Muhammad Risalah Naufal

21051214008

S1 Sistem Informasi 2021

PRAKTIKUM: NLP WORDCLOUD

STUDI KASUS INDIHOME

A. PENDAHULUAN

Permainan mobile merupakan salah satu hiburan favorit bagi banyak pengguna smartphone di era digital saat ini. Mobile Legend, sebagai salah satu game mobile yang populer, tidak hanya menarik perhatian pemain dengan gameplaynya yang seru, tetapi juga menciptakan komunitas pengguna yang besar dan aktif. Dalam konteks ini, analisis sentimen terhadap komentar netizen pada platform Google Playstore dapat memberikan wawasan berharga terkait dengan persepsi dan pengalaman pengguna terhadap game Mobile Legend.

Praktikum Analisis Sentimen ini bertujuan untuk menyelidiki dan menginterpretasi sentimen yang terkandung dalam komentar-komentar netizen terkait Mobile Legend di Google Playstore. Dengan menggunakan metode analisis sentimen dan teknik machine learning, kita dapat mengidentifikasi kecenderungan umum dari pandangan positif, negatif, atau netral yang diungkapkan oleh pengguna game ini.

Analisis ini tidak hanya memberikan gambaran tentang bagaimana pengguna merespons game Mobile Legend, tetapi juga dapat memberikan informasi berharga untuk pengembang game dan pihak terkait dalam meningkatkan kualitas dan kepuasan pengguna. Oleh karena itu, praktikum ini akan melibatkan penggunaan algoritma Machine Learning seperti Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen dari komentar-komentar tersebut.

Melalui praktikum ini, diharapkan dapat diperoleh pemahaman yang lebih mendalam tentang pandangan pengguna terhadap Mobile Legend serta memberikan landasan bagi perbaikan dan pengembangan yang lebih baik di masa depan.

B. METODE

Pada penelitian ini ada beberapa urutan metode yang dipakai:

1. Pengumpulan Data:

   Data yang digunakan dalam analisis sentimen ini diperoleh dari komentar-komentar netizen pada platform Google Playstore yang terkait dengan game Mobile Legend. Komentar-komentar ini mencakup berbagai pandangan, ulasan, dan pengalaman pengguna yang dapat mencerminkan sentimen mereka terhadap permainan.
2. Pembersihan Data:

   Data yang diambil dari Google Playstore mungkin memiliki variasi dalam format dan kualitas. Oleh karena itu, langkah ini melibatkan pembersihan data untuk menghilangkan karakter khusus, tautan, atau elemen lain yang tidak relevan. Proses ini juga dapat mencakup pemilihan fitur dan pemrosesan teks lanjutan seperti lemmatisasi atau stemming.
3. Tokenisasi dan Vektorisasi:

   Komentar-komentar teks kemudian dipecah menjadi token-token yang merupakan unit-unit terkecil dari kata-kata. Selanjutnya, teks tersebut diubah menjadi vektor fitur menggunakan metode seperti CountVectorizer. Ini membantu mengubah teks menjadi representasi numerik yang dapat digunakan oleh algoritma pembelajaran mesin.
4. Pembagian Data:

Data kemudian dibagi menjadi dua bagian: bagian pelatihan (train set) dan bagian pengujian (test set). Pembagian ini dilakukan untuk melatih model pada subset data dan menguji kinerjanya pada data yang tidak terlihat sebelumnya.

5. Model Pembelajaran Mesin:

   Dalam praktikum ini, dua model klasifikasi yang digunakan adalah Naive Bayes dan Random Forest. Model ini dilatih menggunakan bagian pelatihan dari data dan kemudian diuji pada bagian pengujian untuk mengukur kinerja mereka dalam mengklasifikasikan sentimen komentar.

6. Evaluasi Model:

   Kinerja model dievaluasi menggunakan metrik seperti akurasi, presisi, recall, dan F1 score. Metrik-metrik ini memberikan gambaran tentang sejauh mana model mampu mengklasifikasikan sentimen dengan benar.

7. Analisis Sentimen:

   Hasil dari model klasifikasi digunakan untuk menganalisis sentimen secara keseluruhan dari komentar-komentar pengguna. Ini mencakup pemahaman tentang apakah sentimen umumnya positif, negatif, atau netral, dan juga dapat mencakup identifikasi pola atau kata-kata kunci yang sering muncul.

8. Rekomendasi dan Implikasi:

   Berdasarkan hasil analisis sentimen, praktikum ini dapat memberikan rekomendasi dan implikasi bagi pengembang Mobile Legend. Informasi ini dapat digunakan untuk memperbaiki atau meningkatkan aspek-aspek tertentu dari permainan guna meningkatkan kepuasan pengguna.

C. PEMBAHASAN

1. Implementasi Pengumpulan Data dengan google-play-scraper untuk Analisis Sentimen Mobile Legend di Google Playstore.

Pemasangan modul google-play-scraper dilakukan menggunakan perintah !pip install google-play-scraper. Modul ini memungkinkan pengguna untuk mengakses dan mengambil informasi dari Google Playstore, termasuk ulasan dan peringkat pengguna untuk aplikasi tertentu. Penggunaan modul ini mendukung pengumpulan data secara otomatis dari platform distribusi aplikasi, memudahkan proses analisis sentimen.

2. Ekstraksi Informasi Aplikasi dari Google Playstore menggunakan google_play_scraper

Bagian ini berfokus pada penggunaan modul google_play_scraper untuk mengumpulkan informasi terkait aplikasi dari Google Playstore. Fungsi app dari modul ini memungkinkan kita untuk mengekstrak berbagai detail aplikasi, seperti ulasan, peringkat, deskripsi, dan fitur-fitur lainnya. Data ini dapat diambil untuk analisis lebih lanjut, termasuk analisis sentimen, penilaian pengguna, dan karakteristik aplikasi.

