Journal of Computer, Information and Technology Vol: 5, No. 2, 2025,Page: 1-20

Public Opinion Sentiment Analysis of News Trends Using the Random Forest Algorithm

Iqbal Danuraga*, Rudi Heriansyah, Lastri Widya Astuti

Universitras Indo Global Mandiri

DOI:
https://doi.org/10.53697/jkomitek.v5i2.3007
*Correspondence: Iqbal Danuraga
E-mail: 2021110006@Students.uigm.ac.id

Received: 22-10-2025
Accepted: 22-11-2025
Published: 22-12-2025

Abstract: The digital era has fundamentally changed news consumption patterns, with social media becoming the primary platform for information access used by 54% of Americans and 44% of Indonesians. This transformation creates significant challenges for researchers and communication practitioners in understanding public sentiment towards news in the digital era. This study aims to analyze public opinion sentiment towards news trends using the Random Forest algorithm. This study uses an experimental quantitative approach to analyze public opinion sentiment towards political news trends using the Random Forest algorithm. The study population consists of Twitter posts discussing political news topics from 2020 to 2025, with a sample of 1000 tweets selected through purposive sampling. The research instruments include hardware (AMD Ryzen 3 processor, 4GB RAM) and software (Python, Google Collaboratory), with data analysis techniques involving comprehensive preprocessing (cleaning, case folding, tokenization, stemming, stopword removal), TF-IDF transformation, and Random Forest implementation. The results show that the Random Forest model achieved an optimal accuracy of 81% at a 90:10 data split, with an accuracy range of 77-81% across various data split scenarios. This study concludes that Random Forest can be effectively implemented as a public sentiment monitoring instrument for governments, media, and public organizations, although larger datasets and multiple social media platforms are needed for better generalization.

Keywords: Sentiment Analysis, Machine Learning, Social Media, Opinion Mining, Random Forest

Introduction

The digital era has fundamentally changed the way people access and consume news. Social media is now the primary platform for people to obtain the latest information, with over 54% of Americans using social media as their primary news source (Campbell, 2025). In Indonesia, a similar phenomenon is occurring, with social media use for news reaching 44% of the total population. Although this has decreased from 58% in 2021, it remains a dominant information channel (Digital News Report, 2025). This transformation has not only changed news consumption patterns but also created new spaces for the formation and dissemination of more dynamic and interactive public opinion.

The viral phenomenon on social media has become a force capable of shaping public perception on various issues in a very short time. Platforms like Twitter, Facebook, and TikTok function not only as information distribution channels but also as digital public spaces where people actively express their opinions and sentiments on various events (Kim & Anderson, 2025). Recent research shows that social media has a strong agenda-setting

function, where trending issues on digital platforms can influence public perception more significantly than traditional media, with a higher level of polarization in content consumption compared to content production itself (Risnanda et al, 2025).

The complexity of understanding public sentiment toward news in the digital age presents significant challenges for researchers and communication practitioners. The massive volume of data and the rapid spread of information on social media necessitate the development of analytical methods capable of effectively and accurately processing and classifying public opinion (Firdaus et al, 2024). Traditional manual approaches to analyzing public sentiment become impractical when dealing with the millions of posts, comments, and interactions that occur daily across various social media platforms, particularly in the context of viral or trending news.

The need for sophisticated sentiment analysis systems is becoming increasingly urgent, given the potential impact of misinterpretation on public opinion. Sentiment analysis using machine learning techniques has proven effective in classifying public opinion, with the Random Forest algorithm demonstrating superior performance compared to other methods in various research contexts (Jumaryadi et al., 2025). Recent research demonstrates that the Random Forest Classifier can achieve up to 88-92% accuracy in analyzing sentiment on a variety of topics, from e-commerce applications to political issues (Hadi et al, 2025) (Musthafa et al, 2025).

Although various studies have applied sentiment analysis to social media, there is a significant gap in research specifically analyzing public opinion sentiment toward news trends using Random Forest. Previous research has focused more on sentiment analysis of individual commercial products or political candidates, but none has comprehensively explored how the public responds to various news trends developing on social media (Muliana et al., 2024). Furthermore, the majority of existing research uses limited data or focuses on a single social media platform, whereas a holistic understanding of public sentiment requires analysis across time and broader contexts (Pratama, 2025).

This study aims to analyze public opinion sentiment towards news trends using the Random Forest algorithm, utilizing Twitter data from 2020 to 2025, thus providing a comprehensive picture of how the public responds to various emerging news issues. The urgency of this research lies in the need to understand the dynamics of public opinion in the post-truth era, where information and disinformation are mixed in the digital space, as well as the importance of developing instruments that can assist stakeholders in understanding public perceptions of various public issues (Samsir et al, 2024). The novelty of this research lies in the use of a longitudinal dataset spanning five years, the implementation of Random Forest optimized for sentiment analysis of Indonesian-language news, and a holistic approach that not only classifies sentiment but also identifies temporal and thematic patterns in public responses to news trends (Tahyudin et al, 2024).

Methodology

Types and Methods of Research

This study uses a quantitative approach with an experimental method to analyze public opinion sentiment towards news trends using the Random Forest algorithm. According to Sugiyono (2021), quantitative research methods are research methods based on the philosophy of positivism, used to study specific populations or samples, with data collection using research instruments and quantitative or statistical data analysis, with the aim of testing predetermined hypotheses. Experimental quantitative research was chosen because it aims to analyze and compare the performance of machine learning algorithms in classifying sentiment, where researchers can control the variables that influence the results of sentiment analysis (Sudaryono, 2021).

The experimental approach in this study refers to the concept of Creswell & Creswell (2023), which states that experimental research is a quantitative research procedure in which researchers determine whether a treatment affects research results by comparing one or more treatment groups with a control group. In the context of this study, the treatment in question is the application of the Random Forest algorithm to preprocessed text data, while the controlled variables include the text preprocessing method, model parameters, and evaluation techniques used. This method was chosen because it allows researchers to objectively measure the algorithm's performance in classifying sentiment using standardized evaluation metrics such as accuracy, precision, recall, and F1-score (Semary et al, 2024) (Sihombing et al, 2024).

