Journal of Mechatronics and Artificial Intelligence

JURNAL UPI

JMAI JOURNAL OF MECHATRONICS AND ARTIFICIAL INTELLIGENCE

Homepage: http://ejournal.upi.edu/index.php/jmai/

Comparison of Naïve Bayes Classifier and Support Vector Machine Methods for Sentiment Classification of Responses to Bullying Cases on Twitter

Firda Millennianita[1], Ummi Athiyah[2,*], Arif Wirawan Muhammad[3]

[1,3] Fakultas Informatika, Program Studi Informatika, Institut Teknologi Telkom Purwokerto, Indonesia
[2] Fakultas Informatika, Program Studi Sains Data, Institut Teknologi Telkom Purwokerto, Indonesia
**Correspondence author email: ummi@ittelkom-pwt.ac.id*

ARTICLE INFO

Article History:

Submitted/Received 25 May 2024
First Revised 29 May 2024
Accepted 31 May 2024
Publication Date 02 Jun 2024

Keywords:
Bullying,
Sentiment Analysis,
Naive Bayes Classifier,
Support Vector Machine,
Twitter

ABSTRACT

The rapid dissemination of information related to the K-Pop world, facilitated by social media, has made it easier to follow developments and controversies. One notable case that sparked extensive discussion on Twitter was the bullying allegations against Kim Garam of LE SSERAFIM. Researchers, using Twitter data, sought to analyze Indonesian public sentiment regarding this case through sentiment analysis, which classifies opinions as positive or negative. For processing textual data, text mining methods, particularly classification techniques, are employed. Two popular algorithms in text mining are the Naive Bayes classifier and the support vector machine (SVM). The Naive Bayes classifier is favored for its speed, simplicity, and high accuracy, while the SVM excels at identifying a hyperplane that maximizes the margin between classes. In this study, sentiment classification results were labeled as either positive or negative. The comparison between the Naive Bayes classifier and SVM for classifying responses to Kim Garam's bullying case on Twitter showed high accuracy rates: 93% for Naive Bayes and 97% for SVM. The higher accuracy of the SVM algorithm indicates its superiority over the Naive Bayes classifier in this context.

1. Introduction

Hallyu, or the Korean Wave, is a phenomenon referring to the global spread of Korean pop culture across various countries, including Indonesia. Following this phenomenon, consumer interest in Korean products has also increased [1]. The Korean Wave is currently sweeping across Indonesia, affecting a wide range of people from children to adults. One of the most popular aspects of the Korean Wave is K-pop, a genre of popular music originating from South Korea.

The utilization of internet technology, particularly social media, has significantly influenced the development of the K-Pop world [2]. Information regarding K-Pop developments spreads rapidly and is easily accessible through social media. Anything related to K-Pop can become a highly popular topic and headline news among netizens. However, behind the success of K-Pop, questions and issues surrounding the culture and quality of K-Pop idols spark both pros and cons, often leading to debates among K-Pop fans. Given the significant influence of social media in disseminating information and shaping public opinion, analyzing public reactions to controversies within the K-Pop world becomes crucial. Sentiment analysis is a powerful tool to study these reactions, as it helps in understanding the overall sentiment and opinion trends on social media platforms. By analyzing these sentiments, researchers can gauge public opinion and its potential impact on the individuals or entities involved. In this study, the author will focus on examining a specific controversy widely discussed in April 2022, namely the bullying rumors involving LE SSERAFIM member Kim Garam.

The case began when HYBE x Source Music introduced Kim Garam as a member of the group LE SSERAFIM on April 5, 2022, which led to several social media posts on Twitter alleging that Kim Garam had a poor record in middle school. Kim Garam was accused of bullying her schoolmates. Since then, controversial pre-debut photos of Kim Garam began circulating. However, LE SSERAFIM's agency, Source Music, denied the bullying accusations against Kim Garam, stating that she was actually a victim of bullying at her school, including malicious rumors and cyberbullying. The agency's statement, which contradicted the victim's claims and was supported by insufficient evidence, raised various questions among netizens regarding the truth from both sides. With the emergence of evidence and news, the case received attention and responses from the public, including non-K-pop enthusiasts, who commented on the case through their roles as netizens by tweeting on social media platforms like Twitter. You can conduct a sentiment analysis to understand the Indonesian netizens' reactions to the bullying case involving K-pop idol Kim Garam, determining whether the sentiment tends to be positive or negative.

Sentiment analysis is a method used to determine the tendency or opinion about an object or issue, whether it is positive, negative, or neutral [3]. Sentiment analysis enables text processing to understand how Indonesian netizens react to frequently discussed information on Twitter. Several classification methods can be used in sentiment analysis, including Support Vector Machines (SVM) and Naïve Bayes Classifier (NBC) [4]. The Naïve Bayes Classifier and Support Vector Machine methods fall under supervised learning classification. This means that machine learning is trained to recognize data based on the specific labels provided [5]. The Naïve Bayes classifier is a simple probabilistic classification method that calculates probabilities by summing the value-frequency pairs from the given dataset [6]. Support Vector Machine is an algorithm that transforms the original training data into a higher-dimensional space using a non-linear mapping to find the hyperplane or separator that maximizes the distance between classes of data [7].

