PERBANDINGAN ANALISIS *WORDNET* DAN *K-NEAREST NEIGHBOR* PADA ULASAN APLIKASI SIREKAP 2024

SKRIPSI

Diajukan Oleh :

Emyzar Haflida Tanjung

2011102441240

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR

SAMARINDA

JULI 2024

PERBANDINGAN ANALISIS *WORDNET* DAN *K-NEAREST NEIGHBOR* PADA ULASAN APLIKASI SIREKAP 2024

SKRIPSI

Diajukan Sebagai Salah Satu Persyaratan Untuk Memenuhi Gelar Sarjana Fakultas Sains dan Teknologi Universitas Muhammadiyah Kalimantan Timur

Diajukan Oleh :

Emyzar Haflida Tanjung

2011102441240

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR

SAMARINDA

JULI 2024

LEMBAR PERSETUJUAN

PERBANDINGAN ANALISIS *WORDNET DAN K-NEAREST NEIGHBOR* PADA ULASAN APLIKASI SIREKAP 2024

Diajukan Oleh :

EMYZAR HAFLIDA TANJUNG

2011102441240

Disetujui untuk diujikan

Pada tanggal 20 Juni 2024

Pembimbing

Naufal Azmi Verdikha, S.Kom., M.Eng.

NIDN. 1114048801

Mengetahui,

Koordinator Tugas Akhir/Skripsi/Tesis/Disertasi

Abdul Rahim, S.Kom., M.Cs.

NIDN. 111024103

LEMBAR PENGESAHAN

PERBANDINGAN ANALISIS *WORDNET* DAN *K-NEAREST NEIGHBOR* PADA ULASAN APLIKASI SIREKAP 2024

SKRIPSI

Diajukan Oleh :

EMYZAR HAFLIDA TANJUNG

2011102441240

Diseminarkan dan diujikan

Pada tanggal 11 Juli 2024

Penguji 1	Penguji II
Taghfirul Azhima Yoga Siswa, S.Kom., M.Kom **NIDN. 1118038805**	**Naufal Azmi Verdikha, S.Kom., M.Eng.** **NIDN. 1114048801**

Mengetahui,

Ketua

Program Studi Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019203

PERNYATAAN KEASLIAN PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama : Emyzar Haflida Tanjung

NIM : 2011102441240

Program Studi : S1 Teknik Informatika

Judul Penelitian : "Perbandingan Analisis *WordNet* dan *K-Nearest Neighbor* Pada Ulasan Aplikasi Sirekap 2024".

menyatakan bahwa skripsi yang saya tulis ini benar-benar hasil karya saya sendiri, dan bukan merupakan hasil plagiasi/falsifikasi/fabrikasi baik sebagian atau seluruhnya. Atas pernyataan ini, saya siap menanggung risiko atau sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap etika keilmuan dalam tugas skripsi saya ini, atau klaim dari pihak lain terhadap keaslian karya saya ini

Samarinda, 30 Juni 2024

Yang membuat pernyataan

Emyzar Haflida Tanjung

2011102441240

ABSTRAK

Perkembangan model klasifikasi telah mencakup bidang klasifikasi sentimen yang bersumber dari data teks. Penelitian mengenai *Machine Learning* dan *Natural Language Processing* (NLP) yang menggunakan data ulasan terhadap suatu aplikasi sudah pernah dilakukan sebelumnya. Bagi pengguna, ulasan aplikasi sering digunakan sebagai sumber informasi untuk mengetahui lebih lanjut tentang aplikasi tersebut. Tujuan penelitian ini untuk mengetahui hasil perbandingan analisis klasifikasi *WordNet* dan *K-Nearest Neighbor* menggunakan ekstraksi fitur TF-IDF dalam mendapatkan hasil dari nilai evaluasi *F1-Score* pada ulasan Aplikasi "Sirekap 2024". Penelitian ini menggunakan sebanyak 8358 data ulasan, dengan teknik pengambilan data melalui proses *Scraping.* Hasil penelitian berdasarkan uji coba yang telah dilakukan, metode *K-Nearest Neighbor* lebih baik dalam mengklasifikasikan dibanding metode *WordNet,* dengan hasil perbandingan *K-Nearest Neighbor* sebesar 32% sedangkan *WordNet* sebesar 17,50%.

Kata kunci: Sirekap 2024, *K-Nearest Neighbor, WordNet,* Perbandingan.

ABSTRACT

The development of classification models has covered the field of sentiment classification sourced from text data. Research on machine learning and natural language processing (NLP) using review data on an application has been done before. For users, application reviews are often used as a source of information to find out more about the application. The purpose of this study is to determine the comparison results of WordNet and K-Nearest Neighbor classification analysis using TF-IDF feature extraction in obtaining the results of the F1-Score evaluation value on the "Sirekap 2024" application review. This research uses a total of 8358 review data, with data retrieval techniques through the Scraping process. The results of research based on trials that have been carried out, the K-Nearest Neighbor method is better at classifying than the WordNet method, with the results of the K-Nearest Neighbor comparison of 32% while WordNet is 17.50%.

Keywords: Sirekap 2024, K-Nearest Neighbor, WordNet, Comparison.

PRAKATA

Alhamdulillahi Rabbil'alamin. Puji dan syukur atas kehadirat Allah subhanahu wa ta'ala, atas segala rahmat dan karunia-Nya, yang telah memberikan kekuatan, semangat, dan kesabaran, sehingga membantu penulis dalam menyelesaikan karya skripsi ini.

Dengan segala kekurangan dalam penulisan skripsi ini sangat disadari oleh penulis. Penulis menyadari bahwa selama proses penulisan skripsi banyak pihak yang telah memberikan dorongan, bantuan, bimbingan, motivasi, masukan dan do'a yang tiada hentinya diterima oleh penulis sehingga mampu menyelesaikan skripsi ini. Oleh karena itu penulis ingin mengucapkan terimakasih tak terhingga kepada:

1. Orang tua tercinta Ayahanda Ghazali Tanjung dan Ibunda Paridah yang tiada henti selalu memberikan cinta, kasih sayang, dukungan, nasihat, do'a, serta kesabarannya dalam menghadapi setiap langkah hidup penulis.

2. Bapak Arbansyah, S.Kom., M.TI selaku Ketua Program Studi S1 Teknik Informatika.

3. Bapak Naufal Azmi Verdikha, S.Kom., M.Eng selaku dosen pembimbing yang telah dengan sabar memberikan bimbingan, masukan dan saran-saran yang sangat membantu dalam menyempurnakan skripsi ini.

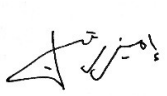
4. Kepada saudara saudari saya kak Zahra Khalida Tanjung, Maulida Ashfiya Tanjung dan adik saya Muhammad Zulfi Naufal Tanjung, yang selalu memberikan dukungan penuh kepada penulis.

5. Ucapan terima kasih juga penulis sampaikan kepada diri sendiri atas dedikasi dan kerja keras yang telah diberikan selama proses penulisan skripsi ini. Terima kasih telah tetap kuat dan semangat meskipun menghadapi banyak tantangan. Perjalanan ini telah mengajarkan banyak hal berharga yang akan selalu diingat.

6. Rekan-rekan seperjuangan prodi Teknik Informatika Universitas Muhammadiyah Kalimantan Timur yang tidak dapat disebutkan satu persatu.

7. Teruntuk seseorang dimasa depan saya yang masih dipersiapkan oleh Allah, terimakasih sudah mendo'akan saya untuk tidak berpacaran hingga saat ini. Semoga kelak kita dipertemukan diwaktu yang terbaik, ketika kita sudah menjadi orang yang sukses dan bisa membahagiakan orang tersayang, sehingga skripsi ini bisa terbentuk karena sembari mewujudkan mimpi dan menunggu mu tanpa berpacaran.

Samarinda, 30 Juni 2024
Penyusun,



Emyzar Haflida Tanjung

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel **Halaman**

DAFTAR GAMBAR

Gambar **Halaman**

DAFTAR LAMPIRAN

Lampiran **Halaman**

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

"Sirekap 2024" merupakan sebuah aplikasi teknologi informasi yang berfungsi sebagai platform untuk mempublikasikan dan merangkum hasil penghitungan suara dalam pemilu tahun 2024. Aplikasi ini juga bertindak sebagai alat bantu dalam proses rekapitulasi suara. Dalam pemilu tersebut, terdapat dua varian sirekap yang digunakan: versi *mobile* untuk Komisi Pemilihan Pemungutan Suara (KPPS) dalam melakukan perhitungan di Tempat Pemungutan Suara (TPS), dan versi web untuk Panitia Pemilihan Kecamatan (PPK) dan anggota Komisi Pemilihan Umum (KPU) di tingkat kota/kabupaten dan provinsi. Aplikasi "Sirekap 2024" dapat dioperasikan melalui ponsel berbasis android, dan dapat diakses baik secara daring maupun luring, hal ini memberikan akses informasi dan layanan yang dikembangkan oleh pemerintah sehingga sangat mudah diakses oleh masyarakat (Hardiyanti *et al.*, 2022).

Aplikasi "Sirekap 2024" sebagai salah satu aplikasi yang tersedia di play store yang dirancang untuk mengotomatisasi dan mempercepat proses pengumpulan dan pengolahan data suara. Namun, seperti halnya teknologi lainnya, penggunaan "Sirekap 2024" menghadapi berbagai tantangan dan masalah yang perlu diatasi. Secara fundamental, sistem elektronik seperti "Sirekap 2024" menawarkan sejumlah manfaat potensial dalam konteks pemilu. Pertama, penggunaan "Sirekap 2024" dapat meningkatkan efisiensi dan akurasi dalam penghitungan suara, mengurangi kesalahan manusia yang mungkin terjadi dalam proses manual. Ini dapat mengurangi potensi konflik atau sengketa terkait hasil pemilu akibat ketidaktepatan dalam penghitungan (Pradesa, 2024).

Hal ini karena "Sirekap 2024" adalah alat vital dalam proses pemilihan umum, memainkan peran penting dalam memfasilitasi partisipasi warga negara dalam proses demokrasi (Herjanto dan Carudin, 2024). Karena analisis sentimen yang bertujuan untuk mengevaluasi penilaian, opini, dan sikap seseorang terhadap organisasi, individu, produk, dan sebagainya dari para pengguna internet maupun layanan teknologi yang terkandung dalam teks (Amrullah, Sofyan Anas dan Hidayat, 2020).

Pada penelitian (Afdal dan Waroka, 2022) yang berjudul *Klasifikasi Ulasan Aplikasi Shopee Menggunakan Algoritma Probabilistic Neural Network Dan K-Nearest Neighbor,* bertujuan untuk membandingkan penggunaan dua algoritma klasifikasi, yaitu *Probabilistic Neural Network* (PNN) dan *K-Nearest Neighbor* (KNN). Metode pembagian data yang digunakan adalah *K-Fold Cross Validation*, kemudian diukur akurasinya pada data ulasan aplikasi dan produk Shopee. Ditemukan bahwa akurasi data ulasan aplikasi menggunakan KNN lebih tinggi daripada PNN, dengan KNN mencapai 77,85% dan PNN mencapai 72,43%. Sedangkan untuk data produk, akurasi KNN juga lebih tinggi dibandingkan PNN, dengan KNN mencapai 91,43% dan PNN mencapai 85,71%. Dari hasil penelitian ini, dapat disimpulkan bahwa algoritma KNN memiliki performa yang lebih baik daripada PNN dalam konteks data yang digunakan.

Pada penelitian (Rahayu *et al.*, 2022) bertujuan untuk menganalisis sentimen yang terdapat pada ulasan pengguna terhadap aplikasi Flip, sehingga dapat memahami apakah ulasan tersebut mencerminkan nilai positif yang serupa dengan rating yang diberikan. Penelitian tersebut menggunakan proses *text mining* pada data ulasan pengguna aplikasi Flip yang tersedia di Google

Play Store, dan algoritma klasifikasi yang dipilih adalah *K-Nearest Neighbor* dengan penerapan pembobotan TF-IDF dan hasil penelitian menunjukkan bahwa 77,67% dari data uji berhasil diklasifikasikan secara akurat ke dalam kelas ulasan positif, dengan nilai presisi dan *recall* yang tinggi, yaitu masing-masing sebesar 82,67% dan 86,92%. Selain itu, dengan menggunakan rasio data latih dan data uji sebesar 80% dan 20%, diperoleh tingkat akurasi klasifikasi sebesar 76,68% menggunakan algoritma *K-Nearest Neighbor*.

Penelitian sebelumnya dengan berbasis *WordNet* yang berjudul *Query Expansion Pada Sistem Temu Kembali Informasi Berbahasa Indonesia Dengan Metode Pembobotan TF-IDF Dan Algoritme Cosine Similarity Berbasis Wordnet* (Dwi Laxmi dan Ali Fauzi, 2019) Penelitian ini menggunakan metode TF-IDF dan algoritma *cosine similarity* berbasis *WordNet*. Dengan menggunakan *WordNet*, penambahan *query* dilakukan untuk menyempurnakan suatu teks tertentu agar sesuai dengan konsep kalimat yang diinginkan. Dalam penelitian ini, *synset* yang berupa relasi kata hiponim akan ditambahkan ke dalam *query*. Berdasarkan hasil pengujian menggunakan *precision* 20 dari 10 *query*, diperoleh nilai presisi rata-rata sebesar 0,7 yang mengindikasikan bahwa probabilitas sistem untuk menemukan kembali dokumen yang relevan tanpa menggunakan ekspansi *query* adalah sebesar 70%. Hasil pengujian lain menunjukkan nilai presisi rata-rata sebesar 0,52 dengan pengujian yang sama.

