ANALISIS SENTIMEN ULASAN PENGGUNA APLIKASI CHATGPT MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE DAN NAIVE BAYES BERBASIS PYTHON

Muh Muzaki, Rudi Kurniawan, Irfan Ali, Mulyawan, Saeful Anwar
Rekayasa Perangkat Lunak, STMIK IKMI Cirebon
Jl. Perjuangan No. 10 B, Kota Cirebon, Indonesia
muhammadmuzaki119@gmail.com

ABSTRAK

Perkembangan kecerdasan buatan generatif seperti ChatGPT mendorong meningkatnya jumlah ulasan pengguna pada platform digital, khususnya Google Play Store. Ulasan tersebut merepresentasikan persepsi pengguna terhadap kualitas dan performa aplikasi sehingga perlu dianalisis secara sistematis. Permasalahan dalam penelitian ini adalah bagaimana mengklasifikasikan sentimen ulasan pengguna ChatGPT serta menentukan algoritma yang paling optimal untuk tugas tersebut. Penelitian ini bertujuan menganalisis sentimen ulasan pengguna dan membandingkan kinerja algoritma Support Vector Machine (SVM) dan Naive Bayes. Data diperoleh melalui web scraping ulasan berbahasa Indonesia, kemudian dilakukan pembersihan dan preprocessing yang meliputi case folding, tokenizing, stopword removal, dan stemming sehingga diperoleh 9.879 data siap analisis. Pelabelan sentimen dilakukan menggunakan pendekatan lexicon-based, sedangkan pembentukan fitur menggunakan TF-IDF. Model dilatih dan diuji dengan pembagian data 80:20 serta dievaluasi menggunakan akurasi, precision, recall, F1-score, dan confusion matrix. Hasil penelitian menunjukkan bahwa SVM memberikan performa terbaik dengan akurasi di atas 90%, sehingga lebih efektif dalam mengklasifikasikan teks ulasan yang bersifat pendek dan sparse dibandingkan Naive Bayes.

Kata kunci : *analisis sentimen, ChatGPT, ulasan pengguna, Support Vector Machine, Naive Bayes*

1. PENDAHULUAN

Perkembangan kecerdasan buatan generatif, khususnya model bahasa besar seperti ChatGPT, telah mengubah cara pengguna berinteraksi dengan sistem digital dalam berbagai konteks, termasuk pencarian informasi, pembelajaran, dan produktivitas berbasis teks. Seiring dengan meningkatnya adopsi aplikasi ini, jumlah ulasan pengguna pada platform distribusi seperti Google Play Store juga mengalami peningkatan signifikan. Ulasan tersebut merepresentasikan pengalaman, tingkat kepuasan, serta kritik terhadap performa aplikasi, sehingga menjadi sumber data yang bernilai untuk evaluasi kualitas perangkat lunak berbasis pengalaman pengguna [1], [2]. Dalam konteks rekayasa perangkat lunak modern, pemanfaatan ulasan pengguna sebagai bahan evaluasi berbasis data semakin relevan untuk mendukung peningkatan kualitas aplikasi secara berkelanjutan [3], [4].

Namun demikian, volume ulasan yang besar dan sifatnya yang tidak terstruktur menimbulkan tantangan dalam proses analisis. Analisis manual tidak hanya memerlukan waktu yang panjang, tetapi juga rentan terhadap subjektivitas. Oleh karena itu, pendekatan otomatis berbasis Natural Language Processing (NLP) diperlukan untuk mengidentifikasi dan mengklasifikasikan opini pengguna secara sistematis [5], [6]. Analisis sentimen menjadi salah satu teknik yang banyak digunakan untuk mengelompokkan teks ke dalam kategori positif, negatif, dan netral. Meskipun berbagai pendekatan telah dikembangkan, pemilihan algoritma klasifikasi yang tepat masih menjadi isu penting, khususnya pada data ulasan yang umumnya pendek, mengandung bahasa informal, dan memiliki representasi fitur yang sparse.

Berbagai penelitian menunjukkan bahwa algoritma machine learning klasik seperti Support Vector Machine (SVM) dan Naive Bayes tetap kompetitif dalam tugas klasifikasi teks, bahkan di tengah berkembangnya model deep learning [7], [8]. SVM dikenal memiliki kemampuan yang kuat dalam menangani data berdimensi tinggi dan memaksimalkan margin pemisah antar kelas, sedangkan Naive Bayes menawarkan efisiensi komputasi dan kemudahan implementasi. Meskipun demikian, kajian komparatif terhadap kedua algoritma pada konteks ulasan ChatGPT berbahasa Indonesia masih terbatas, sehingga diperlukan evaluasi empiris untuk menentukan metode yang paling efektif dalam menangani karakteristik data tersebut.

Berdasarkan kebutuhan tersebut, penelitian ini bertujuan untuk menganalisis sentimen ulasan pengguna aplikasi ChatGPT berbahasa Indonesia serta membandingkan kinerja algoritma Support Vector Machine dan Naive Bayes dalam klasifikasi sentimen berbasis representasi TF-IDF. Untuk mencapai tujuan tersebut, data ulasan dikumpulkan melalui proses web scraping, kemudian dilakukan preprocessing teks yang meliputi case folding, tokenizing, stopword removal, dan stemming. Pelabelan sentimen dilakukan menggunakan pendekatan lexicon-based, sedangkan pembentukan fitur menggunakan metode Term Frequency–Inverse Document Frequency (TF-IDF). Model kemudian dilatih dan diuji menggunakan skema pembagian data 80:20 serta dievaluasi menggunakan metrik akurasi, precision, recall, F1-

score, dan confusion matrix. Hasil penelitian ini diharapkan memberikan kontribusi empiris dalam pemilihan algoritma klasifikasi yang optimal serta mendukung pengembangan sistem analisis sentimen untuk evaluasi aplikasi berbasis kecerdasan buatan.