Based on the explanation above, a study is being conducted to extract data from the social media platform Twitter and perform sentiment analysis on Indonesian netizens' reactions to the bullying case involving K-pop idol Kim Garam. In this study, sentiment analysis is used to process tweet response data to determine whether the responses fall into the positive or negative category. This sentiment analysis applies the simple and highly accurate Naïve Bayes Classifier (NBC) algorithm [8]. Additionally, the study uses the Support Vector Machine (SVM) algorithm as a comparison to the Naïve Bayes Classifier algorithm to achieve more accurate results.

2. Methods

The methods of this research is referred to figure 1.

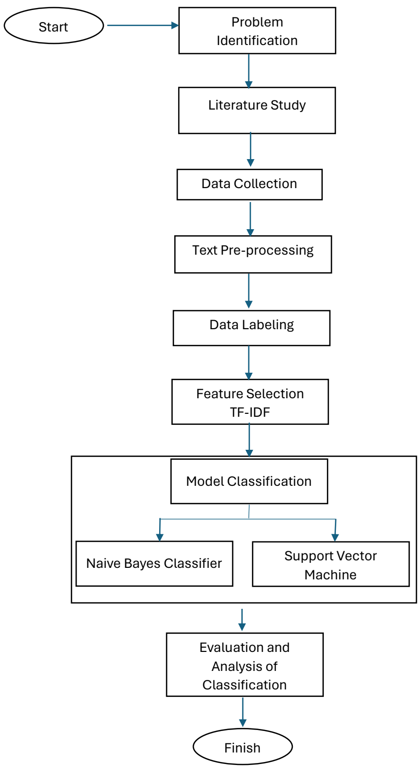

Figure 1: Research flowchart.

2.1. Problem Identification

The initial step of this research is to identify and formulate the issues to be studied. Then, determine the topic and method to be used in the research. The topic can be determined by looking at current issues or topics that are currently being discussed. Additionally, the topic determination can also be based on previous research that already exists, but with differences such as in the methods used.

2.2. Literature Study

This stage is conducted as a literature review. The research is carried out by collecting several data related to the topic of the research problem, which is the classification of bullying of K-Pop idols, and this research uses two methods, namely Naive Bayes Classifier and Support Vector Machine. Data regarding the issue can be obtained from a journal, websites and electronic media, or e-books.

2.3. Data Collection

This research collects tweet data from Twitter related to the bullying case involving K-Pop idol Kim Garam using the keywords 'kim garam', 'bullying', 'bullying case'. The data used is collected based on the trending time from the beginning of the rumor of Kim Garam's bullying case, which started on April 5, 2022. In collecting this data, the researcher retrieves the data by crawling using Google Colaboratory. Twitter data crawling is the process of retrieving or downloading data from the Twitter server. The collected data will be stored in an xlsx file and will undergo a labeling process to determine the response class of the tweets collected. The criteria for selecting tweets include relevance to the specified keywords, ensuring that only tweets containing one or more of these keywords are collected. Additionally, only tweets in the Indonesian language are considered to ensure the relevance of the analysis to the Indonesian public sentiment. The tweets are collected from the period starting April 5, 2022, when the bullying rumors began, to capture both initial and subsequent public reactions.

The crawling process is conducted using Google Colaboratory, a cloud-based platform that allows for the execution of Python code. The steps involved in the crawling process include setting up the necessary libraries such as **'tweppy'** for accessing the Twitter API and '**Pandas'** for data handling, authenticating using Twitter API keys to establish a connection to the Twitter server, and executing a search query with the specified keywords and time frame to retrieve relevant tweets. The search query is configured to filter out retweets and focus on original tweets to capture genuine public sentiment. Finally, the collected tweets are stored in an xlsx file format, including metadata such as tweet text, user handle, timestamp, and tweet ID. This systematic approach ensures that the data collected is relevant, timely, and comprehensive for the sentiment analysis.

2.4. Text Pre-processing

After the data labeling stage, the next step is text preprocessing, which aims to clean unnecessary words and less meaningful words for each tweet data. The text preprocessing steps are illustrated in diagram Figure 2.

