ANALISIS SENTIMEN PENGGUNA APLIKASI LIVIN' BY MANDIRI BERDASARKAN ULASAN PADA *GOOGLE PLAYSTORE* MENGGUNAKAN METODE *RANDOM FOREST*

SKRIPSI

Diajukan untuk Memenuhi Tugas Akhir dan Melengkapi Syarat Guna Memperoleh Gelar Sarjana Strata Satu (S-1) dalam Teknologi Informasi

Oleh:

INDRI AWALIA SEFIANI
NIM : 208096032

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI WALISONGO SEMARANG

TAHUN 2024

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Indri Awalia Sefiani
NIM : 2008096032
Jurusan : Teknologi Informasi

Menyatakan bahwa skripsi yang berjudul:

Analisis Sentimen Pengguna Aplikasi Livin' By Mandiri Berdasarkan Ulasan Pada Google Playstore Menggunakan Metode *Random Forest*

Secara keseluruhan adalah penelitian/karya saya sendiri, kecuali bagian tertentu yang dirujuk sumbernya.

Semarang, 2 Mei 2024
Pembuat pernyataan

Indri Awalia Sefiani
NIM. 2008096032

PENGESAHAN

KEMENTRIAN AGAMA
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof. Dr. Hamka Ngaliyan Semarang
Telp.024-7601295 Fax.7615387

PENGESAHAN

Naskah skripsi berikut ini:

Judul : Analisis Sentimen Pengguna Aplikasi Livin' By Mandiri Berdasarkan Ulasan Pada Google Playstore Menggunakan Metode *Random Forest*
Penulis : **Indri Awalia Sefiani**
NIM : 2008096032
Jurusan : Teknologi Informasi

Telah diujikan dalam sidang tugas akhir oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam Teknologi Informasi.

Semarang, 29 April 2024

DEWAN PENGUJI

Penguji I

Nur Cahyo Hendrowibowo, ST.,M.Kom.
NIP. 197312222006041001

Penguji II

Dr. Khotibul Umam, M.Kom.
NIP.197908272011011007

Penguji III

Hery Mustofa, M.Kom.
NIP. 198703172019031007

Penguji IV

Adzhal Arwani Mahfudh, M.Kom.
NIP. 199107032019031006

Pembimbing I

Dr. Khotibul Umam, M.Kom.
NIP.197908272011011007

Pembimbing II

Mokhamad Iklil Mustofa, M.Kom.
NIP. 19880807201903 1010

NOTA DINAS

Semarang, 1 April 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi
UIN Walisongo Semarang

Assalamu'alaikum. Wr. Wb.

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan :

Judul : ANALISIS SENTIMEN PENGGUNA APLIKASI LIVIN' BY MANDIRI BERDASARKAN ULASAN PADA GOOGLE PLAYSTORE MENGGUNAKAN METODE RANDOM FOREST
Nama : **Indri Awalia Sefiani**
NIM : 2008096032
Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo Semarang untuk diujikan dalam Sidang Munaqosyah.

Wassalamu'alaikum. Wr. Wb.

Pembimbing I,

Dr. Khotibul Umam, ST., M.Kom
NIP. 19790827 201101 1007

NOTA DINAS

Semarang, 1 April 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi
UIN Walisongo Semarang

Assalamu'alaikum. Wr. Wb.

Dengan ini diberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan :

Judul : ANALISIS SENTIMEN PENGGUNA APLIKASI LIVIN' BY MANDIRI BERDASARKAN ULASAN PADA GOOGLE PLAYSTORE MENGGUNAKAN METODE RANDOM FOREST
Nama : **Indri Awalia Sefiani**
NIM : 2008096032
Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Fakultas Sains dan Teknologi UIN Walisongo Semarang untuk diujikan dalam Sidang Munaqosyah.

Wassalamu'alaikum. Wr. Wb.

Pembimbing II,

Mokhamad Iklil Mustofa, M.Kom
NIP. 1980807 201903 1 010

LEMBAR PERSEMBAHAN

Dengan mengucapkan Puji dan Syukur Alhamdulillah penulis ucapkan kepada Allah SWT, penulis dapat menyelesaikan karya tulis sebagai laporan tugas akhir ini dapat dengan baik. Karya tulis ini penulis persembahkan untuk :

1. Bapak Mukhtarom dan Ibu Maryati selaku orang tua dari penulis.
2. Muhammad Syafik Maulana selaku adik penulis.
3. Seluruh dosen Jurusan Teknologi Informasi.
4. Sahabat dan teman – teman seperjuangan khususnya Jurusan Teknologi Informasi Angkatan 2020.
5. Almamater Universitas Islam Negeri Walisongo Semarang.

MOTTO

"Karena sesungguhnya sesudah kesulitan itu ada kemudahan,
sesungguhnya sesudah kesulitan itu ada kemudahan."
Q.S Al Insyirah: 5-6

ABSTRAK

Aplikasi Livin' by Mandiri merupakan aplikasi *m-banking* yang berisi berbagai fitur yang terdapat pada *Google PlayStore* memiliki tujuan untuk mempermudah para nasabahnya. Misalnya fitur yang tersedia seperti halnya fitur yang disediakan untuk memberikan ulasan oleh pengguna. Adanya ulasan yang diberikan oleh pengguna pada aplikasi *mobile banking* yaitu Livin' by Mandiri terdapat dalam jumlah yang banyak dan dalam kedaan yang tidak terstruktur dan ulasan tersebut sulit untuk dipisahkan menjadi ulasan yang mengandung opini negatif atau opini positif. Pada penelitian ini menerapkan analisis sentimen ulasan Aplikasi Livin' by Mandiri yang diambil melalui *Google PlayStore* untuk ditindaklanjuti pada proses pengklasifikasian ulasan tersebut termasuk kedalam ulasan yang negatif, positif ataupun netral.
Pada penelitian ini data yang diambil menggunakan teknik *scrapping* pada ulasan Aplikasi Livin' by Mandiri yang dilanjutkan dengan proses *teks preprocessing* dan tahap melabeli ulasan. Labeling ulasan yang digunakan pada penelitian ini dengan menggunakan teknik *lexicon based* dengan kamus. Kemudian dilanjutkan dengan pembobotan kata TFIDF Selanjutnya didalam penelitian ini proses klasifikasi ulasan dengan menggunakan metode *random forest* yang menghasilkan tiga kelas yaitu kelas negatif, positif ataupun netral. Pengujian yang telah dilakukan pada penelitian ini berdasarkan kinerja model *random forest* maka menghasilkan performa yang baik dengan melakukan perbandingan data training dan data testing sebesar 80:20 menghasilkan nilai akurasi yang sebesar 80%, *precission* sebesar 80% , *recall* sebesar 80% dan *fi-score* juga sebesar 80%.

Kata kunci : Analisis Sentimen, Aplikasi Livin' by mandiri, *Random Forest.*

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT yang memberikan rahmat dan hidayah-Nya kepada penulis serta telah memberikan kemudahan dan kelancaran sehingga penulis dapat menyelesaikan Tugas Akhir sebagai syarat kelulusan dalam menempuh Pendidikan di Progam Studi Teknologi Informasi Fakultas Sains dan Teknologi UIN Walisongo Semarang telah dapat penulis selesaikan dengan judul "**Analisis Sentimen Pengguna Aplikasi Livin' by Mandiri Berdasarkan Ulasan Pada Google Playstore Menggunakan Metode Random Forest**".

Pada kesempatan ini penulis ingin menyampaikan sebuah ucapan terima kasih yang tak terhingga kepada beberapa pihak yang telah banyak memberikan bantuan, arahan dan juga bimbingan sehingga terselesaikannya Tugas Akhir ini. Penyusunan laporan tugas akhir ini tidak terlepas dari bantuan beberapa pihak, oleh karena itu penulis hendak mengucapkan terima kasih kepada:

1. Kedua orang tua penulis yaitu Bapak Mukhtarom dan Ibu Maryati yang selalu memberikan dukungan dan tak lupa selalu mendoakan penulis sehingga terlesesaikannya masa pendidikan S1 dan tugas akhir ini.

2. Ketua Jurusan Prodi Teknologi Informasi UIN Walisongo Semarang, Bapak Nur Cahyo Hendro Wibowo, S.T., M.Kom.
3. Dosen Pembimbing I sekaligus Dosen Wali Bapak Dr. Khotibul Umam, ST., M.Kom yang selalu memberikan arahan dan bimbingan kepada penulis.
4. Dosen Pembimbing II Bapak Iklil Mustofa, M.Kom yang selalu memberikan arahan dan bimbingan juga kepada penulis.
5. Seluruh dosen Teknologi Informasi, staf, karyawan dan dosen di lingkungan UIN Walisongo Semarang yang telah memberikan ilmu pengetahuan yang tidak ternilai selama menempuh pendidikan.
6. Semua pihak yang tidak dapat disebutkan satu per satu yang terlibat dalam penyusunan tugas akhir ini.

Akhir kata, Semoga segala kebaikan dan ketulusan yang diberikan mendapatkan balasan yang setimpal oleh Allah SWT. Semoga skripsi ini bisa bermanfaat bagi para pembaca dan bisa dijadikan bahan rujukan untuk melakukan penelitian selanjutnya.

Semarang, 5 April 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi informasi di era revolusi sudah berkembang dengan pesat, kini keberadaan teknologi berperan penting dalam segala aspek kehidupan manusia sehingga perkembangan teknologi tersebut akan mempermudah manusia dalam melakukan aktivitas dalam kehidupan sehari – hari.

Teknologi yang semakin canggih kini mendorong perkembangan strategi bisnis terutama pada lembaga keuangan yang keberadaanya kini mengubah segala aspek kehidupan manusia menjadi serba digital. Dengan adanya kemajuan teknologi kini berbagi sektor industri telah memanfaatkan hal tersebut untuk menunjang proses pelayanan kepada pelanggan salah satunya sektor perbankan dengan menginovasikan keuangan yakni mengubah proses dari perbankan tradisional menjadi sebuah perbankan yang berbasis pada kemajuan teknologi secara digital yang bertujuan untuk meningkatkan kualitas daya saing dan mencukup kebutuhan dari pelanggan selain itu juga segala aktivitas yang ada akan lebih praktis dan efisien.

Mobile banking atau sering disebut juga sebagai *m-banking* merupakan sebuah aplikasi digital yang diciptakan oleh sektor industri dibidang perbankan dengan memanfaatkan adanya kemajuan teknologi yang bertujuan untuk memudahkan nasabah untuk melakukan berbagai proses transaksi. Berbagai layanan yang terdapat dalam aplikasi *m-banking* kini menjadi kebutuhan bagi nasabah karena keberadaannya dapat membantu dimana dan kapanpun dibutuhkan seperti layanan transfer dana, pengecekan saldo, dan jug adapat melakukan berbagai proses pembayaran tagihan hanya melalui ponsel (Alun Sujjadaa *et al*., 2023).

Aplikasi Mobile Banking Terpopuler di Indonesia (2022)

databoks

No	Nama	Nilai / Skor Top Brand Index (Dalam Persen)
1	m-BCA	47,4
2	BRI Mobile	19,4
3	m-Banking Mandiri	12,9
4	BNI Mobile	11,2
5	CIMB Niaga Mobile	3,8

Gambar 1. 1 Aplikasi M-banking Terpopuler di Indonesia

Berdasarkan data yang diperoleh dari databooks *mobile banking* terpopuler di Indonesia pada tahun 2022 dengan kedudukan pertama yaitu *m-BCA* yang memiliki skor top brand index 47,4%. Kemudian diikuti oleh *BRI Mobile* dengan skor sebesar 19,4%. Di kedudukan ke ketiga

terdapat m-Banking Mandiri dengan skor sebesar 12,9%. Selanjutnya *BNI Mobile* dengan skor 11,2% berada di urutan ke empat. *CIMB Niaga Mobile* berada di urutan terakhir dengan peroleh skor sebesar 3,8% (Kata data 2022, diakses 2 Oktober 2023)

Dalam penggunaan aplikasi *mobile* penggguna dapat memberikan rating disertai ulasan. Melalui fitur rating ini maka pengguna dapat memberikan nilai berupa angka sedangkan pada bagian komentar atau ulsan kini pengguna dapat memberikan berupa kritik dan saran terhadap layanan ataupun fitur yang ada didalam aplikasi tersebut (Khoirul Insan *et al*., 2023). Adanya ulasan yang diberikan oleh pengguna maka dapat dijadikan bahan pertimbangan yang dilakukan oleh pengguna baru sebelum memutuskan untuk menggunakan aplikasi. Selain itu ulasan pengguna. Selain itu ulasan kini dapat dimanfaatkan oleh pengembang aplikasi untuk memperbaiki menjadi aplikasi yang lebih baik lagi (Khoirul Insan *et al*., 2023).

Seperti Firman Allah Subhanahu Wa Ta'ala dalam QS. Al-Hujurat 49: Ayat 6 yang berbunyi:

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْٓا اِنْ جَاۤءَكُمْ فَاسِقٌۢ بِنَبَاٍ فَتَبَيَّنُوْٓا اَنْ تُصِيْبُوْا قَوْمًاۢ بِجَهَالَةٍ فَتُصْبِحُوْا عَلٰى مَا فَعَلْتُمْ نٰدِمِيْنَ

> Artinya: "*Wahai orang-orang yang beriman! Jika seseorang yang fasik datang kepadamu membawa suatu berita, maka telitilah kebenarannya, agar kamu tidak mencelakakan suatu kaum karena kebodohan (kecerobohan), yang akhirnya kamu menyesali perbuatanmu itu*." (QS. Al-Hujurat 49: Ayat 6)

Bahwa pentingnya memeriksa Kembali kebenaran suatu berita atau kritikan sebelum membuat peniaian atau Tindakan. Sangatlah diperlukan memahami konteks dan juga kebenaran informasi sebelum meyimpulkan keputusan berdasarkan sebuah opini publik. Prinsip yang terkandung dalam Q.S Al – Hujurat ayat 6 diperlukan verifikasi dan juga validasi terhadap data yang digunakan pada proses analisis sentiment.

Adanya ulasan yang diberikan oleh pengguna aplikasi *mobile banking* yaitu Livin' by Mandiri pada *Google PlayStore* terdapat dalam jumlah yang banyak dan tidak terstruktur dan ulasan tersebut sulit untuk dipisahkan menjadi ulasan yang mengandung opini negatif atau opini positif (Diki Hendriyanto et al., 2022). Dengan demikian untuk memecahkan masalah tersebut maka diperlukannya sebuah sistem yang berfungsi untuk memisahkan opini baik negatif, positif mapun netral. Analisis sentimen merupakan cabang ilmu dari natural language program, *text mining* dan kecerdasan buatan. Proses mendeteksi opini yang

berbetuk dalam sebuah teks melalui teknik pemrosesan data. Tujuan dari analisis sentimen yaitu untuk mendeteksi teks yang berisi komentar dari pengguna aplikasi termasuk dalam komentaryang negatif atau postif atau bisa saja termasuk kedalam yang netral (Larasati *et al*., 2022).

Terdapat beberapa penerapan algoritma dalam analisis sentimen seperti *random forest* dimana metode ini merupakan salah satu metode dalam melakukan Analisis Sentimen dan masuk ke jenis metode *Decsion Tree.* Algoritma *Random Forest* memiliki peningkatan akurasi sebesar 7,16% yang lebih akurat dibandingkan dengan *Support Vector Machine* dan *Naïve Bayes* (Fitri, 2020). Penelitian sebelumnya Akhmad Miftahusalam dkk menggunakan Algoritma Klasifikasi *Random Forest* dan *Naïve bayes* untuk Analisis Sentimen Review Aplikasi BCA Mobile mendapatkan hasil bahwa metode *Random Forest* menghasilkan prediksi yang lebih baik dari pada metode *Naïve Bayes* dengan dengan nilai akurasi sebesar 93,93%, nilai *precision* 93,02%, nilai *recall* 89,89%, dan nilai *F1-score* 91,43% (Zaki Hariansyah, 2022). Selain itu juga penelitian dilakukan oleh Hasil pengujian dengan algoritma klasifikasi Malina Putri terkait deteksi spammer politik menggunakan *Random Forest* memberikan hasil tingkat

akurasi *F1 score* sebesar 94%, *precision* sebesar 96,1%, dan *recall* sebesar 93,7% (Afemi, 2022). Penelitian sebelumnya Nugraha dkk menggunakan Algoritma Klasifikasi *Random Forest* Untuk Penentuan Kelayakan Pemberian Kredit Di Koperasi Mitra Sejahtera Hasil pengujian dengan algoritma klasifikasi *Random Forest* mampu menganalisis kredit yang bermasalah dan yang debitur yang tidak bermasalah dengan nilai akurasi sebesar 87,88% (Zailani & Hanun, 2020).

Dari beberapa hasil dari penelitian terdahulu metode *Random Forest* menghasilkan akurasi yang baik dan cukup tinggi. Oleh karena itu peneliti tertarik untuk mencoba menggunakan metode *Random Forest* dalam penelitian ini yang memiliki tujuan akhir mendapatkan hasil akurasi dari klasifikasi dengan menggunakan metode *Random Forest*. Berdasarkan penjelasan diatas, maka penulis akan melakukan penelitian yang berjudul "Analisis Sentimen Pengguna Aplikasi Livin' by Mandiri Berdasarkan Ulasan Pada *Google PlayStore* Menggunakan Metode *Random Forest*".

B. Identifikasi Masalah

Berdasarkan latar belakang masalah yang telah diuraikan diatas maka teridentifikasi masalah berikut:

1. Adanya analisis sentimen ini untuk mengidentifikasi ulasan pengguna Aplikasi Livin' by Mandiri yang tidak terstruktur untuk dipisahkan kedalam opini positif, negatif maupun netral.
2. Penerapan metode *Random Forest* untuk menganalisis sentimen pengguna Aplikasi Livin' by Mandiri berdasarkan ulasan pada *Google PlayStore* ini tepat karena metode *Random Forest* ini eror yang dihasilkan relatif rendah, serta menghasilkan performa yang baik dan juga tepat digunakan dalam jumlah data yang besar.

C. Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan diatas, maka peneliti mengambil rumusan masalah sebagai berikut:

1. Bagaimana implementasi yang diberikan oleh metode *Random Forest* dalam membantu analisis sentimen Aplikasi Livin' by Mandiri berdasarkan ulasan pada *Google Playstore*?
2. Bagaimana performa yang dihasilkan oleh Metode *Random Forest* terhadap analisis sentimen Aplikasi Livin' by Mandiri berdasarkan ulasan pada *Google Playstore*?