3. Pengumpulan Jumlah Ulasan Aplikasi Mobile Legends dari Google Playstore menggunakan google_play_scraper

```
[3] #scrape jumlah ulasan yang diinginkan
    from google_play_scraper import Sort, reviews

    result, continuation_token = reviews(
        'com.mobile.legends',
        lang='id',  #disini kita mau men scrape data ulasan aplikasi shopee yang berada di google play store
        country='id', #kita setting bahasa nya menjadi bahasa indonesia
        sort=Sort.MOST_RELEVANT, # # kemudian kita gunakan most_relevan untuk mendapatkan ulasan yang paling relevant
        count=1000, # disini jumlah ulasan yang mau kita ambil ada seribu
        filter_score_with=None # # kemudian di filter_score kita gunakan None untuk mengambil semua score atau ratting bintang 1 sampai 5
    )
```

Pada bagian ini, dilakukan penggunaan fungsi reviews dari modul google_play_scraper untuk mengekstrak jumlah ulasan aplikasi Mobile Legends dari Google Playstore. Pengambilan dilakukan dengan mengatur parameter seperti bahasa, negara, urutan ulasan, jumlah ulasan yang diinginkan, dan filter score.

4. Pemrosesan Data Ulasan Aplikasi Mobile Legends menggunakan Pandas

```
[4] df_busu = pd.DataFrame(np.array(result),columns=['review'])
    df_busu = df_busu.join(pd.DataFrame(df_busu.pop('review').tolist()))
    df_busu.head()
```

	reviewId	userName	userImage	content	score	thumbsUpCount	reviewCreatedVersion	at	replyContent	rep
0	8336e2fb-3770-4e12-96d9-f06ffe38fd4c	Sun thin Then	https://play-lh.googleusercontent.com/a/ACg8oc...	Gamenya sih udah bagus bgt, grafiknya mantap, ...	1	4942	1.8.22.8944	2023-10-29 08:10:42	None	
1	1c26d29b-8f2d-4467-a68b-94bf146ed8dd	Abdul Ghani Rossyidi	https://play-lh.googleusercontent.com/a-/ALV-U...	Untuk event2 sdah oke lah. Tapi tolong priorit...	3	16257	1.8.22.8944	2023-10-29 13:25:49	None	
2	b9cfbc11-a773-478a-a30a-d8f471a8fd2e	Kayna Adiva	https://play-lh.googleusercontent.com/a-/ALV-U...	Saya rank legend 5 malah terus bertemu musuh y...	1	19436	1.8.22.8944	2023-10-31 10:01:24	Dear Hero,\nKami bertekad untuk menciptakan li...	2
3	4d0981a2-1896-44f6-9f62-a1fce0344713	Ina Alfiyanah	https://play-lh.googleusercontent.com/a-/ALV-U...	Setelah di upgrade malah nambah ancur aja, mas...	1	12026	1.8.22.8944	2023-10-22 07:41:54	None	
4	3063a6e4-6cf3-4a2b-b0f8-8b2abcff4440	Riko Novianto	https://play-lh.googleusercontent.com/a/ACg8oc...	Overall game ini udah bagus tetapi yg jadi per...	3	1625	1.8.22.8944	2023-10-27 05:49:47	Dear Hero,\nMohon maaf atas ketidaknyamananya ...	2

Pada bagian ini, dilakukan pembentukan DataFrame menggunakan modul pandas untuk menyimpan hasil ulasan aplikasi Mobile Legends yang telah diambil sebelumnya. Setelah itu, dilakukan pemrosesan data untuk membentuk DataFrame yang terstruktur dengan kolom-kolom yang sesuai.

5. Sortir Data Ulasan Berdasarkan Tanggal

```
[5] #Run This Code to Sort the Data By Date

    new_df = df_busu[['userName', 'score','at', 'content']]
    sorted_df = new_df.sort_values(by='at', ascending=False) #Sort by Newst, change to True if you want to sort by Oldest.
    sorted_df.head()
```

	userName	score	at	content
111	Muhamad Hardiansyah	3	2023-11-07 12:49:39	Min gimana sih update yang kali ini bukannya m...
650	Natalis Meningan	1	2023-11-04 09:49:45	Game udah bagus, tapi ada satu hal yang saya l...
826	Aban Guanteng	1	2023-11-04 09:38:38	Ketemu tim yang sangat berbeda dengan musuh di...
685	Farhan Ramadani	3	2023-11-04 09:07:08	-Minta tolong dong min, jaringannya diperbaiki...
661	Hafizh Mangkona	1	2023-11-04 08:38:14	Kepada Developer Game tolong untuk lebih mempe...

Pada bagian ini, dilakukan proses pengurutan data ulasan aplikasi Mobile Legends berdasarkan tanggal. Fungsi sort_values dari modul pandas digunakan untuk menyusun ulasan berdasarkan tanggal, dan parameter ascending=False digunakan untuk mengurutkan dari yang terbaru. Jika ingin mengurutkan dari yang terlama, parameter tersebut dapat diubah menjadi ascending=True.

6. Seleksi Kolom pada DataFrame Hasil Sortir

```
[6] my_df=sorted_df[['content', 'score']]
    #karena kita hanya membutuhkan kolom content dan score maka kita lakukan filter kolom lgi hingga menyisakan kolom content dan score.

[7] print(my_df)

                                               content  score
111  Min gimana sih update yang kali ini bukannya m...      3
650  Game udah bagus, tapi ada satu hal yang saya l...      1
826  Ketemu tim yang sangat berbeda dengan musuh di...      1
685  -Minta tolong dong min, jaringannya diperbaiki...      3
661  Kepada Developer Game tolong untuk lebih mempe...      1
..                                                 ...    ...
479  Halo moonton, saya ada beberapa keluhan nih. Y...      1
986  Untuk match terlalu ekstrim banyak orang ga je...      1
993  Akhir akhir ini setelah update ada beberapa bu...      1
477  Gimana mau maju . Memori yg di gunakan sampe 4...      5
483  Untuk keseluruhan sudah bagus tapi nih masalah...      3

[1000 rows x 2 columns]
```

Pada bagian ini, dilakukan seleksi kolom pada DataFrame hasil sortir. Kolom-kolom yang dipilih untuk disimpan dalam DataFrame baru (my_df) adalah 'content' (konten ulasan) dan 'score' (peringkat atau rating). Seleksi ini dilakukan karena hanya kolom-kolom tersebut yang dibutuhkan untuk analisis lebih lanjut.