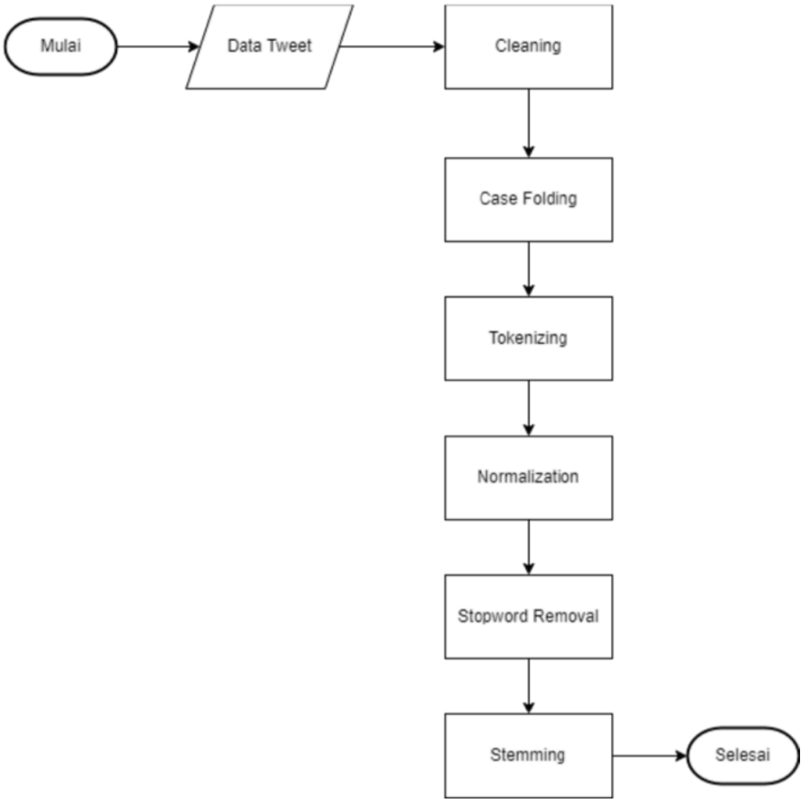

Figure 2: Flowchart of *text preprocessing*

2.4.1. Cleaning

The cleaning stage is carried out with the aim of removing punctuation, numbers, URLs, symbols, tags, usernames or mentions, emoticons, retweets, and other unnecessary characters. An example of the result of the cleaning stage is shown in Table 1.

Table 1: Cleaning stage example.

Data before *cleaning*	Data after *cleaning*
Masih gak percaya kalo bener Kim garam pelaku. Kecewa gue.	Masih gak percaya kalo bener Kim garam pelaku Kecewa gue

2.4.2. Case Folding

Case folding is a stage to change all letters in the text to lowercase. The result of case folding can be seen in Table 2.

Table 2: Case Folding stage example

Data before *case folding*	Data after *case folding*
Masih gak percaya kalo bener Kim garam pelaku Kecewa gue	masih gak percaya kalo bener kim garam pelaku kecewa gue

2.4.3. Tokenizing

The next stage is tokenizing, Breaking down sentences in the text into individual words using the nltk (Natural Language Toolkit) library. Tokenization is essential for processing each word separately and preparing the text for further analysis. An example of the result of the tokenization stage is shown in Table 3. Tokenizing stage example.

Table 3: Case Folding stage example

Data before *tokenizing*	Data after *tokenizing*
masih gak percaya kalo bener kim garam pelaku kecewa gue	‘masih’, ‘gak’, ‘percaya’, ‘kalo’, ‘bener’, ‘kim’, ‘garam’, ‘pelaku’, ‘kecewa’, ‘gue’

2.4.4. Normalization

Normalization is necessary to replace non-standard or non-standard words with standard words, and misspelled or abbreviated words into certain forms. An example of the normalization result is shown in Table 4.

Table 4: Normalization stage example

Data before normalization	Data after normalization
‘masih’, ‘gak’,‘percaya’, ‘kalo’, ‘bener’, ‘kim’, ‘garam’, ‘pelaku’, ‘kecewa’, ‘gue’	‘masih’, ‘tidak’,‘percaya’, ‘kalau’, ‘benar’, ‘kim’, ‘garam’, ‘pelaku’, ‘kecewa’, ‘saya’

2.4.5. Stopword Removal

The purpose of this step is to remove words that have no meaning or significant meaning, such as 'and', 'or', using the nltk.corpus.stopwords library. Stopwords are common words that do not contribute to the sentiment and are therefore removed to reduce noise in the data. Table 5 shows an example result of stopword removal.

Table 5: Example of stopword removal

Data sebelum stopword removal	Data setelah stopword removal
'masih', 'tidak', 'percaya', 'kalau', 'benar', 'kim', 'garam', 'pelaku', 'kecewa', 'saya'	'masih', 'tidak', 'percaya', 'benar', 'pelaku', 'kecewa'

2.4.6. Stemming

Reducing words to their base or root form using the Sastrawi library, which is specifically designed for the Indonesian language. Stemming helps in reducing the complexity of the text data by converting words to their root forms.

2.5. Data Labeling

In data classification, a common problem encountered is data imbalance [9-11]. When one class has a much larger number than the other class, data imbalance can occur, leading to decreased classification performance in the minority class. To avoid this issue, in the data labeling process, the data will only be divided into 2 classes, namely positive and negative classes. To facilitate classification, labeling for the positive class is assigned the label 0, while for the negative class, it is assigned the label 1. The result of the data labeling is shown in Table 6.

Table 6: Data labelling.

No.	Tweet	Label
1	Kecewa kalo kim garam beneran jadi pelaku bullying. Harusnya agency cari tau masa lalu trainee dulu baru dimasukin ke line up debut.	1
2	Agensinya benar-benar buruk, penggemarnya juga, mereka menunjukkan levelnya.	1
3	Kalau rumornya salah, kasihan banget asli kim garam gak salah apa apa tiba tiba dihujat. Mana dihujatnya sama netijen indo doang lagi, parah	0
4	Komentar jahat kalian ke Garam atas tuduhan konyol nggak berdasar udah buat dia tertekan gini. Lu semua yang hate Garam, harus minta maaf.	0

2.5.1. Feature Selection TF-IDF

Data from the preprocessing results then undergo the feature selection process using TFIDF weighting. This step aims to help improve accuracy in classification. TF-IDF calculates how relevant a word is in a document. Basically, the TF-IDF weight calculation works by determining the frequency of a word and comparing it with its proportion in all documents [9]. The process of calculating TFIDF is illustrated in Figure 3.

