

**LAPORAN PROJEK**  
**MATA KULIAH PEMROSESAN TEXT**



**Analisis Sentimen Melalui Pengumpulan Data Twitter  
Terkait Konflik Palestina-Israel**

**KELOMPOK**

<b>Maheza Fiko Pratama</b>	<b>22031554048</b>
<b>Dian Ayu Fauziah</b>	<b>22031554011</b>
<b>Nuhaa Salsabila Shidqiyyah</b>	<b>22031554045</b>

**UNIVERSITAS NEGERI SURABAYA**  
**2023**

## A. PENDAHULUAN

Media sosial adalah sebuah media yang digunakan untuk bersosialisasi dan bertukar informasi oleh para penggunanya dengan menggunakan internet. Analisis sentimen melalui pengumpulan data twitter menjadi metode yang relevan dan inovatif untuk memahami respons publik terhadap perkembangan terkini dalam konflik Palestina-Israel. Twitter memberikan suara kepada jutaan pengguna untuk menyampaikan pendapat, emosi, dan dukungan terhadap suatu isu. Oleh karena itu, analisis sentimen melalui data twitter dapat memberikan wawasan yang berharga tentang bagaimana masyarakat merespons dan mengartikan perkembangan konflik yang terus berlangsung[1]. Dalam era digital ini, media sosial, khususnya Twitter, menjadi platform yang signifikan dalam mengungkapkan pandangan dan perasaan individu terkait konflik tersebut[2].

Konflik antara Palestina dan Israel telah menjadi fokus perhatian dunia internasional selama beberapa dekade terakhir. Ketegangan politik, sosial, dan ekonomi antara kedua pihak telah menciptakan beragam respons dan opini dari masyarakat global[3]. Dalam konteks ini, penelitian ini bertujuan untuk menggali dan menganalisis sentimen masyarakat yang terungkap melalui twit terkait konflik Palestina-Israel di Twitter. Data yang diperoleh dari platform ini akan menjadi sumber informasi yang kaya untuk memahami pola opini, perasaan, dan reaksi masyarakat terhadap konflik yang kompleks ini. Hasil analisis sentimen ini diharapkan dapat memberikan kontribusi dalam pemahaman mendalam tentang dinamika persepsi masyarakat terhadap konflik dan potensialnya memengaruhi pandangan global terhadap isu ini.

Metode yang digunakan dalam projek ini yaitu *Random Forest*. Metode *Random Forest* berisi gabungan dari *Decision Tree* untuk melakukan klasifikasi, *Random Forest* merupakan algoritma ensemble dimana untuk mendapatkan keputusan akhir dilakukan voting majority dari semua model *Decision Tree*[4]. *Naive Bayes* Merupakan salah satu algoritma data mining yang penggunaannya mudah serta pemrosesannya memiliki waktu yang cepat, mudah diimplementasikan dengan struktur yang cukup sederhana dan memiliki tingkat efektifitas yg tinggi[5].

## B. METODE

Pada project ini, tim melakukan beberapa metode dengan beberapa tahapan untuk mencapai hasil yang diinginkan. Berikut tahapan-tahapan yang digunakan :

### 1) Scrapping data

Tim melakukan scrapping pada sosial media dengan menentukan batas untuk jumlah tweet sebanyak 5000 untuk 2 kata kunci. Kata kunci yang dicari “Konflik Palestina” dan “Konflik Israel”, dengan rantang tanggal 7 Oktober 2023 sampai 12 Desember 2023. Dengan total hasil scrapping data Palestina sebanyak 2212 opini dan Israel sebanyak 2038.

## 2) Validasi Data

Setelah mendapatkan data selanjutnya melakukan validasi data. Terdapat beberapa kejanggalan data. Kami menghitung kemunculan akun pada data, dan setelah di check terdapat 1 twet dengan status yang sama dari akun yang berbeda.

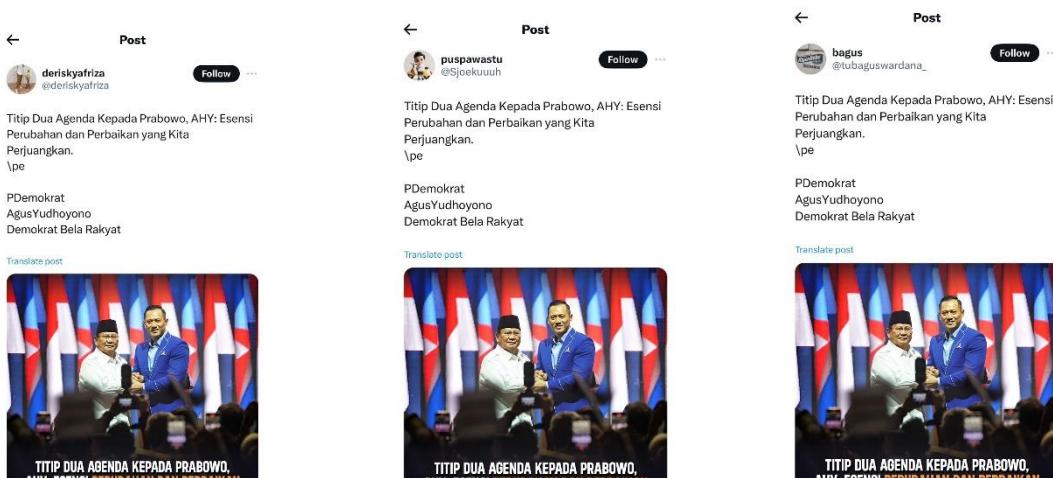
- Ditemukan adanya tweet yang sama, namun jika ditelusuri bukan dari akun yang sama seperti buzzer pada umumnya. Dan jika dilihat beberapa akun dengan tweet yang sama mereka mempunyai pola sama terhadap setiap postingan. **Gambar 1.**