D. Tujuan Penelitian

Berdasarkan rumusan masalah yang telah di berikan, maka penelitian memiliki tujuan sebagai berikut:

1. Mengimplementasikan metode *Random Forest* untuk menganalisis sentimen Aplikasi Livin' by Mandiri berdasarkan ulasan pada *google playstore* termasuk kedalam sentimen positif, negatif maupun netral.
2. Mengetahui performa yang dihasilkan oleh metode *Random Forest* untuk menganalisis sentimen Aplikasi Livin' by Mandiri berdasarkan ulasan pada *google playstore* termasuk kedalam sentimen positif, negatif maupun netral.

E. Batasan Masalah

Didalam penelitian ini menggunakan batasan masalah yang digunakan dalam proses penelitian agar dapat dilakukan secara jelas. Batasan masalah pada penelitian ini sebagai berikut:

1. Dalam penelitian ini menggunakan data dari ulasan pengguna aplikasi livin' by mandiri yang berasal dari *Google Playstore*.
2. Ulasan pengguna aplikasi livin' by mandiri akan melalui proses pengklasifikasian yang memiliki

tiga sentimen yaitu positif, negatif dan juga netral.

3. Analisis sentimen yang digunakan dalam proses penelitian ini menggunakan klasfikasi dengan metode *random forest.*
4. Proses penelitian ini menggunakan bahasa pemrograman *python* dan software *jupyter.*
5. Data yang digunakan dalam penelitian ini sejumlah 10.000 ulasan pengguna Aplikasi Livin' by Mandiri.
6. Data ulasan ulasan pengguna Aplikasi Livin' by Mandiri diambil dari *Google PlayStore* tanggal 12 Oktober 2023 – 20 Oktober 2023

F. Manfaat Penelitian

Penelitian ini memiliki dua manfaat yaitu manfaat secara praktis dan manfaat secara teoritis:

1. Manfaat Praktis

a. Membantu pihak perusahaan dalam mengetahui presepsi pengguna aplikasi dalam bentuk ulasan positif, negatif dan netral terhadap Aplikasi Livin' by Mandiri.
b. Hasil yang didapatkan dari analisis sentimen bisa dijadikan bahan acuan untuk menjaga kualitas yang digunakan untuk memperbaiki

kekurangan yang ada dan dijadikan bahan evaluasi ke arah yang lebih baik lagi.

2. **Manfaat Teoritis**
    a. Membantu dalam proses pengklasifikasian ulasan pada Aplikasi Livin' by Mandiri kedalam ulasan yang negatif, positif atau netral.
    b. Mengetahui hasil performa yang dihasilkan oleh metode *Random Forest* untuk mengklasifikasi ulasan pada Aplikasi Livin' by Mandiri.
    c. Penelitian ini bisa dijadikan refrensi untuk melakukan penelitian berikutnya terkait analisis sentimen.

BAB II

LANDASAN PUSTAKA

A. Kajian Teori

1. Text Mining

Text Mining merupakan sebuah proses pengambilan data dari sebuah dokumen yang berupa teks yang memiliki tujuan untuk menemukan sebuah kata yang mewakili isi dari dokumen tersebut sehingga analisis hubungan dari suatu dokumen bisa dilakukan (Nurjannah & Fitri Astuti, 2013). *Text mining* ini dapat membantu proses pengelompokkan data dengan menggunakan waktu yang lebih singkat. Apabila terdapat kumpulan teks biasanya terdapat sebuah *noise* hal bisa meenerapkan *text mining* dengan melakukan *text processing. Text mining* dilakukan untuk melakukan pengkategorian teks dan juga pengelompokan teks (Afdhal *et al*., 2022).

2. Konsep dan Implementasi Analisis Sentimen

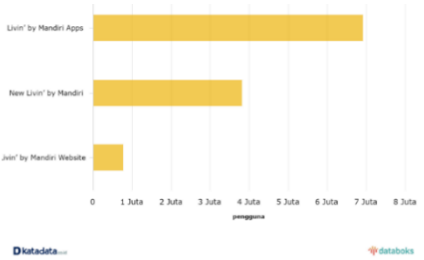
Analisis sentimen merupakan sebuah aktivitas pengelompokan teks yang berisi informasi yang bersifat opini baik negatif dan opini positif. Analisis sentimen ini memiliki tujuan untuk mngubah opini publik terhadap suatu permasalahan yang terdapat dalam sebuah data teks yang tidak beraturan (Alun Sujjadaa *et al*., 2023).

Dalam kehidupan sehari sehari analisis sentimen seringkali digunakan oleh masyarakat sekitar karena adanya pendapat masyarakat merupakan hal yang penting dalam pengambilan keputusan. Analisis sentimen ini biasanya berupa komentar, *review*, dan umpan balik yang didalamnya berupa informasi yang diperlukan (Afdhal *et al*., 2022).

3. Gambaran Umum Livin' by Mandiri

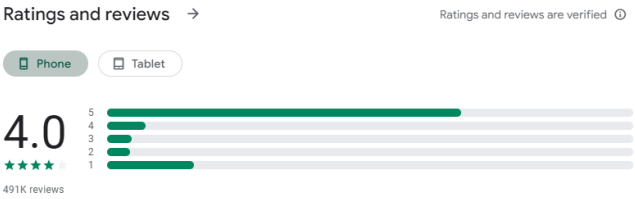
Livin' by Mandiri merupakan sebuah aplikasi *mobile banking* dari Bank mandiri yang menyediakan layanan digital seperti proses transaksi, pengecekan saldo, *top up* dan juga masih banyak yang lainnya. Aplikasi Livin' by Mandiri ini bisa menciptakan kenyamanan untuk melakukan proses transaksi sehingga keberadaannya dapat menarik perhatian dari nasabah baru untuk mencoba proses yang dengan menggunakan aplikasi tersebut. Aplikasi Livin' by Mandiri merupakan aplikasi yang dluncurkan pada tanggal 21 Maret 2017. Layanan ini di buat oleh pihak Bank Mandiri dengan berbagai fitur yang memiliki tujuan untuk memberikan kenyamanan dan kemudahan terhadap nasabah sehingga memudahkan dalam proses transaksi keuangan (Santoso & Rachmawati, 2021).

:



Gambar 2. 1 Jumlah Pengguna Aplikasi M-banking Mandiri

Berdasarkan katadata kini Livin' by Mandiri telah mencapai angka 6,92 jt sebagai pengguna aktif dari Livin by Mandiri (*Apss*) sampai dengan tahun 2021 yang tercatat dalam frekuensi transaksinya sebesar 1,04 miliar yang memiliki nominal sebesar Rp1.455 triliun. Selanjutnya, dalam jumlah 3,81 juta juga dimiliki oleh New Livin' by Mandiri sebagai pengguna aktif dan terdapat 122,57 juta kali yang memiliki nominal Rp123,91 triliun sebagai frekuensi transaksinya. Sementara terdapat 76,12 ribu tercatat sebagai pengguna aktif yang menggunakan Livin' by Mandiri *website*. Memiliki frekuensi transaksi senilai 58,60 juta kali dengan nominal sebesar Rp61,57 triliun. (Kata data 2022, diakses 10 Oktober 2023)



Gambar 2. 2 Rating Aplikasi di Google PlayStore

Aplikasi Livin' by Mandiri ini dapat di unduh melalui *Google PlayStore*. Berdasarkan jumlah unduhan pada *Google PlayStore* hingga bulan September 2023 kini mencapai kurang lebih 10 juta unduhan dan memiliki rating sebesar 4.0 dengan tercatat 491 ribu ulasan komentar pengguna aplikasi Livin' by Mandiri.

4. *Google PlayStore* Sebagai Sumber Data

Google Playstore merupakan aplikasi market yang mengalokasikan perangkat lunak, dimana di dalam *google playstore* pengguna akan mendapatkan aplikasi yang diinginkan dan langsung dari pengembang. Didalam *google playstore* pengguna akan dapat memberikan rating dan ulasan terhadap aplikasi yang digunakan melalui fitur komentar. Biasanya pengguna lainnya akan memperoleh informasi terkait aplikasi yang digunakan melalui komentar yang ada sehingga akan dijadikan bahan pertimbangan untuk

menggunakan aplikasi tersebut. *Google Playstore* menyediakan skala peringkat yang dapat diberikan terhdap *review* aplikasi dari 1 sampai dengan 5. Namun, skala peringkat tersebut tidak dapat dijadikan pacuan untuk menilai apilikasi sehingga terdapat juga ulasan yang diberikan dengan penggambaran aplikasi berupa kalimat yang memiliki sifat positif dan negatif. Dari adanya ulasan tersebut maka akan mempengaruhi pengguna yang akan mengunduh dan menggunakan aplikasi (Alun Sujjadaa *et al*., 2023).

5. Metode *Random Forest* dalam Analisis Sentimen

Machine Learning merupakan bagian dari kecerdasan buatan, dimana sebuah aplikasi komputer dan juga algoritma matematika yang dilakukan dengan cara menerapkan pembelajaran yang berasal dari sebuah data dan akan menghasilkan sebuah prediksi yang akan digunakan di kemudian hari. Proses pembelajaran yang digunakan dalam *machine learning* ini merupakan sebuah usaha dalam memperoleh keerdasan yang terdapat dua tahap yaitu latihan (*training*) dan pengujian (*testing*). Data *training* merupakan data yang digunakan untuk melatih algoritma yang terdapat pada *machine learning* sedangkan data *testing* sendiri merupakan data yang

digunakan untuk mengetahui sejauh mana performa ari sebuah algoritma yang terdapat pada *machine learning* yang sebelumnya telah dilatih (Retnoningsih & Pramudita, 2020).

Random Forest merupakan sebuah algortima pengembangan dari model *Algoritme Decision* dengan cara melatih setiap pohon fikiran yang menggunakan sampel individu (Aldean *et al*., 2022). Kemudian Algoritma *random forest* merupakan dari salah satu metode dari *machine learning* yang digunakan untuk proses klasifikasi sebuah data dalam jumlah yang banyak. J.Ross Quinlan telah mendesain algoritma *random forest* yang merupakan keturunan dari pendekatan ID3 yang digunakan untuk membangun pohon keputusan. Dalam menyelesaikan masalah klasifikasi pad a *machine learning* dan data mining metode *random forest* kini tepat untuk digunakan. Pamuji dan Ramdhan mengemukakan pendapatnya bahwa metode *random forest* ini memiliki kelebihan yaitu eror yang dihasilkan relative rendah, menghasilkan performa yang baik, dan dalam penggunaannya cocok digunakan dalam jumlah data yang besar.

B. Kajian Penelitian yang Relevan

Berikut detail dari rujukan penelitian ditunjukan pada tabel 2.1:

No	Judul	Peniliti, Publikasi, Tahun	Hasil
1	Analisis Sentimen Pada Ulasan Aplikasi Mobile Banking Menggunakan Metode *Naïve Bayes* denga Kamus Inset	Alya Nadira, Nanang Yudi Setiawan, Welly Purnomo, Informatic and Computational Inelligent Journal, 2023	Proses klasifikasi pada sistem menggunakan metode Naïve Bayes. Data latih yang digunakan adalah data ulasan yang telah diberi label secara otomatis menggunakan kamus InSet yang telah melalui penyesuaian kata dan bobot. Algoritma klasifikasi tersebut diuji menggunakan *confusion matrix* dan menghasilkan nilai akurasi 93,1%
2	Analisis Sentimen Pada Ulasan Pengguna Aplikasi Mandiri Online Menggunakan Metode Modified *Term Frequency Scheme* Dan *Naïve Bayes.*	Eka Putri Nirwandai, Indriati, Randy Cahya Wihandika, Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2021	Hasil penelitian ini sebesar *accuracy* 83%, *recall* 86%, *precision* 76%, f-measure 77,70%.

3	Perbandingan Metode Random Forest dan Naive Bayes pada Analisis Sentimen Review Aplikasi BCA Mobile*Random Forest*	Akhmad Miftahusalam, Hasih Pratiwi, Isnandar Slamet, 2023	Hasilnya metode *Random Forest* menghasilkan prediksi yang lebih baik daripada metode *Naïve Bayes* dengan dengan nilai akurasi sebesar 93,93%, nilai presisi 93,02%, nilai recall 89,89%, dan nilai F1-score 91,43%.
4	Analisa Prediksi Kelayakan Pemberian Kredit Pinjaman dengan Metode *Random Forest*	Budi Prasojo, Emy Haryatmi, Jurnal Nasional Teknologi dan Sistem Informasi, 2020	Hasil yang diberikan oleh algoritma *random forest* memiliki tingkat akurasi sebesar 0,83 atau sebesar 83%. Dengan demikian pengujian tersebut termasuk pada kategori klasifikasi model yang sangat bagus.

Tabel 2. 1 Penelitian yang relevan

Berdasarkan tabel diatas maka peneliti akan melakukan penelitian terkait dengan analisis sentimen dengan menggunakan metode *random forest.* Penelitian ini menggunakan ulasan dari aplikasi livin' by mandiri yang diambil melalui *google playstore.* Proses pelabelan yang digunakan pada penelitian ini menggunakan teknik *lexicon based* yaitu dengan menggunakan pada sebuah kamus kata positif dan negatif dan netral yang berbahasa Indonesia sebelum melakukan klasifikasi dengan metode *random forest.*

BAB III

METODOLOGI PENELITIAN

A. Jenis Penelitian

Jenis penelitian yaitu terdapat dua metode, metode pengumpulan data dan metode metode penelitian yang digunakan. Dalam penelitian ini metode pngumpulan data yang digunakan yaitu tiga cara yakni Studi Pustaka & *Litertaur Review*, observasi dan pengambilan data primer yang dilakukan dengan cara pengambilan ulasan pengguna aplikasi melalui *website Google PlayStore*. Metode penelitian yang digunakan metode KDD (*Knowledge Discovery in Database*) dimana terdapat beberapa tahapan seperti *data selection*, *preprocessing*, *transformation*, data mining, dan i*nterpretation*/*evaluation*.

B. Metode Pengumpulan Data

Data dan informasi yang digunakan oleh peneliti pada penelitian ini diperoleh melalui tahpan pengumpulan data sebagai berikut:

1. Studi Pustaka & *Literatur Review*

Didalam tahapan studi pustaka ini peneliti mencari dan mengumpulkan informasi dan juga data yang berkaitan dengan judul penelitian. Selain itu

proses studi pustaka ini juga digunakan untuk mencari informasi terkait penelitian yang dilakukan sebelumnya. Kemudian informasi dan juga data yang terlah dikumpulkan maka dijadikan sebagai data pembanding atau landasan untuk menyelesaikan penelitian yang sedang dilakukan.

2. Observasi

Tahap observasi ini dilakukan oleh peneliti, yang bertujuan untuk mengamati secara langsung pada ulasan yang terdapat pada *Google Playstore* terkait Aplikasi Livin' by Mandiri. Kemudian peneliti juga menjadi pengguna Aplikasi tersebut. Dari observasi yang dilakukan oleh peneliti maka diperoleh hasil sebagai berikut:

a. Terdapat informasi terkait dengan kendala – kendala yang dialami oleh pengguna aplikasi Livin' by Mandiri.
b. Adanya informasi terkait kelebihan dan juga kekurangan dari aplikasi Livin' by Mandiri.

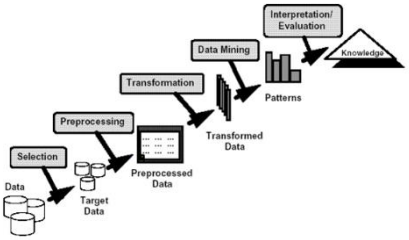
3. Penambangan Data

Tahap penambangan data yang dilakukan pada penelitian ini yaitu dengan cara mengambil dataset yang diperoleh dari ulasan penggunaan aplikasi melalui review yang terdapat pada *Google Playstore*

dengan menggunakan library *Python Scrapper* dengan menggunakan teknik *scrapping* data. Data ulasan yang didapat kemudian disimpan dalam format *Xlsx* yang diolah dengan *Microsoft Excel* dan diubah dalam bentuk format *Comma Separated Value* (CSV).

C. Metode Penelitian

Penelitian ini menggunakan metode KDD (*Knowledge Discovery in Database*). Metode KDD merupakan metode yang berfokus pada analisis pola yang diolah dari data. Penggunaan metode KDD ini tepat karena adanya kesesuian dengan tahapan yang akan dilakukan pada proses penelitian ini (Majid & Sulastri, 2023).



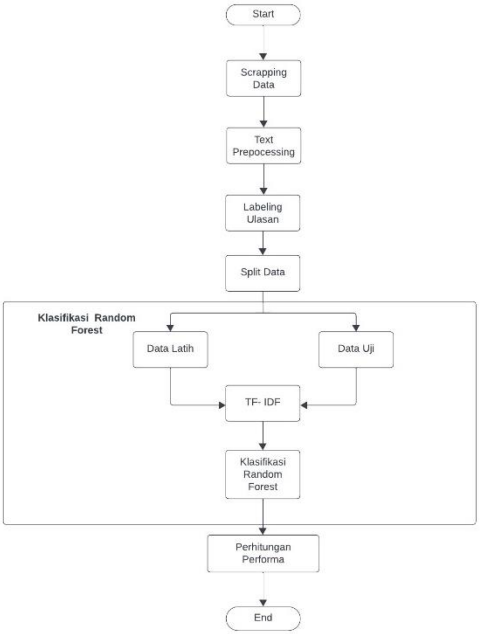
Gambar 3. 1 Alur Metode KDD

Metode KDD ini memiliki beberapa tahapan dalam proses yang akan dilakukan, yaitu:

1. *Selection*: Tahapan awal yang dilakukan dalam proses metode KDD. Didalam tahapan ini akan terdapat proses pengumpulan, sleksi hingga proses pelabelan data.
2. *Prepocessing*: Proses tahap *preprocessing* ini akan mengubah data mentah menjadi data yang siap digunakan untuk proses selanjutnya.
3. *Transformation*: Tahap *transformation* merupakan tahap yang akan mengubah data menjadi bentuk yang dapat diolah pada tahap klasifikasi.
4. *Data Mining*: Pada tahap ini akan terdapat proses dilakukannya klaisifkasi sentimen dengan menggunakan metode atau algoritma tertentu.
5. *Interpretation/Evaluation*: Pada tahap akhir ini akan menginterpretasi atau mengevaluasi dari klasifikasi yang digunakan.

D. Diagram Alur Penelitian

Diagram alur penelitian ini akan memberikan penjelasan terkait proses penelitian ini. Alur penelitian yang akan digunakan dalam penelitian ini mengacu pada diagram dibawah ini:



Gambar 3. 2 Alur Penelitian

Langkah awal yang akan dilakukan dalam penelitian ini ialah mengambil data ulasan oleh pengguna aplikasi livin' by mandiri pada aplikasi *google playstore* melalui cara scrapping data. Proses web scrapping data ini dilakukan dengan cara menggunakan bahasa Pemrograman *Python* yang dimulai dengan penginstalan package *google play scraper*. Dilanjutkan dengan pengambilan ID aplikasi dari *google playstore*. Kemudian akan didapatkan data terkait ulasan aplikasi livin' by mandiri.