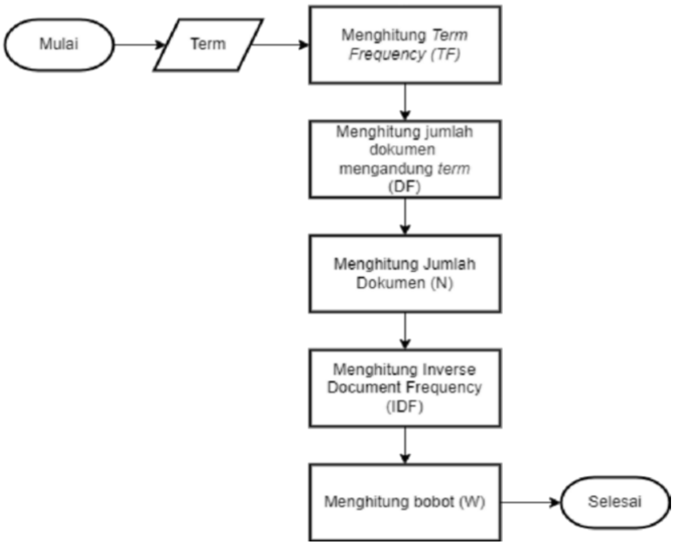

Figure 3: TF-IDF Feature Extraction Steps.

2.6. Classification using Naive Bayes Classifier and Support Vector Machine Models

The next stage of the research is to build a classification model using the Naive Bayes Classifier and Support Vector Machine methods. The implementation of the dataset testing uses the Python programming language. In the classification model, the dataset obtained will be divided into training and testing data with two different ratios, each data split into 70% : 30% and 80% : 20%.

2.6.1. Naive bayes classifier

The application of the Naive Bayes Classifier for classifying Twitter user responses to the bullying case involving Kim Garam. The process is detailed in figure 4.

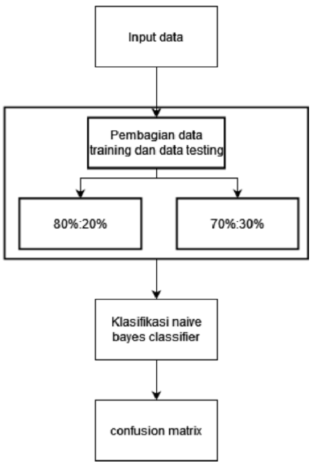

Figure 4: Naive bayes classifier flowchart.

2.6.2. Support vector machine

The application of Support Vector Machine (SVM) for classifying Twitter user responses to the bullying case involving Kim Garam is detailed in figure 5.

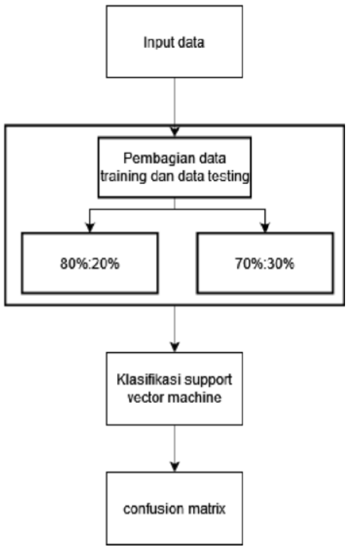

Figure 5: Support vector machine flowchart.

2.7. Evaluation and Analysis of Classification Performance

The parameters used in the evaluation stage are the confusion matrix. The evaluation is performed to see the values of accuracy, recall, precision, and f1-score by looking at the performance results of each classification model of Naive Bayes Classifier and Support Vector Machine from responses to the bullying case involving K-Pop idol Kim Garam. After receiving the evaluation results, conclusions can be drawn from the research.

3. Results and Discussion

3.1. Data Collection

The data collection process through web crawling is done using the Google Colaboratory software. The result of the crawling stage on the Twitter social media platform obtained 1150 tweets based on keywords related to the bullying case involving K-Pop idol Kim Garam, namely 'kim garam', 'bullying', '*kasus* bullying'. The crawled data is saved in an Excel file with the xlsx format.

3.2. Text Pre-processing

Because many tweets use non-standard language, preprocessing consists of several stages. This procedure is done using the Python programming language library. In this study, text preprocessing is carried out through six sequential processes, namely:

3.2.1. Cleaning

The removal of punctuation, numbers, Unicode, and unnecessary characters in a data is done in the cleaning stage. The implementation of the cleaning stage code is as shown in Figure 6.

```python
def cleaning(tweet):

    tweet = tweet.strip()                                                   # menghapus spasi putih di depan dan di belakang.
    tweet = re.sub('@[^\s]+', '', tweet)                                    # Menghapus usernames
    tweet=  re.sub(r'RT','',tweet)                                          # Menghapus retweet atau RT
    tweet = re.sub("#[A-Za-z0-9_]+", "", tweet)                             # Menghapus Hastag
    tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', ' ', tweet)         # Menghapus URLs
    tweet = re.sub(r"\d+", " ", str(tweet))                                 # Menghapus all digit
    tweet = re.sub(r"\b[a-zA-Z]\b", "", str(tweet))                         # Menghapus all single character
    tweet = re.sub(r"[^\w\s]", " ", str(tweet))                             # Menghapus semua tanda baca
    tweet = re.sub(r"\s+", " ", str(tweet))                                 # Merubah spasi ganda dengan spasi tunggal

    return tweet

data['cleaning']=data['tweet'].apply(cleaning)
data
```