Mon Dec 11 16:56:36 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 16:48:40 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 16:25:32 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 16:02:30 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 15:45:51 +0000 2023	1.73E+18 Arab Sebut Israel Mustahil Menang Di Gaza Palestina Sumber : <a href="https://t.co/CiKmJSz">https://t.co/CiKmJSz</a>
Mon Dec 11 15:07:35 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 13:25:03 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY
Mon Dec 11 13:09:53 +0000 2023	1.73E+18 Konflik Hamas-Israel, AHY Dukung Pemerintah Lakukan Langkah Strategis. ira AgusY

**Gambar 1.** Data setelah dilakukan *cleaning*

1548 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN n dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/uaYCwEBcrT">https://t.co/uaYCwEBcrT</a>
1549 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN & dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/yHfFm0iO1">https://t.co/yHfFm0iO1</a>
1550 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/zvtjkRqjR">https://t.co/zvtjkRqjR</a>
1551 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dan DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/ZO10H90tY">https://t.co/ZO10H90tY</a>
1552 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dan DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/M0cfrwDbz">https://t.co/M0cfrwDbz</a>
1553 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/h3hnslGx8u">https://t.co/h3hnslGx8u</a>
1554 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dn DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/KcXtylrAFY">https://t.co/KcXtylrAFY</a>
1555 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN n dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/hkhyp5jI">https://t.co/hkhyp5jI</a>
1556 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN & dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/tl3UDh0f">https://t.co/tl3UDh0f</a>
1557 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/w9cn0HQc">https://t.co/w9cn0HQc</a>
1558 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN n dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/TdVmCrNzL">https://t.co/TdVmCrNzL</a>
1559 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN DAN DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/E3aGYew0CO">https://t.co/E3aGYew0CO</a>
1560 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN & dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/7Q088b1omu">https://t.co/7Q088b1omu</a>
1561 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dn DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/bxUsk9ftN">https://t.co/bxUsk9ftN</a>
1562 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN DAN DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/3tv1ASPO4Q">https://t.co/3tv1ASPO4Q</a>
1563 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dn DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/gopUXk7W5e">https://t.co/gopUXk7W5e</a>
1564 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN n dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/feE3lt5wT">https://t.co/feE3lt5wT</a>
1565 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN & dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/feE3lt5wT">https://t.co/feE3lt5wT</a>
1566 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN DAN DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/oaVlzuulQ">https://t.co/oaVlzuulQ</a>
1567 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/2uNhxStx1">https://t.co/2uNhxStx1</a>
1568 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dn DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/AmeplnZnxw">https://t.co/AmeplnZnxw</a>
1569 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN dn DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/9I4PhNxtrh">https://t.co/9I4PhNxtrh</a>
1570 BERITA PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA DAPAT MEMICU KEPANIKAN DAN DPT MEMBAHAYAKAN KEHIDUPAN ORANG <a href="https://t.co/g7aSeAvcn">https://t.co/g7aSeAvcn</a>
1571 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN n dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/B3mpg6818o">https://t.co/B3mpg6818o</a>
1572 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/jqk7bwv2Z">https://t.co/jqk7bwv2Z</a>
1573 brita PALSU TERKAIT ESKALASI KONFLIK ISRAEL-PALESTINA dpt MEMICU KEPANIKAN dan dpt MEMBAHAYAKAN KEHIDUPAN org <a href="https://t.co/utymBMBw4W">https://t.co/utymBMBw4W</a>

**Gambar 2.** Data setelah dilakukan *cleaning*



**Gambar 3.** Sejumlah akun yang terduga buzzer

### 3) *Pre-processing*

- **Kamus alay**

Kamus alay pada analisis sentimen dapat menjadi alat yang berguna dalam memahami bahasa alay atau bahasa gaul yang digunakan dalam berbagai platform media sosial. Bahasa alay seringkali mencakup kosakata, singkatan, dan konvensi bahasa yang tidak umum atau tidak formal. Analisis sentimen pada bahasa alay dapat membantu dalam memahami perasaan dan opini yang terkandung dalam teks-teks tersebut.

- **Penghapusan Emoticon**

Penghapusan *emoticon* dalam analisis sentimen dapat dilakukan untuk menyederhanakan teks dan fokus pada makna verbalnya. *Emoticon*, meskipun dapat mengekspresikan emosi dengan baik, dapat memperumit analisis sentimen, terutama jika model yang digunakan tidak mendukung interpretasi emosi dari gambar atau karakter grafis.

- **Case folding**

*Case folding* adalah teknik dalam pemrosesan teks yang bertujuan untuk mengurangi kompleksitas dan mengatasi masalah perbedaan huruf besar dan kecil dalam analisis sentimen. Pada dasarnya, *case folding* mengonversi semua karakter dalam teks menjadi huruf kecil (*lowercase*) atau huruf besar (*uppercase*). Ini membantu memastikan konsistensi dalam analisis teks, terutama ketika melakukan komparasi atau ekstraksi fitur.

- **Tokenizing**

Tokenisasi adalah proses memecah teks menjadi unit-unit lebih kecil yang disebut token. Token bisa berupa kata, frasa, atau bahkan karakter, tergantung pada tingkat detail yang diinginkan dalam analisis. Tokenisasi umumnya merupakan langkah pra-pemrosesan penting dalam analisis sentimen untuk mempersiapkan teks untuk pemrosesan lebih lanjut, seperti ekstraksi fitur atau pembuatan model.