Langkah berikutnya, maka akan dilakukan tahap *text preprocessing* dimana akan mengubah data yang masih utuh akan melalui tahap pembersihan sehingga akan bisa dilalukan tahap pengklasifikasian. Pada proses *text preprocessing* ini akan melalui tujuh tahap yang terdiri dari *cleaning, case folding, slang word standardization, stopword removal, stemming, tokenizing.* Dimulai dari tahap *cleaning* yang akan menghapus karakter yang tidak diperlukan. Kemudian dilanjutkan dengan tahap *case folding* yaitu menyamaratakan karakter huruf yang terdapat pada data tersebut menjadi huruf kecil (*lowercase*). Tahap *slang word standardization* dimana akan terjadi proses penyelarasan kata slang yang sudah berbaur pada lingkungan masyarakat. Lalu selanjutnya dilakukan *Stopword Removal* atau bisa disebut juga proses *filtering* merupakan proses penghapusan kata yang tidak mengandung arti. Setelah tahap *stopword removal* selanjutnya akan dilakukan tahap *stemming* yaitu proses penghapusan imbuhan kata yang terdapat pada awalan, sisipan ataupun akhiran. Kemudian tahap terakhir dalam *preprocessing* yaitu *tokenizing* merupakan proses memecah sebuah teks atau kalimat menjadi kata perkata. Langkah selanjutnya yaitu

pemberian label pada data ulasan. Dataset yang sebeumnya belum terdapat label yang menjelaskan positif dan negatif. Dengan demikian pada tahap ini peneliti akan melalukan proses pelabelan dengan cara melalui *dictionary* kosa kata yang memiliki konotasi positif dan negative dengan menggunakan Teknik *Lexicon Based*.

Setelah dilakukannya proses pelabelan maka tahap selanjutnya adalah proses *split data* yaitu proses yang akan membagi data antara data latih dan data uji yang memiliki rasio 80:20 yaitu 80% dari jumlah data yang ada akan digunakan sebagai data latih dan 20% dari jumlah data yang ada akan digunakan sebagai data uji.

Tahap berikutnya ialah proses ekstraksi fitur yang akan mengubah sebuah kata menjadi angka dan dilakukan proses pembobotan nilai yang akan menggunakan TF-IDF yang bertujuan untuk mempermudah proses klasifikasi dengan menggunakan metode *random forest*.

Kemudian proses selanjutnya adalah tahap yang paling inti yaitu proses pengklasifikasian menggunakan metode *random forest*. Proses

pengklasfikasian ini berdasarkan dengan sentimen yang terdapat didalam dokumen. Setelah itu akan dilakukan proses uji model yang bertujuan untuk mengetahui ketepatan dari klasifikasi dan mengetahui kinerja dari model sehingga akan menghasilkan *confusion matrix*. Apabila proses uji model ini telah selesai maka selanjutnya adalah tahap evaluasi model yang akan memberikan hasil *accuracy*, *precission, recall* dan *f1 Score* melalui tabel *confusion matrix* untuk mengathui peroforma dari model yang digunakan.

E. Uraian Metode Penelitian

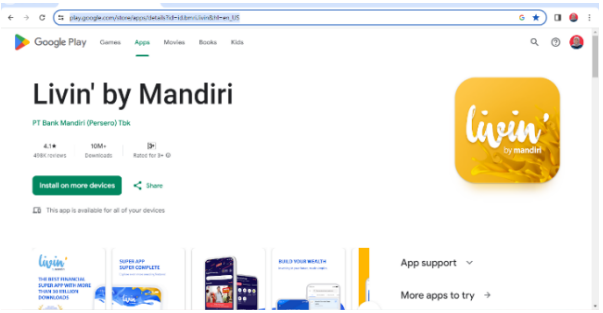
Didalam tahap ini akan dilakukan penjelasan terkait dengan alur pengerjaan penelitian diatas untuk menjelaskan lebih detail sebagai berikut:

1. Pengambilan Data Ulasan Pengguna Aplikasi Livin' by Mandiri

Didalam penelitian ini data yang digunakan oleh peneliti merupakan data ulasan pengguna aplikasi livin' by mandiri yang ambil melalui *google playstore* dengan mengggunakan teknik scraping data. Data Scraping yaitu sebuah teknik yang digunakan dengan cara mengekstrak sebuah data yang diambil melalui *website*. Tujuan dari scrapping data ini yaitu untuk mengambil data

ulasan yang diberikan oleh pengguna aplikasi dan data tersebut akan digunakan dalam penelitian ini (Ahmad, 2022).

Data yang telah di ekstrak tersebut akan disimpan dalam sebuah file yang memiliki format tabular, *spreadsheet*. Pada teknik scrapping data ini dilakukan dengan cara membuat program terlebih dahulu dan dilanjutkan dengan memasukan ID Aplikasi yang diambil melalui *google playstore*.



Gambar 3. 3 Pengambilan ID Aplikasi

2. Labeling Ulasan

Apabila proses *text preposcessing* telah dilakukan semua pada tahap sebelumnya maka proses selanjutnya adalah proses pelabelan pada dataset. Tahapan labeling ulasan pada dataset yang sebelumnya belum terdapat label positif, negatif ataupun netral maka tahap selanjutnya akan dilakukan

proses pelabelan secara otomatis dengan memperhitungkan skor dari sentimen agar mempermudah proses klasifikasi dengan menggunakan kamus *lexicon based*. *Lexicon based* merupakan pendeketan yang berbasis kamus yang digunakan dengan cara menghitung setiap skor sentimen. Didalam kamus tersebut berisikan sebuah Kumpulan kata yang memiliki bobot yang akan menghasilkan nilai polaritas. Nilai polaritas tersebut berasal dari penjumlahan keseluruhan bobot kata pada setiap ulasan. Kemudian hasil akhir yang didapatkan akan menjadikan label kelas sentiment dari ulasan yang ada pada setiap aspek (Budianita et al., 2022).

Penelitian yang dilakukan Sarah Anggina dkk terkait dengan Ulasan Pelanggan Pada Formaggio Coffee and Resto Menggunakan *Multinomial Naïve Bayes Classifier* dengan *Lexicon-Based* dan TF-IDF yang memiliki empat parameter, yaitu accuracy, recall, precision, dan f1-score. Didapatkan nilai yang baik dari setiap parameter sebesar 95%, 68%, 85%, dan 72% (Anggina *et al*., 2022). Selain itu Kolchyna melakukan penelitian dengan menggunakan metode *lexicon* yang dipadukan dengan metode SVM yang dapat mempermudah proses pengolahan data dan hasil yang

diperolah dapat meningkatkan performa dari metode SVM secara keseluruhan (Khoo & Johnkhan, 2018).

3. Text Prepocessing

Tahap pertama yang dilakukan dalam proses text mining yaitu *text prepocessing*. *Text preprocessing* merupakan proses *text mining* terhadap data teks agar menghasilkan data text yang sesuai dengan format dan mudah untuk dipahami (Alun Sujjadaa et al., 2023). Tujuan dari adanya *text prepocessing* ini yaitu untuk meminimalisir adanya sebuah data yang kurang sempurna (Onantya & Adikara, 2019). Dalam tahap text *preprocessing* terdapat beberapa tahapan yang dilakukan yang akan ditunjukkan pada Gambar 2.2 berikut:

a. Cleaning

Pada proses *cleaning* ini merupakan sebuah proses yang dilakukan untuk menghapus karakter yang tidak diperlukan pada dokumen seperti tanda baca, *symbol* dan emoticon. Dalam sebuah ulasan sering ditemukan penggunaan karakter, *symbol, emoticon*. Namun untuk memudahkan dalam proses pengolahan data ketiganya perlu dihapus sehingga diperlukannya proses

cleaning tersebut. Contoh proses *cleaning* terdapat pada Tabel 3.1.

Input Process	***Output Process***
Ini kenapa aplikasi mental Mulu ya, sudah 2 hari. Dan ingin melakukan transaksipun jadi terkendala. Apalagi sehari – hari selalu menggunakan Livin. Tolong lah segera diperbaiki. Untuk saat ini 1 bintang dulu.	Ini kenapa aplikasi mental Mulu ya sudah hari Dan ingin melakukan transaksipun jadi terkendala Apalagi sehari hari selalu menggunakan Livin Tolong lah segera diperbaiki Untuk saat ini bintang dulu

Tabel 3. 1 Penerapan Tahap Cleaning

b. Case Folding

Case Folding merupakan proses mengkonverensi karakter huruf yang terdapat pada data tersebut menjadi huruf kecil (*lowercase*). Tujuan adanya tahapan *case folding* ini yaitu untuk mempermudah proses pencrian kata yang terdapat pada data. Contoh proses *case folding* terdapat pada Tabel 3.2.

Input Process	*Output Process*
Ini kenapa aplikasi mental Mulu ya sudah hari Dan ingin melakukan transaksipun jadi terkendala Apalagi sehari hari selalu menggunakan Livin Tolong lah segera diperbaiki Untuk saat ini bintang dulu	ini kenapa aplikasi mental mulu ya sudah hari dan ingin melakukan transaksipun jadi terkendala apalagi seharihari selalu menggunakan livin tolong lah segera di perbaiki untuk saat ini bintang dulu

Tabel 3. 2 Penerapan Tahap Case Folding

c. Removing Duplicate

Pada ulasan yang diberikan oleh pengguna pada google playstore mungkin saja melakukan berulang-ulang. Kemudian tujuan dari adanya proses ini yaitu untuk menghapus ulasan yang terdapat kesamaan sehingga untuk melakukan proses preprocessing hanya diperlukan satu ulasan.

d. Slang Word Standaridization

Didalam memberikan ulasan pada sebuah aplikasi di *Google playstore* pengguna biasanya menggunakan kata yang dipersingkat dalam pemyampaiannya. *Slang* merupakan bentuk sebuah bahasa yang digunakan secara umum,

penggunanaan bahasa ini dibuat berdasarkan kepopularan dari sebuah kata biasanya pembentukan kata tersebut oleh kelompok social atau kelompol usia tertentu. *Slang Word Standardization* yaitu proses penyelarasan kata *slang* yang sudah berbaur pada lingkungan masyarakat. Pada tahap *slang word standardization* ini akan ada proses penyelarasan kata tidak sesuai dengan kamus besar bahasa Indonesia. Contoh proses *slang word standardization* terdapat pada Tabel 3.3.

Input Process	***Output Process***
ini kenapa aplikasi mental mulu ya sudah hari dan ingin melakukan transaksipun jadi terkendala apalagi seharihari selalu menggunakan livin tolong lah segera di perbaiki untuk saat ini bintang dulu	ini kenapa aplikasi mental mulu ya sudah hari dan ingin melakukan transaksipun jadi terkendala apalagi seharihari selalu menggunakan livin tolong lah segera di perbaiki untuk saat ini bintang dulu

Tabel 3. 3 Penerapan Tahap Slang Word Standaridization

e. Stopword Removal

Stopword Removal atau bisa disebut juga proses *filtering* merupakan proses penghapusan kata yang tidak mengandung arti

yang bermakna sepertikata hubung, kata panggilan dan lain lain. Proses *Stopword Removal* ini memiliki tujuan agar hasil dari data yang dianalisis menghasilkan data yang penting. Contoh proses *stopword removal* terdapat pada Tabel 3.4.

Input Process	***Output Process***
ini kenapa aplikasi mental mulu ya sudah hari dan ingin melakukan transaksipun jadi terkendala apalagi seharihari selalu menggunakan livin tolong lah segera di perbaiki untuk saat ini bintang dulu	aplikasi mental mulu transaksipun terkendala seharihari livin tolong perbaiki bintang

Tabel 3. 4 Penerapan Tahap Stopword Removal

f. Stemming

Stemming merupakan proses penghapusan imbuhan kata yang terdapat pada awalan, sisipan atau akhiran. Proses ini akan manemukan kata dasar dari sebuah kata yang sesuai dengan struktur morfologi Bahasa Indonesia yang benar (Verdaningroem & Saifudin, 2018). Proses *stemming* dengan pengambilan kata dasar yang sesuai dengan Kamus Besar Bahasa Indonesia mendapatkan

akurasi yang lebih cukup baik (Studi *et al*., 2013). Adanya Proses *stemming* akan mempermudah dalam pencarian informasi untuk meningkatkan kualitas informasi yang didapatkan. Dalam tahap *stemming* ini diperlukan sebuah *library. Library* yang akan digunakan dalam proses ini yaitu menggunakan *library* sastrawi yaitu *library Stemming* untuk Bahasa Indonesia. *Library* Sastrawi ini didalam penggunaannya menggunakan *Algoritma Stemming* dari Nazief dan Adriani untuk Bahasa Indonesia (Budiman & Widjaja, 2020).

Contoh proses *stemming* terdapat pada Tabel 3.5.

Input Process	***Output Process***
aplikasi mental mulu transaksipun terkendala seharihari livin tolong perbaiki bintang	aplikasi mental mulu transaksi kendala seharihari livin tolong baik bintang

Tabel 3. 5 Penerapan Tahap Stemming

g. Tokenizing

. Pada tahap *tokenizing* ini karakter yang ada pada suatu kata akan dihilangkan karena tidak berpengaruh terhadap proses

Tokenizing merupakan proses memecah sebuah teks atau kalimat menjadi kata perkata yang menjadi penyusun dari sebuah dokumen pemrosesan teks. Contoh proses *tokenizing* terdapat pada Tabel 3.6.

Input Process	***Output Process***
aplikasi mental mulu transaksi kendala seharihari livin tolong baik bintang	['aplikasi', 'mental', 'mulu', 'transaksi', 'kendala', 'seharihari', 'livin', 'tolong', 'baik', 'bintang']

Tabel 3. 6 Penerapan Tahap Tokenizing

4. Ektrasi Fitur dengan TF - IDF

Pada tahap ini akan dilakukan proses ektrasi fitur dengan menggunakan TF-IDF. *Term Frequency-Invers Document Frequency* (TF-IDF) merupakan metode algoritma yang berfungsi untuk menentukan bobot yang terdapat pada setiap kata dalam suatu dokumen atau artikel. Data yang sebelumnya telah selesai dalam tahap *preprocessing* hasilnya harus berbentuk numerik agar bisa diselesaikan dalam proses klasifikasi. Dengan menggunakan metode pembobotan TF-IDF maka data tersebut bisa diubah menjadi data yang berbentuk numerik. Secara sederhana metode TF-IDF didefinisikan sebagai metode yang berfungsi untuk mengetahui seberapa

sering kata didalam dokumen itu muncul. TF-IDF ini merupakan hasil perhitungan yang dilakukan antara TF (*Term Frequency*) dan IDF (*Invers Document Frequency*). TF (*Term Frequency*) merupakan jumlah frekuensi dari kata yang muncul dalam sebuah dokumen atau artikel. Sedangkan IDF (*Invers Document Frequency*) merupakan proses pengukuran seberapa penting suatu kata yang terdapat dalam sebuah dokumen atau artikel. Rumus TF-IDF sebagai berikut:

$$TFIDF = \text{TF x IDF} \tag{3.5}$$

$$TF = \frac{\text{Jumlah kata t pada dokumen}}{Total\ kata\ dalam\ satu\ dokumen} \tag{3.6}$$

$$IDF = \log \frac{\text{Total dokumen}}{Frekuensi\ dokumen\ mengandung\ term} + 1 \tag{3.7}$$

Proses TF-IDF ini akan menghasilkan bobot dalam sebuah dokumen dari teks atau kata yang ada didalamnya. Jika telah dilakukan pembobotan dengan menggunakan TF-IDF maka dataset akan melalui proses *training* dengan menggunakan klasifikasi *random forest.*

5. Pengklasfikasian *Random Forest*

Apabila data telah melewati proses *preprocessing*, pelabelan ulasan pada dataset dan dilakukan proses pembobotan dengan menggunakan TF-IDF maka tahap selanjutnya yaitu proses pengklasfikasian dengan menggunakan metode *random forest*. Algori tma *random forest* merupakan dari salah satu metode dari *machine learning* yang digunakan untuk proses klasifikasi sebuah data dalam jumlah yang banyak.

Berikut merupakan tahapan dalam melakukan pengklasifikasian dengan menggunakan *random forest*:

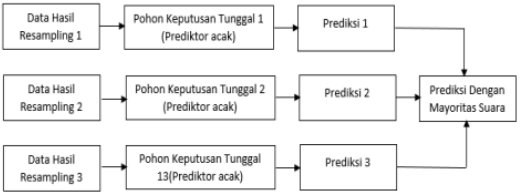
a. Membuat suatu *bosstrap sample* atau pengambilan sampel *Z* dengan *replacement* (pengambilan) dari suatu ukuran N dari gugus data.
b. Memilih m (*mtry*) variabel dengan cara random, dari p variabel, dimana $m \leq p$.
c. Setelah dilakukan pemilihan m secara random, maka pohon di tumbuhkan tanpa *pruning* (pemangkasan).
d. Kemudian langkah 1 sampai dengan 3 dilakukan sebanyak *n* kali hingga terbentuk suatu *forest* sebanyak *n* pohon.

e. Didalam menentukan suatu kelas dilakukan dengan cara *majority vote.*

f. Kemudian setelah terbentuk *forest* maka akan dicari *parameter mtry* yang optimal sehingga diperoleh nilai misklasfikasi error (*Out of Bag Eror*) yang stabil dan juga adanya tingkat kepentingan variabel (*Variabel Importance*).

g. Setelah diperoleh nilai *mtry* optimal kemudian dilakukan prediksi dengan menggunakan data *testing*.

Secara sederhana gambaran algoritma Random Forest ditunjukkan pada gambar 3.4.



Gambar 3. 4 gambar sederhana random forest

6. Pengujian Model

Pengujian model ini akan dilakukan pembagian dataset antara data latih (*training*) dan data uji (*testing*) dengan cara *Split Data. Split Data* merupakan sebuah teknik validasi yang akan membagi menjadi dua bagian yaitu data latih (*training*) dan data uji

(*testing*) secara acak. *Split validation* dilakukan dengan mengunakan jumlah data yang akan dijadikan yang berasal dari data *training* sebesar 20% (Larasati *et al*., 2022). Tujuan dilakukannya proses pengujian model ini agar diketahui seberapa besar performa dari metode yang digunakan dan tentunya akan mengetahui performa dari model yang digunakan juga.