2. TINJAUAN PUSTAKA

Penelitian mengenai analisis sentimen terus berkembang seiring meningkatnya penggunaan aplikasi digital dan media sosial yang menghasilkan data ulasan dalam jumlah besar. Taherdoost & Madanchian, [1] menegaskan bahwa integrasi kecerdasan buatan dengan analisis sentimen kini memainkan peran strategis dalam memahami persepsi pengguna, khususnya pada lingkungan kompetitif yang membutuhkan *respons* cepat terhadap opini publik. Dalam konteks bisnis digital, Nichifor et al.[2] menunjukkan bahwa analisis sentimen dapat mengubah ulasan pengguna menjadi *insight* bisnis yang bernilai, membantu organisasi dalam meningkatkan layanan dan kualitas produk.

2.1. Analisis Sentimen dan *Natural Language Processing (NLP)*

Analisis sentimen merupakan cabang penting NLP yang bertujuan mendeteksi, *mengekstraksi*, dan *mengklasifikasikan* opini atau emosi dalam teks manusia [6]. Tujuan utamanya meliputi *transformasi* teks tidak terstruktur menjadi label sentimen terstruktur, melakukan analisis dalam skala besar, serta memberikan dasar pengambilan keputusan berbasis opini [9]. Mao [5] menekankan bahwa analisis sentimen menghubungkan ekspresi pengguna dengan *insight* yang dapat ditindaklanjuti, misalnya dalam evaluasi kualitas layanan, pengembangan fitur, dan perbaikan pengalaman pengguna.

2.2. *Representasi Teks* (*Bag of Words dan TF-IDF*)

Representasi teks merupakan langkah penting dalam *mengonversi* data *linguistik* menjadi bentuk numerik yang dapat diolah oleh *algoritma machine learning*. Dua pendekatan yang paling umum digunakan adalah *Bag of Words (BoW)* dan *Term Frequency–Inverse Document Frequency* (*TF-IDF*). *BoW* menghitung frekuensi kemunculan kata tanpa mempertimbangkan urutan, sedangkan *TF-IDF* menambahkan bobot yang mencerminkan pentingnya suatu kata terhadap keseluruhan korpus. Kedua metode ini banyak digunakan dalam analisis ulasan dan sentimen karena sesuai dengan karakteristik teks pendek yang *sparse* [5].

Dengan demikian, penelitian ini menggunakan *TF-IDF* untuk membentuk *vektor* fitur yang menggambarkan intensitas kemunculan kata pada ulasan pengguna *ChatGPT*. Pendekatan ini dianggap efisien dan sesuai untuk teks pendek, serta telah terbukti memberikan performa baik dalam berbagai studi analisis sentimen aplikasi *mobile*.

2.3. *Algoritma Support Vector Machine (SVM)*

Support Vector Machine (SVM) merupakan algoritma pembelajaran terawasi yang digunakan secara luas dalam klasifikasi teks. Prinsip kerja SVM adalah mencari *hiperbidang (hyperplane)* yang memisahkan data antar kelas dengan margin maksimal. Menurut Hua et al., 2024 [10], SVM memiliki keunggulan dalam menangani data berdimensi tinggi seperti representasi TF-IDF, sehingga sering menjadi pilihan utama dalam tugas klasifikasi teks pendek.

Sifat *linieritas* dan efisiensi SVM menjadikannya cocok untuk dataset yang besar, seperti data ulasan *Google Play*. Dengan kombinasi representasi TF-IDF, algoritma ini diharapkan dapat memberikan hasil klasifikasi yang stabil dan presisi tinggi.

2.4. Algoritma *Naive Bayes*

Naive Bayes (NB) merupakan algoritma *probabilistik* berbasis *teorema Bayes* yang mengasumsikan independensi antar fitur. Meskipun asumsi ini jarang sepenuhnya benar dalam teks alami, NB telah terbukti sangat efektif untuk tugas klasifikasi teks karena kesederhanaan dan efisiensinya. Menurut Haoues et al., 2023 [3], model *Multinomial Naive Bayes (MNB)* sering digunakan dalam analisis sentimen karena mampu menangani distribusi frekuensi kata dengan baik.

Keunggulan utama NB adalah kemampuannya untuk memberikan hasil cepat dengan sumber daya komputasi minimal, menjadikannya ideal untuk implementasi sistem real-time. Oleh karena itu, dalam penelitian ini NB digunakan sebagai pembanding terhadap SVM untuk mengevaluasi efektivitas metode probabilistik terhadap pendekatan berbasis *margin* maksimum.

3. METODE PENELITIAN

Penelitian ini menggunakan pendekatan *kuantitatif* dengan metode *eksperimental* untuk menganalisis sentimen ulasan pengguna aplikasi *ChatGPT*. Pendekatan ini dipilih karena memungkinkan *evaluasi* kinerja model klasifikasi secara objektif berdasarkan metrik kuantitatif. Fokus utama penelitian adalah membandingkan performa *algoritma Support Vector Machine (SVM)* dan *Naive Bayes* dalam mengklasifikasikan sentimen teks ulasan pengguna.

Pendekatan ini memungkinkan evaluasi kinerja model klasifikasi secara objektif berdasarkan metrik kuantitatif. Proses penelitian dilakukan secara sistematis melalui tujuh tahapan utama, sebagaimana umum digunakan dalam penelitian analisis sentimen berbasis machine learning [7].

3.1. Pengumpulan dan Pembersihan Data

Tahap awal penelitian adalah pengumpulan data ulasan pengguna aplikasi ChatGPT dari *platform Google Play Store* menggunakan teknik *web scraping* dengan bahasa pemrograman *Python*. Data dibatasi

pada ulasan berbahasa Indonesia. Dari sekitar 100.000 data ulasan awal, dilakukan pembersihan data dengan menghapus entri duplikat, data kosong, dan ulasan yang tidak relevan, sehingga diperoleh 9.879 data ulasan bersih. Proses pembersihan data penting untuk mengurangi noise dan meningkatkan kualitas data sebelum analisis lebih lanjut [2], [8].

