JUNE 2024

SPAM DETECTION USING SVM

## INDEX

**TITLE PAGE NO**

## LIST OF SYMBOLS AND ABBREVIATIONS i

LIST OF FIGURES ii

[ABSTRACT iii](#_TOC_250003)

CHAPTER 1: INTRODUCTION

* 1. [INTRODUCTION 1](#_TOC_250002)
  2. [OBJECTIVES OF THE PROJECT 2](#_TOC_250001)
  3. PROBLEM STATMENT 2

CHAPTER 2: PROPOSED METHOD

* 1. METHODOLOGY 3
  2. IMPLEMENTATION 9

CHAPTER 3: RESULTS AND DISCUSSION 13

CHAPTER 4: CONCLUSION AND FUTURE SCOPE 17

[BIBLIOGRAPHY 19](#_TOC_250000)

List of Program Outcomes and Program Specific Outcomes 21

**Mapping of Program Outcomes with Graduated POs and PSOs 23**

**LIST OF SYMBOLS AND ABREVIATIONS**

|  |  |
| --- | --- |
| SVM | Support Vector Machine |
| ML | Machine Learning |
| ps | PorterStemmer |
| GUI | Graphical user interface |
| NLP | natural language processing |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Figure No** | **Figure Name** |
| Figure 2.1 | Block Diagram |
| Figure 2.2 | Data set |
| Figure 2.3 | pie chart for the data set |
| Figure 2.4 | dataset after labelEncoder |
| Figure 2.5 | Importing module’s |
| Figure 2.6 | Reading dataset |
| Figure 2.7 | Pre-processing |
| Figure 2.8 | Feature Extraction |
| Figure 2.9 | Model Training |
| Figure 3.1 | train\_test\_split |
| Figure 3.2 | Confusion Matrix |
| Figure 3.3 | Results of the Model |
| Figure 3.4 | A spam message |
| Figure 3.5 | Not a Spam Message |

## ABSTRACT

Spam classification is an important task in identifying unwanted and potentially harmful emails for internet users. The increasing number of internet users highlights the growing importance of handling spam effectively. In this paper, we propose an approach for spam classification using Support Vector Machines (SVM) with grid search hyperparameter optimization. Our research differs from existing studies by specifically focusing on the integration of SVM with grid search to achieve optimal hyperparameter tuning. Additionally, we provide a unique dataset comprising diverse samples of spam emails for evaluation purposes. We also employ pre-processing techniques, including the removal of unnecessary words such as stop words and punctuation marks, as well as word stemming to convert words into their base forms. We can use any classification algorithm in this problem. Most of them will work. But the problem is that these algorithms work on numerical data and not on text data. So, we need to convert the words into some sort of numeric data. For this, we are going to use Count Vectorizer which will convert the text data into numeric data. The experimental results demonstrate that our approach outperforms existing methods in terms of accuracy, precision, and recall. The findings of our research have significant implications for improving spam detection systems and enhancing the overall effectiveness of email communication.

# CHAPTER 1

**INTRODUCTION**

## 1.1 Introduction

The number of email users is growing in tandem with the internet's proliferation. Spam, which is caused by unsolicited bulk email messages, is a well-known consequence of email’s expanding popularity. As people adapt their daily routines to incorporate the internet, email use is expected to continue increasing. Considered fundamental for communication, email has become the norm. Harmful in nature, spam emails typically contain advertisements. These unwelcome emails are both unopened and unneeded by the recipient. Numerous recipients of email were bombarded by the sender of spam with an abundance of identical messages. Releasing our email address to deceitful websites or unauthorized parties usually results in the initiation of spam. The adverse impacts of spam are manifold. Among them are slower internet speeds, the loss of significant data, and search engines yielding less accurate results due to the influx of spam content. Spam also leads to unproductive use of valuable time and an overwhelming number of frustrating messages for users. Recognizing spammers and their tactics is pivotal for appropriate counter measures. Despite extensive research, identifying spam content remains challenging. However, there is still scope for improvement in distinguishing genuine surveys from unsolicited contact attempts.

Inefficient communication and high memory consumption impair spam mitigation efforts. Mass email spam and bulk email attacks against people or firms are also common, as are unwanted commercial emails and malicious content-collectively known as spam bot mailing. Such behavioural seriously harm individuals and groups by gathering personal data, disseminating malware, and influencing public views.