Data Analysis Instruments and Techniques

The research instruments used in this study consisted of hardware and software that supported the process of collecting and analyzing sentiment data. The hardware used included a computer with an AMD Ryzen 3 processor and 4GB of RAM, sufficient to run machine learning algorithms on medium-sized datasets. According to Sudaryono (2021), the selection of research instruments must consider the technical requirements and computing capabilities required to optimally perform data analysis. The software used included a 64-bit Windows operating system, the Python programming language as the primary platform for algorithm implementation, and Google Colaboratory as a development environment that provides access to sufficient cloud computing resources for training machine learning models.

The data analysis technique in this study adopted a systematic approach consisting of several main stages in accordance with the principles of quantitative data analysis proposed by Emzir (2012). The first stage is data preprocessing, which includes data cleaning to remove noise, case folding for text normalization, tokenization to break text into smaller units, stemming to convert words to their basic form, and stopword removal to eliminate meaningless words. The second stage is data transformation using Term Frequency-Inverse Document Frequency (TF-IDF) to convert text data into numeric representations that can be processed by machine learning algorithms. The third stage is the implementation of the Random Forest algorithm, which uses ensemble learning with multiple decision trees to produce more accurate and robust predictions. Evaluation techniques used include

confusion matrix, accuracy, precision, recall, F1-score, and ROC-AUC curves to comprehensively measure model performance (Jumaryadi et al., 2025; Firdaus et al., 2024).

Population and Sample

The population in this study is all posts or tweets on the social media platform Twitter that discuss political news topics between 2020 and 2025. According to Sugiyono (2021), a population is a generalized area consisting of objects or subjects that have certain qualities and characteristics determined by the researcher to be studied and then drawn conclusions. The characteristics of the population in this study include posts that use Indonesian, have content relevant to political news trends, and are published within a predetermined timeframe. This population was chosen because Twitter is the social media platform most actively used by Indonesians to express political opinions and respond to current news developments, thus representing public sentiment more broadly and in real time.

The research sample was determined using purposive sampling, a technique for selecting data sources based on specific considerations, where researchers have specific objectives in sampling. Creswell & Creswell (2023) state that purposive sampling allows researchers to deliberately select participants and research locations to help researchers understand the problem and research questions. The sample criteria used in this study included: tweets containing opinions or comments on political news, tweets predominantly using Indonesian, tweets that did not contain spam or commercial promotional content, and tweets that had adequate text structure for analysis. The sample size used follows the principle of data sufficiency in machine learning, where, according to Sudaryono (2021), the minimum sample size for machine learning research is 10 times the number of features used. Therefore, considering the complexity of the Random Forest model and the need for validation, this study used a sample of approximately 8,000–10,000 tweets distributed proportionally across positive, negative, and neutral sentiment categories.

Research Procedures

The research procedure was implemented in four main stages, following the experimental quantitative research methodology proposed by Sugiyono (2021). The first stage was data collection, which was carried out through a process of crawling or scraping data from the Twitter platform using APIs or specialized tools to extract tweets relevant to predetermined criteria. The data collection process was carried out systematically, taking into account research ethics and the social media platform's data usage policies. The collected data was then stored in a structured format, and initial validation was conducted to ensure data quality and relevance in accordance with the research objectives. This stage also involved a data selection process based on predetermined criteria to ensure sample homogeneity and representativeness.

The second stage is data preprocessing, which is a crucial step in preparing text data for machine learning analysis. According to Emzir (2012), data preprocessing in both qualitative and quantitative research must be carried out systematically to ensure the validity and reliability of the analysis results. The preprocessing process includes data cleaning to remove unwanted characters, case folding to convert all letters to lowercase,

tokenization to break text into individual tokens, stemming to convert words to their basic form using the Sastrawi algorithm for Indonesian, and stopword removal to remove meaningless words. After preprocessing is complete, the data is transformed using the TF-IDF (Term Frequency-Inverse Document Frequency) technique to convert text data into numeric vector representations that can be processed by machine learning algorithms (Musthafa et al, 2025) (Hadi et al, 2025).

The third stage is the implementation and training of the Random Forest model, which follows the principles of ensemble learning to produce more accurate and robust predictions. The implementation process begins by dividing the dataset into training data (80%) and testing data (20%) using stratified sampling to maintain the proportion of sentiment classes. The Random Forest model is configured with optimal parameters determined through a hyperparameter tuning process using grid search or random search techniques. Model training is performed on the training data using cross-validation to validate model performance and prevent overfitting. The training process involves building multiple decision trees, each trained on a different subset of the data with randomly selected features, in accordance with the characteristics of the Random Forest algorithm (Creswell & Creswell, 2023).

The fourth stage is comprehensive model testing and evaluation, using various evaluation metrics to measure the model's performance in sentiment classification. Testing is conducted on test data never seen by the model during the training process to ensure the objectivity of the evaluation results. The evaluation metrics used include a confusion matrix to visualize the classification results, accuracy to measure the proportion of correct predictions, precision to measure the accuracy of positive predictions, recall to measure the model's ability to detect positive classes, F1-score as the harmonic mean of precision and recall, and the ROC-AUC curve to measure the model's discrimination ability. The evaluation results are then analyzed and interpreted in the context of the research objectives by comparing the model's performance with a predetermined benchmark or baseline. The entire research process is documented in detail to ensure the reproducibility and transparency of the research results (Sudaryono, 2021) (Tahyudin et al, 2024).

Results and Discussion

Data collection

In the previous chapter, this study conducted data analysis by collecting reviews or public opinions regarding Indonesian political news sourced from platform X between 2020 and 2025. In this chapter, the focus of the research is directed at presenting the results and discussing the results obtained from this process. A total of 1,000 review data points were collected by the researcher and used in sentiment analysis using the Random Forest method. This can be seen in Table 1.