3.2. Preprocessing Teks

Tahap *preprocessing* dilakukan untuk meningkatkan kualitas teks sebelum proses klasifikasi. Proses ini meliputi *case folding, cleansing, tokenizing, stopword removal,* dan *stemming*. *Preprocessing* bertujuan untuk menyeragamkan format teks, menghilangkan kata yang tidak bermakna sentimen, serta mengurangi variasi kata. Tahapan ini merupakan prosedur standar dalam analisis sentimen berbasis *Natural Language Processing (NLP)* [4].

3.3. Pelabelan Sentiment

Pelabelan sentimen dilakukan menggunakan pendekatan *lexicon-based* dengan memanfaatkan kamus sentimen Bahasa Indonesia. Setiap ulasan diklasifikasikan ke dalam tiga kelas sentimen, yaitu positif, negatif, dan netral, berdasarkan skor sentimen yang dihasilkan dari kemunculan kata bermuatan sentimen dalam teks. Pendekatan *lexicon-based* dipilih karena efisien dan sesuai untuk dataset berskala besar tanpa anotasi manual [1], [10].

3.4. Pembentukan Fitur

Tahap pembentukan fitur dilakukan menggunakan metode *Term Frequency–Inverse Document Frequency (TF-IDF)* untuk mengonversi teks menjadi *representasi* numerik. TF-IDF memberikan bobot lebih besar pada kata yang memiliki tingkat kepentingan tinggi dalam dokumen relatif terhadap keseluruhan korpus. Metode ini terbukti efektif dalam menangani data teks pendek dan bersifat sparse, serta banyak digunakan dalam penelitian analisis sentimen ulasan aplikasi [8].

3.5. Penerapan Algoritma Klasifikasi

Pada tahap ini, algoritma *Support Vector Machine (SVM)* dengan *kernel linear* dan *Multinomial Naive Bayes* diterapkan untuk mengklasifikasikan sentimen ulasan pengguna. Kedua algoritma dipilih karena sering digunakan sebagai pembanding dalam penelitian klasifikasi teks. Dataset dibagi menjadi data latih dan data uji dengan rasio 80:20 untuk memastikan proses pelatihan dan pengujian model berlangsung secara proporsional [7].

3.6. Evaluasi dan Perbandingan Model

Evaluasi kinerja model dilakukan menggunakan *metrik akurasi, precision, recall*, dan *F1-score*, serta *confusion matrix* untuk menganalisis kesalahan klasifikasi. Perbandingan hasil evaluasi antara SVM dan *Naive Bayes* digunakan untuk menentukan algoritma dengan performa terbaik dalam mengklasifikasikan sentimen ulasan pengguna aplikasi *ChatGPT* [2], [8].

3.7. Visualisasi Hasil

Tahap terakhir adalah visualisasi hasil analisis sentimen untuk mempermudah interpretasi hasil penelitian. Visualisasi yang digunakan meliputi distribusi sentimen, *confusion matrix*, dan *word cloud*. Visualisasi hasil membantu memberikan gambaran yang lebih intuitif mengenai kecenderungan sentimen pengguna dan karakteristik kata yang dominan pada masing-masing kelas sentimen.1

4. HASIL DAN PEMBAHASAN

Hasil analisis terhadap 9.879 ulasan pengguna aplikasi *ChatGPT* menunjukkan bahwa mayoritas ulasan berada pada kategori sentimen positif. Distribusi sentimen memperlihatkan bahwa sekitar dua pertiga data termasuk sentimen positif, diikuti oleh sentimen netral dan negatif dalam proporsi yang lebih kecil. Pola ini mengindikasikan bahwa secara umum pengguna memberikan respons yang baik terhadap performa dan kegunaan aplikasi *ChatGPT*, meskipun masih terdapat sejumlah ulasan yang menyoroti keterbatasan dan kekurangan aplikasi.

Dominasi sentimen positif mencerminkan tingkat kepuasan pengguna yang relatif tinggi, khususnya terkait kemudahan penggunaan dan manfaat aplikasi dalam membantu aktivitas sehari-hari. Sementara itu, ulasan dengan sentimen negatif umumnya berkaitan dengan keterbatasan akses fitur, kecepatan respons, serta isu teknis tertentu. Temuan ini sejalan dengan kecenderungan ulasan aplikasi berbasis kecerdasan buatan yang umumnya didominasi oleh opini positif namun tetap menyisakan *kritik konstruktif* dari pengguna.

4.1. Hasil Pengumpulan dan Pembersihan Data

Pada tahap pengumpulan data, diperoleh sekitar 100.000 ulasan pengguna aplikasi ChatGPT dari Google Play Store. Setelah dilakukan proses pembersihan data berupa penghapusan entri duplikat, data kosong, dan ulasan yang tidak relevan, jumlah data berkurang menjadi 52.118 ulasan bersih. Hasil ini menunjukkan bahwa proses pembersihan data berperan penting dalam meningkatkan kualitas dataset sebelum dilakukan analisis lebih lanjut, sebagaimana disarankan dalam penelitian analisis sentimen berbasis ulasan aplikasi.

```
1 import pandas as pd
2
3 data = pd.read_csv("/content/drive/MyDrive/2 dataset new25/hasil_scraper_ulasan_app_chatgpt.csv")
4 data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 5 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Review ID    100000 non-null  object
 1   Username     100000 non-null  object
 2   Rating       100000 non-null  int64
 3   Review Text  100000 non-null  object
 4   Date         100000 non-null  object
dtypes: int64(1), object(4)
memory usage: 3.8+ MB
```

Gambar 1. Hasil s*crapping* data

Dari Gambar *1*, dapat dilihat bahwa dataset ulasan aplikasi ChatGPT berhasil dimuat menggunakan pustaka *pandas* melalui fungsi *read_csv*. Hasil pemanggilan fungsi *info()* menunjukkan bahwa dataset terdiri dari 100.000 entri data, dengan indeks mulai dari 0 hingga 99.999. Dataset ini memiliki 5 kolom, yaitu *Review ID*, *Username, Rating, Review Text, dan Date*. Seluruh kolom memiliki jumlah data *non-null* sebanyak 100.000, yang mengindikasikan tidak adanya data kosong (*missing values*) pada tahap awal pengolahan. Tipe data yang digunakan meliputi satu kolom bertipe numerik (*int64*) dan empat kolom bertipe objek (*object*), yang merepresentasikan data teks dan tanggal. Informasi penggunaan memori sebesar 3,8 MB menunjukkan bahwa dataset memiliki ukuran yang relatif ringan dan efisien untuk diproses lebih lanjut.

```
1 df.drop_duplicates(subset ="Review Text", keep = 'first', inplace = True)
```