- **Stopwords menggunakan library Sastrawi**

Kata-kata penghubung (*stopwords*) adalah kata-kata umum yang sering muncul dalam teks namun biasanya tidak memberikan kontribusi signifikan terhadap makna atau sentimen. Dalam analisis sentimen, seringkali kata-kata ini diabaikan atau dihapus dari teks karena mereka cenderung tidak membawa informasi yang berguna untuk memahami sentimen.

- **Stemming**

*Stemming* adalah proses menghilangkan afiks (akhiran atau awalan) dari kata untuk menghasilkan bentuk dasar atau kata dasar. Tujuan *stemming* adalah untuk mengurangi variasi kata ke bentuk dasarnya sehingga kata-kata dengan akar yang sama dapat dianggap setara. Dalam analisis sentimen, stemming dapat membantu dalam mengatasi variasi kata yang memiliki makna yang sama sehingga dapat meningkatkan konsistensi dan akurasi dalam mengekstrak fitur atau memahami sentimen.

- **Kamus lexicon**

Lexicon berfungsi sebagai sumber referensi utama bagi pemahaman dan penggunaan kata-kata dalam suatu bahasa, membantu individu untuk memperluas kosakata dan meningkatkan kemampuan komunikasi mereka.

## C. HASIL DAN ANALISIS

### 1. Data Israel

Cross Validation : [0.83846154 0.83076923 0.84496124 0.84496124 0.83076923]

Accuracy : 0.84

Confusiin Matrix :  $\begin{bmatrix} 815 & 0 & 0 \\ 46 & 0 & 0 \\ 111 & 0 & 0 \end{bmatrix}$

Naïve Bayes	Precision	Recall	F1 Score	Support
<b>Negative</b>	0.84	1.00	0.91	815
<b>Neutral</b>	0.00	0.00	0.00	46
<b>Positive</b>	0.00	0.00	0.00	111
<b>Accuracy</b>			0.84	972
<b>Macro Average</b>	0.28	0.33	0.30	972
<b>Weighted Average</b>	0.70	0.84	0.76	972

Cross Validation : [0.67283951 0.68209877 0.72530864 0.66049383 0.69135802]

Accuracy : 0.78

Confusiin Matrix :  $\begin{bmatrix} 146 & 39 & 34 \\ 6 & 64 & 8 \\ 19 & 16 & 78 \end{bmatrix}$

Random Forest	Precision	Recall	F1 Score	Support
<b>Negative</b>	0.85	0.67	0.75	219
<b>Neutral</b>	0.54	0.82	0.65	78
<b>Positive</b>	0.64	0.68	0.66	109
<b>Accuracy</b>			0.70	406
<b>Macro Average</b>	0.68	0.72	0.69	406
<b>Weighted Average</b>	0.74	0.70	0.71	406

### 2. Data Palestina

Cross Validation : [0.67283951 0.68209877 0.72530864 0.66049383 0.69135802]

Accuracy : 0.78

Confusiin Matrix :  $\begin{bmatrix} 146 & 39 & 34 \\ 6 & 64 & 8 \\ 19 & 16 & 78 \end{bmatrix}$

Naïve Bayes	Precision	Recall	F1 Score	Support
<b>Negative</b>	0.56	0.97	0.71	219
<b>Neutral</b>	0.00	0.00	0.00	78
<b>Positive</b>	0.73	0.17	0.28	109
<b>Accuracy</b>			0.57	406

<b>Macro Average</b>	0.43	0.38	0.33	406
<b>Weighted Average</b>	0.50	0.57	0.46	406

Cross Validation : [0.67283951 0.70987654 0.69753086 0.67283951 0.66666667]

Accuracy : 0.66

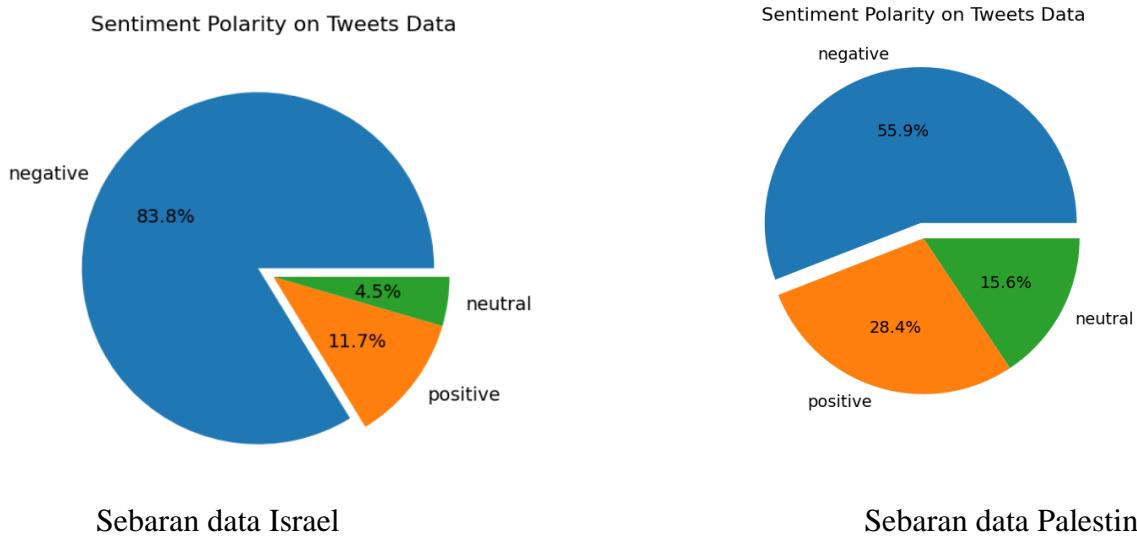
[[800 0 0]]