Table 1. Dataset

No	Data
0	The dongo is in a hurry to knock the hammer, come on, let's get on the HT, come on.
1	Is there a definite date for the discussion in the near future or not???
2	I don't know yet, but I saw this news and felt quite concerned.
3	Don't let services that perform well become weak.
4	Democracy is heading towards a dark period.
….	
995	stupid
996	It is indeed the crust of hell.
997	10 years of raising
998	That's so bad
999	Tai's proposal

Pre-Processing

The Random Forest algorithm was chosen to evaluate the accuracy level of sentiment analysis on public opinion regarding Indonesian Political News on the social media platform X. Before the classification process was carried out, the initial step in this study

```
# meload data
sample_1000data = pd.read_excel('/content/drive/MyDrive/Sample/dataulasan.xlsx')

# menampilkan data
print(sample_1000data)
```

```
                                                Ulasan
0    Si dongo pada kebutan ketok paluuuuu, yok naik...
1    ini tgl pasti buat pembhasannya dlm waktu deka...
2    belum tau tapi aku liat berita ini merasa cuku...
3    Jangan sampai kejasaan yg kinerjanya bagus mal...
4                        Demokrasi menuju masa suram.
..                                                ...
995                                       tolol ah
996                                Memang kerak neraka
997                                  10 tahun melihara
998                                    Asuuuuu dah lah
999                                         Usulan tai

[1000 rows x 1 columns]
```

Figure 1. Data Retrieval

was to call the operator and access the platform. The data file that had been sorted or labeled previously. The data was in XLS format and saved with the name "dataulasam.xlsx." After the data was successfully collected, the next stage was to carry out a preprocessing process to prepare the data for further analysis. The following is Figure 1: Data operator call process

Data Cleaning

Using a dataset of 1,000 reviews, the initial preprocessing step involved data cleaning. The goal of this process was to remove irrelevant characters, such as punctuation and special symbols. The code implementation for this cleaning step is shown in Figure 2.

```
[52] def clean_text(text):
         # Menghapus karakter khusus, angka, dan tanda baca
         text = re.sub(r'[^a-zA-Z\s]', '', text)

         # # Menghapus emoji tertentu
         text = re.sub(r'😋|😻|❤️|👍|😇', '', text)

         return text
```

Figure 2. Cleaning

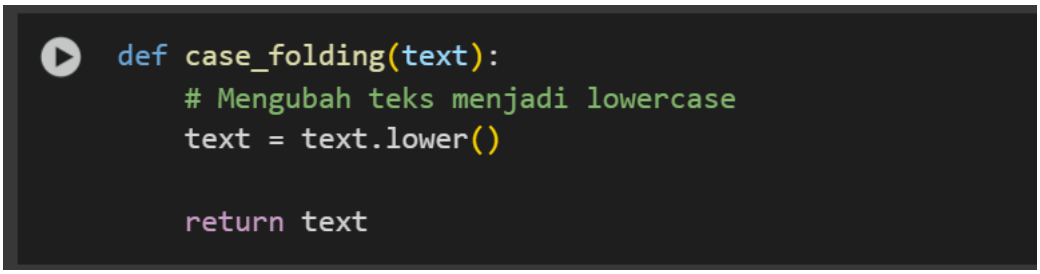
The data cleaning process, as shown in the figure above, involves transforming the review text data to remove irrelevant characters, such as symbols, numbers, punctuation, and common words that do not contribute significantly to sentiment analysis. Furthermore, a word normalization process is also performed to ensure consistency in word representation, as shown in Table 2.

Table 2. Before Cleaning and After Cleaning

No	Data before cleaning	Data after cleaning
0	The dongo is in a hurry to knock the hammer, come on, let's get on the HT, come on.	The dongo is racing to knock on the hammer, come on, let's get on the HT, come on.
1	Is there a definite date for the discussion in the near future or not???	This is the exact date for the discussion in the near future. There is already a date.
2	I don't know yet, but I saw this news and felt quite concerned.	I don't know yet, but I saw this news and felt quite concerned.
3	Don't let services that perform well become weak.	Don't let services that perform well become weak.
4	Democracy is heading towards a dark period.	Democracy is heading towards a dark period.
….		
995	stupid	stupid
996	It is indeed the crust of hell	It is indeed the crust of hell
997	10 years of raising	years of care
998	That's so bad	That's so bad
999	Tai's proposal	Tai's proposal

Folding Case

Case folding is the process of converting all letters to lowercase. This process converts the characters 'A'-'Z' in the data into the characters 'a'-'z'. The following procedure for calling the case folding program code can be seen in Figure 3.



Picture 2. Folding Case

Examples of data after cleaning and before case folding can be seen in Table 3.

Table 1. Data after cleaning and before case folding

No	Data after cleaning has not been case-folded	Data After Folding Case
0	The dongo is racing to knock on the hammer, come on, let's get on the HT, come on.	The dongo is speeding and knocks his hammer, let's get his head up, yuuuuuu
1	This is the exact date for the discussion in the near future. There is already a date.	This is the exact date to discuss in the near future. There is already a task
2	I don't know yet, but I saw this news and felt quite concerned.	I don't know yet, but I saw this news and I feel quite concerned.
3	Don't let services that perform well become weak.	Don't let a service that does good work become weak
4	Democracy is heading towards a dark period	democracy is heading towards a dark period
….		
995	stupid	stupid
996	It is indeed the crust of hell	Indeed, the crust of hell
997	years of care	year of care
998	That's so bad	That's enough
999	Tai's proposal	Tai's proposal

Tokenize

At this stage, each word in a sentence in the document is separated. Word separation is generally done using spaces. While writing styles can vary, the main goal is to break the sentence into its constituent words. The following is the process of calling the tokenize program code, as seen in Figure 4.

```
# kode untuk tokenizing
from nltk.tokenize import RegexpTokenizer

regexp = RegexpTokenizer (r'\w+|$[0-9]+|\S+')
sample_1000data['Token'] = sample_1000data['Ulasan'].apply(regexp.tokenize)

# Save the DataFrame to an Excel file
output_file_path = 'tokenizing.xlsx'
sample_1000data.to_excel(output_file_path, index=False)
print(f"DataFrame berhasil disimpan ke file Excel: {output_file_path}")
```

Figure 4. Tokenize

Examples of data after case folding and before tokenization can be seen in Table 4.