7. Pembuatan Label Sentimen Berdasarkan Peringkat

```
[8] def pelabelan(score):
      if score < 3:
        return 'Negatif'
      elif score == 3 :
        return 'Netral'
      elif score > 3 :
        return 'Positif'
    my_df['Label'] = my_df ['score'].apply(pelabelan)
    my_df.head(50)

<ipython-input-8-674b4ca6ef3f>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  my_df['Label'] = my_df ['score'].apply(pelabelan)
```

	content	score	Label
111	Min gimana sih update yang kali ini bukannya m...	3	Netral
650	Game udah bagus, tapi ada satu hal yang saya l...	1	Negatif
826	Ketemu tim yang sangat berbeda dengan musuh di...	1	Negatif
685	-Minta tolong dong min, jaringannya diperbaiki...	3	Netral
661	Kepada Developer Game tolong untuk lebih mempe...	1	Negatif
269	game yang sangat ramah dan sangat rekomedasi u...	3	Netral
966	Knp ya aku di kasih musuh selalu pro? Pala pun...	3	Netral
651	Dear moonton .. kenapa saya update versi terba...	2	Negatif
909	Ini game cuma ngabisin waktu ajh masa jaringa ...	1	Negatif
833	Saya sangat kecewa dengan game satu ini. Karen...	1	Negatif
605	Kenapa Pas Saya Bermain Di mobile legends Seri...	5	Positif
774	Pebaharuan bukan bener malah sering ngeleg pay...	3	Netral
968	Game apalah ini rank stak legend Mulu ketemu t...	1	Negatif
941	Perbaiki sistem pembagian tim , kasian player ...	1	Negatif
983	Saya sangat menikmati game ini karna menantang...	5	Positif
641	Tolong perbaiki bug sesudah update mau login m...	5	Positif
738	Tolong la mode offline dikembalikan lagi, saya...	1	Negatif
559	Bagus tapi engak adil. Dimana yang keluar game...	2	Negatif
493	Monton saya minta tolong agar jaringanya di pe...	1	Negatif
991	Game Mobile legend ini sangat asik jika di mai...	5	Positif
696	Kenapa setelah update banyak gangguan..misal t...	4	Positif
834	Magic chess permainan yg tidak adanya keseimba...	1	Negatif
757	Jangan asal ban dan kurangin kredit skor min, ...	5	Positif
193	Untuk di gamenya udh bgs, di loby nya juga udh...	3	Netral
337	Untuk di gamenya udh bgs, di loby nya juga udh...	3	Netral
855	Gamenya bagus event2nya oke tapi kalo masalahn...	4	Positif
737	pak montton tolong lah kalau ngasih tim yang b...	1	Negatif

```
[10] my_df.to_excel("scrapping_data_ML.xlsx", index = False)  #kemudian save menjadi file csv
```

Dalam bagian ini, dilakukan pembuatan label sentimen ('Negatif', 'Netral', atau 'Positif') berdasarkan peringkat ('score') ulasan. Fungsi pelabelan didefinisikan untuk mengembalikan label berdasarkan aturan tertentu, dan kemudian diterapkan pada kolom 'score' menggunakan metode apply pada DataFrame my_df.

Kemudian simpan datanya ke "scrapping_data_ML.xlsx".

8. Instalasi Paket NLTK, Sastrawi, dan Emoji

```
[11] !pip install nltk
     !pip install sastrawi
     !pip install emoji

Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.1)
Collecting sastrawi
  Downloading Sastrawi-1.0.1-py2.py3-none-any.whl (209 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 209.7/209.7 kB 4.5 MB/s eta 0:00:00
Installing collected packages: sastrawi
Successfully installed sastrawi-1.0.1
Collecting emoji
  Downloading emoji-2.8.0-py2.py3-none-any.whl (358 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 358.9/358.9 kB 6.6 MB/s eta 0:00:00
Installing collected packages: emoji
Successfully installed emoji-2.8.0
```

Pada bagian ini, dilakukan instalasi paket-paket Python yang dibutuhkan, yaitu NLTK, Sastrawi, dan Emoji. Paket-paket ini sering digunakan dalam pemrosesan teks, pemrosesan bahasa alami, dan penanganan emoji.

9. Import Modul dan Instalasi Berbagai Ressource untuk Pemrosesan Teks

```
[12] import pandas as pd
     import nltk
     import re
     import string
     import emoji
     from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
     from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
     from nltk.corpus import stopwords
     from nltk.tokenize import word_tokenize
     nltk.download('stopwords')
     nltk.download('punkt');

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

Pada bagian ini, dilakukan import modul-modul yang dibutuhkan untuk pemrosesan teks, seperti pandas, nltk, re (regular expression), dan emoji. Selain itu, dilakukan instalasi dan pengunduhan sumber daya (resource) untuk pemrosesan teks menggunakan Sastrawi dan NLTK.

10. Inisialisasi Stemmer Sastrawi

```
[13] # Inisialisasi stemmer Sastrawi
     factory = StemmerFactory()
     stemmer = factory.create_stemmer()
```

Pada bagian ini, dilakukan inisialisasi stemmer dari Sastrawi. Sastrawi menyediakan stemmer untuk Bahasa Indonesia, yang berguna dalam proses pemangkasan kata ke bentuk dasarnya.

11. Pemuatan Data dari Berkas Excel