Figure 6: Process of *cleaning the data.*

3.2.2. Case Folding

The letter case in a text document is changed to lowercase overall. The implementation of the case folding stage is shown in Figure 7.

```python
data['tweet_casefolding'] = data['cleaning'].str.lower()
data
```

Figure 7: Process of *case folding*

3.2.3. Tokenizing

Tokenizing or tokenization aims to separate words and symbols from a given text. With the help of the NLTK library in the Python programming language, the tokenization code is implemented as shown in Figure 8 below.

```python
from nltk.tokenize import word_tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)

data['tweet_tokenize'] = data['tweet_casefolding'].apply(word_tokenize_wrapper)
data.head()
```

Figure 8: Process of *tokenizing*

3.2.4. Normalization

The normalization stage involves cleaning words that are in the form of everyday language or slang words that are still irregular to make them proper and correct Indonesian words. The implementation of normalization using a program is shown in Figure 9 below.

```python
slank_word_dict = {"tl" : "timenline","kocak" :"lucu","elu" :"kamu", "cuma" :"hanya","sampe" : "sampai",
    "amat" : "banget","utk" : "untuk","bhw" : "bahwa","bkn" : "bukan","blm" : "belum","ga" : "tidak","klo" : "kalau",
    "yg" : "yang","dgn" : "dengan","tu" : "itu","tpi" : "tapi","moga" : "semoga","knp" : "mengapa",
    "yaiyalaahhh" : "iya","gimana" : "bagaimana","kl" : "kalau","gak" : "tidak","loe" : "kamu",
    "aja" : "saja","lu" : "kamu","terimakasih" : "terima kasih","mmg" : "memang",
    "bebera" : "beberapa","tp" : "tapi","ttp" : "tetap","tgl" : "tanggal", "udh" : "sudah",
    "lo" : "kamu","ap" : "apa", "gw" : "saya", "tp": "tapi",
    "kalo" : "kalau", "kek" : "seperti", "gini" : "begini", "ya" : "iya", "gak" : "tidak", "udah" : "sudah", "brp" : "berapa", "gk" : "tidak",
    "trimakasih" : "terima kasih","kpada" : "kepada","sy" : "saya","sdh" : "sudah","tdk" : "tidak","krn" : "karena",
    }

def slank_normalized_term(document):
    return [slank_word_dict[term] if term in slank_word_dict else term for term in document]

data['tweet_normalized'] = data['tweet_tokenize'].apply(slank_normalized_term)
data.head()
```

Figure 9: Pocess of *normalization*

3.2.5. Stopword Removal

Conjunctions or stopwords such as 'and', 'but', 'that', and other words are removed in this step. Figure 10 defines some unnecessary words, followed by the implementation of the stopword removal program.

```
from nltk.corpus import stopwords
```

```
list_stopwords = stopwords.words('indonesian')
```

```
#remove stopword pada list token

list_stopwords.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo',
                       'woi', 'amp', 'biar', 'bikin',
                       'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                       'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                       'jd', 'jgn', 'sdh', 'aja', 'loe', 't', 'haduh'
                       'nyg', 'hehe', 'pen', 'u', 'lo', 'loh', 'gue',
                       'wkwk', 'yah', 'co', 'https','http','hadehh','hadehh','lah',
                       'cuyy','yee','dah','noh','cih','sihh','halah','mah','dong',
                       'kok','bacot','anjir','woy','bund','wah','mjb','oh','kan'])

def stopwords_removal(words):
    return [word for word in words if word not in list_stopwords]

data['tweet_stop_removed'] = data['tweet_normalized'].apply(stopwords_removal)
data.head()
```

Figure 10: Process of stopword removal

3.2.6. Stemming

Every affix before and after a word is removed in the stemming process using the Sastrawi library. It starts with installing the Sastrawi library, followed by implementing stemming as shown in the program code in Figure 11 below.

```
# import Sastrawi package
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter

# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# stemmed
def stemmed_wrapper(term):
    return stemmer.stem(term)

term_dict = {}

for document in data['tweet_stop_removed']:
    for term in document:
        if term not in term_dict:
            term_dict[term] = ' '

print(len(term_dict))
print("-----------------------")
```

```
for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
    print(term,":" ,term_dict[term])

print(term_dict)
print("-----------------------")


# apply stemmed term to dataframe
def get_stemmed_term(document):
    return [term_dict[term] for term in document]

data['tweet_Stemmed'] = data['tweet_stop_removed'].swifter.apply(get_stemmed_term)
```

Figure 11: Process of stemming.

Text preprocessing has been completed, and the results are saved in a new file which will be used as the dataset for the classification process. The preprocessed data looks like the one shown in Figure 12 below.