7. Evaluasi Model

Tahap evaluasi model ini bertujuan untuk mengukur kinerja dari suatu model yang digunakan. Didalam mengevaluasi kinerja model maka dilakukan menggunakan *multiclass confusion matrix*.

a. *Confusion Matrix*

Confusion Matrix merupakan tabel yang berisi hasil pengukuran performa yang berasal dari model klasifikasi *machine learning*. *Confusion matrix* ini digunakan untuk menentukan klasifikasi dalam sebuah metode apakah termasuk dalam label baik, buruk ataupun netral. Didalam penelitian ini output yang dihasilkan berupa sentiment yang terdiri dari tiga *class* yaitu sentimen negatif, positif dan netral maka dari itu penelitian ini menggunakan *multiclass confusion matrix* 3x3. Adanya *confusion matrix* yang terdiri dari berbagai macam bagian ini dijadikan evaluasi hasil dari

prediksi yang dilakukan oleh metode *Random Forest.* Berikut merupakan tabel dari *multiclass confusion matrix.*

		Predicted Class		
		Negative	***Netral***	***Positif***
True Class	*Negative*	T Neg	F NegNet	F NegPos
	Netral	F NetNeg	T Net	F Netos
	Positif	F Neg	F PosNet	T Pos

Tabel 3. 7 Multi Class Confusion Matrix

Keterangan:

- TP (*True Positive*) merupakan jumlah data yang diprediksi positif dan faktanya data tersebut positif (Sesuai).
- F PosNeg (*False Positive Negative*), merupakan jumlah data yang diprediksi positif dan faktanya data itu Negatif.
- F PosNet (*False Positive Netral*), merupakan jumlah data yang diprediksi positif dan faktanya data itu Netral.
- F NegPos (*False Negative Positive*), merupakan jumlah data yang diprediksi negatif dan faktanya data itu positif.

- T Neg (*True Negative*), merupakan jumlah data yang diprediksi negatif dan faktanya data itu negatif (Sesuai).
- F NegNet (*False Negative Netral*), merupakan jumlah data yang diprediksi negatif dan faktanya data itu netral.
- F NetPos (*False Netral Positive*), merupakan jumlah data yang diprediksi netral dan faktanya data itu positif.
- F NetNeg (*False Netral Negative*), merupakan jumlah data yang diprediksi netral dan faktanya data itu negatif.
- T Net (True Netral), merupakan jumlah data yang diprediksi netral dan faktanya data itu netral (Sesuai).

Dalam melakukan pengukuran performa dari model yang digunakan maka diperlukannya menghitung yaitu nilai *accuracy, precision, recall,* dan *F1-Score* (Suci Amaliah *et al*., 2022) .

Accuracy adalah variabel yang penggunaannya berdasarkan perhitungan presentase proporsi hasil klasifikasi yang benar.

$$Accuracy = \frac{\text{TPos+TNeg+TNet}}{jumlah\ data\ uji} \quad (3.1)$$

Precision adalah kualitas ketepatan dari sebuah informasi actual dengan adanya prediksi sistem. Berikut persamannya:

$$precision = \frac{\text{tp}}{tp+fp} \quad (3.2)$$

Recall merupakan jumlah adanya keberhasilan sebuah sistem ketika menampilkan informasi yang berbentuk analisis sentimen. Berikut persamannya:

$$recall = \frac{\text{tp}}{tp+fn} \quad (3.3)$$

F1 Score merupakan sebuah perhitungan yang mengkombinasikan nilai *precision* dan *recall* dengan menggunakan perbandingan nilai *precision* dan *recall* tersebut. Berikut persamannya:

$$F1 = 2.\frac{\text{precision . recall}}{precision+recall} \quad (3.4)$$

8. Visualisasi

Didalam proses analisis sentimen ini akan ada tahap yang akan menvisualisasikan data hasil pengklasifikasian yang menggunakan *wrodcloud* dan juga diagram lingkaran. Tujuan visualisasi dengan menggunakan ini untuk mengekstraksikan terkait informasi yang di implementasikan dalam bentuk topik yang paling sering disampaikan oleh pengguna

aplikasi. Sehingga dari banyaknya informasi yang disampaikan terdapat informasi yang dianggap paling penting (Fadilah, 2018).

BAB IV

HASIL DAN PEMBAHASAN

Pada bab ini akan menjelaskan terkait dengan hasil dan pembahasan dari sebuah sistem yang telah dibangun. Dengan demikian maka diperlukannya pengujian terhadap sistem dengan melalui berbagai tahapan yaitu dengan adanya tahap *scraping*, tahap *preprocessing*, kemudian melalui tahapan klaisifikasi dengan menggunakan Metode *Random Forest*, tahap perhitungan akurasi, dan juga tahap visualisasi data. Di dalam penelitian terkait analisis sentiment ini subjek yang digunakan adalah ulasan dari pengguna Aplikasi Livin' by Mandiri yang diambil melalui Google PlayStore.

A. Pengambilan Data Ulasan

Proses pengambilan data ulasan yang berupa dilakukan pada penelitian ini yaitu dengan menggunakan Teknik web *scraping*. Proses pengambilan data dengan web *scraping* menggunakan *python* terdapat beberapa tahapan yaitu:

1. Penginstallan Google Play *Scrapper Package*

Didalam proses scrapping maka tahap awal yang perlu dilakukan penginstalan sebuah package google play scaper.

Gambar 4. 1 Package google playstore

2. Pemanggilan Library

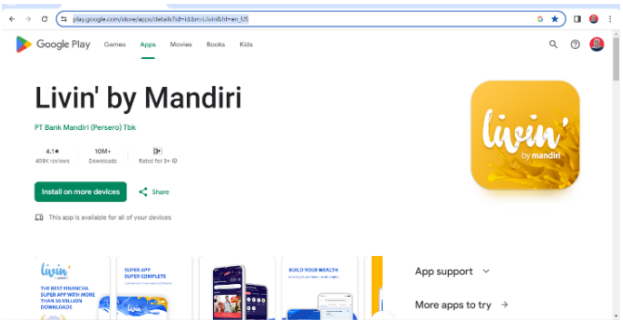
Dalam proses pengambilan data menggunakan teknik *scrapping* maka diperlukan pengambilan library yang dibutuhkan yang berfungsi untuk mengambil feature yang dibutuhkan seperti "review","username","userimage","content","score" dan feature yang lainnya.

```
from google_play_scraper import app
import pandas as pd
import numpy as np
```

Gambar 4. 2 Pemanggilan library scrapping

3. Menyalin ID Aplikasi Livin' by Mandiri pada Google PlayStore

Membuka aplikasi livin' by mandiri dan kemudian disalin ID Aplikasi tersebut yang akan di *scrapping* pada web google playstore lalu dimasukkan pada code yang ada. ID Aplikasi ditunjukkan pada gambar 4.3.



Gambar 4. 3 Tampilan ID Aplikasi Livin' by mandiri melalui google play store

4. *Scrapping* ulasan aplikasi livin' by mandiri

Pada proses sebelumnya telah menyalin ID dari aplikasi livin' by mandiri melalui google playstore lalu sertakan jumlah ulasan yang akan digunakan untuk proses sentimen yang dimasukkan serta dijalankan ke dalam *code* maka lanjutkan dengan memproses tahap *scrapping* data pada gambar 4.4.

```
: #scrape berdasarkan jumlah yang di inginkan
  # run code jika ingin scraping data dengan jumlah tertentu

  from google_play_scraper import Sort, reviews

  result, continuation_token = reviews(
      'id.bmri.livin',
      lang='id', #default to 'en'
      country='id', #default to 'us'
      sort=Sort.MOST_RELEVANT, #defaults to Sort.MOST_RELEVANT
      count=10000,
      filter_score_with=None
```

Gambar 4. 4 Tahap scraping data ulasan

Penggunaan library google_play_scraper dalam bahasa pemrograman Python digunakan untuk

mengambil ulasan dari aplikasi yang akan dilakukan sentimen pada Google Play Store. Pada penelitian ini aplikasi yang diambil ulasannya memiliki ID paket `'id.bmri.livin'`. Kemudian penggunaan parameter `lang='id'` dan `country='id'` mengindikasikan bahwa ulasan yang akan diambil berbahasa Indonesia dan dari pengguna yang berasal dari negara Indonesia. Selanjutnya terkait pengaturan `sort=Sort.MOST_RELEVANT` menentukan bahwa ulasan akan diurutkan berdasarkan relevansinya. '`count = 10.000`' menunjukkan maksimal ulasan yang akan diambil menjadi 10.000 ulasan. Kemudian hasil yang diperoleh dari pemanggilan fungsi '`reviews()`' yaitu daftar ulasan dalam variabel '`result`' dan apabila jumlah ulasan melebihi batas maksimum makapenggunaan token '`continuation_token`' digunakan yang berfungsi untuk melanjutkan pengambilan ulasan tambahan.

5. Membuat Hasil *Scrapping* Menjadi Dataframe

Pada proses *scrapping* yang telah dilakukan maka akan dilakukan proses pengubahan menjadi dataframe yang menggunakan *library* numpy dan juga pandas pada gambar 4.5.

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

Gambar 4. 5 Tahap pengubahan ulasan menjadi data frame

Dataframe dibuat berdasarkan hasil ulasan dengan kolom ‘`review`’ yang dipindahkan kedalam dataframe terbaru Kemudian, `tolist()` digunakan untuk mengubah nilai-nilai dalam kolom `'review'` menjadi sebuah daftar, dan `pd.DataFrame()` digunakan untuk membuat DataFrame baru dari daftar tersebut.

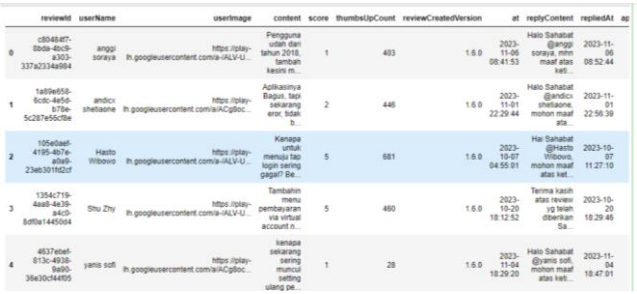
6. Membaca Data Hasil Pengubahan Menjadi Dataframe

Selanjutnya akan dilakukan proses pembacaan data yang sebelumnya telah dilakukan *scrapping* yang ditunjukkan pada gambar 4.6.

```
df = pd.read_csv('data10000.csv')

df.head()
```

Gambar 4. 6 Membaca data frame

Hasil dataframe ditunjukkan pada gambar 4.7.



	reviewId	userName	userImage	content	score	thumbsUpCount	reviewCreatedVersion	at	replyContent	repliedAt	ap
0	c80484f7-88da-4bc9-a303-337a2334a984	anggi soraya	https://play-lh.googleusercontent.com/a-/ALV-U...	Pengguna udah dari tahun 2018, tambah kesini m...	1	493	1.6.0	2023-11-06 08:41:53	Halo Sahabat @anggi soraya, mhn maaf atas keti...	2023-11-06 08:52:44	
1	1a89e658-6cdc-4e5d-b78e-5c287e56cf8e	andicx shetiaone	https://play-lh.googleusercontent.com/a/ACg8oc...	Aplikasinya Bagus, tapi sekarang error, tidak b...	2	446	1.6.0	2023-11-01 22:29:44	Halo Sahabat @andicx shetiaone, mohon maaf ata...	2023-11-01 22:56:39	
2	105e0aef-4195-4b7e-a0a9-23eb3011fd2cf	Hasto Wibowo	https://play-lh.googleusercontent.com/a-/ALV-U...	Kenapa untuk menuju tap login sering gagal? Be...	5	681	1.6.0	2023-10-07 04:55:01	Hai Sahabat @Hasto Wibowo, mohon maaf atas ket...	2023-10-07 11:27:10	
3	1354c719-4aa8-4e39-a4c0-8df0a14450d4	Shu Zhy	https://play-lh.googleusercontent.com/a-/ALV-U...	Tambahin menu pembayaran via virtual account n...	5	460	1.6.0	2023-10-20 18:12:52	Terima kasih atas review yg telah diberikan Sa...	2023-10-20 18:29:46	
4	4637ebef-813c-4938-9a90-36e30cf44f05	yanis sofi	https://play-lh.googleusercontent.com/a/ACg8oc...	kenapa sekarang sering muncul setting ulang pe...	1	28	1.6.0	2023-11-04 18:29:20	Halo Sahabat @yanis sofi, mohon maaf atas keti...	2023-11-04 18:47:01	

Gambar 4. 7 Hasil DataFrame

7. Menyimpan Hasil *Scrapping* Data ke dalam Format CSV

Tahap selanjutnya yaitu mengubah dan menyimpan dataframe dalam format CSV yang bertujuan untuk dijadikan sebagai data pada proses klaisifkasi menggunakan metode *Random Forest* yang dit

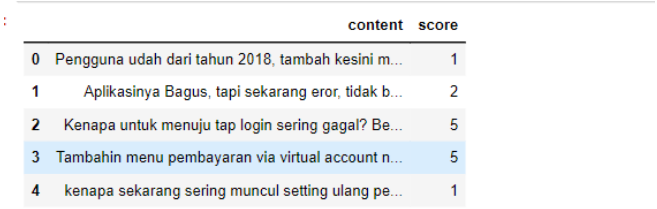
```
df.to_csv("data10000.csv", index = False)
```

Gambar 4. 8 menyimpan data kedalam format csv

Proses penyimpanan dalam format csv yang kemudian. Nama file yang digunakan adalah "data10000.csv". Dengan menetapkan `index = False`, baris indeks dalam DataFrame tidak akan disertakan dalam file CSV yang dihasilkan.

B. Text Preprocessing

Tahap *preprocessing* pada analisis sentimen ini merupakan tahapan yang penting dilakukan karena pada data ulasan memungkinkan banyak terdapat *noise* yang akan mempengaruhi hasil dari akurasi yang diberikan. Tujuan dilakukannya proses *preprocessing* ini akan menghasilkan data yang bersih sehingga dapat dilakukan proses berikutnya yaitu pengklasifikasian. Namun sebelum melanjutkan pada proses preprocessing maka perlu melihat proses pengambilan *dataset* menggunakan teknik *scrapping* yang sudah diambil berdasarkan variabel yang penting ditunjukkan pada gambar 4.9.



	content	score
0	Pengguna udah dari tahun 2018, tambah kesini m...	1
1	Aplikasinya Bagus, tapi sekarang eror, tidak b...	2
2	Kenapa untuk menuju tap login sering gagal? Be...	5
3	Tambahin menu pembayaran via virtual account n...	5
4	kenapa sekarang sering muncul setting ulang pe...	1

Gambar 4. 9 membaca dataset

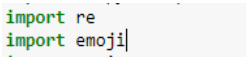
a. Cleaning

Dalam menjalankan proses cleaning maka diperlukannya sebuah *library. library* re dan *emoji*. *Library re* sendiri berfungsi untuk mengetahui deretan dari karakter yang akan digunakan untuk melakukan pencarian teks yang menggunakan pola.

Kemudian selain itu *library emoji* berfungsi untuk mendekteksi emoji yang terdapat pada ulasan sehingga emoji tersebut akan hilang. Pada gambar 4.10 ini menunjukkan proses penginstallan *library emoji.*

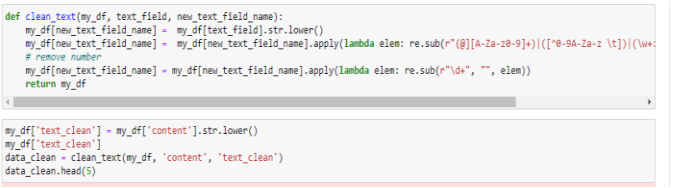
Gambar 4. 10 Install library emoji

Setelah proses penginstallan maka selanjutnya adalah mengimport library re dan juga emoji pada gambar 4.11.



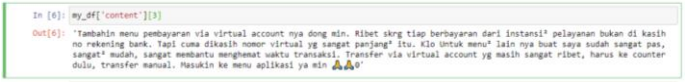
Gambar 4. 11 import library re dan emoji

Proses *cleaning* ini memiki tujuan untuk menghapus karakter yang tidak diperlukan pada dokumen seperti tanda baca, *symbol* dan *emoticon*. Berikut merupakn *source code* dari proses *cleaning* yang dijadikan satu dengan proses *casefolding* pada gambar 4.12.



Gambar 4. 12 Source code tahap cleansing dan case folding

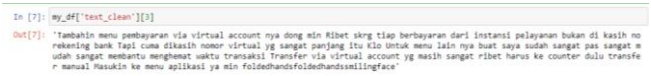
Fungsi `clean_text` di atas menerima tiga parameter `my_df` yang merupakan DataFrame, `text_field` yang merupakan nama kolom di dalam DataFrame yang berisi teks yang akan dibersihkan, dan `new_text_field_name` yang merupakan nama kolom baru tempat hasil teks yang telah dibersihkan akan disimpan. Fungsi '`str.lower()`' digunakan untuk pengubahan menjadi huruf kecil. Selanjutnya proses pembersihan selesai, hasilnya disimpan dalam kolom baru `(new_text_field_name)`. Berikut merupakan salah satu contoh ulasan yang belum melalui proses cleaning ditunjukkan pada gambar 4.13.



Gambar 4. 13 Contoh proses cleaning

Berdasarkan contoh proses cleaning pada gambar 4.13 maka hasil yang diperoleh setelah proses

cleaning seperti pada gambar 4.14 bahwa emoji yang terdapat pada suatu ulasan akan hilang.



Gambar 4. 14 Contoh hasil proses cleaning

Dibawah ini akan ditampilkan hasil dari *casefolding* dari data teratas yang ditunjukkan pada gambar 4.13.

	content	score	text_clean
0	Pengguna udah dari tahun 2018, tambah kesini m...	1	pengguna udah dari tahun tambah kesini malah ...
1	Aplikasinya Bagus, tapi sekarang eror, tidak b...	2	aplikasinya bagus tapi sekarang eror tidak bis...
2	Kenapa untuk menuju tap login sering gagal? Be...	5	kenapa untuk menuju tap login sering gagal beg...
3	Tambahin menu pembayaran via virtual account n...	5	tambahin menu pembayaran via virtual account n...
4	kenapa sekarang sering muncul setting ulang pe...	1	kenapa sekarang sering muncul setting ulang pe...
...	...	...	...
9819	Luar biasa.. Praktis dan nyaman	5	luar biasa praktis dan nyaman
9820	Saya beli pulsa sebesar 100rb tapi tidak masuk...	1	saya beli pulsa sebesar rb tapi tidak masuk d...
9821	Setelah update pembaharuan terbaru kok aplikas...	1	setelah update pembaharuan terbaru kok aplikas...
9822	Begitu daftar livin, ada WA tdk jelas dr nomer...	1	begitu daftar livin ada wa tdk jelas dr nomer ...
9823	Tolong untuk beberapa opsi seperti pengisian e...	3	tolong untuk beberapa opsi seperti pengisian e...