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 52118 entries, 0 to 99998
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Date         52118 non-null  object
 1   Username     52118 non-null  object
 2   Rating       52118 non-null  int64 
 3   Review Text  52118 non-null  object
dtypes: int64(1), object(3)
memory usage: 2.0+ MB
```

Gambar 2. Hasil setelah menghapus data duplikat

Dari Gambar *2*, dapat dilihat bahwa pada tahap pembersihan data telah dilakukan penghapusan data duplikat menggunakan fungsi *drop_duplicates* dengan parameter *subset* pada kolom *Review Text*. Proses ini bertujuan untuk memastikan bahwa setiap ulasan yang dianalisis bersifat unik dan tidak terjadi pengulangan konten yang dapat memengaruhi hasil analisis sentimen. Hasil pemanggilan fungsi *info()* menunjukkan bahwa jumlah data berkurang menjadi 52.118 entri, dengan indeks mulai dari 0 hingga 9.998. Dataset yang tersisa terdiri dari 4 kolom, yaitu *Date, Username, Rating, dan Review Text*, yang seluruhnya memiliki jumlah data non-null sebanyak 52.118, sehingga tidak terdapat data kosong setelah proses pembersihan. Tipe data yang digunakan meliputi satu kolom numerik (*int64*) dan tiga kolom bertipe objek (*object*). Selain itu, penggunaan memori sebesar 2.0 MB menunjukkan bahwa dataset menjadi lebih ringan dan efisien untuk diproses pada tahap preprocessing dan klasifikasi sentimen selanjutnya.

4.2. Hasil *Preprocessing* Teks

Tahap *preprocessing* menghasilkan data teks yang lebih terstruktur dan seragam. Proses *case folding*, *tokenizing*, *stopword removal*, dan *stemming* berhasil mengurangi variasi kata serta menghilangkan elemen yang tidak bermakna pada sentimen. Hasil *preprocessing* ini berdampak pada peningkatan kualitas *representasi* fitur teks, yang selanjutnya mempengaruhi performa model klasifikasi. Temuan ini sejalan dengan praktik umum dalam analisis sentimen berbasis NLP.

```
def case_folding(text):
    if isinstance(text, str):
        lowercase_text = text.lower()
        return lowercase_text
    else:
        return text

df['case_folding'] = df['cleaning'].apply(case_folding)
df.head(5)
```

	Date	Username	Rating	Review Text	cleaning	case_folding
0	2025-11-15 03:28:58	Pengguna Google	5	sangat membantu untuk semua hal	sangat membantu untuk semua hal	sangat membantu untuk semua hal
1	2025-11-15 03:27:26	Pengguna Google	1	diperbaiki lagi app nya masa engabisa membuat ...	diperbaiki lagi app nya masa engabisa membuat ...	diperbaiki lagi app nya masa engabisa membuat ...
2	2025-11-15 03:13:39	Pengguna Google	5	minusnya lama bngt hasilnya padahal jaringan g...	minusnya lama bngt hasilnya padahal jaringan g...	minusnya lama bngt hasilnya padahal jaringan g...
3	2025-11-15 03:12:22	Pengguna Google	2	bikin runyem doang	bikin runyem doang	bikin runyem doang
4	2025-11-15 03:12:20	Pengguna Google	5	baik	baik	baik

Gambar 3. Hasil Case Folding

Dari Gambar *3* dapat dilihat bahwa pada tahap *preprocessing* teks dilakukan proses *case folding* untuk menyeragamkan bentuk huruf pada data ulasan. Proses ini dilakukan dengan mendefinisikan sebuah fungsi *case_folding* yang mengubah seluruh teks ulasan menjadi huruf kecil (*lowercase*) menggunakan fungsi *lower()*. Tahap case folding bertujuan untuk menghindari perbedaan makna semu akibat perbedaan penggunaan huruf kapital dan huruf kecil, sehingga kata dengan makna yang sama dapat diperlakukan secara konsisten dalam proses analisis selanjutnya. Hasil penerapan *case folding* ditampilkan pada kolom baru bernama case_folding, yang menunjukkan bahwa teks ulasan telah berhasil dinormalisasi tanpa mengubah makna asli kalimat.

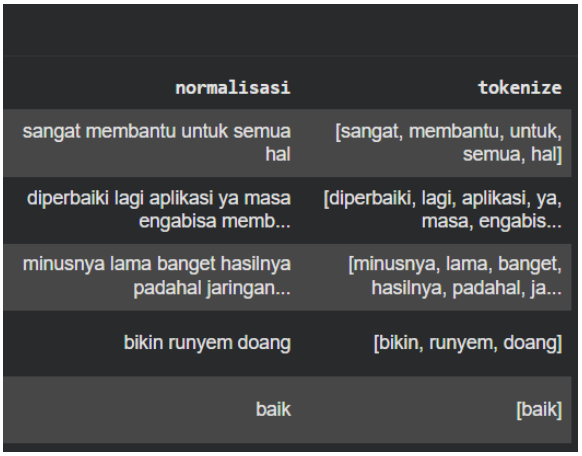
```
data['case_folding'].apply(lambda x: replace_taboo_words(x, kamus_ti