```
[14] # Load data from Excel file
     file_path = 'scrapping_data_ML.xlsx'
     my_df = pd.read_excel(file_path)
```

Pada bagian ini, dilakukan pemuatan data dari berkas Excel ke dalam DataFrame menggunakan modul pandas.

12. Proses Pembersihan Teks (Text Cleaning)

```
[15] # Proses Cleaning

def remove_kata(comment):
  comment = comment.replace('\t', ' ').replace('\n', ' ').replace('\\u', ' ').replace('\\', '')
  comment = comment.encode('ascii', 'replace').decode('ascii')
  comment = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)", " ", comment).split())
  return comment.replace("http://", " ").replace("https://", " ")

my_df['content'] = my_df['content'].apply(remove_kata)


def remove_angka(comment):
    return re.sub(r"\d+", " ", comment)

my_df['content'] = my_df['content'].apply(remove_angka)


def remove_punctuation(comment):
    return comment.translate(str.maketrans("", "", string.punctuation))

my_df['content'] = my_df['content'].apply(remove_punctuation)


def remove_whitespace_LT(comment):
    return comment.strip()

my_df['content'] = my_df['content'].apply(remove_whitespace_LT)


def remove_whitespace_multiple(comment):
    return re.sub('\s+', ' ', comment)

my_df['content'] = my_df['content'].apply(remove_whitespace_multiple)


def remove_single_char(comment):
    return re.sub(r"\b[a-zA-Z]\b", " ", comment)

my_df['content'] = my_df['content'].apply(remove_single_char)


print('Hasil Cleaning: \n')
print(my_df.head(50))
```

```
Hasil Cleaning:

                                              content  score    Label
0   Min gimana sih update yang kali ini bukannya m...      3   Netral
1   Game udah bagus tapi ada satu hal yang saya li...      1  Negatif
2   Ketemu tim yang sangat berbeda dengan musuh di...      1  Negatif
3   Minta tolong dong min jaringannya diperbaikin ...      3   Netral
```

Pada bagian ini, dilakukan serangkaian fungsi pembersihan teks pada kolom 'content' dalam DataFrame my_df. Fungsi-fungsi ini mencakup penghapusan karakter tab, newline, karakter khusus tertentu, URL, angka, tanda baca, dan spasi ganda. Tujuan dari proses ini adalah untuk membersihkan teks sehingga dapat lebih mudah diolah dan mewakili konten yang lebih bersih.

13. Proses Case Folding

```
[16] # Proses Case Folding

import re
import pandas as pd
def casefolding(comment):
    comment = comment.lower()
    comment = comment.strip()
    return comment
my_df['content'] = my_df['content'].apply(casefolding)
print('Hasil Case Folding: \n')
print(my_df.head(50))
```

```
Hasil Case Folding:

                                              content  score    Label
0   min gimana sih update yang kali ini bukannya m...      3   Netral
1   game udah bagus tapi ada satu hal yang saya li...      1  Negatif
2   ketemu tim yang sangat berbeda dengan musuh di...      1  Negatif
3   minta tolong dong min jaringannya diperbaikin ...      3   Netral
```

Pada bagian ini, dilakukan proses case folding pada teks ulasan. Case folding mengubah semua huruf dalam teks menjadi huruf kecil (lowercase) untuk menyamakan format dan mempermudah analisis teks tanpa mempertimbangkan perbedaan huruf besar dan kecil.

14. Penyimpanan DataFrame setelah Pemrosesan Pembersihan dan Case Folding

```
[17] my_df.to_excel('cleaning.xlsx', index=False)

     data_fold = pd.read_excel('cleaning.xlsx')
     print(data_fold.shape)
     print(data_fold.head())

     (1000, 3)
                                                  content  score    Label
     0  min gimana sih update yang kali ini bukannya m...      3   Netral
     1  game udah bagus tapi ada satu hal yang saya li...      1  Negatif
     2  ketemu tim yang sangat berbeda dengan musuh di...      1  Negatif
     3  minta tolong dong min jaringannya diperbaikin ...      3   Netral
     4  kepada developer game tolong untuk lebih mempe...      1  Negatif
```

Pada bagian ini, DataFrame my_df disimpan ke dalam berkas Excel setelah melalui proses pembersihan dan case folding. Selanjutnya, data yang telah dibersihkan dan di-case folding di-load kembali untuk memastikan proses penyimpanan berhasil.

15. Import Modul untuk Tokenisasi dan Distribusi Frekuensi Kata

```
import string
import re

from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
```

Pada bagian ini, dilakukan import modul-modul yang diperlukan untuk melakukan tokenisasi dan menghitung distribusi frekuensi kata pada teks ulasan. Modul-modul tersebut mencakup string, re (regular expression), word_tokenize, dan FreqDist dari NLTK.

16. Proses Tokenisasi

```
[19] # Proses Tokenizing

     import nltk
     nltk.download('punkt')

     def word_tokenize_wrapper(comment):
       return word_tokenize(comment)

     my_df['content'] = my_df['content'].apply(word_tokenize_wrapper)

     print('Hasil Tokenizing: \n')
     print(my_df.head(20))

     [nltk_data] Downloading package punkt to /root/nltk_data...
     [nltk_data]   Package punkt is already up-to-date!
     Hasil Tokenizing:

                                                  content  score    Label
     0   [min, gimana, sih, update, yang, kali, ini, bu...      3   Netral
```

Pada bagian ini, dilakukan proses tokenisasi pada teks ulasan menggunakan modul nltk. Fungsi word_tokenize_wrapper didefinisikan untuk menerapkan tokenisasi pada setiap ulasan dalam kolom 'content' DataFrame my_df.

17. Penyimpanan DataFrame setelah Proses Tokenisas