Table 4. Data after case folding and before tokenization

No	Data after Case folding before Tokenization	Data after tokenization
0	The dongo is speeding and knocks his hammer, let's get his head up, yuuuuuu	['si', 'dongo', 'pada', 'kebut', 'knock', 'palu', 'yok', 'up', 'ht', 'nya', 'yoo']
1	This is the exact date to discuss in the near future. There is already a task	['this', 'date', 'definitely', 'make', 'discuss', 'in', 'time', 'near', 'already', 'there is', 'ta']
2	I don't know yet, but I saw this news and I feel quite concerned.	['not yet', 'know', 'but', 'I', 'see', 'news', 'this', 'feel', 'enough', 'concern']
3	Don't let a service that does good work become weak	['don't', 'until', 'service', 'that', 'work', 'good', 'even', 'so', 'weak']
4	democracy is heading towards a dark period	['democracy', 'goal', 'period', 'gloomy']
….		
995	stupid	['stupid', 'ah']
996	Indeed, the crust of hell	['indeed', 'crust', 'hell']
997	year of care	['year', 'maintain']
998	That's enough	['asu', 'dah', 'lah']
999	Tai's proposal	['proposal', 'tai']

Stemming

Stemming: Converting words in a document to their base or root form. The stemming process in Indonesian documents is quite complex because it requires removing all affixes from the words. The following is an example of the stemming program code call process. It can be seen in Figure 5.

```
# Create a stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# Define the stemming function
def stem_tokens(tokens):
    return [stemmer.stem(token) for token in tokens]

# Assuming sample_['Token'] is a  column containing lists of tokens
sample_1000data['Token'] = sample_1000data['Token'].apply(stem_tokens)

# Display the DataFrame after stemming
print("Data Setelah Proses Stemming:")
print(sample_1000data)

# Save the DataFrame to an Excel file
output_file_path = 'stemming.xlsx'
sample_1000data.to_excel(output_file_path, index=False)
# print(f"DataFrame berhasil disimpan ke file Excel: {output_file_path}")
```

Picture 5. Stemming

Examples of data that have been cleaned, case-folded, tokenized, and stemmed can be seen in Table 5.

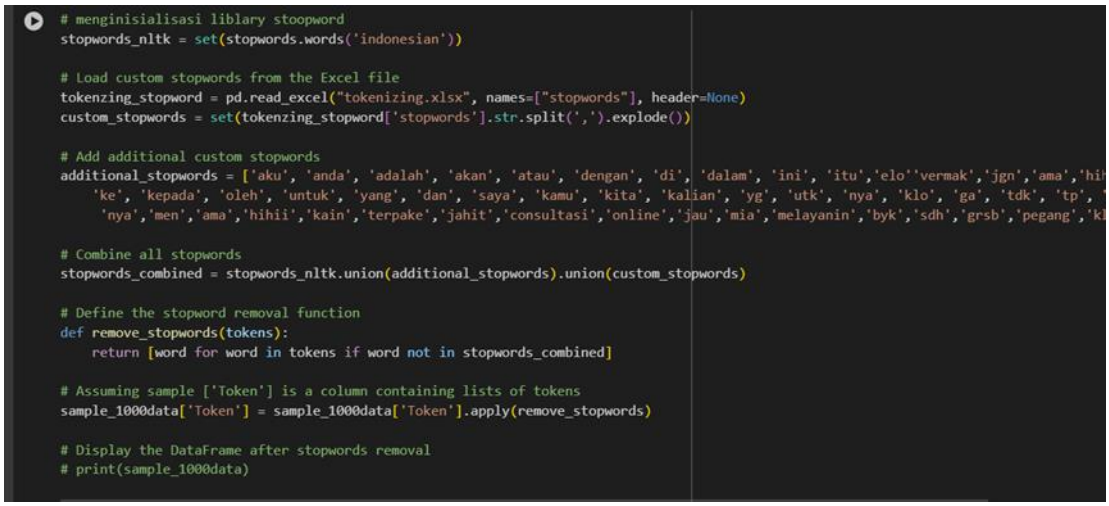
Table 2. Data After Tokenization, and Before Stemming

No	Data After Tokenization before Stemming	Data after Stemming
0	['si', 'dongo', 'pada', 'kebut', 'knock', 'palu', 'yok', 'up', 'ht', 'nya', 'yoo']	['si', 'dongo', 'pada', 'kColab 'knock', 'palu']
1	['this', 'date', 'definitely', 'make', 'discuss', 'in', 'time', 'near', 'already', 'there is', 'ta']	['this', 'date', 'definitely', 'make', 'discuss', 'in', 'time', 'near', 'already', 'there is', 'ta']
2	['not yet', 'know', 'but', 'I', 'see', 'news', 'this', 'feel', 'enough', 'concern']	['not yet', 'know', 'but', 'I', 'see', 'news', 'this', 'feel', 'enough', 'concern']
3	['don't', 'until', 'service', 'that', 'work', 'good', 'even', 'so', 'weak']	['don't', 'until', 'service', 'that', 'work', 'good', 'even', 'so', 'weak']
4	['democracy', 'goal', 'period', 'gloomy']	['democracy', 'goal', 'period', 'gloomy']
….		
995	['stupid', 'ah']	['stupid', 'ah']
996	['indeed', 'crust', 'hell']	['indeed', 'crust', 'hell']
997	['year', 'maintain']	['year', 'maintain']
998	['asu', 'dah', 'lah']	['asu', 'dah', 'lah']
999	['proposal', 'tai']	['proposal', 'tai']

Stopword Remover

At this stage, we filter out commonly used words or words that rarely contribute significantly to meaning, known as stopwords. This process is called "Stopword Removal." By removing these words, this process can improve classification effectiveness, reduce data scatter, and significantly reduce the dimensionality of the feature space.