asi']])
```

case_folding	normalisasi
sangat membantu untuk semua hal	sangat membantu untuk semua hal
diperbaiki lagi app nya masa engabisa membuat ...	diperbaiki lagi aplikasi ya masa engabisa memb...
minusnya lama bngt hasilnya padahal jaringan g...	minusnya lama banget hasilnya padahal jaringan...
bikin runyem doang	bikin runyem doang
baik	baik

Gambar 4. Hasil normalisasi

Dari Gambar *4*, dapat dilihat bahwa pada tahap *preprocessing* teks dilakukan proses normalisasi kata untuk menangani penggunaan kata tidak baku, singkatan, dan variasi bahasa informal yang sering muncul dalam ulasan pengguna. Proses normalisasi dilakukan dengan menerapkan fungsi penggantian kata tidak baku menggunakan kamus kata (*kamus_tidak_baku*), sehingga kata-kata yang bersifat slang atau tidak standar dapat diubah menjadi bentuk baku yang sesuai. Hasil proses normalisasi ditampilkan pada kolom *normalisasi*, yang menunjukkan adanya perbaikan bentuk kata, seperti pengubahan kata singkatan dan istilah informal menjadi kata yang lebih formal tanpa mengubah makna asli kalimat. Tahap normalisasi ini bertujuan

untuk mengurangi variasi kata yang memiliki makna sama, sehingga dapat meningkatkan konsistensi data teks dan membantu algoritma klasifikasi dalam mengenali pola sentimen secara lebih akurat.



normalisasi	tokenize
sangat membantu untuk semua hal	[sangat, membantu, untuk, semua, hal]
diperbaiki lagi aplikasi ya masa engabisa memb...	[diperbaiki, lagi, aplikasi, ya, masa, engabis...
minusnya lama banget hasilnya padahal jaringan...	[minusnya, lama, banget, hasilnya, padahal, ja...
bikin runyem doang	[bikin, runyem, doang]
baik	[baik]

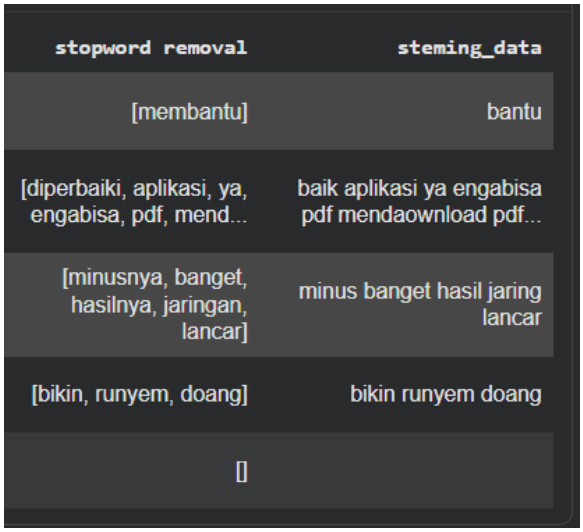
Gambar 5. Hasil tokenisasi

Dari Gambar 5, dapat dilihat bahwa pada tahap *preprocessing* teks dilakukan proses **tokenisasi** untuk memecah teks ulasan menjadi unit-unit kata yang lebih kecil. Proses tokenisasi dilakukan dengan mendefinisikan fungsi *tokenize* yang memisahkan teks berdasarkan spasi menggunakan metode *split()*. Hasil dari proses ini berupa daftar kata (*tokens*) yang merepresentasikan setiap kata penyusun kalimat ulasan. Penerapan fungsi tokenisasi pada kolom *normalisasi* menghasilkan kolom baru bernama *tokenize*, yang menunjukkan bahwa setiap ulasan telah berhasil diubah menjadi sekumpulan token, seperti kata "sangat", "membantu", "untuk", dan "semua". Tahap tokenisasi ini bertujuan untuk mempermudah pemrosesan teks pada tahap selanjutnya, seperti penghapusan stopword, stemming, dan pembentukan fitur TF-IDF.

tokenize	stopword removal
[sangat, membantu, untuk, semua, hal]	[membantu]
[diperbaiki, lagi, aplikasi, ya, masa, engabis...	[diperbaiki, aplikasi, ya, engabisa, pdf, mend...
[minusnya, lama, banget, hasilnya, padahal, ja...	[minusnya, banget, hasilnya, jaringan, lancar]
[bikin, runyem, doang]	[bikin, runyem, doang]
[baik]	[]

Gambar 6. Hasil *stopword removal*

Dari Gambar 6, dapat dilihat bahwa pada tahap preprocessing teks dilakukan proses penghapusan stopword (*stopword removal*) untuk menghilangkan kata-kata umum yang tidak memiliki kontribusi signifikan terhadap penentuan sentimen. Proses ini dilakukan dengan memanfaatkan daftar stopword Bahasa Indonesia yang tersedia pada pustaka NLTK, kemudian diterapkan melalui fungsi *remove_stopwords* untuk menyaring token yang termasuk dalam daftar stopword tersebut. Hasil proses ini ditampilkan pada kolom *stopword removal*, yang menunjukkan bahwa kata-kata seperti "untuk", "yang", dan kata hubung lainnya telah dihilangkan, sementara kata-kata yang memiliki muatan makna utama tetap dipertahankan. Tahap *stopword removal* bertujuan untuk mengurangi *noise* pada data teks dan meningkatkan fokus analisis pada kata-kata yang lebih *informatif*.



stopword removal	steming_data
[membantu]	bantu
[diperbaiki, aplikasi, ya, engabisa, pdf, mend...	baik aplikasi ya engabisa pdf mendaownload pdf...
[minusnya, banget, hasilnya, jaringan, lancar]	minus banget hasil jaring lancar
[bikin, runyem, doang]	bikin runyem doang
[]	

Gambar 7. Hasil *stemming* data

Dari gambar tersebut, dapat dilihat bahwa pada tahap *preprocessing* teks dilakukan proses *stemming* untuk mengubah setiap kata ke bentuk kata dasarnya. Proses stemming dilakukan dengan memanfaatkan *library Sastrawi*, yang dirancang khusus untuk menangani karakteristik *morfologi* Bahasa Indonesia. Tahapan ini diawali dengan pembuatan objek *stemmer* menggunakan *StemmerFactory*, kemudian diterapkan pada setiap token hasil proses *stopword removal*. Hasil stemming ditampilkan pada kolom *stemming_data*, yang menunjukkan bahwa kata-kata berimbuhan berhasil dikembalikan ke bentuk dasar, seperti kata "membantu" menjadi "bantu" dan "minusnya" menjadi "minus". Proses *stemming* bertujuan untuk mengurangi variasi kata yang memiliki makna sama, sehingga dapat meningkatkan konsistensi representasi teks.

4.3. Hasil Pelabelan Sentimen

Pelabelan sentimen menggunakan pendekatan *lexicon-based* menghasilkan tiga kelas sentimen, yaitu *positif*, *negatif*, dan *netral*. Distribusi label menunjukkan bahwa sentimen positif mendominasi ulasan pengguna, diikuti oleh sentimen netral dan negatif. Dominasi sentimen positif mengindikasikan

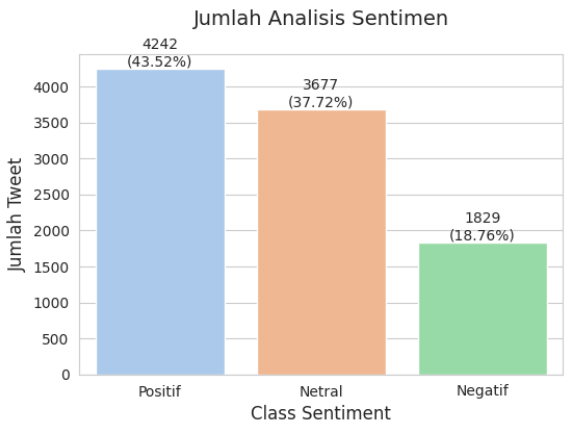
bahwa secara umum pengguna memiliki persepsi yang baik terhadap aplikasi *ChatGPT*.

