates the need for manual tuning and offers a methodical approach to determining the ideal configuration, increasing functionality and model generalizability.

2. Method

Google Collab was used as an instrument to do this research. The flow of the research conducted is described as shown in Figure 1.

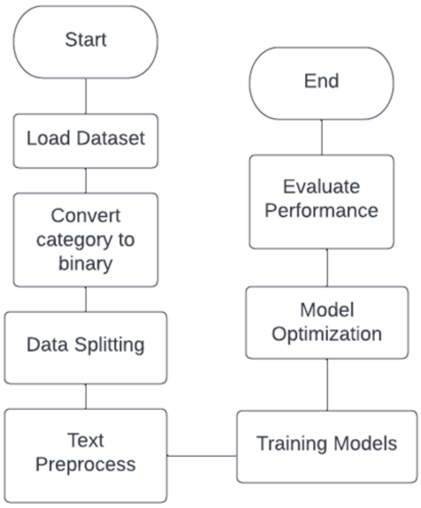

Figure 1. Research flowchart

Dataset

In this study, we used the spam email dataset [21] which is available on Kaggle in csv form, which contains the Message and Category columns. The category consists of spam and ham, the words spam and ham help to distinguish between legitimate (ham) and invalid (spam) messages.

Convert Category To Binary

Spam and non-spam categories need to be converted into binary to facilitate training and evaluation of spam detection models. Machine learning models generally support binary classification tasks with a target value of 0 for non-spam and 1 for spam. Converting spam and non-spam categories into binary is necessary for several reasons. Many machine learning algorithms, particularly those used for classification tasks, are designed to work with binary labels. By converting spam and non-spam categories into binary (0 for non-spam and 1 for spam), we align the data with the format expected by these algorithms, simplifying the training and evaluation processes [22].

Data Splitting

Data splitting is the process of separating the dataset into two subsets, namely the train set and the test set [23]. The train set is used to train the model, while the test set is used to test the performance of the model that has been trained. The goal is to avoid overfitting and ensure that the model can make good predictions on new data that has never been seen before [24]. The portion of the train set and test set used in this study was 75% train set and 25% test set.

Data Preprocessing

Data processing for email spam detection involves several steps. First, the data is cleaned by removing punctuation and converting text to lowercase. This step helps ensure consistency and standardization of text data [24]. Then, the dataset is divided into features and target variables. The features represent purged text messages, while the target variable represents the category label of each message, such as "ham" or "spam". Then, the data is divided into training and test sets. The training set is used to train the model, while the test set serves as an invisible data set to evaluate model performance. This separation ensures that the model can generalize well to new, unseen messages. The next step involves pre-processing the text. Stopwords, which are words that occur frequently and are not very important in the analysis, were omitted from the text [25]. In addition, stemming is done to reduce words to their basic forms or roots [26]. These pre-processing steps aim to remove noise and reduce the dimensionality of the text data, so that the model can focus on important patterns and improve performance. To convert the text data into a format suitable for modelling, the count Vectorizer technique is applied. Count Vectorizer converts text into a numerical representation by counting the frequency of words in the text [27]. This vectorization process allows machine learning algorithms to work with text data effectively and capture the underlying patterns.

Training Models

Model training is the process of training a machine learning model using training data to optimize its performance in studying patterns or relationships between input features and output targets. This process involves adjusting the parameters and weights of the model based on the resulting prediction errors. The goal is to build a model that can make accurate predictions on new data. Model evaluation is carried out using a test set data to ensure good performance.

In this research, Naive Bayes and SVM algorithms are used to detect spam emails.