Karena aplikasi "Sirekap 2024" memainkan peran penting dalam proses pemilu di Indonesia. Memahami sentimen pengguna terhadap aplikasi "Sirekap 2024" menjadi sangat penting untuk meningkatkan kualitas dan kinerja aplikasi. Urgensi penelitian ini terletak pada kebutuhan mendesak untuk memastikan bahwa pemilu yang akan datang dapat dilaksanakan dengan lebih akurat dan efisien, mengurangi potensi kesalahan dan meningkatkan kepercayaan publik terhadap hasil pemilu. Dan adanya berbagai metode mengharuskan kita untuk mengetahui metode mana yang lebih efektif dan akurat dalam konteks ulasan aplikasi. Dengan demikian, penelitian ini tidak hanya akan memberikan kontribusi signifikan terhadap pengembangan dan peningkatan aplikasi "Sirekap 2024", tetapi juga akan menyediakan panduan berharga bagi pengembang dan peneliti dalam memilih metode yang paling efektif dan efisien. Sehingga, penelitian ini bertujuan menyajikan hasil komparasi dari kedua metode tersebut yaitu *K-Nearest Neighbor* (KNN) dan *WordNet* dalam menganalisis sentimen pada ulasan aplikasi "Sirekap 2024" berdasarkan tingkat *rating* yang diberikan oleh pengguna. Penelitian ini dibagi menjadi dua tahapan, pada tahapan pertama yaitu menganalisis sentimen menggunakan *WordNet* dan tahapan kedua menggunakan algoritma klasifikasi KNN dengan ekstraksi fitur TF-IDF. Kemudian kedua tahapan tersebut dikomparasikan hasil evaluasinya menggunakan *f1-score.*

Pada penelitian sebelumnya, yang dilakukan oleh Afdal dengan judul *Klasifikasi Ulasan Aplikasi Shopee Menggunakan Algoritma Probabilistic Neural Network Dan K-Nearest Neighbor*, telah menunjukkan keefektifan *K-Nearest Neighbor* (KNN) dalam analisis sentimen ulasan aplikasi Shopee, mengungguli *Probabilistic Neural Network* (PNN) dalam hal akurasi. Namun, penelitian tersebut terbatas pada konteks aplikasi *e-commerce* dan belum mengeksplorasi penerapannya pada aplikasi dengan konteks berbeda, seperti "Sirekap 2024" yang terkait pemilu. Penelitian ini bertujuan untuk mengisi kesenjangan tersebut dengan membandingkan kinerja KNN dan *WordNet* dalam klasifikasi analisis sentimen ulasan aplikasi "Sirekap 2024".

1.2 Rumusan Masalah

Bagaimana hasil komparasi metode *WordNet* dan klasifikasi *K-Nearest Neighbors* menggunakan ekstraksi fitur TF-IDF menggunakan evaluasi *F1-Score* pada ulasan Aplikasi "Sirekap 2024"?

1.3 Tujuan Penelitian

Tujuan penelitian ini yaitu mengetahui hasil perbandingan *WordNet* dan *K-Nearest Neighbor* menggunakan ekstraksi fitur TF-IDF dalam mendapatkan hasil dari nilai evaluasi *F1-Score* pada ulasan Aplikasi "Sirekap 2024".

1.4 Manfaat Penelitian

1. Penelitian ini memberikan pengetahuan untuk penggunaan metode yang tepat dalam menganalisis sentimen ulasan, terutama pada ulasan yang berbahasa Indonesia dan pada aplikasi "Sirekap 2024"
2. Penelitian ini dapat memberikan wawasan berharga bagi pengembang aplikasi "Sirekap 2024". Informasi yang diperoleh dari penelitian ini dapat digunakan untuk meningkatkan kualitas aplikasi, menyesuaikan fitur, atau merespons ulasan pengguna dengan lebih baik.

1.5 Batasan Masalah

1. Pada penelitian, data "Sirekap 2024" yang dianalisis dan diolah yaitu pada tanggal 6 februari 2024.
2. Jumlah data yang digunakan dan diolah pada penelitian ini sebanyak 8358 dataset.

BAB II

METODE PENELITIAN

2.1 Objek Penelitian

"Sirekap 2024" adalah aplikasi untuk mendokumentasikan formulir hasil penghitungan suara di TPS dan mengirimkannya ke jenjang selanjutnya[1]. Sirekap pada pemilu 2024 sebagai alat bantu berbasis teknologi yang membantu KPPS dalam menyederhanakan proses penginputan hasil perhitungan suara. Sirekap yang dipakai sebagai sarana publikasi hasil pemilihan dan alat bantu dalam pelaksanaan rekapitulasi suara Pilkada Serentak 2020 telah dipersiapkan oleh KPU RI dalam setahun terakhir (Gauru, Martini dan Alfirdaus, 2022). Proses pengambilan data pada tanggal 6 februari 2024 tepatnya pukul 22.00 WITA, terdapat sekitar delapan ribu data ulasan aplikasi "Sirekap 2024" pada google playstore, ulasan tersebut terdiri dari peringkat satu hingga lima.

2.2 Alat dan Bahan

Dalam penelitian ini Alat yang digunakan meliputi perangkat *hardware* dan *software* yang digunakan, diantaranya :

1. Perangkat keras(*Hardware*)
   - Laptop merk Lenovo
     Dengan spesifikasi sebagai berikut :
   - RAM 8192MB RAM
   - Processor Intel Core i3
   - Sistem operasi windows 10 Pro 64-bit

2. Perangkat Lunak (*Software*)

Pada perangkat lunak yang digunakan yaitu, Visual Studio Code versi 1.73.0 dan *library* python. *Library* python yang digunakan :

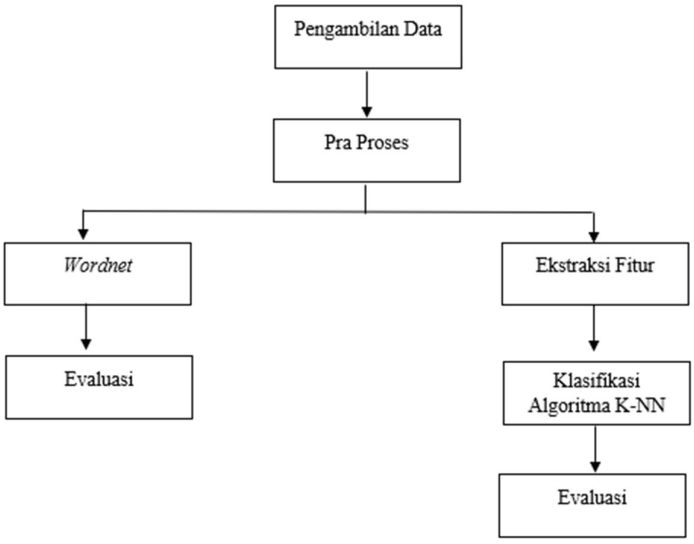
Tabel 2. 1 *Library* Python

***Library* Python**	**Versi**	**Keterangan**
Pandas	1.4.4	Digunakan untuk analisis data
Matplotlib	1.21.5	Digunakan untuk membuat visualisasi data
Numpy	1.21.5	Digunakan untuk mengolah dan memanipulasi data dalam bentuk *array*
Sckitlearn	1.0.2	Pustaka yang dapat digunakan secara gratis untuk pembelajaran mesin dan pemodelan data dalam bahasa pemrograman python.
NLTK(*Natural Language Toolkit*)	3.7	Pustaka yang dirancang untuk digunakan dalam pemrosesan bahasa alami menggunakan python, menyediakan berbagai kumpulan alat untuk analisis teks dan berbagai dataset untuk pengujian.
Textblob	0.18.0	Digunakan untuk mengetahui sebuah *text* bersifat positif, negatif, atau netral dalam analisis sentimen
Google-play-scraper	1.2.6	Digunakan untuk mengakses google play store dan mengekstrak informasi yang diperlukan.
sastrawi	1.0.1	Digunakan untuk memproses kata-kata ke bentuk dasarnya.

[1] https://play.google.com/store/apps/details?id=id.go.kpu.sirekap2024

2.3 Prosedur Penelitian

Penelitian ini menganalisis ulasan pengguna pada aplikasi "Sirekap 2024" menggunakan algoritma *K-Nearest Neighbors* dengan tahapan penelitian yang dapat dilihat pada gambar 2.1



Gambar 2. 1 Diagram Alur Penelitian

Pada gambar alur penelitian diatas, tahapan pertama yang dilakukan adalah pengambilan data yang diambil dari google playstore. Setelah data dikumpulkan kemudian dilakukan tahapan pra proses yang terdiri dari beberapa tahapan. Langkah selanjutnya terbagi menjadi 2 tahapan besar yaitu tahapan *WordNet* dan tahapan klasifikasi K-NN. Kedua tahapan tersebut kemudian dievaluasi menggunakan nilai *F1-Score*.

1. Pengambilan Data

Pada proses pengambilan data, penelitian ini mengambil data sekunder yang diperoleh dari ulasan pengguna "Sirekap 2024". Berikut parameter yang digunakan dalam pengumpulan data :

a. *Add_id*
   Pada proses pengambilan data menggunakan *add_id* dengan nilai parameter "id.go.kpu.sirekap2024" sebagai identifikasi unik atau *ID* yang dimasukkan ke setiap elemen dalam kumpulan data. digunakan untuk tujuan identifikasi, pengelompokkan, atau referensi unik.
b. *Lang*
   Dalam penggunaan *lang* dengan nilai parameternya yaitu "*id*" yang merupakan argumen yang digunakan untuk menentukan bahasa ulasan yang ingin diambil dari Google Playstore.
c. *Country*
   Sedangkan fungsi parameter *country* dengan nilai parameternya yaitu "*id*". Dalam proses pengambilan data, fungsi ini untuk menentukan negara tempat aplikasi tersedia.

d. Sort

Pada parameter *sort* dengan nilai "*Sort.MOST_RELEVAN*" digunakan untuk menentukan metode pengurutan ulasan.

e. Count

Fungsi *count* disini untuk menentukan jumlah ulasan yang ingin diambil. sesuai dengan jumlah yang diinginkan, pada penelitian ini menentukan dengan jumlah 100.000.000.

2. Pra Proses

Pada penelitian ini berikut merupakan tahapan dari pra proses yang digunakan[2] :

a. Add id

Tahap ini menggunakan *library pandas*, '*Add id'* digunakan pada pra proses untuk menambahkan sebuah identifikasi unik atau *ID* ke setiap item dalam kumpulan data. Hal ini dilakukan untuk keperluan identifikasi, pengelompokan, atau referensi yang khas. (*source code* terdapat dalam lampiran 13)

b. Lowercase

Penggunaan *lowercase* merujuk pada konversi atau proses untuk mengubah semua huruf dalam teks menjadi huruf kecil. Ini adalah kebalikan dari *uppercase*, di mana semua huruf dalam teks diubah menjadi huruf besar. Dalam pengolahan teks, yaitu untuk mengkonversi semua huruf menjadi huruf kecil bertujuan untuk memproses data dengan benar dan konsisten. Penelitian ini menggunakan proses *lowercase* untuk mengubah semua huruf menjadi huruf kecil. (*source code* terdapat dalam lampiran 14)

c. Remove Unecessary Character

Proses ini akan menghapus karakter yang tidak diperlukan, dimana proses ini menghilangkan semua karakter dari sebuah teks atau *string* yang bukan huruf atau angka. Proses ini berfungsi untuk membersihkan teks dari simbol, tanda baca, atau karakter spesial lainnya, yang tidak diperlukan dalam analisis atau pemrosesan data. (*source code* terdapat dalam lampiran 15).

d. Spellchecker

Penggunaan *spellchecker* dirancang untuk mendeteksi dan memperbaiki kesalahan pengejaan dalam teks yang dimasukkan pengguna dan mengubah kata yang tidak baku atau kata slang menjadi kata yang sesuai sesuai dengan Kamus Besar Bahasa Indonesia (KBBI). Sebelum masuk proses *spellchecker* terlebih dahulu dibuat kamus_tidak_baku yang berlandaskan dari Kamus Besar Indonesia (KBI) (Hasdiana, 2018). Pada prosesnya *file* csv diinput untuk proses pembuatan kamus tidak baku dan langkah pertama dalam membuat kamus tidak_baku yaitu menginput data csv hasil scraping menggunakan *library pandas*, kemudian membersihkan data dari karakter non-alfabet dan memecah teks menjadi kata-kata, lalu menghapus kata-kata duplikat, lalu membersihkan kata yang mengandung karakter numerik dan menghapus kata yang terdiri dari satu hingga tiga huruf. Langkah selanjutnya mencari kata-kata yang tidak terrdapat di list kamus pada kolom data pandas.Tujuan utama *spellchecker* untuk membantu memperbaiki kesalahan pengejaan yang mungkin terjadi karena ketik cepat, kesalahan aturan pengejaan, atau kesalahan ketik dan kata slank. *Spellchecker* akan memeriksa setiap kata dalam teks untuk melihat apakah ada yang tidak sesuai dengan kamus kata yang sudah disediakan. (*source code* terdapat dalam lampiran 17).