tanggal	tweet	cleaning	tweet_casefolding	tweet_tokenize	tweet_normalized	tweet_stop_removed	tweet_Stemmed	tweet_clean
2022-07-1	Ngapain si	Ngapain si	ngapain sih kalian p	['ngapain', 'sih', 'k	['ngapain', 'sih', 'kalia	['ngapain', 'ngebela', 'pelak	['ngapain', 'ngebela',	ngapain ngebela laku rundung hadehhh banget favorite jaemin
2022-07-0	Mbak kay	Mbak kay	mbak kayaknya bel	['mbak', 'kayakny	['mbak', 'kayaknya', 'b	['mbak', 'kayaknya', 'baca',	['mbak', 'kayak', 'bac	mbak kayak baca tri didik korea nama kim garam catat laku kera
2022-06-2	Emang ad	Emang ad	emang ada berapa	['emang', 'ada', 'b	['emang', 'ada', 'berap	['emang', 'kim', 'garam', 'ke	['emang', 'kim', 'gara	emang kim garam kelas restraining order dokumen op bilang dc
2022-06-2	Di tulisnya	Di tulisnya	di tulisnya it seems	['di', 'tulisnya', 'it'	['di', 'tulisnya', 'it', 'se	['tulisnya', 'it', 'seems', 'hal	['tulis', 'it', 'seems', 'h	tulis it seems halus inti konfirmasi kim garam catat laku anyway
2022-06-2	Haduh Ud	Haduh Ud	haduh udah jelas n	['haduh', 'udah', '	['haduh', 'sudah', 'jela	['haduh', 'nyari', 'pembelaa	['haduh', 'nyari', 'bel	haduh nyari bela orang menteri didik dokumen komite keras se
2022-06-1	Menteri p	Menteri p	menteri pendidikar	['menteri', 'pendi	['menteri', 'pendidika	['menteri', 'pendidikan', 'ng	['menteri', 'didik', 'ng	menteri didik ngga garam laku nyata berita pembullian mana ki
2022-06-0	Pakar Beb	Pakar Beb	pakar beber bukti b	['pakar', 'beber', '	['pakar', 'beber', 'bukt	['pakar', 'beber', 'bukti', 'bu	['pakar', 'beber', 'bul	pakar beber bukti bully kim garam le sserafim nyata sejarah pop
2022-06-0	mau stan	mau stan	mau stan serafim t	['mau', 'stan', 'ser	['mau', 'stan', 'serafim	['stan', 'serafim', 'iya', 'gitu'	['stan', 'serafim', 'iya	stan serafim iya gitu kim garam stay sera males anjerrr stan gru
2022-05-3	Karena ka	Karena ka	karena kasusnya ki	['karena', 'kasusn	['karena', 'kasusnya',	['kasusnya', 'kim', 'garam', '	['kasus', 'kim', 'garan	kasus kim garam korea selatan laku rundung nama masuk dafta
2022-05-2	Tak Beren	Tak Beren	tak berencana kelu	['tak', 'berencana	['tak', 'berencana', 'ke	['berencana', 'keluarkan', 'k	['rencana', 'keluar', '	rencana keluar kim garam hybe tuding lindung laku bully
2022-05-2	Gws deh b	Gws deh b	gws deh buat fans l	['gws', 'deh', 'bua	['gws', 'deh', 'buat', 'fa	['gws', 'deh', 'fans', 'kim', 'g	['gws', 'deh', 'fans', 'l	gws deh fans kim garam laku bully fakta kim garam pembully ko
2022-05-2	Soalnya ya	Soalnya ya	soalnya yang diper	['soalnya', 'yang',	['soalnya', 'yang', 'dip	['dipercaya', 'sisi', 'victim', '	['percaya', 'sisi', 'vict	percaya sisi victim kak since she mentioned that she was going t
2022-05-2	Korelasiny	Korelasiny	korelasinya apa anj	['korelasinya', 'ap	['korelasinya', 'apa', 'a	['korelasinya', 'nct', 'kim', 'g	['korelasi', 'nct', 'kim	korelasi nct kim garam hah coba jelasin hubung renjun jaemin la
2022-05-2	Mulai dari	Mulai dari	mulai dari kim gara	['mulai', 'dari', 'ki	['mulai', 'dari', 'kim', 'g	['kim', 'garam', 'soojin', 'dle	['kim', 'garam', 'sooji	kim garam soojin dle deret idol pop duga laku bullying
2022-05-2	Itu kim ga	Itu kim ga	itu kim garam mem	['itu', 'kim', 'garan	['itu', 'kim', 'garam', 'n	['kim', 'garam', 'member', 'g	['kim', 'garam', 'mem	kim garam member grup le sserafim debut agens bities garam p
2022-05-2	jujur lebih	jujur lebih	jujur lebih gedek ba	['jujur', 'lebih', 'ge	['jujur', 'lebih', 'gedek'	['jujur', 'gedek', 'banget', 'h	['jujur', 'gedek', 'ban	jujur gedek banget hybe kim garam emang laku bullying korban
2022-05-2	semua bu	semua bu	semua bukti sudah	['semua', 'bukti', '	['semua', 'bukti', 'suda	['bukti', 'garam', 'pelaku', 'b	['bukti', 'garam', 'lak	bukti garam laku bullying gatau backingan garam hebat labels h
2022-05-2	@thvthv1	iya lebih t	iya lebih tepatnya	['iya', 'lebih', 'tepa	['iya', 'lebih', 'tepatny	['iya', 'tepatnya', 'dibawah'	['iya', 'tepat', 'bawah	iya tepat bawah source music agens bangsat bubarin gfriend gg
2022-05-2	Kementria	Kementria	kementrian sekolah	['kementrian', 'se	['kementrian', 'sekola	['kementrian', 'sekolah', 'm	['tri', 'sekolah', 'men	tri sekolah mengakatan tingkat hukum beda beda baca telah ke
2022-05-2	@starfess	Di dokum	di dokumen yang c	['di', 'dokumen', '	['di', 'dokumen', 'yang	['dokumen', 'crop', 'tulisan	['dokumen', 'crop', 't	dokumen crop tulis laku kelas kim garam laku tri didik confirm n
2022-07-1	sorry aku	sorry aku	sorry aku ikut kome	['sorry', 'aku', 'iku	['sorry', 'aku', 'ikut', 'k	['sorry', 'komen', 'iya', 'ema	['sorry', 'komen', 'iya	sorry komen iya emang salah garam hukum beneran salah saya
2022-07-1	dari kasus	dari kasus	dari kasus nya kim	['dari', 'kasus', 'n	['dari', 'kasus', 'nya', 'l	['kim', 'garam', 'naeun', 'he	['kim', 'garam', 'nae	kim garam naeun ajar mikir pake kepala dingin langsung ngejud