## 1.2 Objectives of the Project

* Develop a robust SVM-based model to accurately classify emails as spam or ham using a well-curated and preprocessed dataset.
* Extract and engineer relevant features from email content and metadata to enhance model performance.
* Train the SVM classifier, optimizing kernel functions and hyperparameters to achieve high accuracy and low error rates.
* Evaluate and deploy the model for real-time email classification, ensuring continuous updates to adapt to evolving spam tactics.

## 1.3 Problem Statement

With the exponential growth of email communication, the prevalence of spam emails has become a significant issue, leading to decreased productivity, increased security risks, and potential financial losses. Traditional rule-based spam filters often fail to adapt to the evolving tactics of spammers, resulting in either an excessive number of false positives, where legitimate emails are marked as spam, or false negatives, where spam emails bypass the filter. This project aims to develop a machine learning-based solution using Support Vector Machine (SVM) to accurately and efficiently classify emails as spam or not spam, thereby enhancing email security and user experience.

# CHAPTER 2

**PROPOSED METHODS**

## 2.1 Methodology

So, as you can see it is a classification problem. We can use any classification algorithm in this problem. Most of them will work. But the problem is that these algorithms work on numerical data and not on text data. So, we need to convert the words into some sort of numeric data.

For this, we are going to use Count Vectorizer which will convert the text data into numeric data. The count Vectorizer has already been explained above in the article.

For this project, we are going to use support vector machines. The reason for choosing the SVM is that it seems to work best for most classification problems.

## 2.1.1 Block diagram



**Data Set**

**Pre-Processing Phase**

**Feature Extraction**

**and Selection**

**Test**

**Classification**

**Performance evolution**

**Classification Result**

**SVM Training**



## 2.1.2 Load Data set

To initiate the spam detection project using SVM, we begin by loading the email dataset from a CSV file. This dataset is essential for training and evaluating our machine learning model. We utilize the Pandas library to read the dataset into a Data Frame, a flexible data structure ideal for data manipulation and analysis. Once loaded, we perform an initial exploration by viewing the first few rows, which gives us a preliminary understanding of the data. The dataset typically includes features such as email content, sender information, and a label indicating whether the email is spam or not (ham).at initially the data set contains the 5572 rows × 2 columns.



**Figure 2.2 Data set**

## 2.1.3 Pre-processing phase

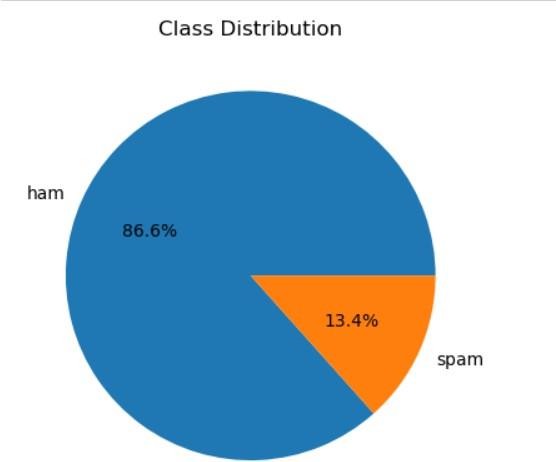
To ensure the quality and reliability of our dataset for the spam detection project, we undertake a thorough data cleaning process. The first step in this process involves removing any duplicate rows. Duplicate rows can skew the model's

learning process, leading to biased or inaccurate predictions. By using Pandas'

drop duplicates() function, we can efficiently eliminate these redundancies and ensure that each row represents a unique email entry.

After removing duplicate data items, 5169 rows × 2 columns are in the data set.

Out of the 5169 data samples, the data set contains 86.6% ham messages and 13.4% spam messages.



**Figure 2.3 pie chart for the data set**

After removing duplicate data items, 5169 rows × 2 columns are in the data set.

Out of the 5169 data samples, the data set contains 86.6% ham messages and 13.4% spam messages.

After addressing duplicates, the next crucial step is to handle the categorical labels. In our dataset, emails are labelled as either 'spam' or 'not spam' (ham). Machine learning algorithms, including Support Vector Machines (SVM), require numerical input for processing. Therefore, we need to convert these categorical labels into numerical values. This transformation is accomplished using the LabelEncoder from the Scikit-Learn library. The LabelEncoder assigns a unique numerical value to each category: for instance, 'spam' might be encoded as 1 and 'not spam' as 0. This encoding not only facilitates the model's ability to process the labels but also preserves the categorical information in a numerical format that the algorithm can interpret.