The following is an example of the process of calling the Stopword Removal program code. It can be seen in Figure 6.



```
# menginisialisasi liblary stoopword
stopwords_nltk = set(stopwords.words('indonesian'))

# Load custom stopwords from the Excel file
tokenzing_stopword = pd.read_excel("tokenizing.xlsx", names=["stopwords"], header=None)
custom_stopwords = set(tokenzing_stopword['stopwords'].str.split(',').explode())

# Add additional custom stopwords
additional_stopwords = ['aku', 'anda', 'adalah', 'akan', 'atau', 'dengan', 'di', 'dalam', 'ini', 'itu','elo''vermak','jgn','ama','hi
    'ke', 'kepada', 'oleh', 'untuk', 'yang', 'dan', 'saya', 'kamu', 'kita', 'kalian', 'yg', 'utk', 'nya', 'klo', 'ga', 'tdk', 'tp',
     'nya','men','ama','hihii','kain','terpake','jahit','consultasi','online','jau','mia','melayanin','byk','sdh','grsb','pegang','k

# Combine all stopwords
stopwords_combined = stopwords_nltk.union(additional_stopwords).union(custom_stopwords)

# Define the stopword removal function
def remove_stopwords(tokens):
    return [word for word in tokens if word not in stopwords_combined]

# Assuming sample ['Token'] is a column containing lists of tokens
sample_1000data['Token'] = sample_1000data['Token'].apply(remove_stopwords)

# Display the DataFrame after stopwords removal
# print(sample_1000data)
```

Figure 6. Stop Word Removal

Examples of data after cleaning, case folding, tokenizing, stemming, and before stopword removal can be seen in Table 6.

Table 3. Data after Stemming and before Stopword Removal

No	Data after Stemming and before Stopword Removal	Data after Stopword Removal
0	['si', 'dongo', 'pada', 'kebut', 'knock', 'palu']	['dongo', 'kebut', 'knock', 'palu']
1	['this', 'date', 'definitely', 'make', 'discuss', 'in', 'time', 'near', 'already', 'there is', 'ta']	["date', 'definitely', 'make', 'discuss', 'time', 'close', 'already', 'there is', 'ta']
2	['not yet', 'know', 'but', 'I', 'see', 'news', 'this', 'feel', 'enough', 'concern']	['not yet', 'know', 'see', 'news', 'feel', 'enough', 'concern']
3	['don't', 'until', 'service', 'that', 'work', 'good', 'even', 'so', 'weak']	['service', 'work', 'good', 'weak']
4	['democracy', 'goal', 'period', 'gloomy']	['democracy', 'goal', 'period', 'gloomy']
….		
995	['stupid', 'ah']	['stupid']
996	['indeed', 'crust', 'hell']	['indeed', 'crust', 'hell']
997	['year', 'maintain']	['year', 'maintain']
998	['asu', 'dah', 'lah']	['asu', 'dah']
999	['proposal', 'tai']	['proposal', 'tai']

TF-IDF labeling process

This label determination process is carried out by utilizing the method TF-IDF. The TF-IDF Weighting Program Code can be seen in Figure 4.7, and the TF-IDF Weighting Results can be seen in Figure 7.

```python
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd

# Read the Excel file into a DataFrame
# Membaca data excel ke dataframe
sample_stemming = pd.read_excel('stemming.xlsx')

# Initialize the TfidfVectorizer object
vectorizer = TfidfVectorizer()

# Perform vectorization and TF-IDF calculation
tfidf_matrix = vectorizer.fit_transform(sample_stemming['Token'])
# Get the list of features (words) used for vectorization
feature_names = vectorizer.get_feature_names_out()

# Create a list of TF-IDF dictionaries for each review
tfidf_dict_list = []
for i in range(len(sample_stemming)):
    feature_index = tfidf_matrix[i, :].nonzero()[1]
    tfidf_scores = zip(feature_index, [tfidf_matrix[i, x] for x in feature_index])
    tfidf_dict = {feature_names[i]: score for i, score in tfidf_scores}
    tfidf_dict_list.append(tfidf_dict)

# Display the TF-IDF calculation results
for i, tfidf_dict in enumerate(tfidf_dict_list, 1):
    print(f'{tfidf_dict}')
```

Figure 7. Weighting Code

After running the code above on the Google Colab platform, you will get the TF-IDF weighting data results, which can be seen in Figure 4.8.

```
{'si': np.float64(0.23254928737321048), 'dongo': np.float64(0.36080578662096513), 'kebut': np.float64(0.36080578662096513), 'ketok': np.float64(0.2981632271259672), 'paluuuuu': np.float64(0.36080578662096513), 'yok': np.float64(0.36080578662096513), 'naikin': np.float64(0.27556237946606676), 'ht': np.float64(0.36080578662096513), 'yuuuuuuuu': np.float64(0.36080578662096513)}
{'tgl': np.float64(0.5232144509523022), 'pembahasannya': np.float64(0.5232144509523022), 'dlm': np.float64(0.4567728456364975), 'ta': np.float64(0.4938135722299442)}
{'tau': np.float64(0.4382021061867628), 'liat': np.float64(0.47662447875265856), 'berita': np.float64(0.40819148795981824), 'concerning': np.float64(0.6435741834042064)}
{'service': np.float64(0.5535645506716506), 'work': np.float64(0.423365710178036), 'good': np.float64(0.5019116702207536), 'weak': np.float64(0.5122620803664414)}
{'democracy': np.float64(0.5951492274006439), 'gloomy': np.float64(0.803615204637404)}
{'ah': np.float64(0.7321096235984696), 'tolol': np.float64(0.6811868312251105)}
{'hell': np.float64(0.6705770844095685), 'crust': np.float64(0.7418398572904819)}
{'maintain': np.float64(1.0)}
{'dah': np.float64(0.579903705149402), 'asuuuuu': np.float64(0.8146850267152302)}
{'tai': np.float64(0.6587529743858644), 'proposed': np.float64(0.7523593016224205)}
```