Gambar 4. 15 Hasil tahapan cleansing dan case folding

b. Casefolding

Tahap *casefolding* ini akan mengubah huruf besar menjadi huruf kecil yang terdapat pada data sejumlah 10.000 ulasan sehingga semua data ulasan akan disamaratakan menjadi huruf kecil. Adapun *sorce code* dari proses *casefolding* digabung menjadi satu pada proses *cleaning*.

c. *Removing Duplicate*

Tahap *removing duplicate* tujuan dari adanya proses ini yaitu untuk menghapus ulasan yang terdapat kesamaan sehingga untuk melakukan proses *preprocessing* hanya diperlukan satu ulasan. Adapun *source code* ditunjukkan pada gambar 4.14.

```
: my_df = my_df.drop_duplicates()
  my_df = my_df.reset_index(drop=True)
  my_df
```

Gambar 4. 16 source code tahap removing duplicate

Pemanggilan '`drop_duplicates()`' digunakan untuk penghapusan data yang terduplikat sedangkan penggunaan '`reset_index(drop=True)`' berfungsi untuk mereset indeks DataFrame dan penghapusan indeks yang berisi duplikat dan mengatur indeks secara berurutan. Berdasarkan *source code* diatas maka dihasilkan penerapan dari tahapan *removing duplicate* yang menghasilkan 10.000 data menjadi 9824 data seperti pada gambar 4.15.

	content	score	text_clean
0	Pengguna udah dari tahun 2018, tambah kesini m...	1	pengguna udah dari tahun tambah kesini malah ...
1	Aplikasinya Bagus, tapi sekarang eror, tidak b...	2	aplikasinya bagus tapi sekarang eror tidak bis...
2	Kenapa untuk menuju tap login sering gagal? Be...	5	kenapa untuk menuju tap login sering gagal beg...
3	Tambahin menu pembayaran via virtual account n...	5	tambahin menu pembayaran via virtual account n...
4	kenapa sekarang sering muncul setting ulang pe...	1	kenapa sekarang sering muncul setting ulang pe...
...	...	...	...
9819	Luar biasa.. Praktis dan nyaman	5	luar biasa praktis dan nyaman
9820	Saya beli pulsa sebesar 100rb tapi tidak masuk...	1	saya beli pulsa sebesar rb tapi tidak masuk d...
9821	Setelah update pembaharuan terbaru kok aplikas...	1	setelah update pembaharuan terbaru kok aplikas...
9822	Begitu daftar livin, ada WA tdk jelas dr nomer...	1	begitu daftar livin ada wa tdk jelas dr nomer ...
9823	Tolong untuk beberapa opsi seperti pengisian e...	3	tolong untuk beberapa opsi seperti pengisian e...

9824 rows × 3 columns

Gambar 4. 17 Hasil dari tahapan removing duplicate

d. Slang Word Standaridization

Tahap *Slang Word Standaridzation* akan menyeleraskan kata pada data yang tidak sesuai dengan kamus besar bahasa Indonesia. Sebelum melalui tahap *Slang Word Standaridization* maka diperlukan *library* yang akan digunakan yang ditunjukkan pada gambar 4.16.

```
import pandas as pd
import numpy as np
import re
import string
import csv
```

Gambar 4. 18 import library tahap slang word standaridization

Pada proses ini tentunya memerlukan data pendukung yang didalamnya menyediakan pengubahan kata yang termasuk dalam kata tidak baku menjadi baku sesuai dengan bahasa Indonesia. Maka dari itu pada tahap ini terdapat data

pendukung berupa beberapa Kumpulan kata yang termasuk dalam kata yang tidak baku menjadi baku dalam format teks dan juga excel. Berikut merupakan *code* proses *import* data pendukung yang ditunjukkan pada gambar 4.17.

```
slang_dictionary = pd.read_csv('colloquial-indonesian-lexicon.csv')
slang_dict = pd.Series(slang_dictionary['formal'].values,index=slang_dictionary['slang']).to_dict()

slang_dictionary1 = pd.read_csv('kbba.txt', sep='\t')
slang_dict1 = pd.Series(slang_dictionary1['tujuan'].values, index=slang_dictionary1['7an'].to_dict())

slang_dictionary2 = pd.read_csv('slangword.txt', sep=':')
slang_dict2 = pd.Series(slang_dictionary2['dan'].values, index=slang_dictionary2['&'].to_dict())

slang_dictionary3 = pd.read_csv('formalizationDict.txt', sep='\t')
slang_dict3 = pd.Series(slang_dictionary3['tujuan'].values, index=slang_dictionary3['7an'].to_dict())
```

Gambar 4. 19 import data pendukung

Pada proses *Slang Word Standaridization* menggunakan 4 file sebagai data pendukung dari pengubahan kata yang tidak baku menjadi baku. `pd.read_csv('colloquial-indonesian-lexicon.csv')` digunakan untuk membaca file CSV yang bernama `'colloquial-indonesian-lexicon.csv'` dan menyimpannya dalam DataFrame `slang_dictionary` . Kemudian code tersebut berlaku pada ada file berikutnya seperti pada "*kbba.txt*" dan "*slangword.txt*" sebagai file kedua dan ketiga serta file keempat yaitu "*formalizationDict.txt*". Berikut merupakan hasil dari

import data pendukung dari file kesatu hingga keempat.

```
slang_dictionary.head()
```

	slang	formal	In-dictionary	context	category1	category2	category3
0	woww	wow	1	wow	elongasi	0	0
1	aminn	amin	1	Selamat ulang tahun kakak tulus semoga panjang...	elongasi	0	0
2	met	selamat	1	Met hari netaas kakt? Wish you all the best @t...	abreviasi	0	0
3	netaas	menetas	1	Met hari netaas kakt? Wish you all the best @t...	afiksasi	elongasi	0
4	keberpa	keberapa	0	Birthday yg keberpa kak?	abreviasi	0	0

```
slang_dictionary1.head()
```

	7an	tujuan
0	@	di
1	ababil	abg labil
2	abis	habis
3	acc	accord
4	ad	ada

Gambar 4. 20 hasil import data pendukung pertama dan kedua

```
slang_dictionary2.head()
```

	&	dan
0	+	tambah
1	/	atau
2	=	sama dengan
3	ababil	anak ingusan
4	abal2	palsu

```
slang_dictionary3.head()
```

	7an	tujuan
0	@	di
1	ababil	abg labil
2	abis	habis
3	acc	accord
4	ad	ada

Gambar 4. 21 hasil import data pendukung ketiga dan keempat

Dari hasil *import* file data pendukung diatas maka tahap selanjutnya yaitu proses pengubahan kata tidak baku menjadi baku berdasarkan keempat file data pendukung diatas menggunakan *source code* dengan fungsi *def* pada gambar 4.20.

```
def Slangwords(text):
    for word in text.split():
        if word in slang_dict.keys():
            text = text.replace(word, slang_dict[word])
            text = re.sub('@[\w]+','',text)
    return text

def Slangwords1(text):
    for word in text.split():
        if word in slang_dict1.keys():
            text = text.replace(word, slang_dict1[word])
            text = re.sub('@[\w]+','',text)
    return text

def Slangwords2(text):
    for word in text.split():
        if word in slang_dict2.keys():
            text = text.replace(word, slang_dict2[word])
            text = re.sub('@[\w]+','',text)
    return text

def Slangwords3(text):
    for word in text.split():
        if word in slang_dict3.keys():
            text = text.replace(word, slang_dict3[word])
            text = re.sub('@[\w]+','',text)
    return text
```

Gambar 4. 22 Source code pertama implementasi tahap slang word standaridization

Fungsi `Slangwords` di atas menerima teks sebagai input dan mengganti kata-kata slang dalam teks tersebut dengan kata-kata formal yang sesuai berdasarkan kamus `slang_dict`. Kemudian pemecahan teks dilakukan dengan menggunakan `'text_split'`. Jika sebuah kata slang ditemukan dalam teks, maka kata tersebut diganti dengan kata formal yang sesuai berdasarkan kamus `slang_dict` dengan menggunakan *source code* `text.replace(word, slang_dict[word]`.

Selanjutnya fungsi tersbut dijalankan juga pada data pendukung yang lainnya. Selain dengan menggunakan keempat file diatas sebagai proses data pendukung tahap *Slang Word Standaridization*

maka menggunakan pula beberapa pendeteksian typo secara manual pada tahap ini. Berikut dibawah ini merupakan *source* yang digunakan pada gambar 4.21.

```
my_df['text_slang'] = my_df['text_clean'].apply(Slangwords)
my_df['text_slang'] = my_df['text_clean'].apply(Slangwords1)
my_df['text_slang'] = my_df['text_clean'].apply(Slangwords2)
my_df['text_slang'] = my_df['text_clean'].apply(Slangwords3)
my_df['text_slang'] = my_df['text_clean'].str.replace('toling', 'tolong')
my_df['text_slang'] = my_df['text_clean'].str.replace('g', 'ngga')
my_df['text_slang'] = my_df['text_clean'].str.replace('mandir', 'mandiri')
my_df['text_slang'] = my_df['text_clean'].str.replace('trsfrx', 'transfer')
my_df['text_slang'] = my_df['text_clean'].str.replace('passw', 'password')
my_df['text_slang']
```

Gambar 4. 23 Source code kedua implementasi tahap slang word standaridization

Dari beberapa tahapan proses *Slang Word Standaridization* maka hasilnya ditunjukan pada gambar 4.22.

```
0       pengguna udah dari tahun  tambah kesini malah ...
1       aplikasinya bagus tapi sekarang eror tidak bis...
2       kenapa untuk menuju tap login sering gagal beg...
3       tambahin menu pembayaran via virtual account n...
4       kenapa sekarang sering muncul setting ulang pe...
                              ...
9819                        luar biasa praktis dan nyaman
9820    saya beli pulsa sebesar rb tapi tidak masuk  d...
9821    setelah update pembaharuan terbaru kok aplikas...
9822    begitu daftar livin ada wa tdk jelas dr nomer ...
9823    tolong untuk beberapa opsi seperti pengisian e...
Name: text_slang, Length: 9824, dtype: object
```

Gambar 4. 24 hasil tahapan slang word standaridization

e. Stopword Removal

Dalam tahap *stopword removal* ini akan terjadi proses penghapusan kata didalam sebuah data yang tidak mengandung arti yang bermakna. Namun sebelum berlanjut pada tahap *stopword removal*

maka diperlukannya pengisntallan sebuah *library* yang akan digunakan yaitu *library* nlp_id. Fungsi dari penginstallan *library* tersebut yaitu mempermudah analisis teks dalam bahasa Indonesia. Berikut merupakan proses penginstallan *library* nlp_id pada gambar 4.23.

```
: pip install nlp-id
```

Gambar 4. 25 penginstallan library nlp-id

Apabila proses penginstallan telah selesai dilakukan maka selanjutnya menimport dan menggunakan modul '`Stopword`' dari puustaka '`nlp_id`' uuntuk mengelola daftar kata – kata stopword berbahasa Indonesia. Ditunjukkan pada gambar 4.24.

```
: from nlp_id.stopword import StopWord
  stopword = StopWord()
```

Gambar 4. 26 import library nlp-id

Setelah penginstallan dan pemanggilan kelas maka dilanjutkan penambahan kolom baru dalam DataFrame '`my_df`' yang didalamnya mengandung teks setelah kata stopword dihapus. m kolom `'text_slang'` dari DataFrame diambil, dan fungsi `stopword.remove_stopword()` diterapkan ke setiap entri untuk menghapus kata-kata stopword

dari teks tersebut. Setelah itu, hasilnya disimpan dalam kolom baru yang dinamai `'text_word_removal'` seperti dibawah ini yang diyunjukkan pada gambar 4.25.

```
my_df['text_word_removal'] = my_df['text_slang'].apply(stopword.remove_stopword)
my_df['text_word_removal']
```

Gambar 4. 27 Source code tahap stopword removal

Lalu hasil dan tahap *stopword removal* ditunjukan padda gambar 4.26.

```
0       pengguna kesini parah jam otomatis transaksi n...
1       aplikasinya bagus eror pakai transaksi pengatu...
2       tap login gagal login bedanya ok fresh dibagia...
3       tambahin menu pembayaran virtual account min r...
4       muncul setting ulang pengaturan jam otomatis s...
                              ...
9819                                    praktis nyaman
9820    beli pulsa sebesar rb masuk laporan email suks...
9821    update pembaharuan terbaru aplikasi gak buka f...
9822    daftar livin wa tdk dr nomer tdk dikenal tlp s...
9823    tolong opsi pengisian e wallet nomonalnya dipa...
Name: text_word_removal, Length: 9824, dtype: object
```

Gambar 4. 28 hasil tahap stopword removal

f. Stemming

Tahap *stemming* yaitu tahap penghapusan imbuhan yang terdapat pada kata sehingga akan berubah menjadi kata dasar. Dalam menjalankan tahap *stemming* ini maka diperlukan penginstallan sebuah *library* yang akan digunakan yaitu *library* sastrawi. Berikut merupakan proses penginstallan *library* sastrawi yang ditunjukkan pada gambar 4.27.

```
pip install sastrawi
```

Gambar 4. 29 Penginstallan library sastrawi

Kemudian apabila proses penginstallan *library* telah selesai dilakukan maka tahap selanjutnya yaitu pemanggilan kelas `stemmerfactory` yang dibuat untuk membuat stemmer seperti pada gambar 4.28.

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
```

Gambar 4. 30 Pemanggilan library sastrawi

Setelah penginstallan dan pemanggilan kelas maka selanjutnya adalah menjalankan proses *stemming* itu sendiri seperti pada gambar 4.29.

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

```
my_df['text_stem'] = my_df['text_word_removal'].apply(stemmer.stem)
my_df['text_stem'].head()
```

Gambar 4. 31 Source code tahap stemming

Penggunaan metode '`apply`' pada kolom '`text_word_removal`' dari DataFrame yang diambil. Selanjutnya fungsi `stemmer.stem()` diterapkan ke setiap entri untuk melakukan stemming, yaitu mengubah kata-kata dalam teks menjadi kata dasarnya. Hasil dari tahapan *stemming* ini ditunjukkan pada gambar 4.30.

```
0    guna kesini parah jam otomatis transaksi ngga ...
1    aplikasi bagus eror pakai transaksi atur jam o...
2    tap login gagal login beda ok fresh bagi teran...
3    tambahin menu bayar virtual account min ribet ...
4    muncul setting ulang atur jam otomatis setting...
Name: text_stem, dtype: object
```

Gambar 4. 32 Hasil tahap stemming

g. Tokenizing

Tahap *tokenizing* yaitu tahap yang akan memecah sebuah teks menjadi kata perkata. Didalam fungsi ini menggunakan parameter '`teks`' yang merupakan pemecahan menjadi token. Penggunaan '`split`' yang membagi menjadi token yang kemudian diambahkan kedalam '`list_teks`' menjadi loop '`for`'. Dibawah ini merupakan *souce code* dari tahap *tokenizing* pada gambar 4.31.

```
def tokenize(teks):
    list_teks = []
    for txt in teks.split(" "):
        list_teks.append(txt)
    return list_teks
```

```
my_df['text_clean_tokenize'] = my_df['text_stem'].apply(tokenize)
my_df['text_clean_tokenize'].head()
```

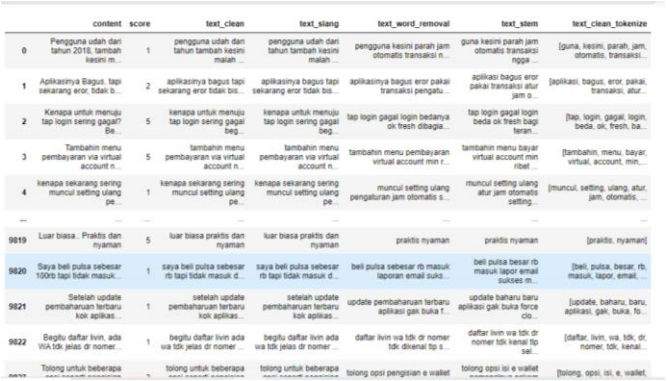
Gambar 4. 33 Source code tahap tokenizing

Kemudian hasil yang diperoleh dari tahap *tokenizing* diatas ditunjukkan pada gambar 4.32 dibawah ini.