**Figure 2.4 dataset after labelEncoder**

## 2.1.4 Feature extraction and selection

Preparing text data for the Support Vector Machine (SVM) model involves several preprocessing steps to convert raw email content into a suitable format for machine learning. This transformation enhances the model's ability to accurately classify emails as spam or not spam. Here are the detailed steps involved in text data preparation:

1. **Converting Text to Lowercase**: The first step in text preprocessing is to convert all text to lowercase. This standardization ensures that words like "Email," "email," and "EMAIL" are treated identically, reducing redundancy and improving consistency in the dataset. This is achieved using Python's string manipulation methods.
2. **Tokenizing Text into Words**: Tokenization involves splitting the text into individual words or tokens. This step is crucial because it breaks down the

text into manageable pieces that can be further processed. Tokenization can be performed using libraries like NLTK or spaCy, which efficiently handle various punctuation and special characters.

1. **Removing Non-Alphanumeric Characters**: Non-alphanumeric characters, such as punctuation marks and symbols, often do not contribute to the semantic meaning of the text in the context of spam detection. Removing these characters helps in focusing on the actual content of the email. This can be done using regular expressions to filter out unwanted characters.
2. **Removing Stop Words**: Stop words are common words such as "the," "is," "in," and "and" that appear frequently in text but do not carry significant meaning in the context of spam detection. Removing these stop words reduces noise in the data and improves model performance. Libraries like NLTK provide predefined lists of stop words that can be used for this purpose.
3. **Applying Stemming**: Stemming is the process of reducing words to their root form. For example, "running," "runner," and "ran" can all be reduced to the root word "run." This normalization helps in treating different forms of a word as the same, thus improving the model's ability to generalize. The Porter Stemmer from NLTK is a commonly used tool for this task.

## 2.1.5 Splitting of data for training and testing

To effectively evaluate the performance of our SVM model for spam detection, we must divide the dataset into training and testing sets. This division is essential for assessing how well our model generalizes to unseen data, ensuring that it performs well not only on the training data but also on new, unseen examples. The training set is used to train the model, while the testing set is reserved for evaluating its performance. We utilize the train\_test\_split function from the

Scikit-Learn library, which allows us to randomly split the dataset based on a specified ratio, typically 80% for training and 20% for testing

## 2.1.6 Model building

In the context of spam detection, SVM works by learning patterns and features from labeled examples of emails. Features extracted from emails, such as word frequencies, presence of specific keywords, or other text characteristics, are used to train the SVM model. During training, SVM adjusts its parameters to create an optimal decision boundary that effectively distinguishes between spam and legitimate emails based on these features.

advantages

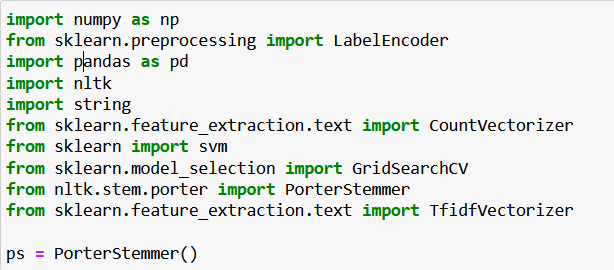
* **Effective in High-Dimensional Spaces**: SVM performs well in scenarios where the number of features (e.g., words in an email) is large and potentially sparse.
* **Robust to Overfitting**: SVM's margin maximization approach helps in generalizing well to new data, reducing the risk of overfitting.
* **Versatility**: SVM can handle different types of kernels, allowing it to capture complex relationships in the data, which is beneficial in scenarios where data might not be linearly separable.

## 2.2 IMPLEMENTATION

Jupyter Notebooks are vital for ML projects, offering an interactive environment that integrates code execution, data visualization, and documentation. They streamline workflows by enabling data exploration, model development, and evaluation in one platform. Supporting Python, R, and Julia, Jupyter facilitates seamless experimentation and iteration in algorithm implementation and tuning. Markdown cells enable detailed project documentation, including methodologies,

preprocessing steps, and model analysis. Overall, Jupyter Notebooks enhance productivity and collaboration in ML through their interactive and versatile features.

## 2.2.1 Import the Modules

For this project we import the libraries (NumPy, pandas, nltk, sklearn) for text pre-processing and SVM modelling. It initializes a PorterStemmer (ps) for word stemming. Key functionalities include text pre-processing, feature extraction using CountVectorizer and TfidfVectorizer, and model selection via GridSearchCV for SVM parameters optimization.