Figure 8. TF-IDF weighting

After TF-IDF word weighting, the next step is the TF-IDF labeling process. This can be seen in Figure 9. Based on the code created, the positive and negative sentiment data can be seen.

```
# Data TF-IDF
data_tfidf = pd.read_excel('tfidf_results.xlsx')

# Mengambil kolom 'combined' dari DataFrame data_tfidf
tfidf_data = data_tfidf['combined']

# Probabilitas prior
p_positif = 0.5
p_negatif = 0.001

# Klasifikasi dokumen
def predict_sentiment(tfidf):
    # Menghitung probabilitas kemunculan fitur pada kelas positif
    p_x_c_pos = np.prod(np.array(list(tfidf.values())))

    # Menghitung probabilitas kemunculan fitur pada kelas negatif
    p_x_c_neg = np.prod(1 - np.array(list(tfidf.values())))

    # Menghitung probabilitas bersyarat
    p_c_x_pos = (p_x_c_pos * p_positif) / (p_x_c_pos * p_positif + p_x_c_neg * p_negatif)
    p_c_x_neg = (p_x_c_neg * p_negatif) / (p_x_c_pos * p_positif + p_x_c_neg * p_negatif)

    # Klasifikasi berdasarkan probabilitas posterior
    if p_c_x_pos > p_c_x_neg:
        return "POSITIF"
    else:
        return "NEGATIF"
```

Figure 9. Labeling

Table 4. TF-IDF Labeling

No	TF-IDF VALUES	Sentiment
0	{'si':(0.23), 'dongo':(0.36), 'kebut': (0.36), 'ketok':(0.29), 'paluuuuu':(0.36), 'yok':(0.36), 'raikin':(0.27), 'ht':(0.36), 'yuuuuuuuu':(0.36)}	NEGATIVE
1	{'date'(0.52) 'discussion':(0.52), 'in':(0.45)}	POSITIVE
2	{'tau':(0.43), 'liat':(0.47), 'berita':(0.40), 'concerning': (0.64)}	POSITIVE
3	{'service': (0.55), 'work': (0.42), 'good':(0.50), 'weak': (0.51)}	POSITIVE
4	{'democracy': (0.59), 'gloomy':(0.80)}	POSITIVE
….		
995	{'ah': (0.73), 'stupid': (0.68)}	POSITIVE
996	{'hell': (0.67), 'crust':(0.74)}	POSITIVE
997	{'maintain': (1.0)}	POSITIVE
998	{'dah': (0.57), 'asuuuuu': (0.81)}	POSITIVE
999	{'tai': (0.65), 'suggestion': (0.75)}	POSITIVE

Random Forest Implementation

In the chapter on the Random Forest implementation process, the primary focus is on evaluating the performance of the trained Random Forest model for sentiment analysis. This evaluation is conducted using a test dataset during the training process, as shown in Figure 10.

```
[ ]  # Meload dataset
     df = pd.read_excel('/content/combined_data.xlsx')

     # Mengabungkan data yang yang ada string menjadi satu
     df['text'] = df['Token'].str.replace(r"\[|\]|\'", "", regex=True)
```

Figure 10. Dataset Utilization

After completing the dataset call, the next step is to integrate the Random Forest function or program code into the Google Colab platform. At this stage, the Random Forest algorithm becomes a key component in the sentiment analysis process, with the role of recognizing and processing patterns in the data. This function is used to classify sentiment in text based on learning from previously input data. This integration process forms the primary basis for building an accurate sentiment prediction model. This stage also serves as a basis for evaluating Random Forest's performance when faced with previously unanalyzed test data, as shown in Figure 11.

```
# Membagi data menjadi set pelatihan dan set pengujian
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)

# Vektorisasi data teks menggunakan TF-IDF
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Membuat dan latih RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train_tfidf, y_train)

# Membuat Prediksi
y_pred = rf.predict(X_test_tfidf)
```

Figure 11. Dataset Calling

To call the Random Forest function used in the classification model training process, by utilizing the RandomForestClassifier class from the scikit-learn library in the Python programming language. Before training begins, an instance of the class is first created using the rf = RandomForestClassifier() command. Then, the fit method is used to train the model with the training data through the syntax rf. Fit (x_train_tfidf, y_train), where x_train_tfidf is the result of text feature transformation using the TF-IDF method, and y_train is the label or target of the training data. After this stage is complete, the RF object contains a trained Random Forest model and is ready to be used to make predictions on previously unknown data.

Random Forest Testing

After the Random Forest method call phase, the next step in the process is to conduct initial testing, which includes using a Confusion Matrix to measure accuracy. The method applied to the test dataset will generate responses that are then evaluated using accuracy metrics. This initial testing provides an overview of the performance of the implemented

Random Forest model and serves as a basis for further development or parameter adjustments to improve its performance. The information obtained from this stage is crucial for understanding the reliability of Random Forest in classifying sentiment on previously unprocessed data. The calculation process and display of Random Forest accuracy can be seen in Figure 12.

```
#implementasi dan evaluasi model Random Forest
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, precision_score, recall_score, f1_score

# Menggabungkan token menjadi satu string per baris
df['text'] = df['Token'].apply(lambda x: ' '.join(eval(x)))

# Membagi data menjadi fitur (X) dan target (y)
X = df['text']
y = df['label_sentimen_encoded']

# Membagi data menjadi set pelatihan dan set pengujian
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)

# Vektorisasi data teks menggunakan TF-IDF
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Membuat dan latih RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train_tfidf, y_train)

# Membuat Prediksi
y_pred = rf.predict(X_test_tfidf)

# Mengevaluasi Model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted', zero_division=0)
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

# Mencetak hasil Evaluasi
print(f'Accuracy       : {accuracy:.4f}')
print(f'Precision      : {precision:.4f}')
print(f'Recall         : {recall:.4f}')
print(f'F1-Score       : {f1:.4f}')
```

Figure 3. The Process of Calculating and Displaying Random Forest Accuracy

Testing showed that the classification process was successfully applied to the test data using the Random Forest method. Researchers conducted this testing using this algorithm, and the best Confusion Matrix results obtained from the Random Forest method can be seen in Figure 13.