```
[20] my_df.to_excel('tokenizing.xlsx', index=False)


     data_token = pd.read_excel("tokenizing.xlsx")
     print(data_token.shape)
     print(data_token.head())

     (1000, 3)
                                                  content  score    Label
     0  ['min', 'gimana', 'sih', 'update', 'yang', 'ka...      3   Netral
     1  ['game', 'udah', 'bagus', 'tapi', 'ada', 'satu...      1  Negatif
     2  ['ketemu', 'tim', 'yang', 'sangat', 'berbeda',...      1  Negatif
     3  ['minta', 'tolong', 'dong', 'min', 'jaringanny...      3   Netral
     4  ['kepada', 'developer', 'game', 'tolong', 'unt...      1  Negatif
```

Pada bagian ini, DataFrame my_df yang telah melalui proses tokenisasi disimpan ke dalam berkas Excel. Selanjutnya, data yang telah ditokenisasi di-load kembali untuk memastikan proses penyimpanan berhasil.

18. Proses Pemfilteran/Stopword Removal

```
[21] # Proses Filtering/Stopword Removal

     import nltk
     nltk.download('stopwords')
     from nltk.corpus import stopwords

     list_stopwords = stopwords.words('indonesian')

     list_stopwords.extend(['yg', 'dg', 'rt', 'dgn', 'ny', 'anj', 'klo',
                            'kalo', 'amp', 'biar', 'bikin', 'bilang',
                            'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                            'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                            'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                            'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                            '&amp', 'yah', 'jgn', 'ga', 'ok', 'bgt', 'banget', 'jg',
                            'gw','guys','gtu','fyi',
                            'iya','aja','sih','iyaa','tpi','udh','ga','ngga','nggak'
                            ,'yeeha','itu','ituu','tpi','giniii','kaan','pas',
                            'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                            'nyg', 'hehe', 'pen', 'u', 'nyesel', 'habis','download','iyaah',
                            'nanya','yaa','tcodtaf','ccq','google','yo','gada','gue','udah','blm','cakep','beneran','dah',
                            'sender','baru','lagi','maen','tbtb','woy','lagi','lg','lgi','njir','kocak','wkwk','naseh','nan',
                            'tcok','cok','nntn','sengaja','bru',
                             ])

     list_stopwords = set(list_stopwords)

     def stopword_removal(comment):
         return [word for word in comment if word not in list_stopwords]

     my_df['content'] = my_df['content'].apply(stopword_removal)
     print('Hasil Filtering: \n')
     print(my_df.head(30))

     Hasil Filtering:

                                                  content  score    Label
     0   [min, gimana, update, kali, bagus, rusak, logi...      3   Netral
     1   [game, bagus, lihat, player, mobile, legends, ...      1  Negatif
     2   [ketemu, tim, berbeda, musuh, ranked, tim, amp...      1  Negatif
     3   [tolong, min, jaringannya, diperbaikin, jaring...      3   Netral
```

19. Penyimpanan DataFrame setelah Proses Pemfilteran Stopword

```
[22] my_df.to_excel('stopword.xlsx', index=False)

     data_filter = pd.read_excel("stopword.xlsx")
     print(data_filter.shape)
     print(data_filter.head())

     (1000, 3)
                                                content  score    Label
     0  ['min', 'gimana', 'update', 'kali', 'bagus', '...      3   Netral
     1  ['game', 'bagus', 'lihat', 'player', 'mobile',...      1  Negatif
     2  ['ketemu', 'tim', 'berbeda', 'musuh', 'ranked'...      1  Negatif
     3  ['tolong', 'min', 'jaringannya', 'diperbaikin'...      3   Netral
     4  ['developer', 'game', 'tolong', 'memperhatikan...      1  Negatif
```

Pada bagian ini, DataFrame my_df yang telah melalui proses pemfilteran stopword disimpan ke dalam berkas Excel. Selanjutnya, data yang telah di-filter stopword di-load kembali untuk memastikan proses penyimpanan berhasil.

20. Proses Stemming

```
[23] # Proses Stemming

     from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

     def stemming(document):
         return [stemmer.stem(term) for term in document]

     my_df['content'] = my_df['content'].apply(stemming)
     print(my_df['content'].head())

     0    [min, gimana, update, kali, bagus, rusak, logi...
     1    [game, bagus, lihat, player, mobile, legends, ...
     2    [ketemu, tim, beda, musuh, ranked, tim, ampas,...
     3    [tolong, min, jaring, diperbaikin, jaring, rom...
     4    [developer, game, tolong, perhati, imbang, mat...
     Name: content, dtype: object
```

Pada bagian ini, dilakukan proses stemming pada teks ulasan menggunakan stemmer dari Sastrawi. Fungsi stemming didefinisikan untuk menerapkan stemming pada setiap ulasan dalam kolom 'content' DataFrame my_df.

21. Penyimpanan DataFrame setelah Proses Stemming

```
[24] my_df.to_excel('final.xlsx', index=False)

     datafinal = pd.read_excel("final.xlsx")
     print(datafinal.shape)
     print(datafinal.head())