- Naïve Bayes
  Naive Bayes Classifier is one of the supervised learning algorithms. Bayes' theorem is used to calculate the probability of an event [13]. The advantages are high independence, the ability to handle large amounts of data, and dependence on probability distributions [14]. Bayes' theorem is used to determine probability distributions based on the frequencies in a data set. The Novel Naïve Bayes classifier chooses the class with the largest posterior probability from the probability distribution. The posterior probability equation is shown in equation (1) [13].
  $$P(A|B) = \frac{P(B|A)P(A)}{P\{B\}} \tag{1}$$
- Support Vector Machine
  Supervised learning technique that is widely used for classification and regression tasks [4]. In classification, SVM assigns a label or class (denoted by y) to the input feature vector (denoted by x). Feature vectors that belong to the same class have the same name. SVM is considered as one of the best parameter classifiers available [8]. The basic rule is to find the hyperplane as far from the training samples as possible to optimize the distance between training samples [10]. In SVM, data points are represented visually as points in a multidimensional space, where each dimension represents a certain feature [9]. Then, the SVM algorithm determines the optimal hyperplane to efficiently separate the two layers in this multidimensional space. Figure 2 shows a simple linear SVM classification with a hyperplane and a line separating the vectors of the two classes.

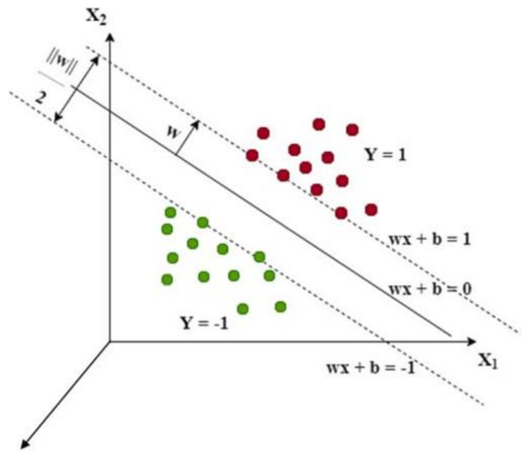

Figure 2. Linear SVM classification

Suppose the equation of a straight hyperplane is:

$$x_2 = ax_1 + b \tag{2}$$

$$ax_1 + b - x_2 = 0 \tag{3}$$

$$wx + b = 0 \tag{4}$$

where $x = (x_1, x_2)$ and $w = (a, 1)$ represent the hyperplane equation for a multidimensional space and b is the bias. SVM uses the following hypothesis function h, to make predictions after determining the hyperplane.

$$h(x_i) = +1, if\ wx + b \geq 0 \tag{5}$$

and

$$h(x_i) = -1, if\ wx + b \leq 0 \tag{6}$$

Optimization Models

To further improve the performance of the model, hyperparameter tuning is performed. Bayesian optimization is used to find optimal hyperparameters for the SVM model. The optimization process aims to maximize the F1 score, which is a metric that balances precision and recall. The same optimization technique is also applied to fine-tune the hyperparameter alpha for Naive Bayes.

After obtaining the optimized hyperparameters, the optimized model is then fitted to the training data, and predictions are made on the test dataset using the optimized model. Grid search is also used to investigate various hyperparameters for SVM and Naïve Bayes. Grid search thoroughly combs the defined hyperparameter space while cross validation assesses the effectiveness of the model. Based on the F1 value, the top hyperparameter and the matching model are found. On the training data set, the best model from grid search and Bayesian optimization is used. This process makes it possible to assess the effectiveness of the model after hyperparameter adjustment.

Evaluate Performance

To analyze the accuracy obtained by the model in classifying test data, evaluation was carried out using a confusion matrix. Accuracy measures the overall accuracy of a model's predictions by calculating the ratio of correctly classified samples to the total number of samples in the test dataset. This gives an indication of the general predictive accuracy of the model. Accuracy can be calculated using formula (7).

$$\text{Accuracy} = (TN + TP) / (TN + FP + FN + TP). \tag{7}$$

The confusion matrix table can be seen in Table 1.