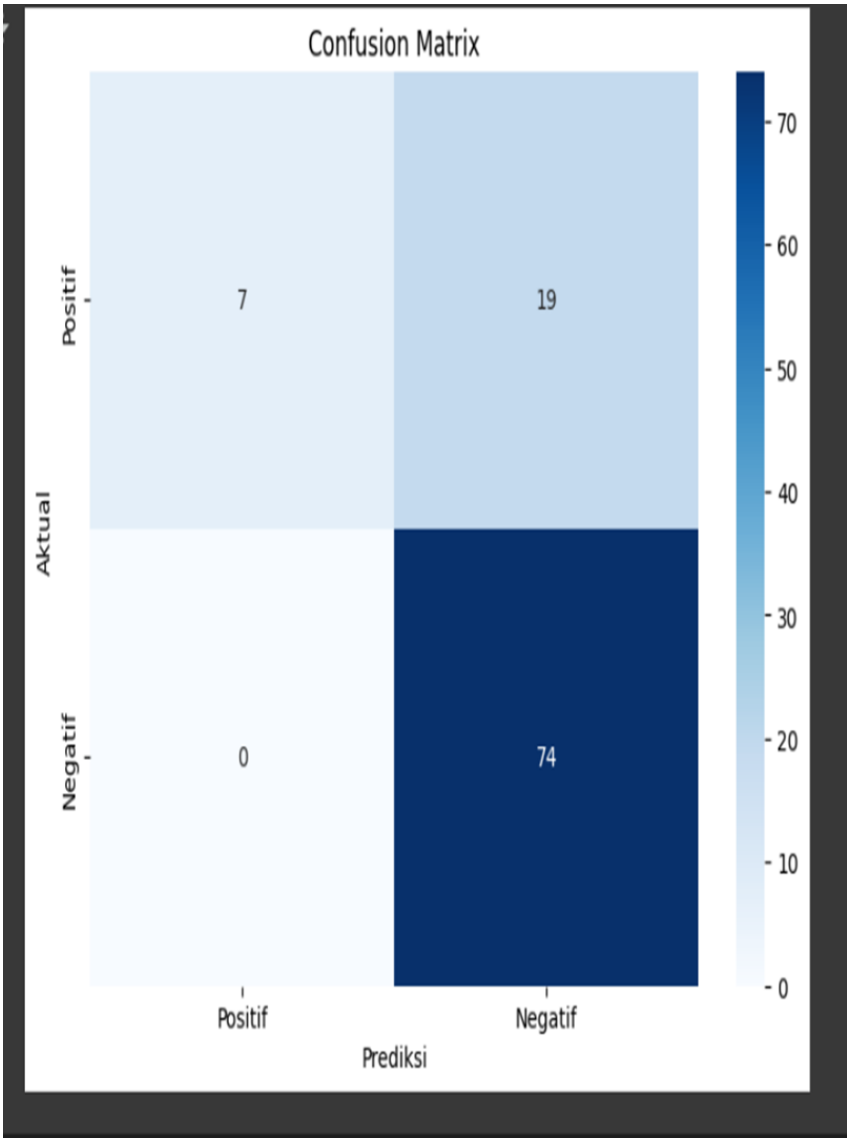


Figure 13. Confusion Matrix

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precission = \frac{TP}{TP + FP}$$

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

$$F1\ Score = 2x\frac{Precission\ X\ Recall}{Preccision + Recall}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{7 + 74}{7 + 74 + 19 + 0} = \frac{81}{100} = 81\%$$

$$Precission = \frac{TP}{TP + FP} = \frac{7}{7 + 19} = \frac{7}{26} = 84\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{7}{7 + 74} = \frac{7}{81} = 81\%$$

$$F1\ Score = 2x\frac{PrecissionxRecall}{Precission + Recall} = 2x\frac{84x81}{84 + 81} = 76\%$$

The accuracy of the Random Forest method was recorded at 0.81 (81%) based on a data split of 80:20. After obtaining the Confusion Matrix results, the next step was to calculate other evaluation metrics such as Precision, Accuracy, Recall, and F1-score. The evaluation results were then presented in the form of a Confusion Matrix table, as shown in Table 8.

Table 8. Confusion Matrix evaluation results

No	*Record Data*	*Training Data*	*Testing Data*	Data Sharing	Presentation Results
1	1000	900	100	90:10	81%
2	1000	800	200	80:20	80%
3	1000	700	300	70:30	78%
4	1000	600	400	60:40	77%
5	1000	500	500	50:50	78%

Based on the results of experiments conducted using variations in the proportion of training and testing data, a Confusion Matrix was obtained that represents the overall performance of the classification model. The Confusion Matrix plays a role in evaluating the model's prediction results by comparing them to actual data. The evaluation of the Random Forest model's performance in this study was carried out through four tests with different data distribution scenarios, as shown in Figure 14 at the root of the Decision tree.

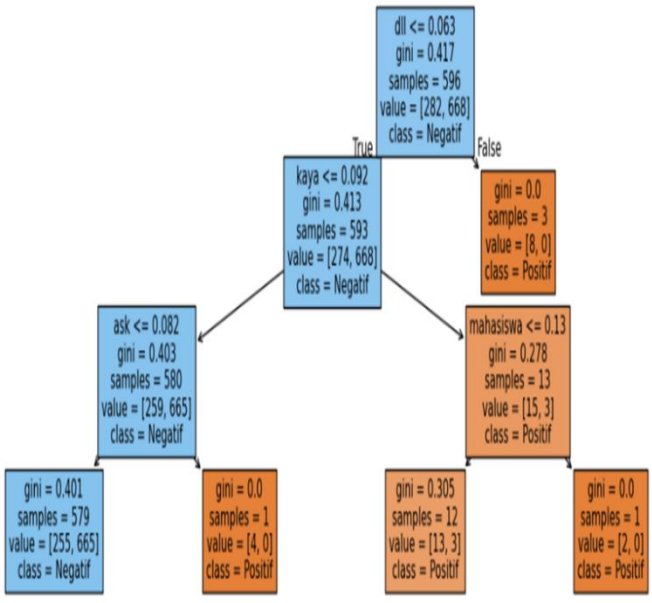


Figure 14. Decision Tree

Test Conclusion

Based on a series of tests conducted using the Random Forest method, it can be concluded that the model has quite reliable performance in classifying public sentiment towards Indonesian political news, although there are still some shortcomings in certain aspects. Testing was conducted using five scenarios of training and testing data proportions, ranging from 90:10 to 50:50. The evaluation results show that the highest accuracy of 81% was achieved in the 90:10 scenario, while the lowest accuracy was recorded at 77% in the 60:40 scenario. This value illustrates that the greater the proportion of training data, the more accurate the prediction results tend to be, because the model can learn sentiment patterns more thoroughly.