Sheet1

Figure 12: Results of *preprocessing*

3.3. Data Labeling

The crawled data that has been saved is then labeled to distinguish positive and negative responses. Manual labeling process would take a considerable amount of time, and since there is a large amount of data to be labeled, manual labeling would be a drawback. Therefore, the labeling process is done with the help of the Textblob library. To facilitate data classification in the system, positive responses are labeled as 0 and negative responses are labeled as 1. The final result of the data labeling process shows 846 comments labeled as positive and 304 comments labeled as negative. The percentage of positive and negative comment data is displayed in the graph in Figure 13.

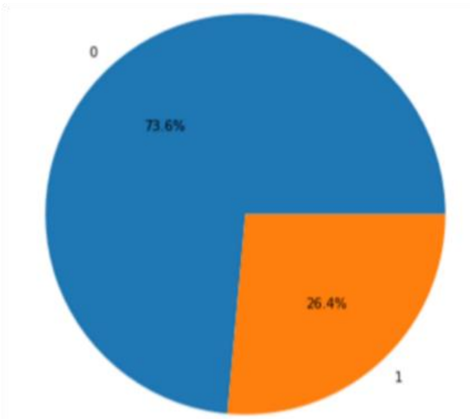

Figure 13: Data labelling results (blue means positive comments).

3.4. Feature Selection TF-IDF

In the feature extraction process, the first step is to convert the dataset into a vector representation using the Count Vectorizer library provided by Python. The documents that have been converted into word vectors are then calculated using the TF-IDF equation, which will result in word vectors that have weight values. The weight calculation for words is done by first calculating the TF or term frequency value.

3.5. Model Classification

The data obtained from the preprocessing stage is then divided into training and testing data with a ratio of 70% : 30% and 80% : 20%, where 70% and 80% are used as training data and 30% and 20% are used as testing data. The purpose of this data division is to train a model from review data and predict to obtain good performance. The machine will train based on the training data and will be matched with the data to be tested. This classification model uses the Naive Bayes Classifier and Support Vector Machine methods.

3.6. Evaluation and Performance Analysis of Classification

To evaluate how well the Naive Bayes Classifier and Support Vector Machine algorithms perform in classifying review data, evaluation needs to be done using a confusion matrix. From Tables 7 and 8, the confusion matrix will be used to calculate the accuracy value of the Naive Bayes Classifier algorithm.

Table 7: Confusion Matrix for Naive Bayes Classifier 1

confusion matrix (70 : 30)		**actual**	
		positive (0)	**negative (1)**
predicted	positive (0)	231	23
	negative (1)	0	91

Based on Table 7, there are a total of 345 tweets divided into two categories: 254 tweets are positive responses and 91 tweets are negative responses. Out of the 254 tweets predicted to contain positive responses, 231 tweets actually contain positive responses and 23 tweets are predicted to contain negative responses. Then, out of the 91 tweets containing negative responses, 91 tweets are predicted to contain negative responses and 0 tweets are predicted to contain positive responses.

Table 8: Confusion matrix of naive bayes classifier 2

confusion matrix (80 : 20)		**actual**	
		positive (0)	**negative (1)**
predicted	positive (0)	155	14
	negative (1)	0	61

Based on Table 8, there are a total of 230 tweets divided into two categories: 169 tweets are positive responses and 61 tweets are negative responses. Out of the 169 tweets predicted to contain positive responses, 155 tweets actually contain positive responses and 14 tweets are predicted to contain negative responses. Then, out of the 61 tweets containing negative responses, 61 tweets are predicted to contain negative responses and 0 tweets are predicted to contain positive responses.

In the calculations above, the accuracy values for the Naive Bayes Classifier model for the 80 : 20 and 70 : 30 data splits are 0.939 and 0.933, respectively. This means that the Naive Bayes algorithm is good to be applied in this research because it can correctly classify 93% of the data.

Table 9: Confusion matrix of support vector machine 1

confusion matrix (80 : 20)		**actual**	
		positive (0)	**negative (1)**
predicted	positive (0)	169	0
	negative (1)	5	56

Based on Table 9, there are a total of 230 tweets divided into two categories: 61 tweets are negative responses and 169 tweets are positive responses. Out of the 169 tweets predicted to contain positive responses, all 169 tweets actually contain positive responses and 0 tweets are predicted to contain negative responses. Then, out of the 61 tweets containing negative responses, 56 tweets are predicted to contain negative responses and 5 tweets are predicted to contain positive responses.