```
0    [guna, kesini, parah, jam, otomatis, transaksi...
1    [aplikasi, bagus, eror, pakai, transaksi, atur...
2    [tap, login, gagal, login, beda, ok, fresh, ba...
3    [tambahin, menu, bayar, virtual, account, min,...
4    [muncul, setting, ulang, atur, jam, otomatis, ...
Name: text_clean_tokenize, dtype: object
```

Gambar 4. 34 hasil tahap tokenizing

Setelah melalui proses tahapan *preprocessing* maka hasil yang akan didapatkan yaitu sebuah data yang sudah dibersihkan dan siap digunakan. Berikut merupakan data yang telah melalui proses tahapan *preprocessing* ditunjukkan pada gambar 4.33.



	content	score	text_clean	text_slang	text_word_removal	text_stem	text_clean_tokenize
0	Pengguna udah dari tahun 2018, tambah kesini m...	1	pengguna udah dari tahun tambah kesini malah ...	pengguna udah dari tahun tambah kesini malah ...	pengguna kesini parah jam otomatis transaksi n...	guna kesini parah jam otomatis transaksi ngga ...	[guna, kesini, parah, jam, otomatis, transaksi...
1	Aplikasinya Bagus, tapi sekarang eror, tidak b...	2	aplikasinya bagus tapi sekarang eror tidak bis...	aplikasinya bagus tapi sekarang eror tidak bis...	aplikasinya bagus eror pakai transaksi pengatu...	aplikasi bagus eror pakai transaksi atur jam o...	[aplikasi, bagus, eror, pakai, transaksi, atur...
2	Kenapa untuk menuju tap login sering gagal? Be...	5	kenapa untuk menuju tap login sering gagal beg...	kenapa untuk menuju tap login sering gagal beg...	tap login gagal login bedanya ok fresh dibagia...	tap login gagal login beda ok fresh bagi feran...	[tap, login, gagal, login, beda, ok, fresh, ba...
3	Tambahin menu pembayaran via virtual account n...	5	tambahin menu pembayaran via virtual account n...	tambahin menu pembayaran via virtual account n...	tambahin menu pembayaran virtual account min r...	tambahin menu bayar virtual account min ribel ...	[tambahin, menu, bayar, virtual, account, min,...
4	kenapa sekarang sering muncul setting ulang pe...	1	kenapa sekarang sering muncul setting ulang pe...	kenapa sekarang sering muncul setting ulang pe...	muncul setting ulang pengaturan jam otomatis s...	muncul setting ulang atur jam otomatis setting...	[muncul, setting, ulang, atur, jam, otomatis, ...
...	...	...	...	...	...	...	...
9819	Luar biasa.. Praktis dan nyaman	5	luar biasa praktis dan nyaman	luar biasa praktis dan nyaman	praktis nyaman	praktis nyaman	[praktis, nyaman]
9820	Saya beli pulsa sebesar 100rb tapi tidak masuk...	1	saya beli pulsa sebesar rb tapi tidak masuk d...	saya beli pulsa sebesar rb tapi tidak masuk d...	beli pulsa sebesar rb masuk laporan email suks...	beli pulsa besar rb masuk lapor email sukses m...	[beli, pulsa, besar, rb, masuk, lapor, email, ...
9821	Setelah update pembaharuan terbaru kok aplikas...	1	setelah update pembaharuan terbaru kok aplikas...	setelah update pembaharuan terbaru kok aplikas...	update pembahanuan terbaru aplikasi gak buka f...	update baharu baru aplikasi gak buka force clo...	[update, baharu, baru, aplikasi, gak, buka, fo...
9822	Begitu daftar livin, ada WA tdk jelas dr nomer...	1	begitu daftar livin ada wa tdk jelas dr nomer ...	begitu daftar livin ada wa tdk jelas dr nomer ...	daftar livin wa tdk dr nomer tdk dikenal tlp s...	daftar livin wa tdk dr nomer tdk kenal tlp sel...	[daftar, livin, wa, tdk, dr, nomer, tdk, kenal...
	Tolong untuk beberapa		tolong untuk beberapa	tolong untuk beberapa	tolong opsi pengisian e wallet	tolong opsi isi e wallet	[tolong, opsi, isi, e, wallet,

Gambar 4. 35 hasil prepocessing

C. Labeling Ulasan

Didalam dataset sebelumnya belum adanya pelabelan pada setiap ulasan komentar sehingga perlu dilakukannya proses pemberian label yaitu agar mengetahui termasuk dalam label negatif, positif, maupun netral. Didalam penelitian ini penulis proses pelabelan yang dilakukan yaitu dengan menggunakan teknik *lexicon based* dimana pelabelan tersebut berdasarkan pada kamus *lexicon* yang memiliki nilai positif, negatif dan netral. Namun apabila terdapat sebuah kata yang tidak termasuk kedalam kata

yang bernilai negatif atau positif maka kata tersebut termasuk kedalam kata yang netral. Dengan menggunakan teknik ini dilihat dari jumlah kata akan diberikan score berdasarkan kamus *lexicon* itu sendiri terprediksi menjadi sebuah ulasan yang negatif, positif dan netral (Ismail *et al*., 2023).

Berikut merupakan *dictionary* kosa kata yang berkonotasi positif dan *dictionary* kosa kata yang berkonotasi negatif.

- *Dictionary* kosa kata positif: https://raw.githubusercontent.com/masdevid/ID-OpinionWords/master/positive.txt
- *Dictionary* kosa kata negatif: https://raw.githubusercontent.com/masdevid/ID-OpinionWords/master/negative.txt

Setelah *Dictionary* kosa kata telah disiapkan kemudian proses selanjutnya yaitu melabeli pada setiap ulasan komentar. Proses pelabelan ini akan akan dilakukan dengan cara memberikan score pada kata disetiap ulasan berdasarkan dengan kamus *lexicon* berbahasa Indonesia. Pada proses pelabelan ini nilai sentimen ini akan dibagi menjadi tigas kelas yaitu sentiment negatif, sentimen positif dan juga netral. Penilaian pada setiap ulasan berbeda- beda, nilai yang dihasilkan pada tiap sentimen dihasilkan dari hasil penjumlahan total nilai

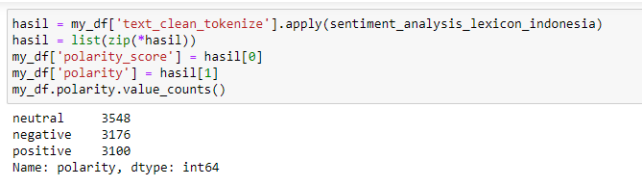
dari kata positif yang ada pada keseluruhan kalimat ulasan yang kemudian dikurangi dengan nilai total kata negatif pada ulasan tersebut maka akan disebut nilai sentiment. Selanjutnya nilai sentimen diatas 0 (sentimen>=0) maka akan dinyatakan sebagai kelas positif, sedangkan apabila nilai sentiment dibawah 0 (sentiment <0) maka dinyatakan sebagai kelas negatif. Selain itu, apabila nilai sentiment sama dengan 0 (sentiment = 0) maka ulasan tersebut dinyatakan sebagai kelas netral(Ahmad, 2022). Pada penelitian ini menggunakan data sejumlah 10.000 ulasan, yang selanjutnya label ini akan diterapkan pada dataset dengan adanya *feature* baru. Pada gambar 4.34. menunjukkan *source code* untuk menjalankan proses pelabelan ulasan.

```
my_df_positive = pd.read_csv('https://raw.githubusercontent.com/masdevid/ID-OpinionWords/master/positive.txt', sep='\t')
list_positive = list(my_df_positive.iloc[:,0])

my_df_negative = pd.read_csv('https://raw.githubusercontent.com/masdevid/ID-OpinionWords/master/negative.txt', sep='\t')
list_negative = list(my_df_negative.iloc[:,0])

def sentiment_analysis_lexicon_indonesia(text):
    score = 0
    for word in text:
        if (word in list_positive):
            score += 1
    for word in text:
        if (word in list_negative):
            score -= 1
    polarity=''
    if (score > 0):
        polarity = 'positive'
    elif (score < 0):
        polarity = 'negative'
    else:
        polarity = 'neutral'
    return score, polarity
```

Gambar 4. 36 Proses pelabelan ulasan

Fungsi‘`sentiment_analysis_lexicon_indonesia`’ yang diterapkan pada kolom

'text_clean_tokenize' dari DataFrame 'my_df' yang menggunakan metode 'apply' hasilnya berupa skor sentiment dan polaritas untuk setiap kelas. Fungsi 'zip(*hasil)' ini untuk memisahkan skor sentiment dan polaritas kedalam dua list terpisah. Kemudian hasil dari proses pelabelan akan disimpan dan akan digabungkan dalam suatu dataframe yang diberi nama feature "polarity". Selanjutnya peneliti akan mengetahui hasil sentiment dari setiap ulasan baik itu ulasan positif, negatif dan juga netral yang telah didapatkan. Berikut merupakan hasil total masing – masing sentimen yang telah diperoleh yang ditunjukkan pada gambar 4.35.



```
hasil = my_df['text_clean_tokenize'].apply(sentiment_analysis_lexicon_indonesia)
hasil = list(zip(*hasil))
my_df['polarity_score'] = hasil[0]
my_df['polarity'] = hasil[1]
my_df.polarity.value_counts()
```

```
neutral     3548
negative    3176
positive    3100
Name: polarity, dtype: int64
```

Gambar 4. 37 menyimpan label dalam dataframe

Berdasarkan dari gambar diatas maka dihasilkan sentiment negatif sejumlah 3176, sentiment positif sejumlah 3100. Dan juga sentiment netral sejumlah 3548. Setelah dilakukan proses pembagian sentiment dari masing – masing kelas maka akan ditampilkan

ulasan yang telah selesai dilabeli dan serta mendapatkan polarity score berdasarkan proses labeling ulasan sebelumnya yang ditunjukkan pada gambar 4.38.

```
In [8]: my_df[['content', 'score', 'polarity_score', 'polarity']]
Out[8]:
```

	content	score	polarity_score	polarity
0	Pengguna udah dari tahun 2018, tambah kesini m...	1	-1	negative
1	Aplikasinya Bagus, tapi sekarang eror, tidak b...	2	2	positive
2	Kenapa untuk menuju tap login sering gagal? Be...	5	0	neutral
3	Tambahin menu pembayaran via virtual account n...	5	-1	negative
4	kenapa sekarang sering muncul setting ulang pe...	1	-1	negative
...	...	...	...	...
9819	Luar biasa.. Praktis dan nyaman	5	1	positive
9820	Saya beli pulsa sebesar 100rb tapi tidak masuk...	1	2	positive
9821	Setelah update pembaharuan terbaru kok aplikas...	1	1	positive
9822	Begitu daftar livin, ada WA tdk jelas dr nomer...	1	0	neutral
9823	Tolong untuk beberapa opsi seperti pengisian e...	3	2	positive

9824 rows × 4 columns

Gambar 4. 38 Hasil pelabelan

D. Ekstrasi Fitur

Didalam tahapan ekstraksi fitur diperlukannya sebuah *library* yang akan digunakan seperti library *sklearn* atau *library sckit-learn. Library* tersebut memiliki fungsi seperti membantu adanya proses *processing* sebuah data dan juga untuk melakukan *training* data sebagai kebutuhan dari *machine learning* dan *data science* (Pedregosa *et al*., 2011). Didalam tahap ekstraksi fitur ini menggunakan *library sklearn* seperti pada kelas *train_test_split*, *Label_encorder*, dan juga *TfidfVectorizer.*

Kemudian data yang sebelumnya akan ada proses *split validation* data yaitu proses pembagian data latih (*training*) dan data uji (*testing*) yang bertujuan untuk mempermudah proses klasifikasi. Pada proses pembagian data ini menggunakan data uji sebesar 20% yang digunakan dari jumlah data keseluruhan. Berikut merupakan proses pembagian data yang ditunjukkan pada gambar 4.36.

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split (my_df['text_stem'],my_df['polarity'],test_size = 0.2,random_state = 42)
```

Gambar 4. 39 Split validation data

Didalam tahap ekstraksi fitur, maka proses yang akan dilakukan oleh sistem dengan meng-*import* `Iabelencorder` untuk mengubah suatu teks menjadi angka untuk mempermudah dalam proses sitem pengklasifikasian. Selanjutnya proses TFIDF untuk melakukan ekstraksi fitur untuk mengetahui bobot dari masing – masing kata. Pada proses pembuatan *word vector* dan juga proses pembobotan kata dengan menggunakan `library sklearn` meng-*import* `TfidfVectorizer.` Berikut merupakan tampilan pembobotan TFIDF ditunjukkan pada gambar 4.37.

```
from sklearn.preprocessing import LabelEncoder
Encoder = LabelEncoder()
y_train = Encoder.fit_transform(y_train)
y_test = Encoder.fit_transform(y_test)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
Tfidf_vect = TfidfVectorizer(max_features=100)
Tfidf_vect.fit(my_df['text_stem'])
Train_X_Tfidf = Tfidf_vect.transform(X_train)
Train_X_Tfidf = Tfidf_vect.transform(X_test)
```

Gambar 4. 40 Proses pembobotan TFIDF

Sehingga hasil yang diperoleh dari tahapan pembobotan dengan menggunakan TFIDF yang menghasilkan kalimat yang menjadi kumpulan *array* yang menjadi sebuah matriks, didalam suatu baris mewakili dari setiap dokumen. Kemudian didalam satu kolom tersebut mewakili dari seluruh kata pada teks. Pada gambar 4.38 merupakan hasil TFIDF.

```
Train_X_Tfidf.toarray()
```

```
array([[0.        , 0.12701529, 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.42055498],
       ...,
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.36969054],
       [0.        , 0.15352436, 0.        , ..., 0.        , 0.        ,
        0.        ]])
```

Gambar 4. 41 Hasil Pembobotan TFIDF

Berikut merupakan contoh dari penelitian yang menggunakan tiga komentar untuk dilakukan pembobotan TFIDF secara manual yaitu sebagai berikut:

(Doc 1) = “luar biasa praktis dan nyaman”.

(Doc 2) = “terlalu sering harus di updet membuat penggunaan terhambat tidak ada perubahan dalam aplikasi masih lebih suka versi lama”.

(Doc 3) = “aplikasinya dong upgrade lagi login jug suka eror masuk akun juga susah”.

Kemudian dibawah ini merupakan hasil preprocessing dari dokumen diatas:

(Doc1) = "['praktis', 'nyaman']"

(Doc 2) ="['updet', 'pengunaan', 'hambat', 'ubah', 'aplikasi', 'suka', 'versi']"

(Doc 3) = "['aplikasi', 'upgrade', 'login', 'suka', 'eror', 'masuk', 'akun', 'susah']"

Proses selanjutnya yaitu perhitungan yang menggunakan metode TFIDF *aray* membentuk word vector yang akan diberi pembobotan nilai. Didalam proses TFIDF ini memiliki dua kata yaitu TF dan IDF. Fungsi dari TF sendiri yaitu untuk menjumlah seluruh kata pada setiap dokumen, kemudian jika IDF ini memiliki fungsi untuk mengurangi dari bobot pada kata apabila didalam suatu dokumen terdapat banyak. Untuk memulai perhitungan TFDIF diawali dengan menghitung TF terlebih dahulu. Berikut merupakan contoh perhitungan TF secara manual pada tabel 4.1.

Token	TF		
	Doc 1	Doc 2	Doc 3
praktis	1	0	0
nyaman	1	0	0
update	0	1	0
penggunaan	0	1	0
hambat	0	1	0
ubah	0	1	0
aplikasi	0	1	1
suka	0	1	1
versi	0	1	0
upgrade	0	0	1
login	0	0	1
eror	0	0	1
masuk	0	0	1
akun	0	0	1
susah	0	0	1

Tabel 4. 1 Contoh perhitungan TF

Berdasarkan tabel diatas maka nilai dari DF telah didapatkan dimana sebagai contoh pengunaan jumlah dokumen pada tabel yaitu sejumlah tiga komentar. Dengan demikian diketahui dokumen atau D = 3. Selanjutnya akan dilakukan perhitungan IDF dan TFIDF menggunakan rumus yang dijelaskan pada bab sebelumnya dan selanjutnya akan di implementasikan pada tabel 4.2.

Token	Df	D/Df (D=3)	IDF (log(D/Df))	IDF + 1	TF*IDF		
					D1	D2	D3
praktis	1	3	Log 3 = 0,477	1,477	1,477	0	0
nyaman	1	3	Log 3 = 0,477	1,477	1,477	0	0
update	1	3	Log 3 = 0,477	1,477	0	1,477	0
penggunaan	1	3	Log 3 = 0,477	1,477	0	1,477	0
hambat	1	3	Log 3 = 0,477	1,477	0	1,477	0
ubah	1	3	Log 3 = 0,477	1,477	0	1,477	0
aplikasi	2	1,5	Log 1,5 = 0,176	1,176	0	1,176	1,176
suka	2	1,5	Log 1,5 = 0,176	1,176	0	1,176	1,176
versi	1	3	Log 3= 0,477	1,477	0	1,477	0
upgrade	1	3	Log 3= 0,477	1,477	0	0	1,477
login	1	3	Log 3= 0,477	1,477	0	0	1,477
eror	1	3	Log 3= 0,477	1,477	0	0	1,477
masuk	1	3	Log 3= 0,477	1,477	0	0	1,477
akun	1	3	Log 3= 0,477	1,477	0	0	1,477
susah	1	3	Log 3= 0,477	1,477	0	0	1,477

Tabel 4. 2 Contoh perhitungan TF dan TFIDF

Dengan demikian setelah diperoleh nilai TFIDF yangdapat dituliskan dalam bentuk *array* akan menjadi sebagai berikut:

Array ([

[1.477, 1.477, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 1.477, 1.477, 1.477, 1.477, 1.176, 1.176, 1.147, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 1.176, 1.176, 0, 1.477, 1.477, 1.477, 1.477, 1.477, 1.477]

Berdasarkan hasil TFIDF diatas maka sudah mengintrepesentasikan pada setiap ketiga dokumen diatas. Kemudian dari setiap kolom pada dokumen tersebut sudah mengintrepesentasikan dari setiap kata yang ada pada seluruh teks.

E. Klasifikasi *Random Forest*

Pada tahap klasifikasi yang menggunakan metode *random forest* ini menggunakan dataset yang sebelumnya telah selesai pada tahap preprocessing dan juga ekstraksi fitur. Adanya data latih sebesar 80% yang selanjutnya akan melalui proses pengujian yang menggunakan data uji yang berfungsi untuk melakukan pengujian pada suatu sistem didalam mengklasifikasikan sebuah data. Dalam menjalankan tahap pengklasifikasian maka diperlukan *libraray* seperti *library sklearn* atau *sckit learn*. Kemudian diperlukan *RandomForestClassifier* dengan menggunakan algoritma pembelajaran ensamble yang digunakan untuk melakukan klasifikasi pohon keputusan. Selain itu menginmport mfungsi *RandomizedSearchCV* yang digunakan untuk pencarian *hyperparameter* secara acak untuk memberikan kinerja terbaik*, Integer, Real* yang digunakan untuk digunakan untuk mendefinisikan ruang pencarian untuk penyetelan *hiperparameter*.

Proses yang dilakukan yaitu menyiapkan *library sklearn.* Apabila sebelumnya telah melakukan penginstallan maka yang diperlukannya yaitu pengimportan *library* dan juga melakukan pemanggilan kelas yang akan dibutuhkan. Pada gambar dibawah ini akan ditunjukkan proses pengimportan *library sklearn* didalam proses klasifikasi pada gambar 4.39.

```
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV
from jcopml.tuning.space import Integer, Real
```

Gambar 4. 42 Import Library sklearn untuk proses klasifikasi

Kemudian selanjutnya akan dilakukan proses pengklasifikasian dengan menggunakan metode *random forest* . Berikut merupakan *source code* untuk melakukan proses klaisifikasi *random forest* yang ditunjukkan pada gambar 4.40.

```
pipeline_rf = Pipeline([
    ('prep', TfidfVectorizer()),
    ('algo', RandomForestClassifier(random_state=42))
])

# Update the parameters for Random Forest
param_rf = {
    'algo__n_estimators': [50, 100, 200, 300],
    'algo__max_depth': [None, 10, 20, 30],
    'algo__min_samples_split': [2, 5, 10],
    'algo__min_samples_leaf': [1, 2, 4],
    'algo__bootstrap': [True, False]
}

# Create RandomizedSearchCV for Random Forest
model_rf = RandomizedSearchCV(pipeline_rf, param_rf, cv=4, n_iter=50, n_jobs=-2, verbose=1, random_state=42)
model_rf.fit(X_train, y_train)

# Print the best parameters and scores
print(model_rf.best_params_)
print(model_rf.score(X_train, y_train), model_rf.best_score_, model_rf.score(X_test, y_test))
```

Gambar 4. 43 Klasfikasi metode random forest

Fungsi `pipeline_rf =Pipeline([...]` untuk membuat sebuah pipeline untuk klasifikasi

menggunakan algoritma Random Forest. Selanjutnya `('prep')` dari pipeline adalah preprocessing menggunakan `TfidfVectorizer`, sementara langkah kedua `('algo')` adalah algoritma klasifikasi RandomForestClassifier. '`param_rf`' mendefinisikan ruang pencarian parameter untuk algoritma *Random Fores*t. Selanjutnya parameter yang akan dioptimalkan meliputi jumlah pohon (`n_estimators`), kedalaman maksimum pohon (`max_depth`), jumlah sampel minimum yang diperlukan untuk membagi simpul internal (`min_samples_split`), jumlah sampel minimum yang diperlukan untuk menjadi sebuah simpul daun (`min_samples_leaf`), dan apakah bootstrap digunakan dalam pembuatan pohon (bootstrap). Pembuatan objek RandomizedSearchCV digunakan untuk mencari parameter terbaik. Model ini dilatih menggunakan data latih '(`X_train dan y_train)',` dan hasilnya digunakan untuk mencetak parameter terbaik yang ditemukan dan skor akurasi dari model terhadap data latih, skor validasi silang terbaik, dan skor akurasi dari model terhadap data uji.