Conclusion

This study successfully demonstrated the effectiveness of the Random Forest algorithm in analyzing public opinion sentiment towards Indonesian political news trends on the Twitter platform using a dataset of 1000 tweets from the period 2020-2025. The main findings showed that the Random Forest model was able to achieve optimal accuracy of 81% at a 90:10 data split, indicating that a larger proportion of training data provides better sentiment classification performance. A comprehensive preprocessing process including cleaning, case folding, tokenization, stemming, and stopword removal proved effective in preparing Indonesian text data for machine learning analysis, while the TF-IDF technique successfully converted text representations into numeric features that can be processed by the algorithm. However, this study has limitations such as a relatively small dataset size with only 1000 sample tweets, an accuracy range that can still be improved (77-81%), and a limited focus on one social media platform, namely Twitter. Therefore, generalization of the results to other platforms and more diverse news topics still requires further validation.

The practical implications of this research indicate that the Random Forest algorithm can be implemented as an effective public sentiment monitoring instrument for governments, media, and public organizations in understanding public responses to policies or political news in real time. For further research, it is recommended to use a larger and more diverse dataset with a minimum of 10,000 tweet samples to improve model robustness, integrate data from multiple social media platforms such as Facebook, Instagram, and TikTok to provide a more holistic picture of sentiment, and implement other ensemble learning techniques such as XGBoost or BERT for performance comparison. Furthermore, the development of a real-time sentiment monitoring system with an interactive visualization dashboard will provide significant added value for political communication practitioners and policymakers in monitoring the dynamics of public opinion on current issues. Further research should also consider temporal analysis to identify patterns of sentiment change over time, as well as the implementation of deep learning approaches to capture more complex contextual nuances in Indonesian-language texts.

References

Aufar, M. (2020). Sentiment Analysis on Youtube Social Media Using Decision Tree and Random Forest Algorithm: A Case Study. *2020 International Conference on Data Science and Its Applications Icodsa 2020*, https://doi.org/10.1109/ICoDSA50139.2020.9213078

Campbell, S. W. (2025). Social (media) psychology of the "news-finds-me" perception. Journal of Computer-Mediated Communication, 30(5), zmaf010.https://doi.org/10.1093/jcmc/zmaf010

Chen, T. (2021). Monitoring and recognizing enterprise public opinion from high-risk users based on user portrait and random forest algorithm. *Axioms*, *10*(2), ISSN 2075-1680, https://doi.org/10.3390/axioms10020106

Creswell, J. W., & Creswell, J. D. (2023). Research design: Qualitative, quantitative, and mixed methods approaches (6th ed.). SAGE Publications.

Digital News Report. (2025). Reuters Institute Digital News Report 2025. Reuters Institute for the Study of Journalism, University of Oxford.

Emzir. (2012). Educational research methodology: Quantitative and qualitative (Revised Edition). Rajawali Pers.

Firdaus, AA, Berlilana, & Tahyudin, I. (2024). Application of sentiment analysis as an innovative approach to policy making: A review. Journal of Research and Application of Informatics Students, 5(6), 813-820.

Hadi, W., Putra, TD, & Oktafiani, D. (2025). Sentiment analysis of Lazada Indonesia product reviews using Random Forest Classifier. Innovative: Journal Of Social Science Research, 5(1), 2348-2357.

Jumaryadi, Y., Meiyanti, R., Fajriah, R., Mahsyar, AN, & Anggraeni, PS (2025). Implementation of the Random Forest algorithm for sentiment analysis of user reviews of the Merdeka Mengajar application. Bulletin of Computer Science Research, 5(4), 813-820.

Kim, R. M., & Anderson, A. (2025). The agenda-setting function of social media. In Proceedings of the ACM Web Conference 2025 (WWW '25), April 28-May 2, 2025, Sydney, NSW, Australia.

Muliana, AS, Ramadhan, S., & Pratama, D. (2024). Analysis of public sentiment on election results using Naïve Bayes to social media data from Indonesian netizens. Journal of Applied Data Sciences, 5(2), 145-156.

Musthafa, A., Berlilana, & Tahyudin, I. (2025). Utilization of machine learning in analyzing sentiment towards the TAPERA program on the digital platform X. MALCOM: Indonesian Journal of Machine Learning and Computer Science, 5(2), 456-467.

Natarajan, R. (2022). Intelligent gravitational search random forest algorithm for fake news detection. *International Journal of Modern Physics C*, *33*(6), ISSN 0129-1831, https://doi.org/10.1142/S012918312250084X

Pratama, MRS (2025). Sentiment analysis on Twitter social media regarding law enforcement in Indonesia in 2024 using the Support Vector Machine (SVM) method. Global Research and Innovation Journal, 1(3), 43-51.

Risnanda, R., Alfifa, AN, & Wibowo, S. (2025). The role of mass media in shaping public opinion through the hashtag phenomenon #kaburajadulu. Journal of Education and Teaching Review, 8(2), 5201-5215.

Samsir, S., Ritonga, WA, Aditya, R., & Watrianthos, R. (2024). Machine learning-driven sentiment analysis of social media data in the 2024 US Presidential Race. Bulletin of Information Technology, 5(4), 326-332.

Semary, N.A., Fadl, A., Essa, E., & Gad, A.G. (2024). Enhancing machine learning-based sentiment analysis through advanced feature extraction methods. PLOS ONE, 19(2), e0294968.https://doi.org/10.1371/journal.pone.0294968

Sihombing, E., Nasution, H., & Siregar, I. (2024). Comparison of machine learning algorithms in public sentiment analysis: A case study of 2024 Indonesian presidential election. International Journal of Science, Technology & Management, 5(5), 1408-1418.

Sudaryono. (2021). Quantitative, qualitative, and mixed method research methodology (4th Edition). Rajawali Pers.

Sugiyono. (2021). Quantitative, qualitative, and R&D research methods (2nd Edition, 3rd Printing). Alfabeta.

Tahyudin, I., Berlilana, & Prastyo, PA (2024). Sentiment analysis of Indonesian ministries' social media. Jurnal Sosioteknologi, 23(2), 287-302.