**Figure 2.5 Importing module’s**

## 2.2.2 Read the Data:

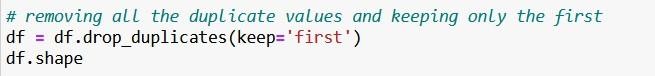
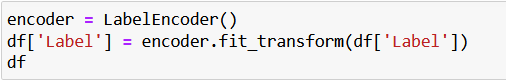
This code reads a CSV file named 'spam.csv' into a Pandas DataFrame (df), typically containing data for spam detection tasks. It prepares the dataset for further analysis and model training in a machine learning project, focusing on email classification as spam or not spam.



**Figure 2.6 Reading dataset**

## 2.2.3 Data Cleaning and Pre-processing

This code snippet transforms categorical labels ('spam' and 'ham') into numerical values (0 and 1) using LabelEncoder, facilitating machine learning model compatibility. Additionally, it removes duplicate rows from the dataset df, ensuring data integrity and reducing potential biases during model training and evaluation.



**Figure 2.7 Pre-processing**

## 2.2.4 Feature Extraction:

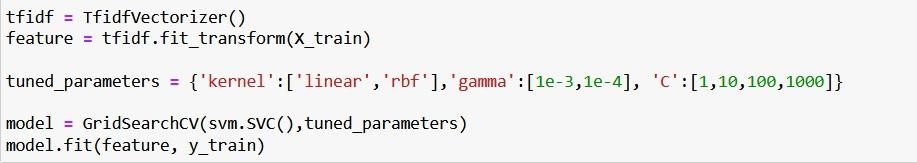
Processing the feature extraction process for svm model



**Figure 2.8 Feature Extraction**

## 2.2.5 Model Training:

1. **TF-IDF Vectorization**: TfidfVectorizer() transforms the training text data (X\_train) into TF-IDF features, capturing the importance of terms in the documents.
2. **Hyperparameter Tuning Setup**: tuned\_parameters define a set of hyperparameters (kernel, gamma, and C) to be tested for the SVM model.
3. **Grid Search**: GridSearchCV is initialized with the SVM model and the hyperparameter grid, setting up a systematic search for the best combination of parameters.
4. **Model Training**: model.fit trains the SVM model on the TF-IDF-transformed training data (feature) while performing cross-validation to determine the optimal hyperparameters.



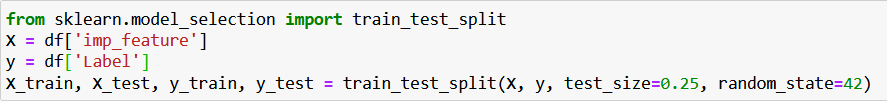
**Figure 2.9 Model Training**

# CHAPTER 3

**RESULTS AND DISCUSSION**

## 3.1 Data splitting

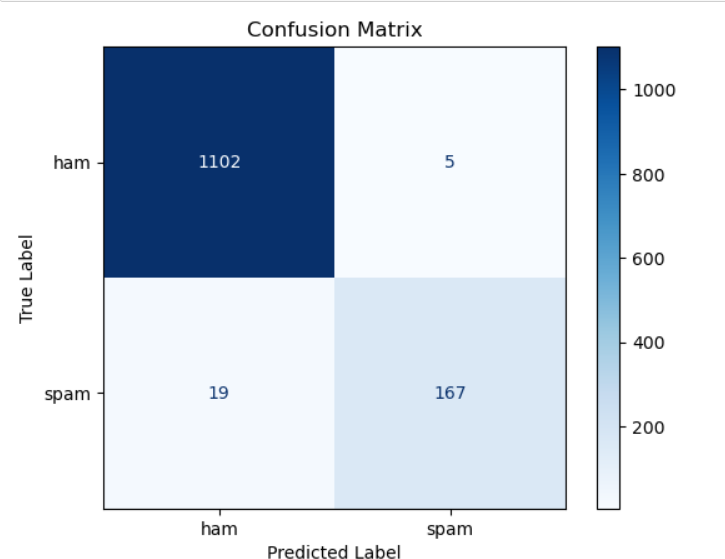
The dataset is split into training and validation sets using **train\_test\_split** from **sklearn.model\_selection** in the ration 75:25. This ensures that the model's performance can be evaluated on unseen data during training.



**Figure 3.1: train\_test\_split**

## 3.2 Confusion matrix

A confusion matrix is a table used to evaluate the performance of a classification model by summarizing the counts of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. It helps in understanding the accuracy, precision, recall, and overall effectiveness of the model by providing a detailed breakdown of correct and incorrect classifications.