F. Pengujian Model

Setelah dilakukannya proses pengkalsifikasian maka selanjutnya akan dilakukan pengujian model yang berfungsi untuk menguji ketepatan dari model yang digunakan untuk proses pengklasifikasian dan jug akan didapatkan hasil dari klasifikasi yang akan diperlihatkan melalui *multiclass confusion matrix.* Penelitian ini menggunakan tiga kelas yaitu kelas negatif , kelas netral dan juga kelas positif maka dari itu menggunakan *multiclass confusion matrix* 3x3 yang dapat dilihat pada tabel dibawah ini

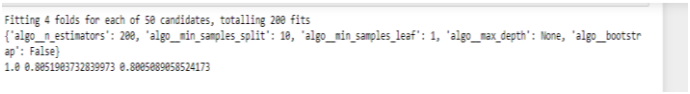
		Predicted Class		
		Negative	***Netral***	***Positif***
True class	*Negative*	T Neg	F NegNet	F NegPos
	Netral	F NetNeg	T Net	F Netos
	Positif	F Neg	F PosNet	T Pos

Tabel 4. 3 Multiclass confusion matrix 3x3

Selanjutnya akan dilakukan perhitungan nilai akurasi yaitu dengan menggunakan rumus dibawah ini :

$$Accuracy = \frac{\text{TPos} + \text{TNeg} + \text{TNet}}{jumlah\ data\ uji}$$

Maka hasil yang didapatkan dari pengujian model dengan menggunakan *multiclass confusion matrix* 3x3 seperti gambar dibawah ini.



Gambar 4. 44 Perolehan hasil akurasi

Berdasarkan hasil dari *multiclass confusion matrix* dapat disimpulkan bahwa pengujian model diperoleh akurasi sebesar 0.80 atau 80% . Setelah didapatkan nilai akurasi maka akan dilakukan proses evaluasi dari model yang digunakan.

G. Evaluasi Model

Tujuan dilakukannya evaluasi model yaitu untuk nilai dari performa pada model yang digunakan. Perhitungan performa ini meliputi nilai *acuraccy, precision, recall* dan juga *f1 score* berdasarkan *multiclass confusion matrix.* Pada tabel dibawah ini menunjukkan hasil dari *multiclass confusion matrix* 3x3.

		Predicted Class		
		Negative	***Netral***	***Positif***
True Class	*Negative*	569	69	15
	Netral	120	479	95
	Positif	29	64	525

Tabel 4. 4 Hasil Multiclass confusion matrix 3x3

Setelah didapatkan nilai dari *multiclass confusion matrix* maka akan dilakukan proses perhitungan performa dari model yang digunakan secara manual. Hasil dari perhitungan *akurasi,* yang dilakukan secara manual akan dijelaskan pada proses dibawah ini.

$$Accuracy = \frac{\text{TPos+TNeg+TNet}}{jumlah\ data\ uji} \text{ X } 100$$

$$Accuracy = \frac{151 + 479 + 525}{549 + 69 + 15 + 120 + 479 + 95 + 29 + 64 + 5} \text{ X } 100$$

$$Accuracy = \frac{1553}{1944} \text{ X } 100$$

$$Accuracy = 0.80 \text{ X } 100$$

$$Accuracy = 80\%$$

Setelah diketahui nilai akurasi nya maka selanjutnya yaitu melakukan perhitungan terhadap performal model yang lainnya seperti nilai dari *precission, recall*, dan juga *f1 score* berdasarkan nilai *true positif*, nilai *false positif*, dan juga nilai *false negatif*. Karena dalam menentukan nilai *true positif*, nilai *false positif*, dan juga nilai *false negatif* pada *multiclass confusion matrix* 3x3 sangat sulit maka solusi yang diberikan untuk mempermudah pencarian masing – masing nilai yaitu dengan memcah kolom 3X3 menjadi kolom 2X2 terlebih dahulu yang akan ditunjukkan pada tabel tabel dibawah ini:

a. Kelas Negatif

True\predic	Negatif	Bukan Negatif
Negatif	TP = 569	FN = 69 + 15 = 84
Bukan Negatif	FP = 120 + 29 = 149	TN = 479 + 95 + 64 + 525 = 731

Tabel 4. 5 Perhitungan kelas negatifl

b. Kelas Netral

True\predic	Netral	Bukan Negatif
Netral	TP = 479	FN = 120 + 95 = 215
Bukan Netral	FP = 69 + 64 = 133	TN = 569 + 15 + 29 + 64 + 525 = 1202

Tabel 4. 6 Perhitungan kelas netral

c. Kelas Positif

True\predic	Positif	Bukan Positif
Positif	TP = 525	FN = 29 + 64 = 93
Bukan Positif	FP = 15 + 95 = 110	TN = 569 + 69 + 120 + 479 = 1237

Tabel 4. 7 Perhitungan kelas positif

Berdasarkan beberapa tabel diatas maka diperoleh nilai *true* positif, *true* negatif, *false* positif dan juga *false* negatif. Sehingga akan mempermudah pencarian nilai *precission, recall*, dan juga *f1 score* dengan menggunakan

rumus yang telah dijelaskan pada bab sebelumnya. Berikut merupakan hasil dari perhitungan yang diperoleh pada tabel 4.8.

Kelas	TP	FP	FN	*Precision*	*Recall*	*F1 Score*
Negatif	569	149	84	79%	87%	82%
Netral	479	133	215	78%	69%	73%
Positif	525	110	93	82%	85%	84%

Tabel 4. 8 Hasil perhitungan performa

Berdasarkan tabel 4.8 perolehan nilai pada kelas negatif *precission* sebesar 79%, recall sebesar 87%, f1 score sebesar 82%. Pada kelas netral memperoleh nilai *precission* sebesar 78%, *recall* sebesar 69%, *f1 score* sebesar 85%. Pada kelas positif memperoleh nilai *precission* sebesar 82%, *recall* sebesar 85%%, *f1 score* sebesar 84%.

Setelah didapatkan nilai dari tabel 4.8 maka akan dilakukan proses perhitungan *precission, recall dan f1 score* dari model yang digunakan secara manual. Hasil dari perhitungan yang dilakukan secara manual akan dijelaskan pada proses dibawah ini.

Perhitungan *precission* secara manual :

$$Precission = \frac{\text{TP}}{TP + FP} X100$$

$$Precission = \frac{569 + 479 + 525}{569 + 479 + 525 + 149 + 133 + 110} X100$$

$$Precission = \frac{1573}{1965} X100$$

$$Precission = 0{,}800 \text{ x } 100$$

$$Precission = 80\%$$

Perhitungan *recall* secara manual :

$$Recall = \frac{\text{TP}}{TP + FN} X100$$

$$Recall = \frac{569 + 479 + 525}{569 + 479 + 525 + 84 + 215 + 93} X100$$

$$Recall = \frac{1573}{1965} X100$$

$$Recall = 0{,}800 \text{ x } 100$$

$$Recall = 80\%$$

Perhitungan *F1 Score* secara manual:

$$f1\ score = \frac{2. Precission. Recall}{Precission + Recall} X100$$

$$f1\ score = \frac{2\ .\ 0{,}800\ .\ 0{,}800}{0{,}800\ a = 0{,}800} X100$$

$$f1\ score = \frac{1{,}28}{1{,}6} X100$$

$$f1\ score = 0{,}80 \text{ x } 100$$

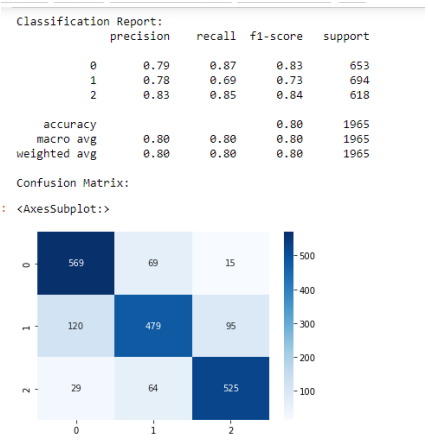
$$f1\ score = 80\%$$

Adapun perhitungan nilai accuracy, *precission, recall*, dan juga *f1 score* yang dilakukan melalui sistem ditunjukan pada gambar 4.41.

```
from sklearn.metrics import classification_report, confusion_matrix

# Menggunakan model terbaik untuk membuat prediksi pada data uji
y_pred = model_rf.predict(X_test)

# Membuat classification report
print("Classification Report:")
print(classification_report(y_test, y_pred))

# Membuat confusion matrix
conf_mat = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
sns.heatmap(conf_mat, annot=True, fmt='d', cmap='Blues', cbar=True)
```

Gambar 4. 45 Source code multiclass confusion matrix

Penggunaan `'y_pred = model_rf.predict(X_test)'` ini digunakan untuk pencarian parameter terbaik yang telah dilatih sebelumnya dan `'conf_mat = confusion_matrix(y_test, y_pred)'` perhitungan *confusion matrix* dengan menggunakan fungsi *'confusion matrix'*. *'sns.heatmap(conf_mat, annot=True, fmt='d', cmap='Blues', cbar=True)* digunakan untuk proses visualisasi confusion matrix dalam bentuk heatmap menggunakan pustaka Seaborn (sns). Hasil *classification report* dan *confusion matrix* yang ditunjukkan pada gambar 4.42.



Gambar 4. 46 Hasil perhitungan performa

Hasil pengukuran evaluasi performa yang dihasilkan dari model *random forest* pada setiap kelasnya memperoleh nilai yang baik. Dengan demikian dapat disimpulkan bahwa tingkat keberhasilan dari sistem tersebut untuk mencari ketepatan informasi yang didminta (*precision)* pada kelas negatif sebesar 79%, kelas netral sebesar 78% dan pada kelas positif sebesar 82% sehingga dapat disimpulkan bahwa proposi label yang diprediksi positif memiliki nilai yang tinggi dibandingkan pada kelas negatif dan netral. Selanjutnya tingkat keberhasilan dari sistem tersebut untuk menemukan kembali informasi *(recall)* pada kelas negatif memperoleh nilai sebesar 87%, kelas positif sebesar 69% dan kelas positif sebesar 84% sehingga dapat disimpulkan bahwa

proposi label yang diprediksi netral memiliki nilai yang rendah dibandingkan pada kelas negatif dan positif. Maka diperoleh total keseluruhan nilai *precission* sebesar 80%, *recall* 80%, dan juga *f1 score* 80%.

H. Visualisasi

Setelah didapatkan nilai *accuracy*, *precission*, *recall* dan juga *f1 score* maka selanjutnya yaitu mengimplementasikan hasil dari sentiment kedalam bentuk *wordcloud.* Tujuan dari pengimplementasian terhadap *wordcloud* yaitu untuk mengetahui dari jumlah sentimen serta kata yang sering diberikan pada ulasan aplikasi oleh pengguna yang informasinya akan ditampilkan dalam bentuk *wordcloud.*

Gambar 4. 47 Wordcloud pada data ulasan

Pada gambar dibawah ini merupakan kata yang sering diberikan pada ulasan aplikasi livin' by mandiri seperti "transaksi","update","aplikasi" dan masih banyak yang lainnya.

Selanjutnya ada juga *wordcloud* yang ditampilkan berdasarkan sentimen negatif dan juga positif yang ditunjukkan pada gambar dibawah ini.

Gambar 4. 48 Wordcloud sentimen positif dan negatif

Berdasarkan hasil implementasi *wordcloud* pada sentimen posistif kata yang sering digunakan pada ulasan oleh pengguna aplikasi livin' by mandiri seperti "transaksi","aplikasi","mudah", "bagus", "mandiri", dan yang lainnya. Sedangkan pada sentimen negatif "transaksi","kecewa","ribet" ,"update" dan kata – kata yang lainnya.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil pembahasan dari penelitian yang dilakukan oleh peneliti dapat disimpulkan sebagai berikut:

1. Penggunaan Metode *Random Forest* dalam melakukan klaisifikasi pada ulasan Aplikasi Livin' by mandiri dengan menggunakan 10.000 ulasan dapat menganalisis sentiment dengan hasil klasifikasi cukup baik. Proses yang diawali dengan tahapan *preprocessing,* kemudian melabeli ulasan, dan melakukan pembobotan dengan menggunakan TFIDF. Klasifikasi dengan menggunakan metode *Random Forest* dilanjutkan dengan pengujian serta evaluasi yang menghasilkan tiga kelas yaitu negatif, netral dan positif.
2. Hasil penerapan *Random Forest* pada klasifikasi ulasan aplikasi livin' by mandiri mendapat nilai akurasi sebesar *precission* sebesar 80%, *recall* 80%, dan juga *f1 score* 80%.
3. Hasil yang diperoleh pada penelitian ini bisa dijadikan sebagai bahan acuan untuk tetap menjaga kualitas serta untuk mengetahui persepsi

dari pengguna aplikasi livin' by mandiri dan bisa dijadikan sebagai evaluasi menjadi lebih baik.

B. Saran

Berdasarkan hasil penelitian yang sudah dilaksanakan, peneliti menyadari bahwa masih terdapat kekurangan sehingga pada penelitian berikutnya bisa dikembangkan kembali. Adapun saran yang bisa diberikan seperti dibawah ini:

1. Menerapkan penggunaan algoritma yang lainnya seperti *Decission Tree*, *K-Means Clustering*, *Support Vector Machines (SVM*) atau beberapa algoritma yang lainnya agar menghasilkan performa agar dapat dilakukan perbandingan untuk mengetahui klasifikasi terbaik.
2. Didalam penelitian ini teknik pelabelan yang digunakan yaitu menggunakan teknik *lexicon based* yang hanya mengandalkan sebuah kamus kata negatif dan positif sehingga pada penelitian berikutnya bisa menggunakan teknik pelabelan yang lain seperti teknik manualisasi ataupun teknik rating.

DAFTAR PUSTAKA

Afdhal, I., Kurniawan, R., Iskandar, I., Salambue, R., Budianita, E., & Syafria, F. (2022). Penerapan Algoritma Random Forest Untuk Analisis Sentimen Komentar Di YouTube Tentang Islamofobia. *Jurnal Nasional Komputasi Dan Teknologi Informasi*, *5*(1), 122–130. http://ojs.serambimekkah.ac.id/jnkti/article/view/4004/pdf

Afemi, M. P. (2022). *DETEKSI SPAMMER POLITIK PADA TWITTER DI INDONESIA MENGGUNAKAN RANDOM FOREST PROGRAM STUDI MATEMATIKA 2022 M / 1443 H DETEKSI SPAMMER POLITIK PADA TWITTER DI*.

Ahmad, F. (2022). Analisis sentimen terhadap ulasan aplikasi. *ANALISIS SENTIMEN TERHADAP ULASAN APLIKASI MOBILE MENGGUNAKAN METODE SUPPORT VECTOR MACHINE (SVM) DAN PENDEKATAN LEXICON BASED Diajukan*.

Aldean, M. Y., Paradise, P., & Setya Nugraha, N. A. (2022). Analisis Sentimen Masyarakat Terhadap Vaksinasi Covid-19 di Twitter Menggunakan Metode Random Forest Classifier (Studi Kasus: Vaksin Sinovac). *Journal of Informatics, Information System, Software Engineering and Applications (INISTA)*, *4*(2), 64–72. https://doi.org/10.20895/inista.v4i2.575

Alun Sujjadaa, Somantri, Juwita Nurfazri Novianti, & Indra Griha Tofik Isa. (2023). Analisis Sentimen Terhadap Review Bank Digital Pada Google Play Store Menggunakan Metode Support Vector Machine (Svm). *Jurnal Rekayasa Teknologi Nusa Putra*, *9*(2), 122–135. https://doi.org/10.52005/rekayasa.v9i2.345

Anggina, S., Setiawan, N. Y., & Bachtiar, F. A. (2022). Analisis Ulasan Pelanggan Menggunakan Multinomial Naïve Bayes Classifier dengan Lexicon-Based dan TF-IDF Pada

Formaggio Coffee and Resto. *Is The Best Accounting Information Systems and Information Technology Business Enterprise This Is Link for OJS Us*, *7*(1), 76–90. https://doi.org/10.34010/aisthebest.v7i1.7072

Budianita, E., Cynthia, E. P., Pranata, A., & Abimanyu, D. (2022). Pendekatan berbasis Machine Learning dan Leksikal Pada Analisis Sentimen. *Seminar Nasional Teknologi Informasi, Komunikasi Dan Industri (SNTIKI)*, 99–104. https://ejournal.uin-suska.ac.id/index.php/SNTIKI/article/view/19137

Budiman, A. E., & Widjaja, A. (2020). Analisis Pengaruh Teks Preprocessing Terhadap Deteksi Plagiarisme Pada Dokumen Tugas Akhir. *Jurnal Teknik Informatika Dan Sistem Informasi*, *6*(3), 475–488. https://doi.org/10.28932/jutisi.v6i3.2892

Diki Hendriyanto, M., Ridha, A. A., & Enri, U. (2022). Analisis Sentimen Ulasan Aplikasi Mola Pada Google Play Store Menggunakan Algoritma Support Vector Machine Sentiment Analysis of Mola Application Reviews on Google Play Store Using Support Vector Machine Algorithm. *Journal of Information Technology and Computer Science (INTECOMS)*, *5*(1), 1–7.

Fadilah, L. (2018). *Klasifikasi Random Forest pada Data Imbalanced Program Studi Matematika Universitas Islam Negeri Syarif Hidayatullah 2018 / 1439 H Klasifikasi Random Forest*.

Fitri, E. (2020). Analisis Sentimen Terhadap Aplikasi Ruangguru Menggunakan Algoritma Naive Bayes, Random Forest Dan Support Vector Machine. *Jurnal Transformatika*, *18*(1), 71. https://doi.org/10.26623/transformatika.v18i1.2317

Ismail, A. R., Bagus, R., & Hakim, F. (2023). *Implementasi Lexicon Based Untuk Analisis Sentimen Dalam Mengetahui*

Trend Wisata Pantai Di DI Yogyakarta Berdasarkan Data Twitter. *1*(1), 37–46.