[2] https://github.com/gioprana89/scraping-google-play

e. *Stemming*

tahapan ini digunakan untuk menghilangkan atau memotong akhiran pada kata-kata dalam teks dengan tujuan untuk menghasilkan bentuk kata dasar atau kata akar. Hal ini dilakukan untuk mengurangi variasi dalam kata-kata yang muncul dalam teks, sehingga kata-kata yang sebenarnya memiliki makna yang sama dapat dipresentasikan secara konsisten sebagai satu entitas (*source code* terdapat dalam lampiran 18)

3. *WordNet*

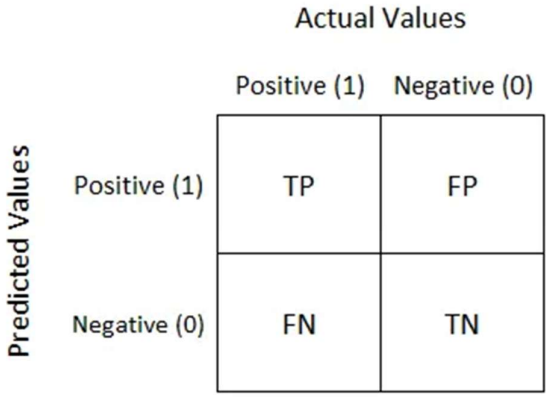
WordNet merupakan pusat data yang memuat istilah-istilah dalam bahasa Inggris beserta keterkaitannya. Istilah ini mencakup kata benda, kata kerja, kata sifat, atau kata keterangan, serta hubungan-hubungan antara istilah tersebut (Siahaan *et al.*, 2023). Dalam metode *WordNet* yang pertama dilakukan dalam penelitian ini yaitu data diterjemah dengan API Google Translate yang kemudian dilakukan perhitungan menggunakan metode *WordNet* dengan *Library Textblob*. Tabel 2.2 terdapat kolom indeks yang merupakan nilai polaritas sentimen dari -1 (sangat negatif) hingga 1 (sangat positif). Dalam tabel ini, polaritas yang lebih besar dari 0,5 dikategorikan sebagai sangat positif dengan peringkat 5. Polaritas antara 0,2 dan 0,5 diberi peringkat 4, menunjukkan sentimen positif. Polaritas netral, yang berkisar antara -0,1 dan 0,1, diberi peringkat 3. Sentimen negatif dengan polaritas antara -0,5 dan -0,2 diberi peringkat 2, sementara polaritas kurang dari -0,5, yang menunjukkan sentimen sangat negatif, diberi peringkat 1. Sehingga pada tahap ini akan menghasilkan label ranking yang terdiri dari ranking 1-5 dari ulasan "Sirekap 2024" (*Source code* terdapat dalam lampiran 22).

Tabel 2. 2 Kriteria Indeks

Indeks	Peringkat
>0,5	5
0,2 – 0,5	4
(-0,1)-0,1	3
(-0,5)-(-0,2)	2
<(-0,5)	1

4. Evaluasi *WordNet*

Pada evaluasi tahapan *WordNet, F1-Score* diperoleh dari nilai yang terdapat di *confusion matrix* yang merupakan pengukuran yang sering digunakan dalam masalah klasifikasi, dimana output dapat terdiri dari dua kelas atau lebih yang memiliki 4 nilai yaitu *true positive* (TP), *false positive* (FP), *true negative* (TN) dan *false negative* (FN) (Istighfarizky *et al.*, 2022). (*Source code* terdapat pada lampiran 24).



Gambar 2. 2 Dasar Confusion Matrix

Pada gambar diatas merupakan dasar *confusion matrix* 2x2, pada penelitian ini menggunakan lebih dari 2 kelas, yang berarti masuk kedalam *confusion matrix multi class.*

	Rating 1	Rating 2	Rating 3	Rating 4	Rating 5
Rating 1	TP	FP	FP	FP	FP
Rating 2	FN	TN	TN	TN	TN
Rating 3	FN	TN	TN	TN	TN
Rating 4	FN	TN	TN	TN	TN
Rating 5	FN	TN	TN	TN	TN

Gambar 2. 3 *Confusion matrix 5x5*

Dari gambar 2.3, menunjukkan *Confusion matrix* klasifikasi *multi class* pada sistem evaluasi rating. *Matrix* ini digunakan untuk mengevaluasi kinerja metode klasifikasi yang memprediksi lima tingkat rating (rating 1 hingga rating 5).*Confusion matrix* adalah tabel yang menggambarkan jumlah data uji yang diklasifikasikan dengan benar dan jumlah data uji yang diklasifikasikan dengan salah (Normawati dan Prayogi, 2021).

Tabel 2. 3 Penjelasan nilai konfusi matriks

Nilai	Keterangan
True Positive (TP)	Data *Positive* yang diprediksi benar
True Negative (TN)	Data *Negative* yang diprediksi benar
False Positive (FP)	Data *Negative* namun diprediksi sebagai data positif
False Negative (FN)	Data *positve* namun diprediksi sebagai data negatif

Dari keempat nilai tersebut akan menjadi dasar dalam perhitungan yaitu:

a. *Precision*

Precision merupakan proporsi dari *True Positive* (TP) terhadap total prediksi positif. Sehingga *precision* bertujuan untuk mengurangi jumlah *False Positive* (FP). Berikut rumus *precision*:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (1)$$

b. *Recall*

Recall merupakan proporsi dari *True Positive* (TP) dengan hasil data yang positif . sehingga *recall* bertujuan untuk mengurangi jumlah *False Negative* (FN). Berikut rumus *recall*:

$$\text{Recall} = \frac{TP}{TP+FP} \quad (2)$$

b. *F1-Score*

F1-score merupakan matrik evaluasi yang menggabungkan *precision* dan *recall* menjadi satu nilai. Berikut rumus F1-Score:

$$\text{F1-Score} = \frac{2 \times precision \times recall}{precision+recall} = \frac{2TP}{2TP+FP+FN} \quad (3)$$

c. *Average Macro*

Pada *avarage macro* merujuk untuk menghitung rata-rata dari *presisi,recall dan F1-Score*, berikut cara mencari rata-rata *macro*:

$$MAF = \frac{\sum_{K=1}^{K} F1Score_k}{K} \quad (4)$$

Keterangan:

MAF = *Macro Average F1-Score*

K = Jumlah kelas pada klasifikasi *multiclass*

5. Tahapan Klasifikasi KNN

a. Ekstraksi Fitur TF-IDF

Tahap selanjutnya yaitu ekstraksi fitur yang digunakan pada penelitian ini adalah *Term Frequency –Inverse Document Frequency* (TF-IDF) .TF-IDF merupakan teknik pembobotan kata yang menggabungkan perhitungan nilai *Term Frequency* (TF) dan jumlah kemunculan kata pada seluruh koleksi dokumen (Karo Karo *et al.*, 2023). *Term Frequency* mengukur frekuensi kemunculan sebuah kata dalam dokumen tertentu, dimana semakin sering kata tersebut muncul, semakin besar nilai TF (Amly, Yusra dan Fikry, 2023). Sementara itu, *Inverse Document Frequency* (IDF) mengukur jumlah dokumen yang mengandung kata tersebut dalam seluruh dataset, sehingga semakin jarang kata tersebut muncul, semakin besar nilai IDF-nya (Syahrandi, Latipah and Verdikha, 2023). Hasil dari pembobotan kata adalah perkalian antara nilai TF dan IDF, dimana bobotnya akan lebih kecil jika kata tersebut muncul lebih sering, dan sebaliknya, akan lebih besar jika kata tersebut muncul lebih jarang (Umar, Riadi dan Purwono, 2020). Berikut rumus persamaan TF-IDF: (*Source code* terdapat pada lampiran 27).

$$tfidf(t,d) = tf(t,d) \times idf(t) \quad (5)$$

Keterangan :

tfidf(t,d) = bobot *term*

tf(t,d) = *term* frekuensi kata t pada dokumen d

idf(t) = *Invers* dokumen frekuensi kata t

kemudian utnuk mencari nilai IDF menggunakan persamaan berikut :

$$idf(t) = \log\left(\frac{N+1}{Nt+1}\right) + 1 \quad (6)$$

Keterangan :
t = *term*
N = total keseluruhan dokumen
Nt = total dokumen dengan *term*

Pada ekstraksi fitur TF-IDF pada penelitian ini berikut parameter yang digunakan dari *library* scikit-learn :

a. ngram_range : pada parameter ngram_range digunakan untuk menetapkan rentang nilai n yang digunakan dalam proses ekstraksi fitur. Pengguna dapat menentukan nilai minimum dan maksimum dari n-gram yang diinginkan. Penelitian ini menggunakan nilai (1,1).
b. norm : pada parameter norm proses normalisasi dilakukan dengan cara menghitung jumlah kuadrat dari setiap elemen pada vektor, mengambil akar kuadrat, dan dari hasil penjumlahan tersebut, dan kemudian membagi setiap nilai elemen pada vektor dengan nilai akar kuadrat tersebut. Penelitian ini menggunakan nilai l2.

b. Normalisasi Tf-IDF

Setelah mendapatkan nilai TF-IDF kemudian proses normalisasi. Normalisasi data merupakan metode yang digunakan untuk mengubah skala nilai data menjadi rentang 0 sampai 1. Proses ini penting sebelum melakukan data *mining* agar tidak terjadi dominasi oleh parameter tertentu. Dalam penelitian ini, digunakan metode L2-*Norm* dengan rumus berikut:

$$v_{norm} = \frac{\vec{v}}{||v||} = \frac{\vec{v}}{\sqrt{v1^{2+v^2+v^2} \ldots\ldots v^n}} \quad (7)$$

Keterangan:
$\vec{v}$ = Nilai vektor yang dinormalisasikan
$\overrightarrow{||v||}$p = $\vec{v}$ pada dokumen dengan nilai p = 2

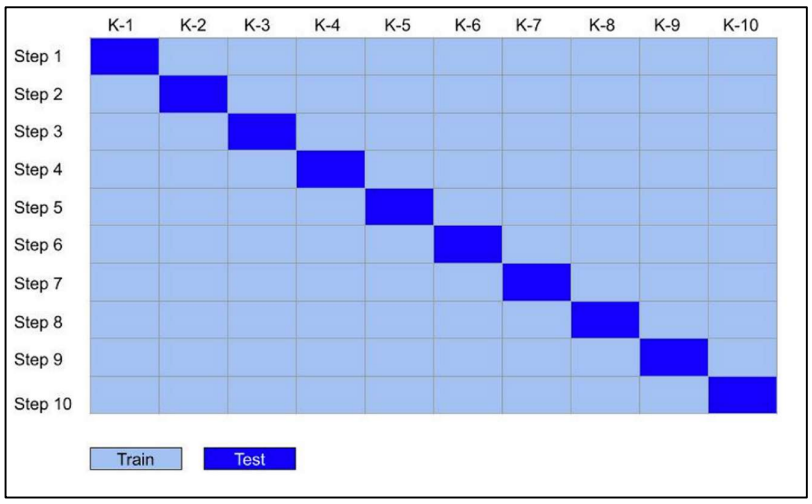
c. Klasifikasi Algoritma *K-Nearest Neighbors*

K-Nearest Neighbor (KNN) adalah algoritma yang mampu mengklasifikasikan objek berdasarkan data pelatihan yang terkait dengan objek tersebut (Barus, 2022). Prinsip kerja *K-Nearest Neighbor* (K-NN) adalah menemukan jarak terdekat antara data yang akan dievaluasi dengan k tetangga terdekat dalam data *training.* Dalam penggunaannya, algoritma KNN memiliki beberapa keunggulan, termasuk kesederhanaan dan kemudahan pemahaman, sifat non-parametrik, kemudahan dalam penyesuaian model, serta ketahanannya terhadap *noise*. (Saifurridho, Martanto & Hayati, 2024). Pada penelitian ini setelah tahap ekstraksi fitur TF-IDF, dilakukan analisis terhadap teknik klasifikasi yang telah diperoleh dengan memanfaatkan data *training*. Berikut klasifikasi algoritma *K-Nearest Neighbor* (KNN) yang menggunakan *library* sklearn dengan parameter dibawah ini (*Source code* terdapat pada lampiran 30):

a. n_neighbor : merupakan jumlah tetangga yang akan digunakan untuk menentukan label kelas suatu sampel. Pada penelitian ini nilai k yang digunakan yaitu k = 10.
b. P : parameter yang digunakan untuk menentukan jenis jarak yang digunakan. Pada penelitian ini menggunakan jarak *Euclidean*.

6. Evaluasi KNN

Pada tahap ini evaluasi *f1-score* hampir sama dengan evaluasi *WordNet*, Akan tetapi tahapan evaluasi ini *f1-score* menggunakan *K-fold Cross Validation*. Dalam penggunaan rumus, evaluasi *F1-Score* tahap ini terdapat pada rumus (1) - (4) . *K-fold cross validation* merupakan salah satu dari teknik yang difungsikan untuk memilah data menjadi data *training* serta data *testing* (Ridwansyah, 2022). Nilai yang akan diperoleh dengan *K-Fold Cross Validation* dengan menerapkan beberapa nilai K yang dilakukan sebanyak 10-*fold validation* (Fikriani, Asror, dan Murti 2019). 10-*fold cross validation* digunakan dalam membagi *dataset* ke data *training* dan data *testing*. Evaluasi tersebut bertujuan untuk menilai seberapa akurat sebuah model yang telah dibuat. nilai K= 10, Angka 10 digunakan sebagai batas akhir karena metode 10-*fold cross validation* merupakan metode yang paling umum digunakan dan memiliki estimasi performa yang akurat (Refaeilzadeh, et al., 2020)Berikut penggunaan *K-Fold Cross Validation* dengan nilai K= 10 pada gambar 2.4.