Table 1. Confusion matrix

	Predicted Negatives	Predicted Positives
Actual Negatives	TN	FP
Actual Positives	FN	TP

Model evaluation also will be carried out by analyzing model performance based on the variables F1 score, recall and also precision. The F1 score is a metric commonly used to evaluate the performance of a classification model. This score combines precision and recall into one score, thus providing a balanced measure of model accuracy. The formula for calculating the F1 score is as follows: F1 = 2 * (precision * recall) / (precision + recall) [28]. The performance of the trained model is evaluated using various metrics, including accuracy, precision, recall and F1 score. These metrics provide insight into how well the model is performing in classifying email messages as spam or not spam. Precision focuses on the model's ability to correctly identify spam emails among predicted positive emails. It measures the proportion of spam emails that are correctly classified from all emails that are predicted to be spam. A high precision value indicates a low false positive rate. Recall, also known as sensitivity or true positive rate, measures a model's ability to correctly identify all true positive samples. It calculates the ratio of correctly classified spam emails to the total number of actual spam emails. A high recall value indicates a low false negative rate.

The F1 score is a harmonized average of precision and recall, providing a balanced measure of model performance [8]. This score takes both precision and recall into account to assess the model's accuracy in identifying positive and negative examples. For the SVM and Naive Bayes models, the accuracy, precision, recall and F1 scores were calculated based on the predicted labels and the actual labels of the test dataset. These values indicate the model's performance in classifying email messages. In addition, for the SVM and Naive Bayes models optimized through Bayesian optimization, the accuracy, precision, recall, and F1 values are calculated using predicted labels and actual labels from the test dataset, respectively.

Furthermore, accuracy, precision, recall, and F1 values were calculated for the SVM and Naive Bayes models which were obtained through a grid search. In the case of email spam detection, the F1 score is often considered more important than other metrics such as accuracy, precision and recall. The reason for this is due to the unbalanced nature of the spam detection dataset. Email datasets usually have a large number of non-spam messages (ham) compared to spam messages. This class imbalance can lead to high accuracy values even though the model performs poorly in detecting spam. For example, if the dataset consists of 90% ham and 10% spam, a classifier that predicts all emails as ham will achieve 90% accuracy without effectively identifying spam. In such unbalanced scenarios, precision and recall alone may not provide a complete picture of model performance. Precision focuses on the accuracy of positive predictions, whereas recall focuses on being able to correctly identify all positive examples. However, high precision values can be achieved by labelling very few examples as spam, while high recall values can be obtained by labelling almost all examples as spam, including many false positives. The F1 value, which is the harmonic average of precision and recall, takes both metrics into account and balances their importance. This value gives equal weight to precision and recall and provides a single value that reflects the trade-off between the two. This is particularly useful in spam detection, where correctly identifying as many spam messages as possible (high recall) while minimizing false positives (high precision) is critical.

3. Results and Discussion

From the research that has been carried out, the first step taken is to analyze the results of the accuracy of the model built. The accuracy results of the model built can be seen in Table 2.

Table 2. Comparison table

Algorithm	Accuracy	Precision	Recall	F1 Score
SVM	98.4207%	99.3976%	88.7097%	93.75%
NB	98.5642%	96.6292%	92.4731%	94.5055%
Bayes SVM	98.5642%	99.4048%	89.7849%	94.3503%
Bayes NB	98.5642%	96.1111%	93.0108%	94.5055%
SVM Grids	98.4207%	99.3976%	88.7097%	93.7500%
Grid NB	98.5642%	96.1111%	93.0108%	94.5355%

Based on table 2, the classifier with untuned parameters that has the highest accuracy score is the NB classifier with a score of 98.5642%, while the SVM classifier has a score of 98.4207%. The highest precision score resulted from the untuned classifier, namely the SVM classifier with a score of 99.3976% followed by the NB classifier with a score of 96.6292%. The highest recall and f1 scores for the untuned classifier are those owned by the NB classifier with scores of 92.4731% and 94.5055%. For the SVM classifier, the recall and f1 scores are 88.7097% and 93.7500%. For accuracy metrics. The classifier that has done Bayesian Optimization or Bayes NB tuning parameters achieves an accuracy score of 98.5642%, while the SVM classifier obtains the same accuracy score of 98.5642%. However, if you look at the precision score, the SVM classifier shows the best results with a score of 99.4048%, followed by the NB classifier with a score of 96.1111%. For recall and F1 scores, the NB classifier achieved the highest scores of 93.0108% and 94.5055%, while the SVM classifier achieved a recall score of 89.7849% and an F1 score of 94.3503%. For the classifier that has performed parameter tuning using the grid search method or called gird SVM with the parameters adjusted it achieves an accuracy score of 98.4207%, while the NB classifier with the parameters adjusted achieves an accuracy score of 98.5642%. When looking at the precision score, the NB grid with tuned parameters achieved the highest score of 99.3976%, while the SVM grid with tuned parameters achieved a precision score of 96.1111%. For recall and F1 scores, the NB grid obtained a recall score of 93.0108% and an F1 score of 94.5355%. Meanwhile, the SVM grid has a recall score of 88.7097% and an F1 score of 93.7500%.