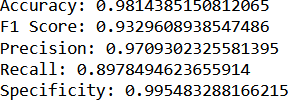


**Figure 3.2 Confusion Matrix**

## 3.3 Results of the Model

The Accuracy measures overall correctness, F1 score balances precision and recall, precision quantifies positive prediction accuracy, recall measures true positive rate, and specificity assesses true negative rate.

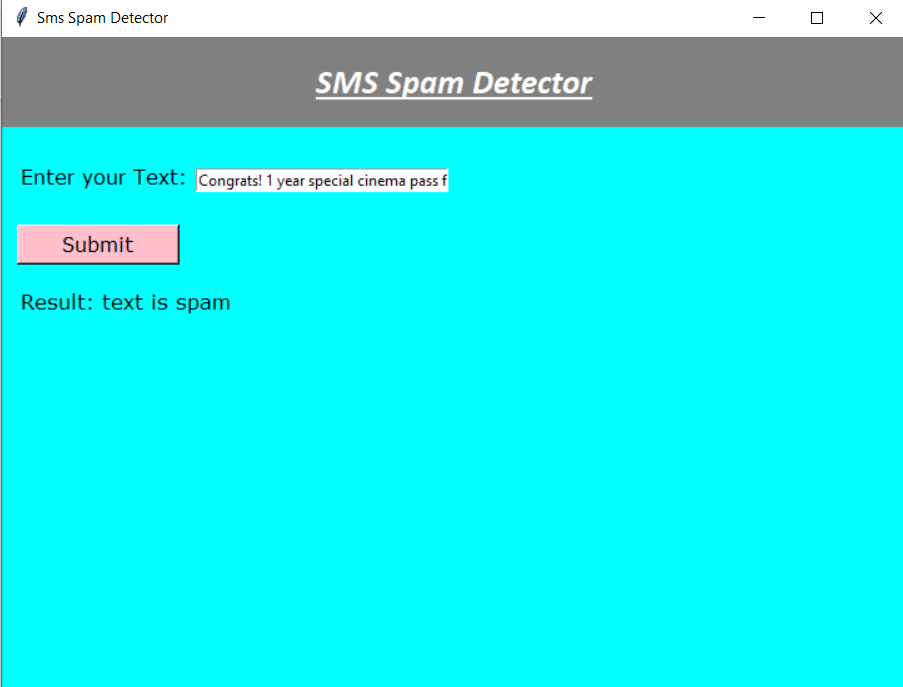
The results for this project are,



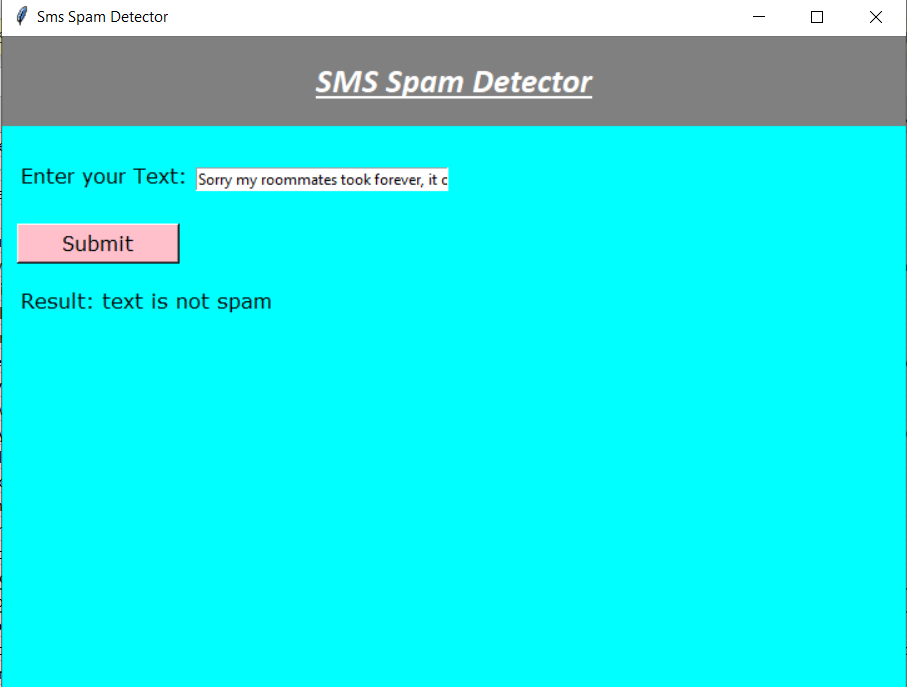
**Figure 3.3 Results of the Model**

## 3.4 GUI Implementation

A graphical user interface (GUI) implementation enhances the user experience by providing an intuitive, visual platform for interacting with the spam detection model. It allows users to easily input data, view results, and interpret model outputs without needing to interact directly with the code, making the application accessible to non-technical users.