Gambar 2. 4 *Cross Validation k = 10*

Berikut parameter yang digunakan dalam proses *cross validation* (*Source code* terdapat pada lampiran 28).

a. y_true : Parameter ini sekumpulan nilai aktual atau target dari variabel dependen yang ada dalam dataset, dan dipakai dalam evaluasi.
b. y_pred : parameter ini digunakan untuk membandingkan hasil prediksi model dengan nilai target sebenarnya(y_true).
c. *Average* : parameter ini digunakan untuk mencari rata-rata dalam perhitungan *f1-Score.*

BAB III

HASIL DAN PEMBAHASAN

3.1 Hasil

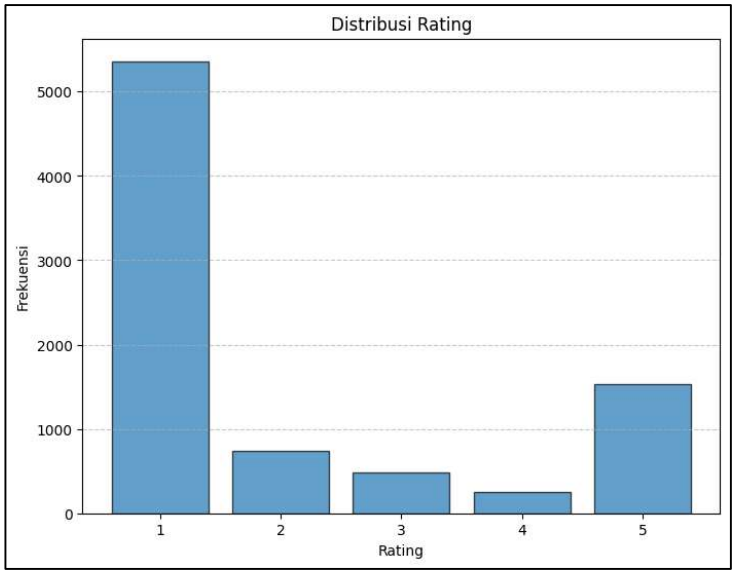
3.1.1 Pengambilan Data

Tahapan pertama yaitu pengambilan data melalui proses *scraping*. Pada prosesnya data *scraping* dimasukan ke *dataframe pandas*. Gambar 3.1 merupakan hasil dari *scraping* yang telah dilakukan proses *filltering* kolom, sehingga terdapat 5 kolom yang digunakan.

	userName	score	at	content	thumbsUpCount
6671	Nency Miranda	5	2024-02-05 14:07:12	Mantap	0
5783	ichal sabiel	1	2024-02-05 14:06:17	Sudah tau pemakainya masyarakat biasa Malah sp...	0
7909	Evandra Aditya	1	2024-02-05 14:06:16	Apk nya nggk bisa buat log in	0
6307	Ganesh insann	1	2024-02-05 14:05:38	Susah masuk dih	0
8008	Yohana Frediana	1	2024-02-05 14:04:32	Tidak bisa masuk inisiliasi	0

Gambar 3. 1 Hasil *Scraping* Data

Pada 5 kolom diatas terdapat nama pengguna, rating, waktu, komentar, dan jumlah like. Selanjutnya, gambar 3.2 merupakan hasil dari distribusi rating yang digambarkan dengan visualisasi *chart*. Data yang didapatkan dari proses *scraping* sebelumnya yaitu 8358 *dataset*. kemudian, dari hasil tersebut terdapat frekuensi data terbanyak pada *rating* 1, yaitu sebanyak 5361 ulasan. Dan frekuensi data yang paling sedikit terdapat pada rating 4, sebanyak 255 ulasan.



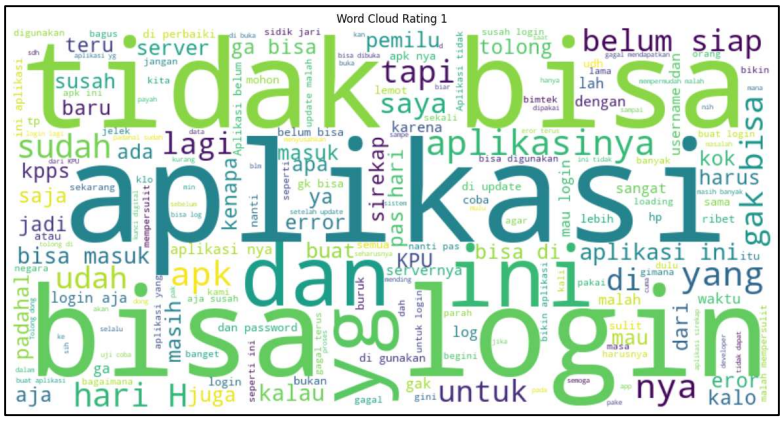
Gambar 3. 2 Hasil Distribusi *Rating*

Penjelasan hasil frekuensi setiap *rating* terdapat pada tabel 3.1 dibawah ini. Dari tabel tersebut dapat dilihat hasil frekuensi setiap rating mulai dari yang terbanyak, sedang dan yang paling sedikit.

Tabel 3. 1 Frekuensi Masing-Masing *Rating*

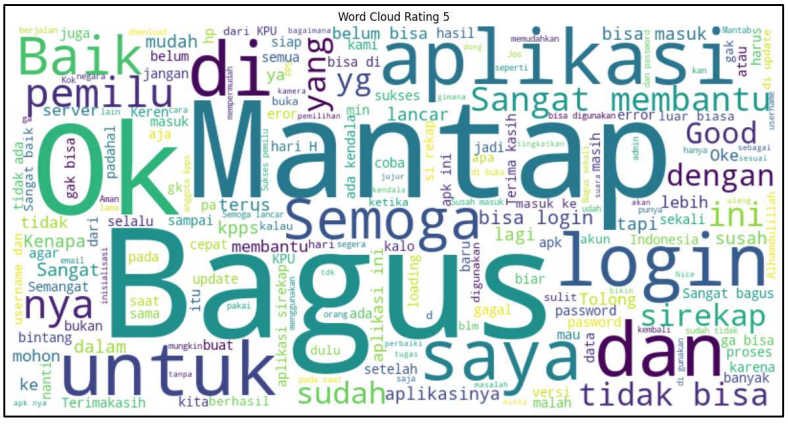
Rating	**Frekuensi**
1	5361
2	736
3	478
4	255
5	1528
Total	8358

Dari tabel diatas, terdapat frekuensi data ulasan dari setiap *rating* memiliki jumlah yang berbeda. *Rating* 1 memiliki frekuensi 5361 data ulasan, kemudian pada *rating* 2 terdapat 736 ulasan, *rating* 3 terdapat 478 ulasan, *rating* 4 terdapat 255 ulasan, dan *rating* 5 yaitu sebanyak 1528 ulasan.



Gambar 3. 3 Hasil *Wordcloud rating* 1

Selanjutnya, untuk memvisualisasikan frekuensi kata yang sering muncul pada ulasan *rating* 1, dengan menggunakan teknik "*wordcloud"*. Pada gambar 3.3 diatas, frekuensi kata yang sering muncul dan yang paling banyak dibahas pada *rating* 1 adalah "aplikasi", "bisa","login", "tidak", "bisa" maka konteks yang sering muncul pada *rating* 1 yaitu "tidak bisa login aplikasi".



Gambar 3. 4 Hasil *Wordcloud rating* 5

Sedangkan untuk frekuensi kata yang paling sering muncul pada ulasan *rating* 5 yaitu "Ok", "Bagus" dan "Mantap". Maka konteks yang sering muncul dan yang paling banyak dibahas pada *rating* 5 yaitu "aplikasinya bagus".

3.1.2 Pra Proses Data

Tahap pertama pra proses yaitu *add id.* Proses ini memberikan identifikasi unik pada setiap baris. Panjang *id* ditentukan berdasarkan jumlah digit dari total jumlah baris. Berdasarkan gambar dibawah, jika data memiliki baris 8358 baris, maka *ID* nya ‘d0001’, ‘d0002’ hingga ‘d8357’. Proses ini memastikan bahwa setiap baris memiliki *id* yang mudah diidentifikasi dan terstruktur secara konsisten.

	ID	Nama	Rating	Waktu	Komentar	Like
0	d0001	Nency Miranda	5	05/02/2024 14:07	Mantap	0.0
1	d0002	ichal sabiel	1	05/02/2024 14:06	Sudah tau pemakainya masyarakat biasa Malah sp...	0.0
2	d0003	Evandra Aditya	1	05/02/2024 14:06	Apk nya nggk bisa buat log in	0.0
3	d0004	Ganesh insann	1	05/02/2024 14:05	Susah masuk dih	0.0
4	d0005	Yohana Frediana	1	05/02/2024 14:04	Tidak bisa masuk inisiliasi	0.0
...	...	...	...	...	...	...
8353	d8354	Tn A Muntaha	5	23/01/2024 09:05	Semoga Tambah Baik,Mudah Dalam Input Dan Share...	21.0
8354	d8355	Adhi Nugroho	1	23/01/2024 08:51	Cara masuk nya bagaimana? Pakai email kok gag ...	448.0
8355	d8356	TATANG RUSDIANA	5	23/01/2024 07:03	Mbuh	6.0
8356	d8357	vie	5	23/01/2024 06:50	Semoga tidak seperti sirekap 2020 yang saat di...	42.0
8357	d8358	Ahmad ferdiansyah	5	23/01/2024 04:26	Semoga dengan adanya aplikasi ini pemilu akan ...	186.0

[8358 rows x 6 columns]

Gambar 3. 5 Hasil *Add Id*

Gambar 3.5 diatas merupakan hasil dari *add id,* dari 8358 baris, dan 6 kolom. Pelabelan berdasarkan komentar dari pengguna “Sirekap 2024”. Label *id* yang dimulai dari baris 0 hingga 8357.

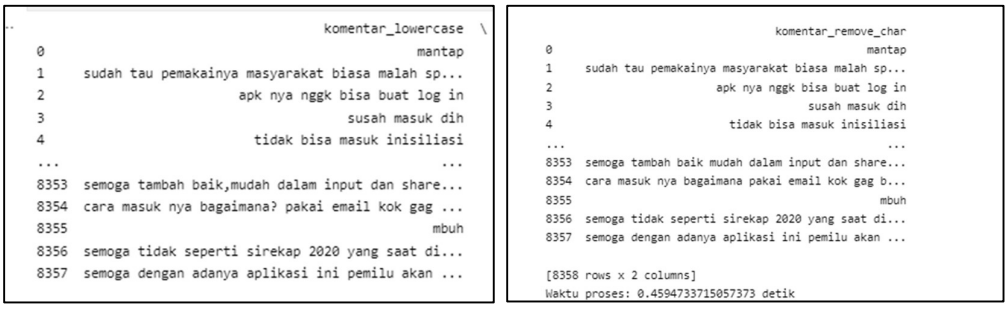
	Mantap
1	Sudah tau pemakainya masyarakat biasa Malah sp...
2	Apk nya nggk bisa buat log in
3	Susah masuk dih
4	Tidak bisa masuk inisiliasi
...	...
8353	Semoga Tambah Baik,Mudah Dalam Input Dan Share...
8354	Cara masuk nya bagaimana? Pakai email kok gag ...
8355	Mbuh
8356	Semoga tidak seperti sirekap 2020 yang saat di...
8357	Semoga dengan adanya aplikasi ini pemilu akan ...

	komentar_lowercase
0	mantap
1	sudah tau pemakainya masyarakat biasa malah sp...
2	apk nya nggk bisa buat log in
3	susah masuk dih
4	tidak bisa masuk inisiliasi
...	...
8353	semoga tambah baik,mudah dalam input dan share...
8354	cara masuk nya bagaimana? pakai email kok gag ...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

[8358 rows x 2 columns]
Waktu proses: 0.0249330997467041 detik

Gambar 3. 6 Sebelum dan Setelah *Lowercase*

Tahap berikutnya mengubah semua data di kolom ‘komentar’ menjadi huruf kecil. Seperti pada komentar dengan kata “Semoga” diubah menjadi “semoga”, huruf “S” yang awalnya huruf kapital berubah menjadi huruf kecil “s”. Pada penelitian ini hanya menggunakan kolom ‘komentar’ untuk proses klasifikasi teks dari ulasan pengguna. Gambar 3.6 gambar sebelah kiri merupkan sebelum *lowercase dataset* dan sebelah kanan merupakan hasil setelah proses *lowercase*. Proses ini membutuhkan durasi waktu untuk merubah data teks komentar menjadi huruf kecil selama 0,009 detik.



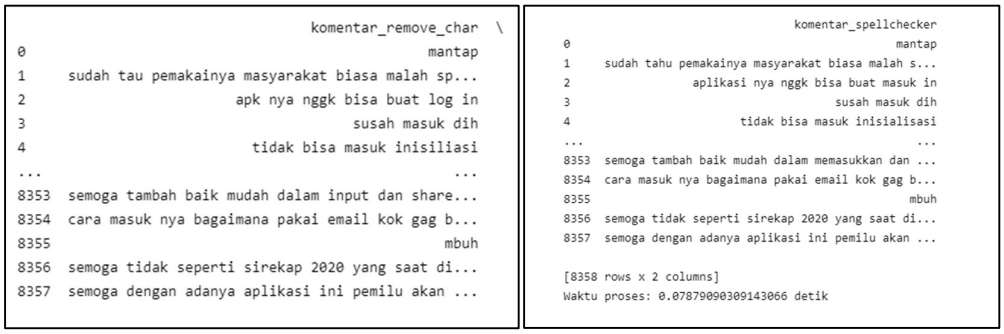
	komentar_lowercase
0	mantap
1	sudah tau pemakainya masyarakat biasa malah sp...
2	apk nya nggk bisa buat log in
3	susah masuk dih
4	tidak bisa masuk inisiliasi
...	...
8353	semoga tambah baik,mudah dalam input dan share...
8354	cara masuk nya bagaimana? pakai email kok gag ...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

	komentar_remove_char
0	mantap
1	sudah tau pemakainya masyarakat biasa malah sp...
2	apk nya nggk bisa buat log in
3	susah masuk dih
4	tidak bisa masuk inisiliasi
...	...
8353	semoga tambah baik mudah dalam input dan share...
8354	cara masuk nya bagaimana pakai email kok gag b...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

[8358 rows x 2 columns]
Waktu proses: 0.4594733715057373 detik

Gambar 3. 7 Hasil Sebelum dan Setelah *Remove Char*

Tahap berikutnya yaitu membersihkan data dari karakter-karakter yang tidak diperlukan seperti simbol, emoji, angka dengan menggunakan fungsi *'remove_unnecesary_char'*. Dari gambar 3.7 diatas pada baris 8353 yang sebelumnya terdapat tanda baca ',' dan pada baris 8354 yang sebelumnya terdapat tanda baca'?' setelah proses *remove char* tanda tersebut hilang. Gambar 3.7 sebelah kiri merupakan sebelum *remove char* dan gambar sebelah kanan merupakan hasil dari *remove char.* Durasi proses ini selama 0,459 detik.