```python
import pandas as pd
import requests

# Unduh kamus leksikon positif dan negatif dari GitHub
#positive_url = "https://raw.githubusercontent.com/fajri91/InSet/master/positive.tsv"
#negative_url = "https://raw.githubusercontent.com/fajri91/InSet/master/negative.tsv"

positive_lexicon = set(pd.read_csv('/content/drive/MyDrive/2 dataset new25/positive.tsv', sep="\t", header=None)[0])
negative_lexicon = set(pd.read_csv('/content/drive/MyDrive/2 dataset new25/negative.tsv', sep="\t", header=None)[0])

# Fungsi untuk menentukan sentimen dan menghitung skornya
def determine_sentiment(text):
    if isinstance(text, str):
        positive_count = sum(1 for word in text.split() if word in positive_lexicon)
        negative_count = sum(1 for word in text.split() if word in negative_lexicon)
        sentiment_score = positive_count - negative_count
        if sentiment_score > 0:
            sentiment = "Positif"
        elif sentiment_score < 0:
            sentiment = "Negatif"
        else:
            sentiment = "Netral"
        return sentiment_score, sentiment
    return 0, "Netral"

# Tentukan sentimen dan skor untuk setiap ulasan
data[['Score', 'Sentiment']] = data['steming_data'].apply(lambda x: pd.Series(determine_sentiment(x)))

# Tampilkan hasilnya
data.head(5)
```

	steming_data	Score	Sentiment
0	bantu	1	Positif
1	baik aplikasi ya engabisa pdf mendaownload pdf...	0	Netral
2	minus banget hasil jaring lancar	2	Positif
3	bikin runyem doang	0	Netral
4	bayar mulu	0	Netral

Gambar 8. Hasil pelabelan metode *lexicon-based*

Pada Gambar *8*, tahap pelabelan sentimen digunakan pendekatan *lexicon-based* dengan memanfaatkan kamus kata positif dan negatif berbahasa Indonesia. Kamus sentimen diperoleh dari repositori publik dan dimuat ke dalam sistem sebagai daftar kata positif (*positive_lexicon*) dan kata negatif (*negative_lexicon*). Setiap ulasan yang telah melalui tahap *preprocessing* dan *stemming* dianalisis dengan menghitung jumlah kemunculan kata positif dan kata negatif. Skor sentimen ditentukan berdasarkan selisih antara jumlah kata positif dan negatif yang ditemukan dalam teks ulasan. Apabila skor sentimen bernilai positif, ulasan diklasifikasikan sebagai sentimen positif, apabila bernilai negatif, diklasifikasikan sebagai sentimen negatif, sedangkan apabila skor bernilai nol, ulasan dikategorikan sebagai sentimen netral. Hasil pelabelan ditampilkan dalam kolom *Score* dan *Sentiment*, yang menunjukkan bahwa setiap ulasan telah berhasil diberi label sentimen secara otomatis.



Gambar 9. Hasil preprocessing data dalam diagram batang

Pada Gambar *9* dapat di ketahui dari 9.748 ulasan yang diproses, diperoleh hasil:

a. Positif: 4.242 ulasan (43,52%)
b. Negatif: 1.829 ulasan (18,76%)
c. Netral: 3.677 ulasan (37,72%)

4.4. Hasil Pembentukan Fitur

Pembentukan fitur menggunakan metode TF-IDF menghasilkan representasi numerik dari data teks ulasan. TF-IDF mampu menyoroti kata-kata yang memiliki tingkat kepentingan tinggi dalam *korpus* ulasan, sehingga membantu algoritma klasifikasi dalam membedakan karakteristik setiap kelas sentimen. TF-IDF menghasilkan *matriks* berukuran 9.748 × 5.000 (9.748 dokumen dan 5.000 fitur unik). Vektor hasil transformasi inilah yang digunakan sebagai masukan untuk algoritma *SVM* dan *Naive Bayes.*

```python
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

cleaned_data = data.dropna(subset=['steming_data'])