     (1000, 3)
                                                  content  score    Label
     0  ['min', 'gimana', 'update', 'kali', 'bagus', '...      3   Netral
     1  ['game', 'bagus', 'lihat', 'player', 'mobile',...      1  Negatif
     2  ['ketemu', 'tim', 'beda', 'musuh', 'ranked', '...      1  Negatif
     3  ['tolong', 'min', 'jaring', 'diperbaikin', 'ja...      3   Netral
     4  ['developer', 'game', 'tolong', 'perhati', 'im...      1  Negatif
```

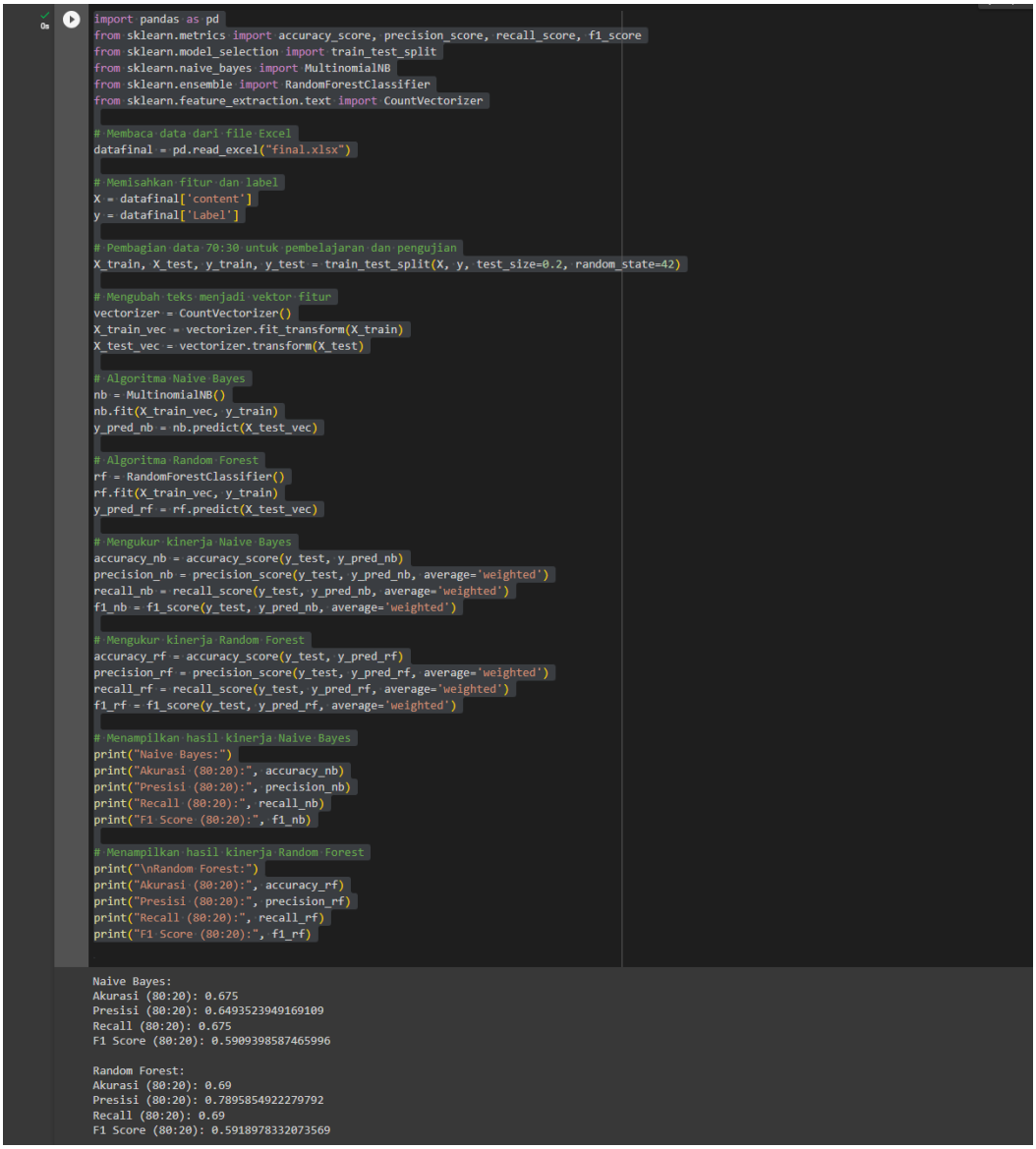
Pada bagian ini, DataFrame my_df yang telah melalui proses stemming disimpan ke dalam berkas Excel. Selanjutnya, data yang telah di-stem di-load kembali untuk memastikan proses penyimpanan berhasil.

22. Proses TF-IDF

```
[25] # Proses TF-IDF

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

     datafinal = pd.read_excel("final.xlsx")
     datafinal = datafinal.astype({'Label': 'category', 'content': 'string', 'score': 'int'})

     tf = TfidfVectorizer()

     text_tf = tf.fit_transform(datafinal['content'].astype('U'))

     result = pd.DataFrame(text_tf.toarray(), columns=tf.get_feature_names_out()).join(datafinal['Label'])

     print(result)

          aamiin  aamon  aba  abadi  abai  abis  abyss  acara  accountsudah  \
     0       0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     1       0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     2       0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     3       0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     4       0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     ..      ...    ...  ...    ...   ...   ...    ...    ...           ...
     995     0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     996     0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     997     0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     998     0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0
     999     0.0    0.0  0.0    0.0   0.0   0.0    0.0    0.0           0.0

          achievement  ...  youll  youre  youtube  youtubeigdll   yt  yth  yutuban  \
     0            0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     1            0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     2            0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     3            0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     4            0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     ..           ...  ...    ...    ...      ...           ...  ...  ...      ...
     995          0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     996          0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     997          0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     998          0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0
     999          0.0  ...    0.0    0.0      0.0           0.0  0.0  0.0      0.0

          zaman  zhask    Label
     0      0.0    0.0   Netral
     1      0.0    0.0  Negatif
     2      0.0    0.0  Negatif
     3      0.0    0.0   Netral
     4      0.0    0.0  Negatif
     ..     ...    ...      ...
     995    0.0    0.0  Negatif
     996    0.0    0.0  Negatif
     997    0.0    0.0  Negatif
     998    0.0    0.0  Positif
     999    0.0    0.0   Netral