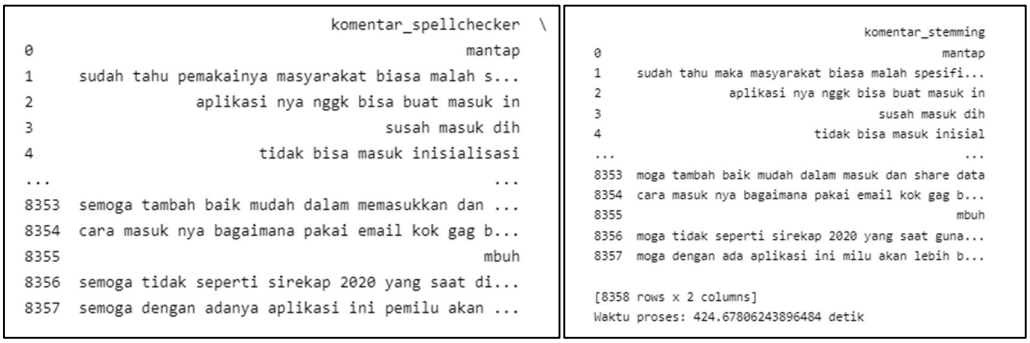
	komentar_remove_char
0	mantap
1	sudah tau pemakainya masyarakat biasa malah sp...
2	apk nya nggk bisa buat log in
3	susah masuk dih
4	tidak bisa masuk inisiliasi
...	...
8353	semoga tambah baik mudah dalam input dan share...
8354	cara masuk nya bagaimana pakai email kok gag b...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

	komentar_spellchecker
0	mantap
1	sudah tahu pemakainya masyarakat biasa malah s...
2	aplikasi nya nggk bisa buat masuk in
3	susah masuk dih
4	tidak bisa masuk inisialisasi
...	...
8353	semoga tambah baik mudah dalam memasukkan dan ...
8354	cara masuk nya bagaimana pakai email kok gag b...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

[8358 rows x 2 columns]
Waktu proses: 0.07879090309143066 detik

Gambar 3. 8 Hasil Sebelum dan Setelah *Spellchecker*

Selanjutnya pada tahap ini sebelum masuk tahap *spellchecker,* terlebih dahulu membuat kamus tidak baku yang berlandaskan dari Kamus Bahasa Indonesia (KBI) yang kemudian dimasukkan kedalam proses *spellchecker*. Sebagai contoh pada salah satu ulan pada gambar 3.8, sebelah kiri kata yang sebelumnya "apk" berubah menjadi "aplikasi" . Proses *spellchecker* membutuhkan waktu proses kurang lebih 0,078 detik.



	komentar_spellchecker
0	mantap
1	sudah tahu pemakainya masyarakat biasa malah s...
2	aplikasi nya nggk bisa buat masuk in
3	susah masuk dih
4	tidak bisa masuk inisialisasi
...	...
8353	semoga tambah baik mudah dalam memasukkan dan ...
8354	cara masuk nya bagaimana pakai email kok gag b...
8355	mbuh
8356	semoga tidak seperti sirekap 2020 yang saat di...
8357	semoga dengan adanya aplikasi ini pemilu akan ...

	komentar_stemming
0	mantap
1	sudah tahu maka masyarakat biasa malah spesifi...
2	aplikasi nya nggk bisa buat masuk in
3	susah masuk dih
4	tidak bisa masuk inisial
...	...
8353	moga tambah baik mudah dalam masuk dan share data
8354	cara masuk nya bagaimana pakai email kok gag b...
8355	mbuh
8356	moga tidak seperti sirekap 2020 yang saat guna...
8357	moga dengan ada aplikasi ini milu akan lebih b...

[8358 rows x 2 columns]
Waktu proses: 424.67806243896484 detik

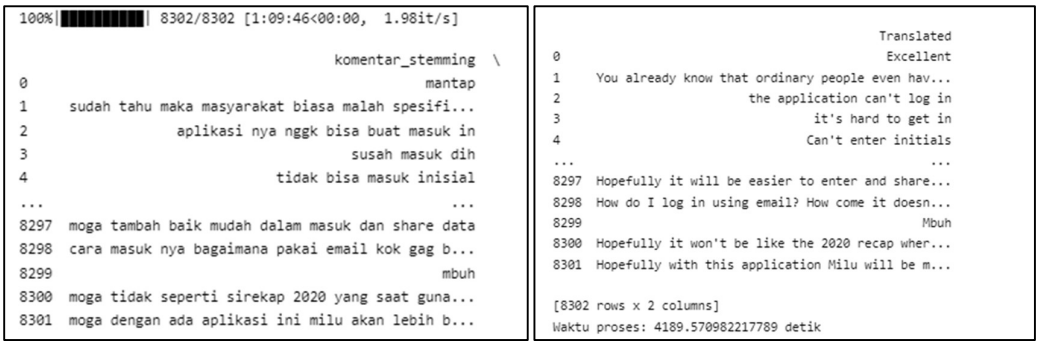
Gambar 3. 9 Hasil Sebelum dan Setelah *Stemming*

Pada tahapan ini merubah setiap kata pada data menjadi bentuk dasar. Bentuk dasar kata yang diubah berlandaskan dari kamus tidak baku yang telah dibuat dari proses *spellchecker*. Proses *stemming* ini guna menormalisasi data teks. Pada gambar 3.9 sebelah kanan merupakan hasil

setelah *stemming,* terdapat pada ulasan yang awalnya kata "semoga" berubah menjadi kata dasarnya yaitu "moga". Proses ini memakan waktu kurang lebih selama 7 menit.

3.1.3 WordNet

Pada tahapan ini peneliti memasukkan *library `deep_translator`* untuk menerjemahkan teks dari bahasa Indonesia ke bahasa Inggris. Karena *WordNet* hanya dapat memproses teks dalam bahasa Inggris.



```
100%|██████████| 8302/8302 [1:09:46<00:00,  1.98it/s]

                                  komentar_stemming  \
0                                             mantap
1     sudah tahu maka masyarakat biasa malah spesifi...
2                 aplikasi nya nggk bisa buat masuk in
3                                   susah masuk dih
4                             tidak bisa masuk inisial
...                                                ...
8297  moga tambah baik mudah dalam masuk dan share data
8298  cara masuk nya bagaimana pakai email kok gag b...
8299                                             mbuh
8300  moga tidak seperti sirekap 2020 yang saat guna...
8301  moga dengan ada aplikasi ini milu akan lebih b...
```

```
                                           Translated
0                                           Excellent
1     You already know that ordinary people even hav...
2                         the application can't log in
3                                 it's hard to get in
4                               Can't enter initials
...                                              ...
8297  Hopefully it will be easier to enter and share...
8298  How do I log in using email? How come it doesn...
8299                                             Mbuh
8300  Hopefully it won't be like the 2020 recap wher...
8301  Hopefully with this application Milu will be m...

[8302 rows x 2 columns]
Waktu proses: 4189.570982217789 detik
```

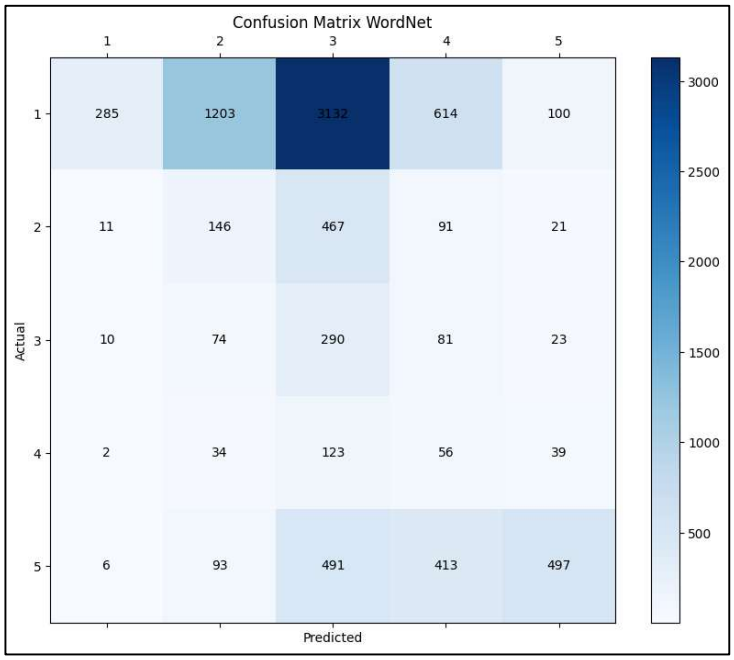
Gambar 3. 10 Hasil Sebelum dan Setelah *Translated*

Pada Gambar 3.10 sebelah kiri merupakan proses sebelum *translated* sedangkan gambar sebelah kanan merupakan hasil setelah *translated* dari bahasa Indonesia ke bahasa Inggris. Proses ini memakan waktu yang paling lama kurang lebih selama 1 jam 9 menit.

	ID	Rating	komentar_stemming	Translated	Score_Wordnet	Sentiment_Score
0	d0001	5	mantap	Excellent	5	1.000000
1	d0002	1	sudah tahu maka masyarakat biasa malah spesifi...	You already know that ordinary people even hav...	3	-0.045000
2	d0003	1	aplikasi nya nggk bisa buat masuk in	the application can't log in	3	0.000000
3	d0004	1	susah masuk dih	it's hard to get in	2	-0.291667
4	d0005	1	tidak bisa masuk inisial	Can't enter initials	3	0.000000

Gambar 3. 11 Hasil Hitung *Fungsi Score*

Selanjutnya pada hitung *score,* kolom '*komentar_stemming'* berisi ulasan asli pengguna yang telah melalui proses *stemming* sebelumnya. Dari gambar 3.11 diatas pada *id* 'd0001' dengan *'Score_Wordnet'* 5 dan *'Sentiment_Score'* '1,000000' yang berarti ulasan tersebut positif, dan pada *id* 'd0003' dengan *'Score_Wordnet'* 2 dan *'Sentiment_Scorre'* '-0,291667' yang berarti ulasan tersebut negatif, dan *id* 'd0005' dengan *'Score_Wordnet'* 3 dan *'Sentiment_Scorre'* '0,000000' yang berarti ulasan tersebut netral.



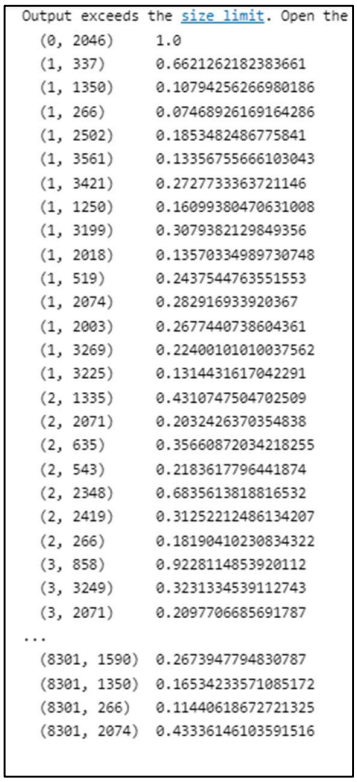
Gambar 3. 12 Hasil *Confusion Matrix Wordnet*

Selanjutnya, pada gambar 3.12 klasifikasi tertinggi terdapat pada warna biru gelap (*Navy)* yaitu sebanyak 3132 data *test* dengan nilai aktual *rating* 1, yang diklasifikasikan kedalam *rating* 3. Untuk klasifikasi paling rendah terdapat pada warna yang paling cerah (*light blue)* yaitu 2 data *test* dengan nilai aktual *rating* 4, yang diklasifikasikan pada *rating* 1.

Dari hasil gambar diatas, *rating* 1 yang berhasil diklasifikasikan benar sebanyak 285 data *test*, *rating* 2 yang berhasil diklasifikasikan benar sebanyak 146 data *test*, *rating* 3 berhasil diklasifikasikan benar sebanyak 290 data *test*, pada *rating* 4 yang berhasil diklasifikasikan benar 56 data *test*, dan yang terakhir pada *rating* 5 yang berhasil diklasifikasikan benar yaitu sebanyak 497 data *test*. Maka disimpulkan dari klasifikasi tersebut, menunjukan banyak kesalahan besar, terutama dalam memprediksi *rating* 1 yang diklasifikasikan kedalam *rating* 3 yaitu sebanyak 3132 data *test*, sehingga hasil nilai performa menggunakan *F1-Score* sebesar 17,50%.

3.1.4 Klasifikasi Algoritma *K-Nearest Neighbor*

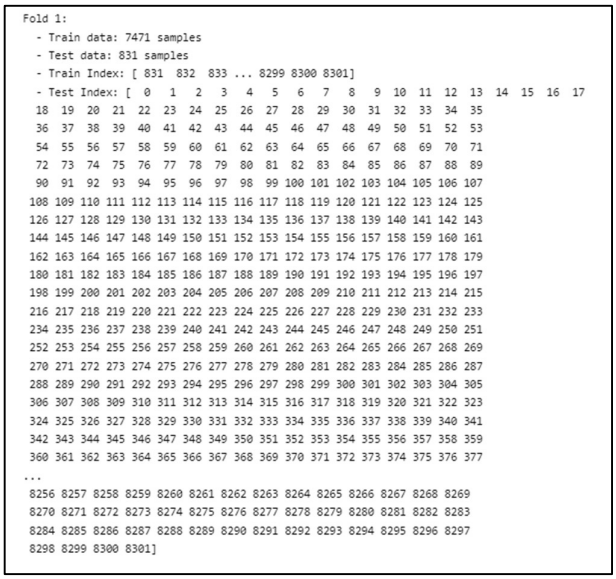
Pada tahap pertama klasifikasi ini, yaitu ekstraksi fitur TF-ID, proses dilakukan menggunakan *library 'TfidfVectorizer'* yang berfungsi untuk mengubah teks menjadi fitur numerik berbasis TF-IDF dan untuk memasukkan nilai X.