Confusiin Matrix : [208 0 0]  
[208 0 0]]

<b>Random Forest</b>	<b>Precision</b>	<b>Recall</b>	<b>F1 Score</b>	<b>Support</b>
<b>Negative</b>	0.66	1.00	0.79	800
<b>Neutral</b>	0.00	0.00	0.00	208
<b>Positive</b>	0.00	0.00	0.00	208
<b>Accuracy</b>			0.66	1216
<b>Macro Average</b>	0.22	0.33	0.26	1216
<b>Weighted Average</b>	0.43	0.66	0.52	1216

Berdasarkan analisis sentimen dari data yang diberikan, dapat disimpulkan bahwa sentimen terkait dengan Israel memiliki sebaran yang lebih merata antara positif, negatif, dan netral. Sebaliknya, sentimen terkait dengan Palestina cenderung lebih dominan pada aspek negatif. Untuk data Israel, mayoritas sentimen adalah negatif dengan persentase sekitar 55.88%, diikuti oleh sentimen positif sekitar 28.44%, dan sentimen netral sekitar 15.68%. Sementara itu, data Palestina menunjukkan mayoritas sentimen negatif yang signifikan, mencapai sekitar 83.70%, dengan sentimen positif sekitar 11.73%, dan sentimen netral sekitar 4.52%. Perbandingan antara kedua negara menunjukkan perbedaan yang mencolok dalam persepsi sentimen, dengan sentimen terkait Palestina lebih didominasi oleh aspek negatif dibandingkan dengan Israel. Analisis ini memberikan gambaran umum tentang pandangan dan opini masyarakat terhadap kedua entitas tersebut.

Dalam menganalisis hasil dari sentimen analisis Israel dua model, yakni Naive Bayes dan Random Forest, kita menemukan bahwa keduanya memiliki akurasi sekitar 86%. Meskipun demikian, Naive Bayes mengalami kesulitan dalam mengenali perasaan "netral" dan "positif," seperti terlihat dari hasil yang rendah untuk dua kategori tersebut. Sementara itu, Random Forest menunjukkan peningkatan dalam mengenali "netral" dan "positif," tetapi masih ada kekurangan pada pengenalan "netral."



## D. KESIMPULAN DAN EVALUASI

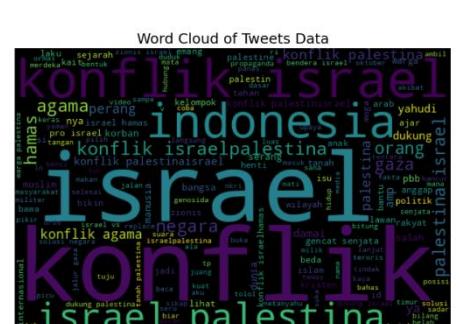
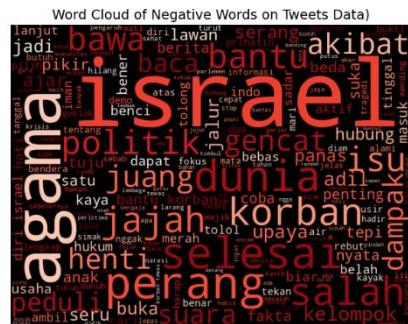
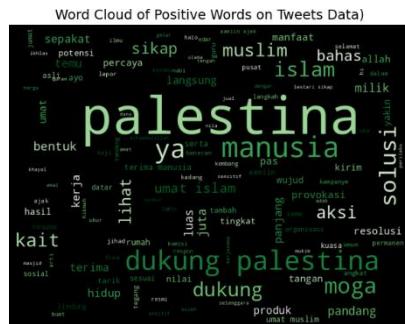
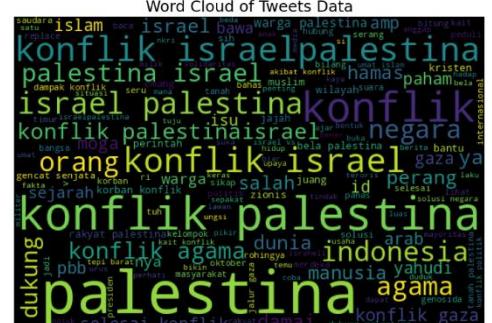
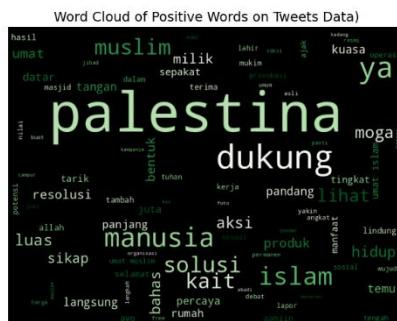
Dari hasil evaluasi dua model klasifikasi, yaitu Naïve Bayes dan Random Forest, pada dua dataset yang berbeda, yaitu Data Israel dan Data Palestina, terdapat beberapa temuan menarik. Pertama, pada Data Israel, model Naïve Bayes menunjukkan akurasi tinggi, terutama untuk kelas Negative, namun kesulitan memprediksi kelas Neutral dan Positive. Sebaliknya, model Random Forest memberikan akurasi yang seimbang untuk ketiga kelas. Kedua, pada Data Palestina, keduanya menghadapi kesulitan dalam memprediksi kelas selain Negative, dengan akurasi yang cenderung rendah. Rekomendasi untuk penanganan ketidak seimbangan kelas, seperti oversampling atau undersampling, serta fine-tuning model dapat diambil sebagai langkah-langkah untuk meningkatkan performa model dalam kedua kasus tersebut. Evaluasi lebih lanjut diperlukan untuk memahami dan mengatasi tantangan yang muncul dalam memprediksi kelas minoritas.

## E. REFERENSI

- [1] P. Pasek, O. Mahawardana, G. A. Sasmita, P. Agus, dan E. Pratama, “Analisis Sentimen Berdasarkan Opini dari Media Sosial Twitter terhadap ‘Figure Pemimpin’ Menggunakan Python,” 2022.
- [2] Indra Maulana, “Dinamika Konflik Palestina dan Israel Konten ini telah tayang di Kompasiana.com dengan judul ‘Dinamika Konflik Palestina dan Israel’, Klik untuk baca: <https://www.kompasiana.com/indramaulana2704/65712501c57afb0178058352/dinamika-konflik-palestina-dan-israel> Kreator: Indra Maulana Kompasiana adalah platform blog. Konten ini menjadi tanggung jawab bloger dan tidak mewakili pandangan redaksi Kompas. Tulis opini Anda seputar isu terkini di Kompasiana.com.”
- [3] Admin Staff, “Konflik Palestina Dan Israel: Pentingnya Memahami Jenis Konflik Internasional Dari Sudut Pandang Hukum Dan Politik.”
- [4] T. B. Rohman, D. Dwi Purwanto, dan J. Santoso, “Sentiment Analysis Terhadap Review Rumah Makan di Surabaya Memanfaatkan Algoritma Random Forest.”
- [5] E. Fitri, Y. Yuliani, S. Rosyida, dan W. Gata, “Analisis Sentimen Terhadap Aplikasi Ruangguru Menggunakan Algoritma Naive Bayes, Random Forest Dan Support Vector Machine,” *TRANSFORMTIKA*, vol. 18, no. 1, hlm. 71–80, 2020, [Daring]. Tersedia pada: [www.nusamandiri.ac.id](http://www.nusamandiri.ac.id),

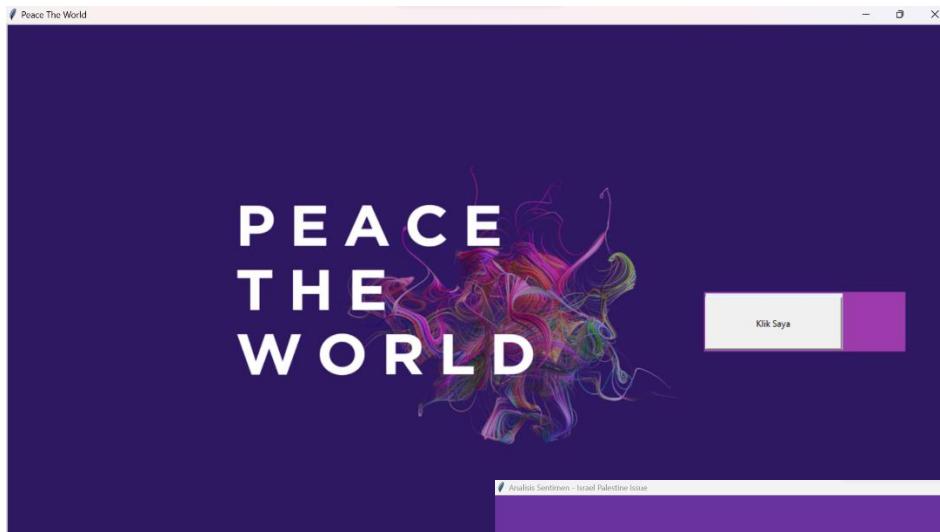
## F. LAMPIRAN

- Hasil Wordcloud

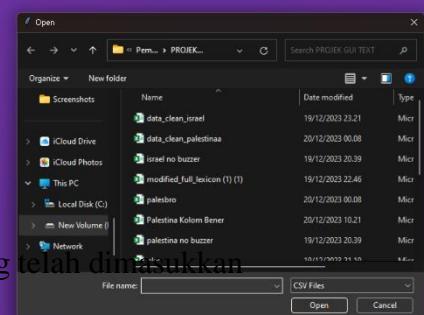


- Tampilan GUI

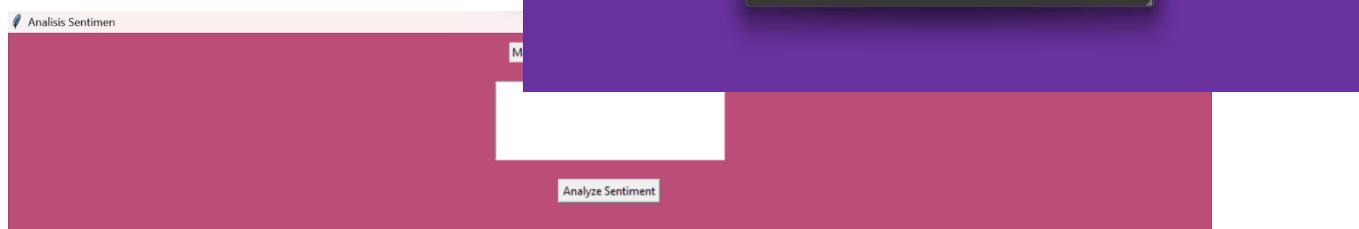
- Tampilan Awal, terdapat button klik untuk memasukkan file csv