     [1000 rows x 3601 columns]
```

Pada bagian ini, dilakukan proses pembuatan matriks TF-IDF menggunakan modul TfidfVectorizer dari scikit-learn. Teks yang telah melalui proses stemming dibaca dari DataFrame datafinal, kemudian diubah menjadi representasi vektor TF-IDF.

23. Analisis Sentimen menggunakan Naive Bayes dan Random Forest



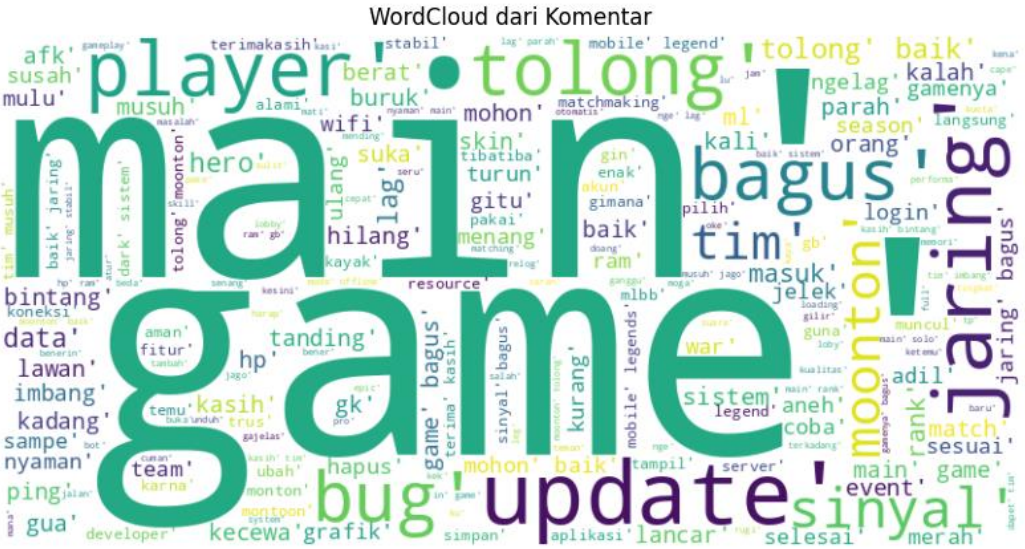
```python
import pandas as pd
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_extraction.text import CountVectorizer

# Membaca data dari file Excel
datafinal = pd.read_excel("final.xlsx")

# Memisahkan fitur dan label
X = datafinal['content']
y = datafinal['Label']

# Pembagian data 70:30 untuk pembelajaran dan pengujian
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Mengubah teks menjadi vektor fitur
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

# Algoritma Naive Bayes
nb = MultinomialNB()
nb.fit(X_train_vec, y_train)
y_pred_nb = nb.predict(X_test_vec)

# Algoritma Random Forest
rf = RandomForestClassifier()
rf.fit(X_train_vec, y_train)
y_pred_rf = rf.predict(X_test_vec)

# Mengukur kinerja Naive Bayes
accuracy_nb = accuracy_score(y_test, y_pred_nb)
precision_nb = precision_score(y_test, y_pred_nb, average='weighted')
recall_nb = recall_score(y_test, y_pred_nb, average='weighted')
f1_nb = f1_score(y_test, y_pred_nb, average='weighted')

# Mengukur kinerja Random Forest
accuracy_rf = accuracy_score(y_test, y_pred_rf)
precision_rf = precision_score(y_test, y_pred_rf, average='weighted')
recall_rf = recall_score(y_test, y_pred_rf, average='weighted')
f1_rf = f1_score(y_test, y_pred_rf, average='weighted')

# Menampilkan hasil kinerja Naive Bayes
print("Naive Bayes:")
print("Akurasi (80:20):", accuracy_nb)
print("Presisi (80:20):", precision_nb)
print("Recall (80:20):", recall_nb)
print("F1 Score (80:20):", f1_nb)

# Menampilkan hasil kinerja Random Forest
print("\nRandom Forest:")
print("Akurasi (80:20):", accuracy_rf)
print("Presisi (80:20):", precision_rf)
print("Recall (80:20):", recall_rf)
print("F1 Score (80:20):", f1_rf)
```

```
Naive Bayes:
Akurasi (80:20): 0.675
Presisi (80:20): 0.6493523949169109
Recall (80:20): 0.675
F1 Score (80:20): 0.5909398587465996