Based on the result above, it can be noticed that the Naive Bayes classifier dominates almost all measurement metrics compared to the SVM classifier which only excels in precision scores. For the Naive Bayes classifier on the accuracy score, the Bayesian optimization tuning parameters and grid search do not affect the increase or decrease in performance. On precision and recall metrics, Naïve Bayes shows the highest performance without any parameter tuning at all. while for the f1 metric, Naïve Bayes has the highest performance by tuning the grid search. For the SVM classifier, this classifier has the highest performance compared to other parameter tinting methods. after Bayes optimization is done except for the f1 metric which has the highest score after tuning grid search parameters.

4. Conclusion

According to the study findings, the Naive Bayes classifier generally outperforms the SVM classifier in identifying spam emails, except for precision where SVM performs better. Grid Search Multi Nominal Naive Bayes yields better results compared to Bayesian Optimization. However, the SVM classifier with Bayesian optimization achieves the highest precision, indicating fewer false positives. Overall, Grid Search Naive Bayes exhibits superior performance in terms of accuracy, recall, and F1-score, making it a preferred choice for spam email detection tasks. These results provide valuable insights for researchers and organizations aiming to enhance email security.

Future research can explore additional data processing techniques and hyperparameter tuning methods to further improve the effectiveness of both SVM and Naive Bayes classifiers in spam email detection. Additionally, investigating ensemble methodologies and addressing issues with unbalanced datasets will contribute to developing more reliable spam detection models, ultimately improving email communication security.

REFERENCES

[1] K. Debnath and N. Kar, "Email spam detection using deep learning approach," in 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON), IEEE, 2022, pp. 37–41.

[2] D. Aprilianto, "SVM Optimization with Correlation Feature Selection Based Binary Particle Swarm Optimization for Diagnosis of Chronic Kidney Disease," J. Soft Comput. Explor., vol. 1, no. 1, Sep. 2020, doi: 10.52465/joscex.v1i1.1.

[3] A. Nurdina and A. B. I. Puspita, "Naive Bayes and KNN for Airline Passenger Satisfaction Classification: Comparative Analysis," J. Inf. Syst. Explor. Res., vol. 1, no. 2, Jul. 2023, doi: 10.52465/joiser.v1i2.167.

[4] S. Nandhini and D. J. Marseline, "Performance Evaluation of Machine Learning Algorithms for Email Spam Detection," Int. Conf. Emerg. Trends Inf. Technol. Eng. ic-ETITE 2020, pp. 1–4, 2020, doi: 10.1109/ic-ETITE47903.2020.312.

[5] N. Reska and K. Tsabita, "Comparison of KNN, naive bayes, and decision tree methods in predicting the accuracy of classification of immunotherapy dataset," J. Student Res. Explor., vol. 1, no. 2, pp. 104–121, Jul. 2023, doi: 10.52465/josre.v1i2.170.

[6] S. O. Olatunji, "Improved email spam detection model based on support vector machines," Neural Comput. Appl., vol. 31, pp. 691–699, 2019.

[7] A. Ghosh and A. Senthilrajan, "Comparison of machine learning techniques for spam detection," Multimed. Tools Appl., pp. 1–28, 2023.

[8] S. M. M. Hossain and I. H. Sarker, "Content-based Spam Email Detection Using N-gram Machine Learning Approach," 2021.

[9] D. Mallampati and N. P. Hegde, "Feature Extraction and Classification of Email Spam Detection Using IMTF-IDF+ Skip-Thought Vectors.," Ingénierie des Systèmes d′ Information, vol. 27, no. 6, 2022.