X = cleaned_data['steming_data']
y = cleaned_data['Sentiment']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

print("Jumlah data latih:", len(X_train))
print("Jumlah data uji:", len(X_test))
print('==========================')

vectorizer = TfidfVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

X_train_vec.shape, X_test_vec.shape
```

```
Jumlah data latih: 7798
Jumlah data uji: 1950
==========================
((7798, 5774), (1950, 5774))
```

Gambar 10. Pembentukan fitur *TF-IDF*

Dari Gambar *10*, dapat dilihat bahwa setelah proses *preprocessing* dan pelabelan sentimen selesai, data yang digunakan kemudian dibagi menjadi data latih dan data uji menggunakan metode *train-test split*. Pembagian data dilakukan dengan rasio 80% data latih dan 20% data uji, serta menerapkan parameter *stratify* pada label sentimen untuk menjaga proporsi kelas sentimen pada kedua subset data. Hasil pembagian menunjukkan bahwa jumlah data latih yang digunakan sebanyak 7.798 data, sedangkan data uji berjumlah 1.950 data. Selanjutnya, data teks pada kolom *stemming_data* direpresentasikan ke dalam bentuk numerik menggunakan metode *Term Frequency–Inverse Document Frequency (TF-IDF)*. Proses ini menghasilkan matriks fitur berdimensi 5.774 fitur kata, baik pada data latih maupun data uji.

4.5. Hasil Penerapan Algoritma SVM dan Naïve Bayes

Model *Support Vector Machine* dan *Multinomial Naive Bayes* dilatih menggunakan data latih dan diuji menggunakan data uji dengan rasio 80:20. Hasil pengujian menunjukkan bahwa kedua algoritma mampu mengklasifikasikan sentimen ulasan pengguna, namun dengan tingkat performa yang berbeda. SVM menunjukkan kemampuan klasifikasi yang lebih stabil dibandingkan *Naive Bayes*, khususnya dalam menangani data dengan dimensi fitur yang tinggi.

Tabel 1. Classification Report for SVM

No.	kelas	precision	recall	f1-score	support
1	Negatif	0.895	0.855	0.875	429.000
2	Netral	0.872	0.915	0.893	840.000
3	Positif	0.945	0.913	0.929	681.000
4	accuracy	0.902	0.902	0.902	0.902
5	macro avg	0.904	0.895	0.899	1950.000
6	weighted avg	0.903	0.902	0.902	1950.000

Tabel *1* menunjukan bahwa hasil evaluasi kinerja *algoritma Support Vector Machine (SVM)* dalam mengklasifikasikan sentimen ulasan pengguna mempunyai performa yang cukup tinggi pada seluruh kelas sentimen. Untuk kelas negatif, model SVM memperoleh nilai *precision* sebesar 0,895, recall sebesar 0,855, dan *f1-score* sebesar 0,875 dengan jumlah data uji sebanyak 429 ulasan. Pada kelas *netral*, nilai *precision* yang diperoleh sebesar 0,872, *recall* 0,915, dan *f1-score* 0,893 dengan 840 ulasan, yang menunjukkan bahwa model memiliki kemampuan yang baik dalam mengenali ulasan netral. Sementara itu, kelas positif menghasilkan performa tertinggi dengan nilai *precision* 0,945, *recall* 0,913, dan *f1-score* 0,929 dari 681 ulasan, menandakan bahwa model SVM sangat efektif dalam mengidentifikasi sentimen positif. Secara keseluruhan, model SVM mencapai tingkat akurasi sebesar 90,2%, dengan nilai macro average *f1-score* sebesar 0,899 dan *weighted average f1-score* sebesar 0,902. Hasil ini menunjukkan bahwa algoritma SVM memiliki performa yang stabil dan andal dalam mengklasifikasikan sentimen ulasan aplikasi ChatGPT berbasis representasi fitur TF-IDF.

Tabel 2. Classification Report for Naive Bayes

	precision	recall	f1-score	support
Negatif	0.917	0.336	0.491	429.000
Netral	0.660	0.787	0.718	840.000
Positif	0.729	0.847	0.784	681.000
accuracy	0.709	0.709	0.709	0.709
macro avg	0.769	0.657	0.664	1950.000
weighted avg	0.741	0.709	0.691	1950.000

Berdasarkan gambar tersebut, dapat dilihat bahwa hasil evaluasi kinerja algoritma Naive *Bayes* dalam mengklasifikasikan sentimen ulasan pengguna menunjukkan performa yang lebih rendah dibandingkan algoritma SVM. Pada kelas negatif, Naive Bayes menghasilkan nilai precision sebesar 0,917, namun nilai *recall* relatif rendah yaitu 0,336, sehingga *f1-score* yang diperoleh hanya 0,491 dari 429 data uji. Hal ini menunjukkan bahwa meskipun model cukup tepat ketika memprediksi sentimen negatif, masih banyak data negatif yang tidak berhasil teridentifikasi. Untuk kelas netral, diperoleh nilai *precision* 0,660, recall 0,787, dan *f1-score* 0,718 dengan jumlah 840 data, yang menandakan kemampuan klasifikasi yang cukup namun belum optimal. Sementara itu, pada kelas positif, model memperoleh nilai precision 0,729, recall 0,847, dan f1-score 0,784 dari 681 data, yang menunjukkan bahwa Naive Bayes masih mampu mengenali sentimen positif dengan cukup baik. Secara keseluruhan, algoritma *Naive Bayes* mencapai tingkat akurasi sebesar 70,9%, dengan nilai *macro average f1-score* 0,664 dan weighted average *f1-score* 0,691. Hasil ini menunjukkan bahwa meskipun *Naive Bayes* memiliki keunggulan dari sisi kesederhanaan dan efisiensi komputasi, performanya dalam mengklasifikasikan sentimen ulasan aplikasi ChatGPT masih berada di bawah algoritma *Support Vector Machine*.

4.6. Hasil Evaluasi dan Perbandingan Model

Evaluasi kinerja model menggunakan metrik akurasi, precision, recall, dan *F1-score* menunjukkan bahwa algoritma SVM menghasilkan performa yang lebih baik dibandingkan *Naive Bayes*. Tingkat akurasi SVM mencapai lebih dari 90%, sedangkan *Naive Bayes* menunjukkan akurasi yang lebih rendah. *Confusion matrix* memperlihatkan bahwa SVM memiliki tingkat kesalahan klasifikasi yang lebih kecil, terutama pada kelas sentimen netral dan negatif.