Analisis Sentimen - Israel Palestine Issue

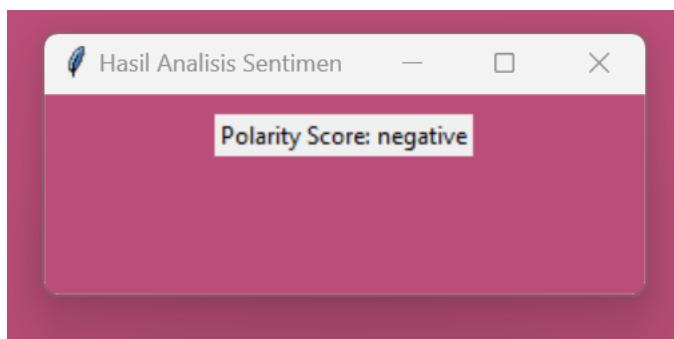


Open CSV



- Memasukkan teks sesuai dengan csv yang telah dimasukkan

- Hasil



## G. LISTINGCODE

```
"""pales.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1nafdufqqr9IUBNFYVhrEe41
i9cY7rsbPi

"""

import nltk

from nltk.tokenize import word_tokenize
nltk.download('punkt')

def tokenizingText(text):
```

```
text = word_tokenize(text)
return text

import nltk
from nltk.tokenize import word_tokenize
nltk.download('stopwords')

def tokenizingText(text):
    text = word_tokenize(text)
    return text

import pandas as pd
pd.options.mode.chained_assignment = None
import numpy as np
seed = 0
np.random.seed(seed)

import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
import re
import string

from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from wordcloud import WordCloud
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
```

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,
classification_report, confusion_matrix
from sklearn.model_selection import cross_val_score
import csv

input_file_path = '/content/palestina no buzzer.csv'
output_file_path = 'ples.csv'

try:
    with open(input_file_path, 'r') as input_file,
        open(output_file_path, 'w', newline='') as output_file:
        pembaca = csv.reader(input_file)
        penulis = csv.writer(output_file)
        header = next(pembaca)
        jumlah_kolom = len(header)
        penulis.writerow(["full_text"])

        for nomor_baris, baris in enumerate(pembaca, start=2):
            kolom_gabungan = ','.join(baris)
            penulis.writerow([kolom_gabungan])

        print(f"File CSV yang sudah diperbaiki disimpan di:\n{output_file_path}")

except Exception as e:
    print(f"Terjadi kesalahan: {e}")

tweets_data = pd.read_csv('/content/ples.csv')
tweets = tweets_data[['full_text']]
tweets

import pandas as pd
import re

from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
```

```
import string

alay_dict = pd.read_csv('/content/pleskms.csv', header=None,
names=['awal', 'replace'], index_col='awal')

def replace_alay(text):
    for original, replacement in alay_dict.iterrows():
        text = re.sub(r'\b{}\b'.format(re.escape(original)), replacement['replace'], text)
    return text

def cleaningText(text):
    text = re.sub(r'@[A-Za-z0-9]+', '', text)
    text = re.sub(r'#([A-Za-z0-9]+)', '', text)
    text = re.sub(r'RT[\s]', '', text)
    text = re.sub(r"http\S+", '', text)
    text = re.sub(r'[0-9]+', '', text)

    text = text.replace('\n', ' ')
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = text.strip(' ')
    return text

def casefoldingText(text):
    text = text.lower()
    return text

def tokenizingText(text):
    text = word_tokenize(text)
    return text

def filteringText(text):
    listStopwords = set(stopwords.words('indonesian'))
    filtered = [txt for txt in text if txt not in listStopwords]
    return filtered

def stemmingText(text):
```

```

factory = StemmerFactory()
stemmer = factory.create_stemmer()
text = [stemmer.stem(word) for word in text]
return text

def toSentence(list_words):
    sentence = ' '.join(word for word in list_words)
    return sentence

tweets['text_clean'] = tweets['full_text'].apply(cleaningText)
tweets['text_clean'] =
tweets['text_clean'].apply(casefoldingText)
tweets.drop(['full_text'], axis=1, inplace=True)
tweets['text_clean'] = tweets['text_clean'].apply(replace_alay)
tweets['text_preprocessed'] =
tweets['text_clean'].apply(tokenizingText)
tweets['text_preprocessed'] =
tweets['text_preprocessed'].apply(filteringText)
tweets['text_preprocessed'] =
tweets['text_preprocessed'].apply(stemmingText)
tweets.drop_duplicates(subset='text_clean', inplace=True)
tweets.to_csv(r'data_clean.csv', index=False, header=True,
index_label=None)

import pandas as pd
import re
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import string

alay_dict = pd.read_csv('/content/singkatan-lib.csv',
header=None, names=['awal', 'replace'], index_col='awal')

def replace_alay(text):
    for original, replacement in alay_dict.iterrows():
        text = re.sub(r'\b{}\b'.format(re.escape(original)),
replacement['replace'], text)

```

```
return text

def cleaningText(text):
    text = re.sub(r'@[A-Za-z0-9]+', '', text)
    text = re.sub(r'#([A-Za-z0-9]+)', '', text)
    text = re.sub(r'RT[\s]', '', text)
    text = re.sub(r"http\S+", '', text)
    text = re.sub(r'[0-9]+', '', text)
    emoticons_pattern = (
```

A horizontal row of various emoji characters, including faces with different expressions (angry, surprised, etc.), a bee, and several cat faces.

A horizontal row of various emoji characters, including lions, monkeys, people with different expressions, and a dog, arranged in two rows.

A horizontal row of various emoji icons, including a yellow balloon, a red owl, a smiling face, a green face with a tongue out, an orange owl, a purple person, a yellow person, two hands, a blue phone, a king, a man in a suit, an elderly woman, a person with a speech bubble, a person with arms raised, a person in a hat, a person running, a person with a backpack, a person with arms crossed, a person with a hand up, and a person with a hand down.

A horizontal row of various food and drink emojis, including a sandwich, a donut, a milkshake, a cookie, a green smoothie, a sandwich, a pizza slice, a bowl of ramen, and a bowl of cereal.

A horizontal row of various food and drink icons, including a sandwich, salad, smiley face, taco, cookie, milk, and a green vegetable.

A horizontal row of various food and drink emojis, including a sandwich, a bowl of cereal, a slice of pizza, a water bottle, a coconut, a broccoli, a pretzel, a piece of meat, a sandwich, a can of soda, a leafy green, a mango, a bun, a bowl of ramen, and a donut.

A horizontal row of various emoji characters, including faces, objects, and symbols.

A horizontal row of various emoji characters, including faces with different expressions and colors.

A horizontal row of various emoji characters, including faces with different expressions (surprised, crying, etc.) and animals like cats and a bee.

A horizontal row of various emoji characters, including faces, animals, and objects.

A horizontal row of various food and drink emojis, including a glass of milk, a peanut, a green donut, a chocolate cake, a pie, a pacifier, a helmet, a red ribbon, a blue bowl, a coconut, a broccoli, a cookie, a pretzel, a steak, a sandwich, a tomato, and a lettuce leaf.

A horizontal row of various emoji characters, including people, animals, and objects.

A horizontal row of various emoji characters, including cats, faces with different expressions, and people in various poses.

)

t

```
text = re.sub(emoticons_pattern, '', text)
```

```
    text = text.replace('\n', ' ')
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = text.strip(' ')
    return text

def casefoldingText(text):
```

```

text = text.lower()
return text

def tokenizingText(text):
    text = word_tokenize(text)
    return text

def filteringText(text):
    listStopwords = set(stopwords.words('indonesian'))
    filtered = [txt for txt in text if txt not in
listStopwords]
    return filtered

def stemmingText(text):
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    text = [stemmer.stem(word) for word in text]
    return text

def toSentence(list_words):
    sentence = ' '.join(word for word in list_words)
    return sentence

tweets['text_clean'] = tweets['full_text'].apply(cleaningText)
tweets['text_clean'] =
tweets['text_clean'].apply(casefoldingText)
tweets.drop(['full_text'], axis=1, inplace=True)
tweets['text_clean'] = tweets['text_clean'].apply(replace_alay)
tweets['text_preprocessed'] =
tweets['text_clean'].apply(tokenizingText)

tweets['text_preprocessed'] =
tweets['text_preprocessed'].apply(filteringText)

tweets['text_preprocessed'] =
tweets['text_preprocessed'].apply(stemmingText)

tweets.drop_duplicates(subset='text_clean', inplace=True)
tweets.to_csv(r'data_clean.csv', index=False, header=True,
index_label=None)

tweets = pd.read_csv('/content/data_clean.csv')

```

```

for i, text in enumerate(tweets['text_preprocessed']):
    tweets['text_preprocessed'][i] =
tweets['text_preprocessed'][i].replace("""", "") \
.replace(',', '').replace(']', '').replace('[', '')
list_words=[]
for word in tweets['text_preprocessed'][i].split():
    list_words.append(word)
tweets['text_preprocessed'][i] = list_words
tweets

lexicon = dict()
import csv
with open('/content/modified_full_lexicon (1).csv', 'r') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)
    for row in reader:
        lexicon[row[0]] = int(row[1])

def sentiment_analysis_lexicon_indonesia(text):
    weight = 0
    for word in text:
        if word in lexicon:
            weight += lexicon[word]
    polarity = 'positive' if weight > 0 else ('negative' if
weight < 0 else 'neutral')
    return weight, polarity
results =
tweets['text_preprocessed'].apply(sentiment_analysis_lexicon_in
donesia)
results = list(zip(*results))
tweets['polarity_score'] = results[0]

```

```

tweets['polarity'] = results[1]
print(tweets['polarity'].value_counts())

tweets.to_csv(r'sentimen.csv', index = False, header =
True,index_label=None)

fig, ax = plt.subplots(figsize = (6, 6))
sizes = [count for count in tweets['polarity'].value_counts()]
labels = list(tweets['polarity'].value_counts().index)
explode = (0.1, 0, 0)
ax.pie(x = sizes, labels = labels, autopct = '%1.1f%%', explode =
explode, textprops={'fontsize': 14})
ax.set_title('Sentiment Polarity on Tweets Data', fontsize =
16, pad = 20)
plt.show()

pd.set_option('display.max_colwidth', 3000)
positive_tweets = tweets[tweets['polarity'] == 'positive']
positive_tweets = positive_tweets[['text_clean',
'polarity_score', 'polarity']].sort_values(by =
'polarity_score', ascending=False).reset_index(drop = True)
positive_tweets.index += 1
positive_tweets[0:5]
pd.set_option('display.max_colwidth', 3000)
negative_tweets = tweets[tweets['polarity'] == 'negative']
negative_tweets = negative_tweets[['text_clean',
'polarity_score', 'polarity']].sort_values(by =
'polarity_score', ascending=True)[0:10].reset_index(drop =
True)
negative_tweets.index += 1
negative_tweets[0:5]
list_words=''
for tweet in tweets['text_preprocessed']:
    for word in tweet:
        list_words += ' '+(word)

```