[10] G. A. Reddy and B. I. Reddy, "Classification of Spam Text using SVM," J. Univ. Shanghai Sci. Technol., vol. 23, no. 8, pp. 616–624, 2021.

[11] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," ACM Trans. Intell. Syst. Technol., vol. 2, no. 3, pp. 1–27, 2011.

[12] P. Kaviani and S. Dhotre, "Short survey on naive bayes algorithm," Int. J. Adv. Eng. Res. Dev., vol. 4, no. 11, pp. 607–611, 2017.

[13] T. M. Hansen and C. C. Finlay, "Use of machine learning to estimate statistics of the posterior distribution in probabilistic inverse problems–An application to airborne EM data," J. Geophys. Res. Solid Earth, vol. 127, no. 11, p. e2022JB024703, 2022.

[14] R. A. Cahya and F. A. Bachtiar, "Weakening Feature Independence of Naïve Bayes Using Feature Weighting and Selection on Imbalanced Customer Review Data," in 2019 5th International Conference on Science in Information Technology (ICSITech), IEEE, 2019, pp. 182–187.

[15] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of Bayesian optimization," Proc. IEEE, vol. 104, no. 1, pp. 148–175, 2015.

[16] V. Plevris, G. Solorzano, N. P. Bakas, and M. E. A. Ben Seghier, "Investigation of performance metrics in regression analysis and machine learning-based prediction models," in 8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2022), European Community on Computational Methods in Applied Sciences, 2022.

[17] Y. Zhang, N. H. Kim, C. Park, and R. T. Haftka, "Multifidelity surrogate based on single linear regression," AIAA J., vol. 56, no. 12, pp. 4944–4952, 2018.

[18] Y. Fang, C. Liu, and Z. Li, "Optimization method of aluminum electrolysis current efficiency

based on LightGBM-TPE," in Second Guangdong-Hong Kong-Macao Greater Bay Area Artificial Intelligence and Big Data Forum (AIBDF 2022), SPIE, 2023, pp. 158–163.
[19] M. M. Ramadhan, I. S. Sitanggang, F. R. Nasution, and A. Ghifari, "Parameter tuning in random forest based on grid search method for gender classification based on voice frequency," DEStech Trans. Comput. Sci. Eng., vol. 10, no. 2017, 2017.
[20] W. Nugraha and A. Sasongko, "Hyperparameter Tuning on Classification Algorithm with Grid Search," Sist. J. Sist. Inf., vol. 11, no. 2, pp. 391–401, 2022.
[21] F. Qureshi, "Spam Emails."
[22] P. Flach and M. Kull, "Precision-recall-gain curves: PR analysis done right," Adv. Neural Inf. Process. Syst., vol. 28, 2015.
[23] Q. H. Nguyen et al., "Influence of data splitting on performance of machine learning models in prediction of shear strength of soil," Math. Probl. Eng., vol. 2021, pp. 1–15, 2021.
[24] P. Rouzrokh et al., "Mitigating bias in radiology machine learning: 1. Data handling," Radiol. Artif. Intell., vol. 4, no. 5, p. e210290, 2022.
[25] A. Datta, B. Jena, A. K. Dash, and R. Pradhan, "A Comprehensive Analytical Study of Traditional and Recent Development in Natural Language Processing," Int. J., vol. 10, no. 5, 2021.
[26] I. Boban, A. Doko, and S. Gotovac, "Sentence retrieval using stemming and lemmatization with different length of the queries," Adv. Sci. Technol. Eng. Syst., vol. 5, no. 3, pp. 349–354, 2020.
[27] H. A. Almuzaini and A. M. Azmi, "Impact of stemming and word embedding on deep learning-based Arabic text categorization," IEEE Access, vol. 8, pp. 127913–127928, 2020.
[28] I. Song and S. Kim, "AVILNet: A new pliable network with a novel metric for small-object segmentation and detection in infrared images," Remote Sens., vol. 13, no. 4, p. 555, 2021.