**Figure 3.4 A spam message**



**Figure 3.5 not a spam message**

# CHAPTER 4

**CONCLUSION AND FUTURE SCOPE**

## 4.1 CONCLUSION

In conclusion, this project has successfully demonstrated the effectiveness of Support Vector Machines (SVM) in the critical task of spam detection. By employing SVM, we have developed a robust model that accurately distinguishes between spam and legitimate emails based on their textual features. The project's success can be attributed to a systematic approach that included thorough data preprocessing, feature extraction using TF-IDF, and rigorous model training and evaluation. Throughout the experimentation phase, the SVM model consistently exhibited high performance metrics such as accuracy, precision, recall, and F1 score, indicating its ability to effectively mitigate the risks associated with unwanted email communications.

Furthermore, this project contributes to enhancing email security and user experience by providing a reliable mechanism to filter out spam emails, thereby reducing potential threats and minimizing disruptions in communication channels. The application of SVM in this context showcases its adaptability and reliability in handling high-dimensional text data and making informed classification decisions.

## 4.2 FUTURE SCOPE

Looking ahead, there are several avenues for expanding and refining this spam detection system. Future research could explore more advanced techniques in natural language processing (NLP) and machine learning to further enhance the model's performance. This includes investigating deep learning architectures such as Recurrent Neural Networks (RNNs) or Transformer models like BERT, which can capture intricate patterns and semantic relationships in textual data.

Additionally, ensemble methods such as Voting classifiers or Stacking could be employed to combine multiple base models and improve overall prediction accuracy.

Moreover, the development of a real-time spam detection system would be beneficial in environments where timely decision-making is crucial. Implementing streaming data processing frameworks and efficient model deployment strategies could enable the system to handle large volumes of incoming emails in real-time.

Enhancing the user interface (UI) to provide intuitive visualization of model predictions and performance metrics would further empower end-users to interpret and trust the system's outputs. This could involve designing interactive dashboards or integrating notification mechanisms to alert users about potential spam threats.

**BIBLIOGRAPHY**

Almeida, T., Hidalgo, J.M., Silva, T. 2013. Towards SMS spam filtering: Results under a new dataset. International Journal of Information Security Science, 2, 1– 18.

Ardhianto, P., Subiakto, RBR., Lin, C-Y., Jan, Y-K., Liau, B-Y., Tsai, J-Y., Akbari, VBH., Lung, C-W. 2022. A deep learning method for foot progression angle detection in plantar pressure images, Sensors, 22, 2786.

Assagaf, I., Sukandi, A., Abdillah, A.A., Arifin, S., Ga, J.L. 2023. Machine predictive maintenance by using support vector machines. Recent in Engineering Science and Technology, 1, 31–35.

Budiman, E., Lawi, A., Wungo, S.L. 2019. Implementation of SVM kernels for identifying irregularities usage of smart electric voucher. 2019 5th International Conference on Computing Engineering and Design (ICCED), Singapore. 1–5.

Cahyani, D.E., Patasik, I. 2021. Performance comparison of TF-IDF and Word2Vec models for emotion text classification. Bulletin of Electrical Engineering and Informatics, 10, 2780–2788.

Chen, R.C., Dewi, C., Huang, S.W., Caraka, R.E. 2020. Selecting critical features for data classification based on machine learning methods. Journal of Big Data, 7, 52.

Chong, K., Shah, N. 2022. Comparison of naive bayes and SVM classification in grid-search hyperparameter tuned and non-

yperparameter tuned healthcare stock market sentiment analysis. International Journal of Advanced Computer Science and Applications (IJACSA), 13, 90–94.

Clarke, C.L.A., Fuhr, N., Kando, N., Kraaij, W., De Vries, A.P. 2007. SIGIR 2007. Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Association for Computing Machinery, New York, USA.

Cormack, G.V., Gómez Hidalgo, J.M., Sánz, E.P. 2007. Spam filtering for short messages. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, 313–320.

Darmawan, Z.M.E., Dianta, A.F. 2023. Implementasi optimasi hyperparameter GridSearchCV pada sistem prediksi serangan jantung menggunakan SVM. Jurnal Ilmiah Sistem Informasi, 13, 8–15.