Table 10: Confusion matrix of support vector machine 2

confusion matrix (70 : 30)		**actual**	
		positive (0)	**negative (1)**
predicted	positive (0)	254	0
	negative (1)	8	83

Based on Table 10, there are a total of 345 tweets divided into two categories: 91 tweets are negative responses and 254 tweets are positive responses. Out of the 254 tweets predicted to contain positive responses, all 254 tweets actually contain positive responses and 0 tweets are predicted to contain negative responses. Then, out of the 91 tweets containing negative responses, 83 tweets are predicted to contain negative responses and 8 tweets are predicted to contain positive responses.

In the calculations above, the accuracy values for the Support Vector Machine (SVM) model for the 80 : 20 and 70 : 30 data splits are 0.978 and 0.976, respectively. This indicates that the SVM algorithm is good to be applied in this research because it can correctly classify 97% of the data.

Table 11: Comparison of Naive Bayes and Support Vector Machine Algorithms.

Split Data	**Accuracy**	
	Naïve Bayes Classifier	***Support Vector Machine***
70 : 30	93,3%	97,6%
80 : 20	93,9%	97,8%

Looking at the accuracy values from Table 11 above, it shows that the SVM algorithm is better at classification compared to Naive Bayes. For the data split into testing and training, it produces better accuracy results with an 80 : 20% split.

4. Conclusion

Based on the research conducted, two main conclusions can be drawn. Firstly, the sentiment analysis classification results in this study contain more positive comments on Twitter posts related to the bullying case of Kim Garam. This is related to the case's final outcome, which found Kim Garam not guilty, leading to many people showing support for Kim Garam, who was previously accused of bullying. Secondly, the comparison of the Naive Bayes Classifier and Support Vector Machine methods for classifying responses to the bullying case by Kim Garam on Twitter resulted in high accuracy values of 93% for the Naive Bayes Classifier and 97% for the Support Vector Machine. Based on the accuracy values, it can be concluded that the Support Vector Machine algorithm is better than the Naive Bayes Classifier algorithm as it has a higher accuracy.

References

[1] E. Fachrosi *et al.*, “Dinamika fanatisme penggemar k-pop pada komunitas btsarmy medan,” *J. Divers.*, vol. 6, no. 2, pp. 194– 201, 2020.

[2] S. C. M. Putri and S. I. Savira, “Gambaran Citra Diri Fanboy Kpop (Sebuah Studi Kasus Pada Penggemar Laki Laki Musik Korea Dalam Komunitas Fandom),” *Character J. Penelit. Psiokologi*, vol. 8, no. 6, p. 13, 2021.

[3] R. Rosdiana, T. Eddy, S. Zawiyah, and N. Y. U. Muhammad, “Analisis Sentimen pada Twitter terhadap Pelayanan Pemerintah Kota Makassar,” *Proceeding SNTEI*, no. June 2020, pp. 87–93, 2019.

[4] T. Mardiana, H. Syahreva, and T. Tuslaela, “Komparasi Metode Klasifikasi Pada Analisis Sentimen Usaha Waralaba Berdasarkan Data Twitter,” *J. Pilar Nusa Mandiri*, vol. 15, no. 2, pp. 267–274, 2019, doi: 10.33480/pilar.v15i2.752.

[5] M. Hengki and M. Wahyudi, “Klasifikasi Algoritma Naïve Bayes dan SVM Berbasis PSO Dalam Memprediksi Spam Email Pada Hotline-Sapto,” *Paradig. - J. Komput. dan Inform.*, vol. 22, no. 1, pp. 61–67, 2020, doi: 10.31294/p.v22i1.7842.

[6] M. F. Rifai, H. Jatnika, and B. Valentino, “Penerapan Algoritma Naïve Bayes Pada Sistem Prediksi Tingkat Kelulusan Peserta Sertifikasi Microsoft Office Specialist (MOS),” *Petir*, vol. 12,

no. 2, pp. 131–144, 2019, doi: 10.33322/petir.v12i2.471.

[7] M. I. Fikri, T. S. Sabrila, and Y. Azhar, “Perbandingan Metode Naïve Bayes dan Support Vector Machine pada Analisis Sentimen Twitter,” *Smatika J.*, vol. 10, no. 02, pp. 71–76, 2020, doi: 10.32664/smatika.v10i02.455.

[8] S. Lestari and S. Saepudin, “Analisis Sentimen Vaksin Sinovac Pada Twitter Menggunakan Algoritma Naive Bayes,” *SISMATIK (Seminar Nas. Sist. Inf. dan Manaj. Inform.*, pp. 163–170, 2021.

[9] S. Kaparang, D. R. Kaparang, and V. P. Rantung, “Analisis Sentimen New Normal Pada Masa Covid-19 Menggunakan Algoritma Naive Bayes Classifier,” *Jointer - J. Informatics Eng.*, vol. 2, no. 01, pp. 16–23, 2021, doi: 10.53682/jointer.v2i01.33.

[10] F. Ratnawati, “Implementasi Algoritma Naive Bayes Terhadap Analisis Sentimen Opini Film Pada Twitter,” *INOVTEK Polbeng - Seri Inform.*, vol. 3, no. 1, p. 50, 2018, doi: 10.35314/isi.v3i1.335.

[11] Aditya, C.S.K., 2024, March. Sentiment analysis regarding the impact of Covid-19 on education in Indonesia with the naïve bayes classifier. In AIP Conference Proceedings (Vol. 2927, No. 1). AIP Publishing.