Khoirul Insan, M. K., Hayati, U., & Nurdiawan, O. (2023). Analisis Sentimen Aplikasi Brimo Pada Ulasan Pengguna Di Google Play Menggunakan Algoritma Naive Bayes. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(1), 478–483. https://doi.org/10.36040/jati.v7i1.6373

Khoo, C. S. G., & Johnkhan, S. B. (2018). Lexicon-based sentiment analysis: Comparative evaluation of six sentiment lexicons. *Journal of Information Science*, *44*(4), 491–511. https://doi.org/10.1177/0165551517703514

Larasati, F. A., Ratnawati, D. E., & Hanggara, B. T. (2022). Analisis Sentimen Ulasan Aplikasi Dana dengan Metode Random Forest. *… Teknologi Informasi Dan …*, *6*(9), 4305–4313. http://j-ptiik.ub.ac.id

Majid, F. N., & Sulastri. (2023). Analisa Sentimen Aplikasi Pedulilindungi Dengan Metode Nbc Dan Svm. *Jurnal Elektronika Dan Komputer*, *16*(1), 100–108. https://journal.stekom.ac.id/index.php/elkom■ page100

Nurjannah, M., & Fitri Astuti, I. (2013). PENERAPAN ALGORITMA TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY (TF-IDF) UNTUK TEXT MINING Mahasiswa S1 Program Studi Ilmu Komputer FMIPA Universitas Mulawarman Dosen Program Studi Ilmu Komputer FMIPA Universitas Mulawarman. *Jurnal Informatika Mulawarman*, *8*(3), 110–113.

Onantya, I. D., & Adikara, P. P. (2019). Analisis Sentimen Pada Ulasan Aplikasi BCA Mobile Menggunakan BM25 Dan Improved K-Nearest Neighbor. *J-Ptiik.Ub.Ac.Id*, *3*(3), 2575–2580.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion,

B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Retnoningsih, E., & Pramudita, R. (2020). Mengenal Machine Learning Dengan Teknik Supervised Dan Unsupervised Learning Menggunakan Python. *Bina Insani Ict Journal*, *7*(2), 156. https://doi.org/10.51211/biict.v7i2.1422

Santoso, A. A., & Rachmawati, I. (2021). Analisis Minat Pengguna Layanan M-Banking Livin' by Mandiri di Indoneisa Menggunakan Model Modiffikasi UTAUT 2. *E-Proceeding of Management*, *8*(5), 4316–4322.

Studi, P., Informatika, T., Sains, F., Teknologi, D. A. N., Negeri, U. I., & Hidayatullah, S. (2013). *Aplikasi Stemming Pada Kamus Bahasa Indonesia Dengan Pendekatan Algoritma Nazief & Adriani*.

Suci Amaliah, Nusrang, M., & Aswi, A. (2022). Penerapan Metode Random Forest Untuk Klasifikasi Varian Minuman Kopi di Kedai Kopi Konijiwa Bantaeng. *VARIANSI: Journal of Statistics and Its Application on Teaching and Research*, *4*(3), 121–127. https://doi.org/10.35580/variansiunm31

Verdaningroem, N. J. M., & Saifudin, A. (2018). Penerapan Kamus Dasar pada Algoritma Porter untuk mengurangi kesalahan Stemming Bahasa Indonesia. *Jurnal Teknologi*, *10*(2), 103–112.

Zailani, A. U., & Hanun, N. L. (2020). Penerapan Algoritma Klasifikasi Random Forest Untuk Penentuan Kelayakan Pemberian Kredit Di Koperasi Mitra Sejahtera. *Infotech: Journal of Technology Information*, *6*(1), 7–14. https://doi.org/10.37365/jti.v6i1.61

Zaki Hariansyah, M. (2022). Implementasi Metode Multinomial Naive Bayes pada Analisis Sentimen Terhadap Layanan Aplikasi Livin by Mandiri Implementation of Naive Bayes Multinomial Method on Sentiment Analysis of Livin by Mandiri Application Services. *Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI) Jakarta-Indonesia*, *September*, 517–524. https://senafti.budiluhur.ac.id/index.php

DAFTAR LAMPIRAN

LAMPIRAN 1: Contoh Data Pendukung "FormalizationDict.txt".

7an	tujuan
@	di
ababil	abg labil
abis	habis
acc	accord
ad	ada
adlah	adalah
adlh	adalah
adoh	aduh
afaik	as far as i know
aha	tertawa
ahaha	haha
aing	saya
aj	saja
aja	saja
ajep-ajep	dunia gemerlap
ajj	saja
ak	saya
aka	dikenal juga sebagai
……	
ybs	yang bersangkutan
yg	yang
yi	yaitu
yl	yang lain
yng	yang
yo	iya
yoha	iya
yowes	ya sudah

LAMPIRAN 2: Contoh Data Pendukung "Slangword.txt".

```
&:dan
+:tambah
/:atau
=:sama dengan
ababil:anak ingusan
abal2:palsu
abal:palsu
ad:ada
akooh:aku
alay:norak
albm:album
ampe:sampai
anjir:waw
anyway:ngomong-ngomong
aq:aku
asap:secepatnya
ato:atau
atw:atau
ava:foto profil
baget:keras kepala
……
ttg:tentang
ttp:tetap
tw:tau
typo:salah tulis
u:you
unyu:menggemaskan
with:dengan
woles:santai
wtf:apa-apaan
x:kali
y:ya
yg:yang
```

LAMPIRAN 3: Contoh Data Pendukung "KBBA.txt".

@	di
ababil	abg labil
abis	habis
acc	accord
ad	ada
adlah	adalah
adlh	adalah
adoh	aduh
afaik	as far as i know
aha	tertawa
ahaha	haha
aing	saya
aj	saja
aja	saja
ajj	saja
ak	saya
aka	dikenal juga sebagai
akika	aku
akko	aku
…….	
dungu	bodoh
lemot	lambat
capek	lelah
kurng	kurang
bajir	banjir
ista	nista
istaa	nista
less	kurang
bebal	bodoh
koar	teriak
muna	munafik
lelet	lambat
naas	nahas

LAMPIRAN 4: Source Code

```
from google_play_scraper import app

import pandas as pd

import numpy as np

import re

import string

import csv

import emoji

import unicodedata

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import
train_test_split

from sklearn.pipeline import Pipeline

from sklearn.compose import ColumnTransformer


#scrape berdasarkan jumlah yang di inginkan

from google_play_scraper import Sort, reviews

result, continuation_token = reviews(

    'id.bmri.livin',

    lang='id',

    country='id',

    sort=Sort.MOST_RELEVANT, #defaults to
Sort.MOST_RELEVANT
```

```
    count=10000,
    filter_score_with=None
)


#SIMPAN DATASET YANG DISCRAPPING
df.to_csv("data10000.csv", index = False)


# MEMBACA DATASET SCARAPPING
df = pd.read_csv('data10000.csv')
df.head()


# MEMFILTER KOLOM YANG INGIN DIPAKAI
df =
pd.DataFrame(np.array(result),columns=['review']
)
df =
df.join(pd.DataFrame(df.pop('review').tolist()))
df.head()


# BUAT VARIABEL BARU UNTUK MEMBACA DATA KOLOM
YANG SUDAH DIFILTER
my_df = df[['content', 'score']]
my_df.head()
```

```
# PROSES CLEAN TEXT & CASE FOLDING

def clean_text(my_df, text_field,
new_text_field_name):

    my_df[new_text_field_name] =
my_df[text_field].str.lower()

    my_df[new_text_field_name] =
my_df[new_text_field_name].apply(lambda elem:
re.sub(r"(@)[A-Za-z0-9]+)|([^0-9A-Za-z
\t])|(\w+:\/\/\S+)|^rt|http.+?", "",elem))

    # remove number

    my_df[new_text_field_name] =
my_df[new_text_field_name].apply(lambda elem:
re.sub(r"\d+", "", elem))

    return my_df


# MEMBUAT KOlOM BARU YANG SUDAH DI PROSES CLEAN
TEXT & CASE FOLDING777

my_df['text_clean'] =
my_df['content'].str.lower()

my_df['text_clean']

data_clean = clean_text(my_df, 'content',
'text_clean')

data_clean.head(5)


# data duplicate

duplicates = my_df[my_df.duplicated()]

print(duplicates)
```

```
# MENGHAPUS KATA YANG DUPLICATE

my_df = my_df.drop_duplicates()

my_df = my_df.reset_index(drop=True)

my_df


# PENYELARASAN KATA SESUAI KAMUS DAN DATA
PENDUKUNG

slang_dictionary = pd.read_csv('colloquial-
indonesian-lexicon.csv')

slang_dict =
pd.Series(slang_dictionary['formal'].values,inde
x=slang_dictionary['slang']).to_dict()

slang_dictionary1 = pd.read_csv('kbba.txt',
sep='\t')

slang_dict1 =
pd.Series(slang_dictionary1['tujuan'].values,
index=slang_dictionary1['7an'].to_dict())

slang_dictionary2 = pd.read_csv('slangword.txt',
sep=':')

slang_dict2 =
pd.Series(slang_dictionary2['dan'].values,
index=slang_dictionary2['&'].to_dict())

slang_dictionary3 =
pd.read_csv('formalizationDict.txt', sep='\t')

slang_dict3 =
pd.Series(slang_dictionary3['tujuan'].values,
index=slang_dictionary3['7an'].to_dict())
```

```
#MENAMPILKAN SEBAGIAN DATA

slang_dictionary.head()

slang_dictionary1.head()

slang_dictionary2.head()

slang_dictionary3.head()


#PEMBUATAN FUNGSI SLANGWORDS

def Slangwords(text):

    for word in text.split():

        if word in slang_dict.keys():

            text = text.replace(word,
slang_dict[word])

            text = re.sub('@[\w]+','',text)

    return text


def Slangwords1(text):

    for word in text.split():

        if word in slang_dict1.keys():

            text = text.replace(word,
slang_dict1[word])

            text = re.sub('@[\w]+','',text)

    return text
```

```
def Slangwords2(text):

    for word in text.split():

        if word in slang_dict2.keys():

            text = text.replace(word,
slang_dict2[word])

            text = re.sub('@[\w]+','',text)

    return text


def Slangwords3(text):

    for word in text.split():

        if word in slang_dict3.keys():

            text = text.replace(word,
slang_dict3[word])

            text = re.sub('@[\w]+','',text)

    return text


# PEMBUATAN KOLOM BARU YANG SUDAH DI PROSES
SLANGWORDS

my_df['text_slang'] =
my_df['text_clean'].apply(Slangwords)

my_df['text_slang'] =
my_df['text_clean'].apply(Slangwords1)

my_df['text_slang'] =
my_df['text_clean'].apply(Slangwords2)

my_df['text_slang'] =
my_df['text_clean'].apply(Slangwords3)
```

```
my_df['text_slang'] =
my_df['text_clean'].str.replace('toling',
'tolong')

my_df['text_slang'] =
my_df['text_clean'].str.replace('g', 'ngga')

my_df['text_slang'] =
my_df['text_clean'].str.replace('mandir',
'mandiri')

my_df['text_slang'] =
my_df['text_clean'].str.replace('trsfrx',
'transfer')

my_df['text_slang'] =
my_df['text_clean'].str.replace('passw',
'password')

my_df['text_slang']



#IMPOR LIBRARY STOPWORD

from nlp_id.stopword import StopWord

stopword = StopWord()

#PEMBUATAN KOLOM BARU YANG SUDAH DIPROSES
STOPWORD

my_df['text_word_removal'] =
my_df['text_slang'].apply(stopword.remove_stopwo
rd)

my_df['text_word_removal']
```

```
# IMPORT LIBRARY STEMMING

from Sastrawi.Stemmer.StemmerFactory import
StemmerFactory

#create stemmer

factory = StemmerFactory()

stemmer = factory.create_stemmer()


#PEMBUATAN KOLOM BARU YANG SUDAH DIPROSES
STEMMING

my_df['text_stem'] =
my_df['text_word_removal'].apply(stemmer.stem)

my_df['text_stem'].head()


#PEMBUATAN FUNGSI TOKENIZE

def tokenize(teks):

    list_teks = []

    for txt in teks.split(" "):

        list_teks.append(txt)

    return list_teks


#PEMBUATAN KOLOM BARU YANG SUDAH DIPROSES
TOKENIZE

my_df['text_clean_tokenize'] =
my_df['text_stem'].apply(tokenize)

my_df['text_clean_tokenize'].head()
```

```
# FUNGSI PELABELAN

my_df_positive =
pd.read_csv('https://raw.githubusercontent.com/m
asdevid/ID-OpinionWords/master/positive.txt',
sep='\t')

list_positive = list(my_df_positive.iloc[::,0])

my_df_negative =
pd.read_csv('https://raw.githubusercontent.com/m
asdevid/ID-OpinionWords/master/negative.txt',
sep='\t')

list_negative = list(my_df_negative.iloc[::,0])

def sentiment_analysis_lexicon_indonesia(text):

    score = 0

    for word in text:

        if (word in list_positive):

            score += 1

    for word in text:

        if (word in list_negative):

            score -= 1

    polarity=''

    if (score > 0):

        polarity = 'positive'

    elif (score < 0):

        polarity = 'negative'

    else:

        polarity = 'neutral'
```

```
    return score, polarity


#PEMBUATAN KOLOM BARU YANG SUDAH DIPROSES
PELABELAN

hasil =
my_df['text_clean_tokenize'].apply(sentiment_ana
lysis_lexicon_indonesia)

hasil = list(zip(*hasil))

my_df['polarity_score'] = hasil[0]

my_df['polarity'] = hasil[1]

my_df.polarity.value_counts()


# MENAMPILKAN DATA DARI POLARITY SCORE

my_df['polarity_score']


# MENAMPILKAN SELURUH DATA YANG SUDAH DIPROSES

my_df.head()


# SIMPAN DATA MENJADI DATASET BARU

my_df.to_csv('hasil10k.csv')


# SPLIT DATA

from sklearn.model_selection import
train_test_split
```

```
X_train,X_test,y_train,y_test = train_test_split
(my_df['text_clean_tokenize'],my_df['polarity'],
test_size = 0.2,random_state = 42)

# Mengimport label encoder untuk mengubah teks
menjadi angka

from sklearn.preprocessing import LabelEncoder

Encoder = LabelEncoder()

y_train = Encoder.fit_transform(y_train)

y_test = Encoder.fit_transform(y_test)


# Proses TFIDF untuk melakukan ekstraksi fitur

from sklearn.feature_extraction.text import
TfidfVectorizer

Tfidf_vect = TfidfVectorizer(max_features=100)

Tfidf_vect.fit(my_df['text_clean_tokenize'])

Train_X_Tfidf = Tfidf_vect.transform(X_train)

Train_X_Tfidf = Tfidf_vect.transform(X_test)

Train_X_Tfidf.toarray()


# KLASIFIKASI RANDOM FOREST

from sklearn.svm import SVC

from sklearn.ensemble import
RandomForestClassifier

from sklearn.model_selection import
RandomizedSearchCV

from jcopml.tuning.space import Integer, Real
```

```
# Update the 'algo' step in the pipeline to use
RandomForestClassifier

pipeline_rf = Pipeline([

    ('prep', TfidfVectorizer()),

    ('algo',
RandomForestClassifier(random_state=42))

])

# Update the parameters for Random Forest

param_rf = {

    'algo__n_estimators': [50, 100, 200, 300],

    'algo__max_depth': [None, 10, 20, 30],

    'algo__min_samples_split': [2, 5, 10],

    'algo__min_samples_leaf': [1, 2, 4],

    'algo__bootstrap': [True, False]

}

# Create RandomizedSearchCV for Random Forest

model_rf = RandomizedSearchCV(pipeline_rf,
param_rf, cv=4, n_iter=50, n_jobs=-2, verbose=1,
random_state=42)

model_rf.fit(X_train, y_train)

# Print the best parameters and scores

print(model_rf.best_params_)

print(model_rf.score(X_train, y_train),
model_rf.best_score_, model_rf.score(X_test,
y_test))
```

```
# CONFUSION MATRIX & CLASSIFICATION REPORT

from sklearn.metrics import
classification_report, confusion_matrix

# Menggunakan model terbaik untuk membuat
prediksi pada data uji

y_pred = model_rf.predict(X_test)

# Membuat classification report

print("Classification Report:")

print(classification_report(y_test, y_pred))

# Membuat confusion matrix

conf_mat = confusion_matrix(y_test, y_pred)

print("Confusion Matrix:")

sns.heatmap(conf_mat, annot=True, fmt='d',
cmap='Blues', cbar=True)



# UBAH TIPE DATA MENJADI STRING

my_df['text_clean_tokenize']
=my_df['text_clean_tokenize'].astype('str')

my_df['text_clean_tokenize']
=my_df['text_clean_tokenize'].astype(pd.StringDt
ype())

my_df.dtypes

# WORDCLOUD SEMUA DATA

all_words_p =' '.join([word for word in
my_df['text_clean_tokenize']])

%matplotlib inline
```

```
import matplotlib.pyplot as plt

from wordcloud import WordCloud

wordcloud =
WordCloud(background_color='white',width=800,
height=500, random_state=21,
max_font_size=130).generate(all_words_p)

plt.figure(figsize=(20,10))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off');

#WORDCLOUD POSITIF

my_df_p=my_df[my_df['polarity_score']>0]

all_words_p =' '.join([word for word in
my_df_p['text_clean_tokenize']])

%matplotlib inline

import matplotlib.pyplot as plt

from wordcloud import WordCloud

wordcloud =
WordCloud(background_color='white',width=800,
height=500, random_state=21,
max_font_size=130).generate(all_words_p)

plt.figure(figsize=(20,10))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off');

# WORDCLOUD NEGATIF

my_df_n=my_df[my_df['polarity_score']<0]

all_words_n =' '.join([word for word in
my_df_n['text_clean_tokenize']])
```

```
%matplotlib inline

import matplotlib.pyplot as plt

from wordcloud import WordCloud

wordcloud =
WordCloud(background_color='white',width=800,
height=500, random_state=21,
max_font_size=130).generate(all_words_n)

plt.figure(figsize=(20,10))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off');

# WORDCLOUD NETRAL

my_df_net=my_df[my_df['polarity_score']==0]

all_words_net =' '.join([word for word in
my_df_net['text_clean_tokenize']])

%matplotlib inline

import matplotlib.pyplot as plt

from wordcloud import WordCloud

wordcloud =
WordCloud(background_color='white',width=800,
height=500, random_state=21,
max_font_size=130).generate(all_words_net)

plt.figure(figsize=(20,10))

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis('off');
```