```
wordcloud = WordCloud(width = 600, height = 400,
background_color = 'black', min_font_size =
10).generate(list_words)

fig, ax = plt.subplots(figsize = (8, 6))

ax.set_title('Word Cloud of Tweets Data', fontsize = 18)
ax.grid(False)
ax.imshow( (wordcloud) )
fig.tight_layout(pad=0)
ax.axis('off')
plt.show()

def words_with_sentiment(text):
    positive_words = []
    negative_words = []
    for word in text:
        score_pos = 0
        score_neg = 0
        if word in lexicon:
            score_pos = lexicon[word]
        if score_pos > 0:
            positive_words.append(word)
        elif score_pos < 0:
            negative_words.append(word)

    return positive_words, negative_words

sentiment_words =
tweets['text_preprocessed'].apply(words_with_sentiment)
sentiment_words = list(zip(*sentiment_words))
positive_words = sentiment_words[0]
negative_words = sentiment_words[1]
fig, ax = plt.subplots(1, 2, figsize = (12, 10))
list_words_positive=''

for row_word in positive_words:
```

```

        for word in row_word:
            list_words_postive += ' '+(word)

wordcloud_positive = WordCloud(width = 800, height = 600,
background_color = 'black', colormap = 'Greens'
                                , min_font_size =
10).generate(list_words_postive)

ax[0].set_title('Word Cloud of Positive Words on Tweets Data)', fontsize = 14)

ax[0].grid(False)

ax[0].imshow((wordcloud_positive))

fig.tight_layout(pad=0)

ax[0].axis('off')

list_words_negative=''

for row_word in negative_words:
    for word in row_word:
        list_words_negative += ' '+(word)

wordcloud_negative = WordCloud(width = 800, height = 600,
background_color = 'black', colormap = 'Reds'
                                , min_font_size =
10).generate(list_words_negative)

ax[1].set_title('Word Cloud of Negative Words on Tweets Data)', fontsize = 14)

ax[1].grid(False)

ax[1].imshow((wordcloud_negative))

fig.tight_layout(pad=0)

ax[1].axis('off')

plt.show()

data = pd.read_csv('/content/sentimen.csv')

data.head()

X_train, X_test, y_train, y_test =
train_test_split(data['text_preprocessed'], data['polarity'],
test_size=0.2, random_state=0)

tfidf_vectorizer = TfidfVectorizer(max_features=2000)

```

```

X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
feature_names = tfidf_vectorizer.get_feature_names_out()
naive_bayes = MultinomialNB()
naive_bayes.fit(X_train_tfidf, y_train)
y_pred = naive_bayes.predict(X_test_tfidf)
accuracy = accuracy_score(y_test, y_pred)

cv_scores =
cross_val_score(RandomForestClassifier(random_state=0),
X_train_tfidf, y_train, cv=5)

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Score:", np.mean(cv_scores))
print(f'Accuracy: {accuracy:.2f}')

print('\nClassification Report:')
print(classification_report(y_test, y_pred))

print('\nConfusion Matrix:')
print(confusion_matrix(y_test, y_pred))

X_train, X_test, y_train, y_test =
train_test_split(data['text_preprocessed'], data['polarity'],
test_size=0.2, random_state=0)

bow_vectorizer = CountVectorizer(max_features=1000)
X_train_bow = bow_vectorizer.fit_transform(X_train)
X_test_bow = bow_vectorizer.transform(X_test)

random_forest = RandomForestClassifier(n_estimators=100,
random_state=0)

random_forest.fit(X_train_bow, y_train)
y_pred = random_forest.predict(X_test_bow)

accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy:.2f}')
print('\nClassification Report:')
print(classification_report(y_test, y_pred))

```

```

print('\nConfusion Matrix:')
print(confusion_matrix(y_test, y_pred))

• Listing Code GUI

from tkinter import *
from tkinter import filedialog
import pandas as pd

class SentimentAnalysisApp:

    def __init__(self, master):
        self.master = master
        master.title("Analisis Sentimen - Israel Palestine Issue")
        master.configure(bg='#6a329f')

        self.open_csv_button = Button(master, text="Open CSV",
command=self.open_csv_file, width=25, height=4)
        self.open_csv_button.place(relx=0.81, rely=0.68,
anchor=CENTER)

        self.csv_data = pd.DataFrame()

    def open_csv_file(self):
        file_path =
filedialog.askopenfilename(filetypes=[("CSV Files", "*.csv")])

        if file_path:
            self.csv_data = pd.read_csv(file_path,
delimiter=';')

        self.show_sentiment_analysis_window()

```

```
def show_sentiment_analysis_window(self):
    new_window = Toplevel(self.master)
    new_window.title("Analisis Sentimen")
    new_window.geometry("400x200")
    new_window.configure(bg='#ba4e78')

    text_label = Label(new_window, text="Masukkan teks
untuk analisis sentimen:")
    text_label.pack(pady=10)

    input_textbox = Text(new_window, height=5, width=30)
    input_textbox.pack(pady=10)

    analyze_button = Button(new_window, text="Analyze
Sentiment", command=lambda:
self.analyze_sentiment(input_textbox.get("1.0", "end-1c")))
    analyze_button.pack(pady=10)

def analyze_sentiment(self, text_to_analyze):
    print(f"Text to analyze: {text_to_analyze}")
    polarity_score =
self.get_sentiment_score(text_to_analyze)

    result_window = Toplevel(self.master)
    result_window.title("Hasil Analisis Sentimen")
    result_window.geometry("300x100")
    result_window.configure(bg='#ba4e78')

    score_label = Label(result_window, text=f"Polarity
Score: {polarity_score}")
    score_label.pack(pady=10)

def get_sentiment_score(self, text):
    text = text.strip()
```

```
try:
    row = self.csv_data[self.csv_data['text_clean'] == text]
    if not row.empty:
        sentiment_score = row['polarity'].values[0]
        return sentiment_score
    else:
        return "Data tidak ditemukan. Teks: {}".format(text)
except Exception as e:
    return "Terjadi kesalahan: {}".format(str(e))

root = Tk()
sentiment_app = SentimentAnalysisApp(root)
root.mainloop()
```