Gambar 3. 13 Hasil Ekstraksi Fitur TF-IDF

Dari gambar 3.13 setiap baris menampilkan indeks*, term*, kemudian nilai TF-IDF, seperti pada baris awal terdapat *index* 0 dengan kata 'mantap' berada pada *term* 2046 dengan nilai TF-IDF 1,0. Pada baris kedua terdapat *index* 1 dengan kata 'badut', berada pada *term* 337 dan nilai TF-IDF 0,6621.

Selanjutnya yaitu tahap *cross validation* yang menampilkan isi data setiap pembagian *fold* yang berisi *train* data, *test* data, *train index* dan *test index*. Terdapat perbedaan hasil pada *test* data di beberapa *fold*. *Fold* 1-2 memiliki *test* data 831 *samples,* sedangkan *fold* 3-10 memiliki *test* data 830 *samples*. Hasil *test index* yang berbeda karena pembagian data ke *fold* yang berbeda-beda.



Gambar 3. 14 Hasil Nilai *K-Fold 1*

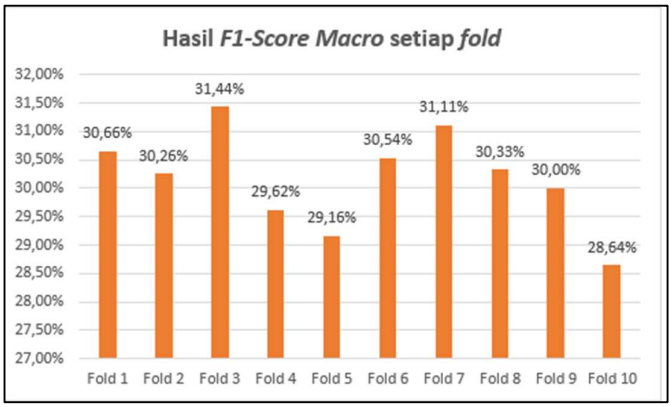
Dari gambar 3.14 yang merupakan hasil dari nilai *K-fold* 1 terdapat *train* data sebanyak 7471 *samples* sebagai data latih model, *test* data sebanyak 831 *samples* sebagai data uji model, untuk *test* data yang digunakan pada *fold* 1 dari *index* 0 sampai 8301.

Gambar 3. 15 Hasil Nilai *K-Fold 10*

Sedangkan hasil dari *K-fold 10* pada gambar 3.15, *train data* 7472 *samples* sebagai data latih model, *test* data 830 *samples* sebagai data uji model, untuk *test* data yang digunakan pada *fold* 10 dari *index* 7472 sampai 8301. Fungsi dari proses *cross validation* yaitu untuk mengevaluasi performa model klasifikasi yang digunakan untuk mengklasifikasikan sentimen dari teks (ulasan "Sirekap 2024").

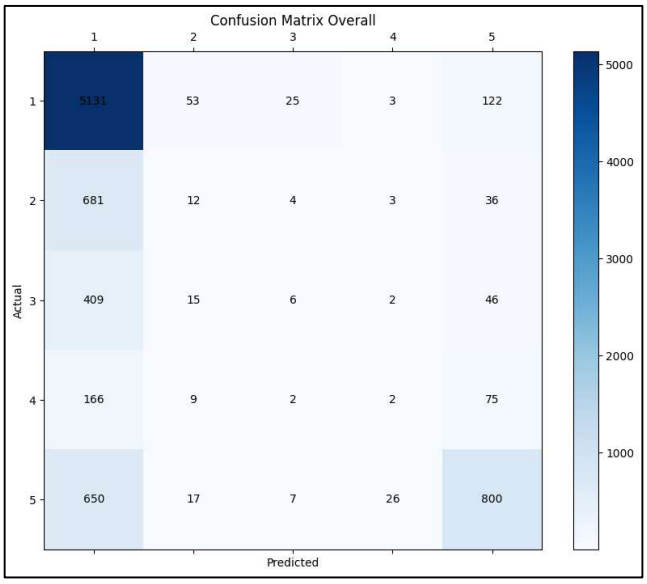
Tabel 3. 2 Hasil *F1 Score* Setiap *Fold*

Fold	***F1-Score***
Fold-1	30,66%
Fold-2	30,26%
Fold-3	31,44%
Fold-4	29,62%
Fold-5	29,16%
Fold-6	30,54%
Fold-7	31,11%
Fold-8	30,33%
Fold-9	30,00%
Fold-10	28,64%



Gambar 3. 16 Hasil *F1-Score Macro* Setiap *Fold*

Dari tabel 3.2 dan gambar 3.16 merupakan hasil dari setiap *fold*, hasil yang paling tinggi terdapat pada *fold* 3 yaitu sebesar 31,44%, sedangkan hasil yang paling rendah yaitu pada *fold* 10 sebesar 28,64%,



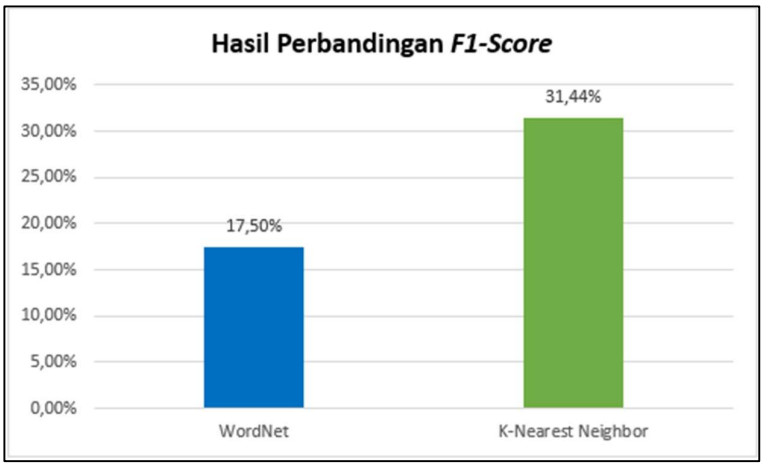
Gambar 3. 17 Hasil *Confusion Matrix Overall*

Selanjutnya, pada gambar 3.17 klasifikasi tertinggi terdapat pada warna biru gelap (*Navy)* yaitu sebanyak 5131 data *test* dengan nilai aktual *rating* 1, yang diklasifikasikan kedalam *rating* 1. Untuk klasifikasi paling rendah terdapat pada warna yang paling cerah (*light blue)* yaitu 2 data *test* dengan nilai aktual *rating* 4, yang diklasifikasikan pada *rating* 3.

Dari hasil gambar diatas, *rating* 1 yang berhasil diklasifikasikan benar sebanyak 5131 data *test*, *rating* 2 yang berhasil diklasifikasikan benar sebanyak 12 data test, *rating* 3 berhasil diklasifikasikan benar sebanyak 6 data *test*, pada rating 4 yang berhasil diklasifikasikan benar 2 data *test*, dan yang terakhir pada *rating* 5 yang berhasil diklasifikasikan benar yaitu sebanyak 800 data *test*. Maka disimpulkan dari klasifikasi tersebut, menunjukan jumlah prediksi benar pada diagonal relatif tinggi. terutama dalam memprediksi *rating* 1 yang diklasifikasikan kedalam *rating* 1 yaitu sebanyak 5131 data *test*. Sehingga hasil nilai performa menggunakan *F1-Score* sebesar 31,44%.

Tabel 3. 3 Hasil Perbandingan 2 Metode

Metode	Hasil
WordNet	17,50%
K-Nearest Neighbor	31,44%



Gambar 3. 18 Hasil Perbandingan 2 Metode

Dari tabel 3.3 dan gambar 3.18 hasil yang diperoleh dari perbandingan kedua metode tersebut dalam melakukan klasifikasi data ulasan aplikasi "Sirekap 2024" menggunakan evaluasi *f1-score* menunjukkan, hasil evaluasi metode *K-Nearest Neighbor* lebih baik dibandingkan dengan hasil evaluasi metode *WordNet.*

3.2 Pembahasan

Berdasarkan penyajian data dan hasil analisis data, maka pada bab ini akan di deskripsikan temuan penelitian dan hasil pengujian yang telah diuji pada bab sebelumnya guna menjawab rumusan masalah yaitu, bagaimana hasil komparasi metode *WordNet* dan klasifikasi *K-Nearest Neighbor* menggunakan evaluasi *f1-score* pada ulasan aplikasi sirekap 2024. Pengambilan data pada penelitian ini yang berhasil di *scraping* sebanyak 8358 dataset dan menggunakan 5 kolom, namun data yang di*scraping* tidak seimbang, seperti yang dapat dilihat pada distribusi *rating* gambar 3.2,

dimana *rating* satu memiliki dataset yang paling banyak yang berarti pengguna sangat tidak puas dengan aplikasi atau pelayanannya.

Pada tahap pra proses dalam penelitian ini untuk mempersiapkan data agar siap digunakan dalam analisis dan modeling. *Spellchecker* pada penelitian ini untuk memperbaiki kesalahan pengejaan dalam teks, namun pada hasil *spellchecker* terdapat kata yang tidak berhasil berubah sesuai dengan kamus yang telah dibuat, terdapat pada gambar 3.8 baris 2 dengan kata 'nggk' dan 8335 dengan kata 'mbuh' . Kemudian *stemming* yang berfungsi untuk merubah setiap kata pada data menjadi bentuk dasar, tetapi karena memotong kata ke bentuk dasarnya tanpa memperhatikan konteks, bisa menyebabkan kehilangan makna spesifik dari kata tersebut, terdapat pada gambar 3.9 baris 8356 dengan kata 'semoga' kemudian menjadi kata 'moga' dan baris 8357 dengan awalnya kata 'pemilu' menjadi kata 'milu'.

Pada proses *WordNet*, dalam mengklasifikasikan hanya dengan melakukan perhitungan menggunakan nilai polaritas, dan dari nilai tersebut ulasan diklasifikasikan kedalam peringkat sesuai nilai yang dihasilkan, namun pada proses *translated WordNet* termasuk salah satu tahap yang memakan waktu cukup lama yaitu selama 1 jam 9 menit. Hasil performa wordnet *menggunakan f1-score* sangat rendah yaitu hanya sebesar 17,50% karena dalam mengklasifikasian menggunakan evaluasi *f1-score* terdapat banyak kesalahan dalam memprediksi, yang dapat dilihat pada *confusion matriks* gambar 3.12, yaitu terutama dalam memprediksi *rating* 1 yang diklasifikasikan kedalam rating 3 yaitu sebanyak 3132 data *test* sedangkan yang diprediksi benar dalam *rating* 1 sampai 5 hanya sebanyak 1274 data *test*, dan yang paling banyak diklasifikasikan benar yaitu pada *rating* 1, sebanyak 285 data *test.*

Pada tahap KNN dalam terdapat tahap *cross validation* dengan nilai K= 10, karena pada penelitian ini data yang digunakan tidak seimbang sehingga tahap *cross validation* untuk mengurangi resiko *overfitting*. hasil performa klasifikasi knn menggunakan *f1-score* lebih baik dari *WordNet* yaitu sebesar 31,44% karena dalam mengklasifikasikan menggunakan evaluasi *f1-score* lebih banyak data yang diprediksi benar, terutama dalam memprediksi *rating* 1 yang diklasifikasikan kedalam *rating* 1, yaitu sebanyak 5131 data *test*, sedangkan yang prediksi salah paling banyak pada *rating* 5 yang diklasifikasikan kedalam *rating* 1 yaitu sebanyak 650 data *test* . Hasil klasifikasi dapat dilihat pada *confusion matriks* gambar 3.17, untuk melihat hasil perbandingan dari dua metode terdapat pada tabel 3.3 dan gambar 3.18.

BAB IV

KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan analisis yang telah dilakukan oleh peneliti, dalam rangka menjawab tujuan penulisan yang telah dipaparkan pada pendahuluan, peneliti menarik kesimpulan perbandingan dua metode tersebut. Dari hasil analisis, metode *K-Nearest Neighbor* dalam klasifikasi data menggunakan *f1-score* diperoleh hasil sebesar 0,31442430244371733 atau 31%. Sedangkan hasil metode *WordNet* lebih rendah dari K-NN, yaitu sebesar 0,17505636718982256 atau 18%. Hasil komparasi antara metode *WordNet* dan K-NN menunjukkan bahwa K-NN memiliki keunggulan dalam hal evaluasi menggunakan *F1-Score*.

Metode *WordNet* yang diterapkan dalam penelitian ini mampu mengklasifikasikan ulasan aplikasi "Sirekap 2024" berdasarkan indeks sentimen. Hasil analisis menunjukkan bahwa *WordNet* efektif dalam mengidentifikasi sentimen berdasarkan nilai polaritas. Namun, performa *WordNet* masih terdapat kekurangan karena *WordNet* merupakan leksikon yang hanya dirancang untuk bahasa Inggris. Sedangkan metode K-NN dengan ekstraksi fitur TF-IDF menunjukkan bahwa K-NN memiliki keunggulan dalam hal evaluasi menggunakan *F1-Score*.

Penelitian ini dapat memberikan wawasan yang berharga bagi pengembang aplikasi "Sirekap 2024" dalam memahami sentimen pengguna. Informasi yang diperoleh dari analisis sentimen ini dapat digunakan untuk meningkatkan kualitas aplikasi, menyesuaikan fitur-fitur yang ada, dan merespons ulasan pengguna dengan lebih baik. Dengan memahami pola sentimen dan umpan balik pengguna, pengembang dapat membuat keputusan yang lebih tepat untuk pengembangan dan perbaikan aplikasi di masa depan.

4.2 Saran

Dari hasil penelitian mengenai "Perbandingan Analisis *WordNet* Dan *K-Nearest Neighbor* pada Ulasan Aplikasi "Sirekap 2024", karena data yang digunakan pada penelitian ini tidak seimbang, maka disarankan bagi peneliti selanjutnya untuk melakukan tahapan *undersampling* guna untuk menangani masalah ketidakseimbangan pada data.