4.7. Visualisasi Hasil Analisis

Visualisasi hasil analisis sentimen dilakukan untuk mempermudah interpretasi temuan penelitian. Visualisasi distribusi sentimen menunjukkan dominasi sentimen positif pada ulasan pengguna *ChatGPT*. *Confusion matrix* memberikan gambaran perbandingan kesalahan klasifikasi antara SVM dan Naive Bayes, sedangkan word cloud membantu mengidentifikasi kata-kata yang sering muncul pada masing-masing kelas sentimen. Visualisasi ini mendukung pemahaman kualitatif terhadap hasil analisis sentimen dan memberikan konteks tambahan terhadap hasil kuantitatif.

Perbedaan performa antara algoritma *Support Vector Machine* dan *Naive Bayes* menunjukkan bahwa karakteristik data ulasan pengguna sangat mempengaruhi efektivitas model klasifikasi. SVM mampu membangun batas pemisah yang lebih optimal antar kelas sentimen, sehingga lebih stabil dalam menangani data dengan distribusi kelas yang tidak seimbang. Sebaliknya, *Naive Bayes* mengasumsikan independensi antar fitur, yang pada praktiknya tidak selalu terpenuhi pada data teks ulasan. Temuan ini mengindikasikan bahwa pemilihan algoritma klasifikasi harus mempertimbangkan sifat data dan tujuan analisis, khususnya pada konteks evaluasi aplikasi berbasis kecerdasan buatan yang memiliki dinamika opini pengguna yang kompleks

5. KESIMPULAN DAN SARAN

Penelitian ini berhasil membangun pipeline analisis sentimen ulasan pengguna aplikasi ChatGPT berbasis Python yang terstruktur, mulai dari

pengumpulan data, preprocessing teks, pelabelan sentimen menggunakan metode lexicon-based InSet, pembentukan fitur TF-IDF, hingga pelatihan dan evaluasi model machine learning. Hasil penelitian menunjukkan bahwa Support Vector Machine (SVM) memberikan performa terbaik dibandingkan Multinomial Naive Bayes (MNB), dengan nilai akurasi, precision, recall, dan F1-score yang lebih tinggi serta prediksi yang lebih stabil, terutama pada data teks pendek berbahasa Indonesia yang bersifat sparse dan berdimensi tinggi. Meskipun demikian, MNB tetap menunjukkan performa yang cukup kompetitif sebagai model baseline, namun mengalami keterbatasan pada kelas negatif akibat ketidakseimbangan label dan keberagaman bahasa informal. Evaluasi menggunakan metrik precision, recall, dan F1-score terbukti penting untuk memberikan gambaran performa model yang lebih komprehensif dibandingkan hanya mengandalkan akurasi. Untuk pengembangan selanjutnya, disarankan penggunaan metode pelabelan yang lebih akurat seperti labeling manual, hybrid lexicon–machine learning, atau crowdsourcing, serta eksplorasi model yang lebih beragam seperti Logistic Regression, Random Forest, XGBoost, deep learning, dan model berbasis transformer. Selain itu, perlu dilakukan augmentasi dan normalisasi data untuk menangani bahasa informal, pengembangan analisis sentimen berbasis aspek, integrasi hasil ke dalam sistem atau dashboard analitik, serta penambahan dataset multilingual agar analisis sentimen menjadi lebih komprehensif dan bernilai tinggi dalam konteks evaluasi pengalaman pengguna dan pengembangan aplikasi kecerdasan buatan.

DAFTAR PUSTAKA

[1] H. Taherdoost and M. Madanchian, "Artificial intelligence and sentiment analysis: A review in competitive research," *Computers*, vol. 12, no. 2, p. 37, 2023, doi: 10.3390/computers12020037.

[2] E. Nichifor *et al.*, "Utilising artificial intelligence to turn reviews into business enhancements through sentiment analysis," *Electronics (Basel).*, vol. 12, no. 21, p. 4538, 2023, doi: 10.3390/electronics12214538.

[3] M. Haoues, R. Mokni, and A. Sellami, "Machine learning for mHealth apps quality evaluation," *Software Quality Journal*, vol. 31, no. 4, pp. 1179–1209, Dec. 2023, doi: 10.1007/s11219-023-09630-8.

[4] V. H. Pranatawijaya, N. N. K. Sari, R. A. Rahman, E. Christian, and S. Geges, "Unveiling user sentiment: Aspect-based analysis and topic modelling of ride-hailing and Google Play app reviews," *Journal of Information Systems Engineering and Business Intelligence*, vol. 10, no. 3, pp. 328–339, 2024, doi: 10.20473/jisebi.10.3.328-339.

[5] Y. Mao, "Sentiment analysis methods, applications, and challenges," *Expert Syst. Appl.*, vol. 223, 2024, doi: 10.1016/j.eswa.2024.119862.

[6] M. Bordoloi, S. Patowary, and M. Choudhury, "Sentiment analysis: A survey on design framework, algorithms and applications," *Applied Sciences*, vol. 13, no. 7, p. 4550, 2023, doi: 10.3390/app13074550.

[7] L. Ashbaugh and Y. Zhang, "A comparative study of sentiment analysis on customer reviews using machine learning and deep learning," *Computers*, vol. 13, no. 12, p. 340, 2024, doi: 10.3390/computers13120340.

[8] P. S. Ghatora, S. E. Hosseini, S. Pervez, M. J. Iqbal, and N. Shaukat, "Sentiment analysis of product reviews using machine learning and pre-trained LLM," *Big Data and Cognitive Computing*, vol. 8, no. 12, p. 199, 2024, doi: 10.3390/bdcc8120199.

[9] H. Li, X. Li, F. Dunkin, Z. Zhang, and X. Lu, "Adaptive multi-granularity trust management scheme for UAV visual sensor security under adversarial attacks," *Comput. Secur.*, vol. 148, p. 104108, Jan. 2025, doi: 10.1016/j.cose.2024.104108.

[10] Y. C. Hua, P. Denny, J. Wicker, and K. Taskova, "A systematic review of aspect-based sentiment analysis: domains, methods, and trends," *Artif. Intell. Rev.*, vol. 57, no. 11, p. 296, Sep. 2024, doi: 10.1007/s10462-024-10906-z.