Random Forest:
Akurasi (80:20): 0.69
Presisi (80:20): 0.7895854922279792
Recall (80:20): 0.69
F1 Score (80:20): 0.5918978332073569
```

Pada bagian ini, dilakukan analisis sentimen menggunakan dua model klasifikasi, yaitu Naive Bayes dan Random Forest. Langkah-langkahnya melibatkan pembacaan data dari berkas Excel, pemisahan fitur dan label, pembagian data menjadi set pembelajaran dan pengujian, serta transformasi teks menjadi vektor fitur menggunakan CountVectorizer. Selanjutnya, dilakukan pelatihan model Naive Bayes dan Random Forest, pengujian model, dan pengukuran kinerja menggunakan metrik seperti akurasi, presisi, recall, dan F1-score.

24. Visualisasi WordCloud dari Komentar

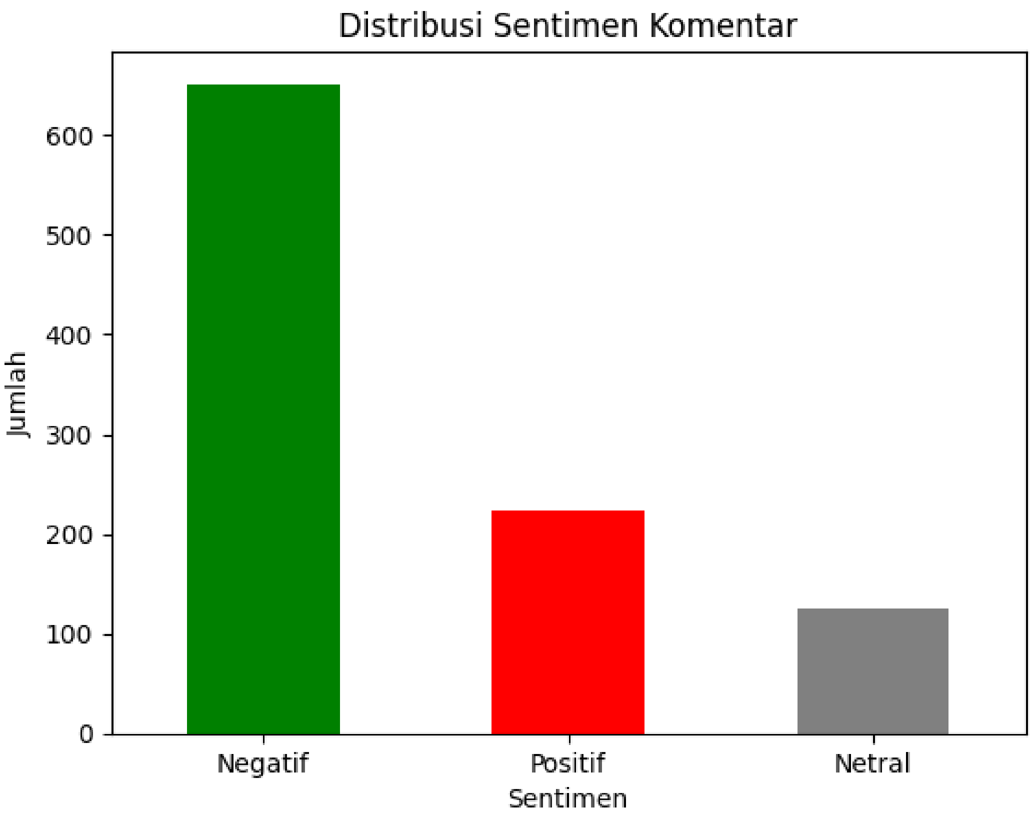
```python
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from sklearn.feature_extraction.text import TfidfVectorizer

# Proses TF-IDF
datafinal = pd.read_excel("final.xlsx")
datafinal = datafinal.astype({'Label': 'category', 'content': 'string', 'score': 'int'})

tf = TfidfVectorizer()
text_tf = tf.fit_transform(datafinal['content'].astype('U'))
result = pd.DataFrame(text_tf.toarray(), columns=tf.get_feature_names_out()).join(datafinal['Label'])

# Visualisasi WordCloud
text = ' '.join(datafinal['content'])

# Menggunakan daftar stop words dari NLTK
stop_words = set(stopwords.words('indonesian'))

# Menghapus stop words dari teks
filtered_text = ' '.join(word for word in text.split() if word.lower() not in stop_words)

# Buat word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(filtered_text)

# Tampilkan word cloud dengan matplotlib
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('WordCloud dari Komentar')
plt.show()
```



Pada bagian ini, dilakukan visualisasi WordCloud dari komentar yang telah melalui proses TF-IDF. Pertama, dilakukan pembacaan data dari berkas Excel dan pembuatan matriks TF-IDF menggunakan modul TfidfVectorizer. Selanjutnya, dibuat WordCloud dari seluruh teks komentar dengan menghapus stop words. Terakhir, WordCloud ditampilkan menggunakan matplotlib.

25. Visualisasi Distribusi Sentimen Komentar

```python
[30] import pandas as pd
import matplotlib.pyplot as plt

# Proses TF-IDF
datafinal = pd.read_excel("final.xlsx")
datafinal = datafinal.astype({'Label': 'category', 'content': 'string', 'score': 'int'})

# Visualisasi Distribusi Sentimen
sentiment_distribution = datafinal['Label'].value_counts()
sentiment_distribution.plot(kind='bar', rot=0, color=['green', 'red', 'grey'])
plt.title('Distribusi Sentimen Komentar')
plt.xlabel('Sentimen')
plt.ylabel('Jumlah')
plt.show()
```



Pada bagian ini, dilakukan visualisasi distribusi sentimen komentar berdasarkan label (positif, negatif, netral). Proses dimulai dengan membaca data dari berkas Excel dan mengonversi tipe data kolom Label. Selanjutnya, dilakukan perhitungan jumlah sentimen untuk setiap kategori dan ditampilkan dalam bentuk diagram batang.

D. KESIMPULAN

- Data komentar pengguna dari Google Play Store Mobile Legends telah berhasil diambil menggunakan google-play-scraper.Tidak ada data yang hilang dalam dataset
- Data berisi kolom-kolom seperti reviewId, userName, content, score, dan lainnya.
- Dilakukan pembersihan data seperti penghapusan karakter khusus, angka, dan tanda baca.
- Proses tokenisasi untuk membagi teks menjadi token atau kata-kata.
- Implementasi filtering untuk menghapus stopwords (kata-kata umum yang tidak memberikan makna signifikan).
- Proses stemming untuk mengubah kata-kata menjadi bentuk dasarnya.
- Sentimen pada setiap komentar diberikan label "Positif", "Negatif", atau "Netral" berdasarkan skor yang diberikan.
- Berdasarkan analisis sentimen pada 1.000 komentar, digunakan metode Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen.
- Berdasarkan analisis sentimen pada 1.000 komentar, digunakan metode Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen.
- Berdasarkan analisis sentimen pada 1.000 komentar, digunakan metode Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen.
- Berdasarkan analisis sentimen pada 1.000 komentar, digunakan metode Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen.
- Berdasarkan analisis sentimen pada 1.000 komentar, digunakan metode Naive Bayes dan Random Forest untuk mengklasifikasikan sentimen.
- Random Forest memberikan akurasi sekitar 69%, dengan presisi sekitar 78.9%.
- Distribusi sentimen komentar diisi oleh 65% komentar negatif, 23% komentar positif dan 12% komentar netral.