DAFTAR RUJUKAN

Afdal, M. and Waroka, L. (2022) 'Shopee Application Review Classification Using Probabilistic Neural Network Algorithm And K-Nearest Neighbor', *Indonesian Journal of Informatic Research and Software Engineering*, 2(1), pp. 49–58.

Amly, M.R., Yusra, Y. and Fikry, M. (2023) 'Penerapan Metode Naïve Bayes Classifier Pada Klasifikasi Sentimen Terhadap Anies Baswedan Sebagai Bakal Calon Presiden 2024', *Jurnal Sistem Komputer dan Informatika (JSON)*, 4(4), p. 621. Available at: https://doi.org/10.30865/json.v4i4.6214.

Amrullah, A.Z., Sofyan Anas, A. and Hidayat, M.A.J. (2020) 'Analisis Sentimen Movie Review Menggunakan Naive Bayes Classifier Dengan Seleksi Fitur Chi Square', *Jurnal*, 2(1), pp. 40–44. Available at: https://doi.org/10.30812/bite.v2i1.804.

Barus, S.G. (2022) 'Klasifikasi Sentimen Data Tidak Seimbang Menggunakan Algoritma Smote Dan K-Nearest Neighbor Pada Ulasan Pengguna Aplikasi Pedulilindungi', *Senamika*, pp. 162–173.

Dwi Laxmi, M. and Ali Fauzi, M. (2019) 'Query Expansion Pada Sistem Temu Kembali Informasi Berbahasa Indonesia Dengan Metode Pembobotan TF-IDF Dan Algoritme Cosine Similarity Berbasis Wordnet', *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 3(1), pp. 2548–964. Available at: http://j-ptiik.ub.ac.id.

Fikriani, A., Asror, I. and Murti, Y.R. (2019) 'Klasifikasi Kepribadian Berdasarkan Data Twitter dengan Menggunakan Metode Support Vector Machine', *e-Proceeding of Engineering*, 6(3), pp. 10436–10450.

Gauru, C.C., Martini, R. and Alfirdaus, L.K. (2022) 'Implementasi Sirekap Dalam Pilkada 2020 Kabupaten Semarang', *Reformasi*, 12(2), pp. 224–230. Available at: https://doi.org/10.33366/rfr.v12i2.3874.

Hardiyanti, M. *et al.* (2022) 'Urgensi Sistem E-Voting Dan Sirekap Dalam Penyelenggaraan Pemilu 2024', *Journal Equitable*, 7(2), pp. 249–271. Available at: https://doi.org/10.37859/jeq.v7i2.4257.

Hasdiana, U. (2018) *No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title*, *Analytical Biochemistry*. Available at: http://link.springer.com/10.1007/978-3-319-59379-1%0Ahttp://dx.doi.org/10.1016/B978-0-12-420070-8.00002-7%0Ahttp://dx.doi.org/10.1016/j.ab.2015.03.024%0Ahttps://doi.org/10.1080/07352689.2018.1441103%0Ahttp://www.chile.bmw-motorrad.cl/sync/showroom/lam/es/.

Herjanto, M.F.Y. and Carudin, C. (2024) 'Analisis Sentimen Ulasan Pengguna Aplikasi Sirekap Pada Play Store Menggunakan Algoritma Random Forest Classifer', *Jurnal Informatika dan Teknik Elektro Terapan*, 12(2), pp. 1204–1210. Available at: https://doi.org/10.23960/jitet.v12i2.4192.

Istighfarizky, F. *et al.* (2022) 'Klasifikasi Jurnal menggunakan Metode KNN dengan Mengimplementasikan Perbandingan Seleksi Fitur', *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana)*, 11(1), p. 167. Available at: https://doi.org/10.24843/jlk.2022.v11.i01.p18.

Karo Karo, I.M. *et al.* (2023) 'Analisis Sentimen Ulasan Aplikasi Info BMKG di Google Play Menggunakan TF-IDF dan Support Vector Machine', *Journal of Information System Research (JOSH)*, 4(4), pp. 1423–1430. Available at: https://doi.org/10.47065/josh.v4i4.3943.

Normawati, D. and Prayogi, S.A. (2021) 'Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter', *Jurnal Sains Komputer & Informatika (J-SAKTI*, 5(2), pp. 697–711.

Pradesa, I.A. (2024) 'Analisis Penggunaan Sistem Rekapitulasi Suara (Sirekap) Dalam Menghadapi Problematika Pemilu 2024', *Triwikrama: Jurnal Multidisiplin Ilmu Sosial*, 03(04), pp. 47–57.

Rahayu, S. *et al.* (2022) 'Implementasi Metode K-Nearest Neighbor (K-NN) untuk Analisis Sentimen Kepuasan Pengguna Aplikasi Teknologi Finansial FLIP', *Edumatic: Jurnal Pendidikan Informatika*,

6(1), pp. 98–106. Available at: https://doi.org/10.29408/edumatic.v6i1.5433.

Ridwansyah, T. (2022) 'KLIK: Kajian Ilmiah Informatika dan Komputer Implementasi Text Mining Terhadap Analisis Sentimen Masyarakat Dunia Di Twitter Terhadap Kota Medan Menggunakan K-Fold Cross Validation Dan Naïve Bayes Classifier', *Media Online*, 2(5), pp. 178–185. Available at: https://djournals.com/klik.

Saifurridho, M., Martanto, M. and Hayati, U. (2024) 'Analisis Algoritma K-Nearest Neighbor terhadap Sentimen Pengguna Aplikasi Shopee', *Jurnal Informatika Terpadu*, 10(1), pp. 21–26. Available at: https://doi.org/10.54914/jit.v10i1.1054.

Siahaan, L. *et al.* (2023) 'Keterampilan Membaca Pada Pengajaran Bipa Menggunakan Media Digitalisasi', *Journal of Science and Social Research*, 6(1), p. 160. Available at: https://doi.org/10.54314/jssr.v6i1.1186.

Syahrandi, A., Latipah, A.J. and Verdikha, N.A. (2023) 'Multilayer Perceptron and TF-IDF in the Classification of Hate Speech on Twitter in Indonesian', *JSE Journal of Science and Engineering*, 1(1), pp. 17–22. Available at: https://doi.org/10.30650/jse.v1i1.3773.

Umar, R., Riadi, I. and Purwono, P. (2020) 'Klasifikasi Kinerja Programmer pada Aktivitas Media Sosial dengan Metode Support Vector Machines', *Cybernetics*, 4(01), p. 32. Available at: https://doi.org/10.29406/cbn.v4i01.2042.

LAMPIRAN

Lampiran 1 Pengambilan Data

```
from google_play_scraper import Sort, reviews

result, continuation_token = reviews(
'id.go.kpu.sirekap2024',
lang='id',
country='id',
sort=Sort.MOST_RELEVANT,
count=10000000,
filter_score_with=None
)
```

Scraping Data

```
import pandas as pd
import numpy as np

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

df.head()
```

Dataframe Pandas

```
df = df[['userName', 'score', 'at', 'content',
'thumbsUpCount']]
df.sort_values(by='at', ascending=False)
df.head()
```

Filtering dan *Sorting* Kolom

Lampiran 2 Analisis Data

```
import pandas as pd

file_csv = 'scrapped_data.csv'

df = pd.read_csv(file_csv, sep='|')

print(df)
```

Input Data File CSV

```
deskripsi_statistik = df.describe()

mean_rating = df['Rating'].mean()
median_rating = df['Rating'].median()
mode_rating = df['Rating'].mode()[0]

print("Statistik Deskriptif untuk Dataframe berdasarkan
kolom:")
print(deskripsi_statistik)

print("\nStatistik Deskriptif untuk Kolom 'Rating':")
print(f"Mean Rating: {mean_rating}")
print(f"Median Rating: {median_rating}")
print(f"Mode Rating: {mode_rating}")
```

Statistik Desktiptif

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
ratings, counts =
zip(*sorted(dict(df['Rating'].value_counts()).items()))
plt.bar(ratings, counts, alpha=0.7, edgecolor='black',
align='center')
plt.title('Distribusi Rating')
plt.xlabel('Rating')
plt.ylabel('Frekuensi')
plt.xticks(ratings)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()

rating_counts = df['Rating'].value_counts().sort_index()

for rating, count in rating_counts.items():
print(f"Rating {rating}: {count} kali")
```

Distribusi *Rating*

```
import seaborn as sns

df['Panjang Komentar'] = df['Komentar'].apply(lambda x:
len(x.split()))
```

Relasi *Rating* Dengan Panjang Komentar

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Rating', y='Panjang Komentar', data=df,
color='blue', alpha=0.7)
plt.title('Hubungan antara Rating dan Panjang Komentar')
plt.xlabel('Rating')
plt.ylabel('Panjang Komentar')
plt.show()
```

Scatter Plot

```
plt.figure(figsize=(10, 6))
sns.boxplot(x='Rating', y='Panjang Komentar', data=df,
color='blue')
plt.title('Distribusi Panjang Komentar untuk Setiap Rating')
plt.xlabel('Rating')
plt.ylabel('Panjang Komentar')
plt.show()
```

Box Plot

```
info_shortest_comments =
df.loc[df.groupby('Rating')['Panjang
Komentar'].idxmin()][['Rating', 'Panjang Komentar',
'Komentar']]
info_longest_comments = df.loc[df.groupby('Rating')['Panjang
Komentar'].idxmax()][['Rating', 'Panjang Komentar',
'Komentar']]

for index, row in info_shortest_comments.iterrows():
print(f"Rating {row['Rating']}: Komentar Paling Pendek -
Index: {index}, Panjang: {row['Panjang Komentar']},
Komentar: {row['Komentar']}")

print("\n")

for index, row in info_longest_comments.iterrows():
```

```
10 print(f"Rating {row['Rating']}: Komentar Paling Panjang -
   Index: {index}, Panjang: {row['Panjang Komentar']},
   Komentar: {row['Komentar']}")
```

Kalimat Terpanjang dan Terpendek Setiap *Rating*

```
1  from wordcloud import WordCloud
2
3  def generate_wordcloud(text, title):
4  wordcloud = WordCloud(width=800, height=400,
   background_color='white').generate(text)
5
6  plt.figure(figsize=(15, 10))
7  plt.imshow(wordcloud, interpolation='bilinear')
8  plt.axis('off')
9  plt.title(title)
10 plt.show()
11
12 sorted_ratings = df['Rating'].unique()[::-1]
13
14 for rating in sorted_ratings:
15 comments = ' '.join(df[df['Rating'] == rating]['Komentar'])
16 generate_wordcloud(comments, f'Word Cloud Rating {rating}')
```

Word Cloud Setiap *Rating*

Lampiran 3 Pra Proses

```
1  import pandas as pd
2
3  file_csv = 'scrapped_data.csv'
4
5  df = pd.read_csv(file_csv, sep='|')
6
7  print(df)
```

Input Data CSV

```
1  digit_count = len(str(len(df)))
2
3  df.insert(0, 'ID', df.index.map(lambda x: 'd' +
   str(x+1).zfill(digit_count)))
4
5  print(df)
```

Memberikan *ID* Disetiap Data

```
1  import time
2
3  def lowercase(text):
4  return text.lower()
5
6  start_time = time.time()
7  df['komentar_lowercase'] = df['Komentar'].apply(lowercase)
8  end_time = time.time()
9  time_taken = end_time - start_time
10
11 print(df[['Komentar', 'komentar_lowercase']])
12 print("Waktu proses:", time_taken, "detik")
```

Lower Case

```
1  import re
2
```

```
def remove_unnecessary_char(text):
text =
re.sub('((www\.[^\s]+)|(https?://[^\s]+)|(http?://[^\s]+))',' ',text)
text = re.sub('\n',' ',text)
text = re.sub('\r',' ',text)
text = re.sub(r'\\x..',' ',text)
text = re.sub('  +', ' ', text)
text = re.sub('[^0-9a-zA-Z]+', ' ', text)
return text

start_time = time.time()
df['komentar_remove_char'] =
df['komentar_lowercase'].apply(remove_unnecessary_char)
end_time = time.time()
time_taken = end_time - start_time


print(df[['komentar_lowercase', 'komentar_remove_char']])
print("Waktu proses:", time_taken, "detik")
```

Remove Unecessary Character

```
alay_dict = pd.read_csv('kamus_tidak_baku.csv', sep=";",
encoding='latin-1', header=None)
alay_dict = alay_dict.rename(columns={0: 'original',
                                      1. 1: 'replacement'})

alay_dict_map = dict(zip(alay_dict['original'],
alay_dict['replacement']))
```

Input Kamus Tidak Baku

```
df['komentar_remove_char'] =
df['komentar_remove_char'].astype(str)
def normalize_alay(text):
return ' '.join([str(alay_dict_map.get(word, word)) for word
in text.split (' ')])

start_time = time.time()
df['komentar_spellchecker'] =
df['komentar_remove_char'].apply (normalize_alay)
end_time = time.time()
time_taken = end_time - start_time

print(df[['komentar_remove_char', 'komentar_spellchecker']])
print("Waktu proses:", time_taken, "detik")
```

Proses *Spellchecker*

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stemming(text):
return stemmer.stem(text)

start_time = time.time()

df['komentar_stemming'] =
df['komentar_spellchecker'].apply(stemming)

end_time = time.time()
time_taken = end_time - start_time
```

```
13
14 print(df[['komentar_spellchecker', 'komentar_stemming']])
15 print("Waktu proses:", time_taken, "detik")
```

Stemming Sastrawi

```
1  import numpy as np
2
3  df['komentar_stemming'] = df['komentar_stemming'].replace('',
   np.nan)
4
5  baris_nan = df[df['komentar_stemming'].isnull()]
6  jumlah_nan = df['komentar_stemming'].isnull().sum()
7  print(f"Jumlah NaN pada kolom 'komentar_stemming':
   {jumlah_nan}")
8  print(baris_nan)
9  #print("ID dengan 'komentar_stemming' NaN:",
   baris_nan['ID'].tolist())
```

Cek Data Kosong

```
1  df_cleaned = df.dropna(subset=['komentar_stemming'])
2  df_cleaned
```

Hapus Data Kosong

Lampiran 4 *WordNet*

```
1  import pandas as pd
2
3  file_csv = 'output_praproses.csv'
4  3
5  df = pd.read_csv(file_csv, sep='|')
6
7  print(df)
```

Input Data

```
1  from deep_translator import GoogleTranslator
2  import time
3
4  def translate_komentar(text):
5  text_translated = GoogleTranslator(source='id',
   target='en').translate(text)
6  return text_translated
7
8  from tqdm import tqdm
9
10 tqdm.pandas()
11
12 start_time = time.time()
13 df['komentar_stemming'] = df['komentar_stemming'].fillna('')
14
15 df['Translated'] =
   df['komentar_stemming'].progress_apply(translate_komentar)
16 end_time = time.time()
17 time_taken = end_time - start_time
18
19 print(df[['komentar_stemming', 'Translated']])
20 print("Waktu proses:", time_taken, "detik")
```

Fungsi *Translated*

```
1  from textblob import TextBlob
2
3  def scoring_sentiment(text):
```

```
    analysis = TextBlob(text)
    sentiment_score = analysis.sentiment.polarity
    # print("Sentiment polarity:", sentiment_score)
    score_wordnet = 0

    if sentiment_score > 0.5:
            score_wordnet = 5
    elif 0.2 <= sentiment_score <= 0.5:
            score_wordnet = 4
    elif -0.2 < sentiment_score < 0.2:
            score_wordnet = 3
    elif -0.5 <= sentiment_score <= -0.2:
          score_wordnet = 2
    else:
          score_wordnet = 1

    return score_wordnet, sentiment_score
```

Hitung *Score*

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, f1_score
from matplotlib.ticker import FixedLocator

cm = confusion_matrix(df['Rating'], df['Score_Wordnet'])

macro_f1 = f1_score(df['Rating'], df['Score_Wordnet'],
average='macro')
print(f"Macro F1-Score: {macro_f1}")

fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
cax = ax.matshow(cm, cmap=plt.cm.Blues)
fig.colorbar(cax)

tick_marks = np.arange(len(np.unique(df['Rating'])))
ax.set_xticks(tick_marks)
ax.set_yticks(tick_marks)
ax.set_xticklabels(list(range(1, 6)))
ax.set_yticklabels(list(range(1, 6)))

plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix WordNet')

for i in range(len(cm)):
for j in range(len(cm[i])):
ax.text(j, i, str(cm[i][j]), va='center', ha='center',
color='black')

plt.show()
```

Confusion Matrix F1-Score

Lampiran 5 Klasifikasi

```
import pandas as pd

file_csv = 'output_praproses.csv'

df = pd.read_csv(file_csv, sep='|')
```

```
6
7 print(df)
```

Input Data

```
1 X = df['komentar_stemming']
2 y = df['Rating']
```

Menentukan *Input* dan Kelas

```
1 from sklearn.feature_extraction.text import TfidfVectorizer
2
3 X.fillna('', inplace=True)
4 vectorizer = TfidfVectorizer()
5 features = vectorizer.fit_transform(X)
```

Ekstraksi Fitur TF-IDF

```
1 from sklearn.model_selection import KFold
2 from sklearn.model_selection import cross_val_predict,
  StratifiedKFold
3 from sklearn.metrics import accuracy_score, precision_score,
  recall_score, f1_score, confusion_matrix
4 import numpy as np
5 import matplotlib.pyplot as plt
6
7 def cross_validation(model, _X, _y, _cv):
8 kf = KFold(n_splits=_cv, shuffle=True)
9 fold_count = 1
10 results = {}
11 fold_predictions = []
12 overall_cm = np.zeros((len(np.unique(_y)),
  len(np.unique(_y))), dtype=int)
13
14 for train_index, test_index in kf.split(_X):
15 X_train, X_test = _X[train_index], _X[test_index]
16 y_train, y_test = _y[train_index], _y[test_index]
17
18 model.fit(X_train, y_train)
19 y_pred = model.predict(X_test)
20 fold_predictions.append(y_pred)
21 cm = confusion_matrix(y_test, y_pred)
22
23 results[f"Fold {fold_count}"] = {
24 "Confusion Matrix": cm,
25 "Precision": precision_score(y_test, y_pred,
  average='macro'),
26 "Recall": recall_score(y_test, y_pred, average='macro'),
27 "F1 Score": f1_score(y_test, y_pred, average='macro')
28 }
29 overall_cm += cm
30 fold_count += 1
31
32 for key, value in results.items():
33 print(key + ":")
34 print("Confusion Matrix:")
35 print(value["Confusion Matrix"])
36 print("Precision:", value["Precision"])
37 print("Recall:", value["Recall"])
38 print("F1 Score:", value["F1 Score"])
39 print()
40
41 fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
```

```
    im = ax.imshow(value["Confusion Matrix"],
    interpolation='nearest', cmap=plt.cm.Blues)
    ax.figure.colorbar(im, ax=ax)

    class_names = np.unique(_y)
    ax.set(xticks=np.arange(len(class_names)),
    yticks=np.arange(len(class_names)),
    xticklabels=class_names, yticklabels=class_names,
    title=f'Confusion Matrix - {key}',
    ylabel='True label',
    xlabel='Predicted label')

    thresh = value["Confusion Matrix"].max() / 2.
    for i in range(value["Confusion Matrix"].shape[0]):
    for j in range(value["Confusion Matrix"].shape[1]):
    ax.text(j, i, format(value["Confusion Matrix"][i, j], 'd'),
    ha="center", va="center",
    color="white" if value["Confusion Matrix"][i, j] > thresh
    else "black")

    plt.show()

    print("Overall Confusion Matrix:")
    print(overall_cm)
    print()

    overall_precision = np.diag(overall_cm).sum() /
    overall_cm.sum(axis=0).sum()
    overall_recall = np.diag(overall_cm).sum() /
    overall_cm.sum(axis=1).sum()
    overall_f1_score = 2 * (overall_precision * overall_recall) /
    (overall_precision + overall_recall)

    #print("Overall Evaluation:")
    #print("Precision:", overall_precision)
    #print("Recall:", overall_recall)
    #print("F1 Score:", overall_f1_score)

    y_pred = cross_val_predict(model, _X, _y, cv=_cv)

    cm_overall = confusion_matrix(_y, y_pred)
    #print("Confusion Matrix:")
    #print(cm_overall)

    fig, ax = plt.subplots(figsize=(10, 8), dpi=100)
    cax = ax.matshow(cm_overall, cmap=plt.cm.Blues)
    fig.colorbar(cax)

    tick_marks = np.arange(len(np.unique(_y)))
    ax.set_xticks(tick_marks)
    ax.set_yticks(tick_marks)
    ax.set_xticklabels(list(range(1, 6)))
    ax.set_yticklabels(list(range(1, 6)))

    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.title('Confusion Matrix Overall')

    thresh = overall_cm.max()/2.

    for i in range(cm_overall.shape[0]):
```

```
for j in range(cm_overall.shape[1]):
ax.text(j, i, format(overall_cm[i][j], 'd'), va='center',
   ha='center', color='black' if overall_cm[i,j] > thresh else
   "black")

      plt.show()

      precision_macro = precision_score(y, y_pred,
   average='macro')
      print(f"\nPrecision Macro: {precision_macro}")

      recall_macro = recall_score(y, y_pred, average='macro')
      print(f"Recall Macro: {recall_macro}")

      f1_score_macro = f1_score(y, y_pred, average='macro')
      print(f"F1-Score Macro: {f1_score_macro}")


      return results, fold_predictions
```

Cross Validation 10 Fold

```

kfold = KFold(n_splits=10, shuffle=False)
for fold, (train_index, test_index) in
enumerate(kfold.split(X, y), 1):
if fold == 10:

X_train, X_test = X[train_index], X[test_index]
y_train, y_test = y[train_index], y[test_index]

print(f'Fold {fold}:')
print(f'  - Train data: {len(X_train)} samples')
print(f'  - Test data: {len(X_test)} samples')
print(f'  - Train Index: {train_index}')
print(f'  - Test Index: {test_index}')
```

Menampilkan Isi Data Setiap Pembagian *Fold*

```
from sklearn.neighbors import KNeighborsClassifier
# from sklearn.neighbors import NearestNeighbors
knn_model = KNeighborsClassifier()
```

Klasifikasi K-NN

```
import time

start_time = time.time()
cv_results, fold_predictions = cross_validation(knn_model,
features, y, _cv=10)

end_time = time.time()
time_taken = end_time - start_time
print("Waktu proses:", time_taken, "detik")
```

Menjalankansss *Cross Validation*

Lampiran 6 Kartu Kendali Bimbingan

KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH

Nama	:	Emyzar Haflida Tanjung
NIM	:	2011102441240
Nama Dosen Pembimbing	:	Naufal Azmi Verdikha, S.Kom., M.Eng.
Judul Penelitian	:	Perbandingan Analisis *Wordnet* dan *K-Nearest Neighbor* Pada Ulasan Aplikasi Sirekap 2024.

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	27 Januari 2024	penentuan algoritma yang akan digunakan dan pembuatan judul besar.	
2	29 januari 2024	arahan proses codingan awal dan pembuatan kamus tidak baku	
3	31 Januari 2024	1. koreksi pembuatan kamus tidak baku 2. penjelasan codingan dan mencari alasan mengapa harus melakukan klasifikasi sentimen.	
4	1 Febuari 2024	arahan point penting yang harus ditulis diproposal.	
5	18 Febuari 2024	1. koreksi penulisan latar belakang 2. pembuatan tahap alur penelitian (metodologi)	
6	19 Febuari 2024	koreksi bab 1 dan bab 2 proposal	
7	21 Februari 2024	bimbingan koreksi revisi proposal	
8	11 Maret 2024	arahan membuat kerangka codingan	
9	25 Juni 2024	koreksi revisi proposal	
10	13 Juni 2024	revisi bab 3	
11	14 Juni 2024	1. koreksi penyesuaian penulisan bab 3 2. koreksi revisian bab 3	
12	22 Juni 2024	koreksi revisian bab 3	

Dosen Pembimbing

(.....)
Naufal Azmi Verdikha, S.Kom., M.Eng
NIDN. 1114048801

Mengetahui
Ketua Program Studi

(.....)
Ardiansyah, S.Kom., M.T
NIDN. 1118019203

Lampiran 7 Hasil Uji Turnitin

SKRIPSI EMYZAR HAFILDA TANJUNG

by Teknik Informatika Universitas Muhammadiyah Kalimantan Timur

Submission date: 25-Jul-2024 02:23PM (UTC+0800)
Submission ID: 2422166292
File name: SKRIPSI_EMYZAR_HAFILDA_TANJUNG.docx (1.95M)
Word count: 5788
Character count: 36142

SKRIPSI EMYZAR HAFILDA TANJUNG

ORIGINALITY REPORT

18%	17%	7%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	repository.ub.ac.id Internet Source	2%
2	e-journal.hamzanwadi.ac.id Internet Source	1%
3	media.neliti.com Internet Source	1%
4	core.ac.uk Internet Source	1%
5	docplayer.info Internet Source	1%
6	dspace.umkt.ac.id Internet Source	1%
7	scholar.unand.ac.id Internet Source	1%
8	digilib.polban.ac.id Internet Source	1%
9	repo.iain-tulungagung.ac.id Internet Source	<1%

Lampiran 8 Surat Ijin Penelitian

UNIVERSITAS MUHAMMADIYAH Kalimantan Timur
Berkarakter | Berwawasan | Berkemajuan
UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi
Telp. 0541-748511 Fax. 0541-766832
Website http://informatika.umkt.ac.id
email: informatika@umkt.ac.id

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Nomor : 055-001/KET/FST.1/A/2024
Lampiran : -
Perihal : **Keterangan Pengambilan Data Sekunder**

Assalamu'alaikum Warrahmatullahi Wabarrakatuh

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Amin.

Dengan surat ini, kami menerangkan bahwa mahasiswa berikut:

No	Nama	NIM
1	Emyzar Haflida Tanjung	2011102441240
2	Lilis Sagita	2011102441198
3	Sri Ramadani	2011102441177
4	Muhammad Fariz Ijlal Rafi	2011102441124
5	Nurlita	2011102441070

Melakukan penelitian dengan pengambilan data sekunder di Google Playstore data yang diambil yaitu data ulasan aplikasi "Sirekap" dari rating 1-5.

Demikian hal ini disampaikan, atas kerjasamanya kami ucapkan terima kasih.

Wassalamu'alaikum Warrahmatullahi Wabarrakatuh

Samarinda, 18 Dzulhijjah 1445 H
25 Juni 2024 M

Ketua Program Studi S1 Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019203

RIWAYAT HIDUP

Penulis bernama lengkap Emyzar Haflida Tanjung dilahirkan di Kisaran 23 Agustus 2001, dan merupakan anak ketiga dari 4 bersaudara dari pasangan Ghazali Tanjung,.SE dan Paridah. Mengawali pendidikan formal di Pendidikan Sekolah Dasar di MIN Tanah Paser (2007-2013) dan melanjutkan Pendidikan Sekolah Menengah Pertama di Mts Bina Islam Tanah Paser (2013-2016). Penulis melanjutkan jenjang pendidikan formal Sekolah Menengah Atas di SMKN 1 Tanah Paser (2016-2019). Penulis masuk di Fakultas Sains dan Teknologi Universitas Muhammadiyah Kalimantan Timur pada tahun 2020.

Untuk menyelesaikan studi di Fakultas Sains dan Teknologi Jurusan Teknik Informatika UMKT, penulis melakukan penelitian dengan judul "**Perbandingan Analisis *Wordnet* Dan *K-Nearest Neighbor* Pada Ulasan Aplikasi Sirekap 2024"** sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer.