DAFTAR ISI

Daftar Tabel

Daftar Gambar

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Salah satu perubahan gaya hidup masyarakat akibat pesatnya perkembangan internet adalah semakin cepatnya penyebaran informasi melalui media sosial. Aplikasi Tiktok adalah salah satu aplikasi yang memiliki banyak pengguna, juga sangat disukai, dan aplikasi ini berbagi informasi dengan cepat, karena popularitas aplikasi Tiktok. Aplikasi TikTok sendiri merupakan alat untuk membuat dan membagikan berbagai jenis video pendek dengan gaya vertikal yang hanya bisa dilihat dengan menggeser layar ke atas atau ke bawah (Sola et al., 2021).

Perbandingan metode *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentimen komentar pada aplikasi tiktok di google play store, kategori Ulasan Baik, Negatif, dan Netral digunakan untuk memfilter komentar di aplikasi Tiktok. Analisis sentimen yang akan dirinci dalam analisis teks adalah proses pengkategorian teks atau emosi (positif, negatif, dan netral).Memahami, mengekstraksi, dan mengolah data tekstual, serta secara otomatis mengevaluasi informasi sentimen yang terdapat dalam kata-kata atau pendapat, perilaku, dan perasaan seseorang terhadap suatu entitas, adalah definisi lain dari analisis sentimen (menggambarkan individu, peristiwa atau topik). Tujuannya adalah untuk menawarkan data yang berguna kepada seseorang dari kumpulan data yang tidak terstruktur (Herlinawati et al., 2020).

Penelitian yang dilakukan oleh (Zulqornain & Adikara, 2021) Analisis sentimen aplikasi Tiktok menggunakan metode *Naive Bayes* dan *Categorial Propotional Difference* (CPD). Aplikasi Tiktok ini memungkinkan pengguna untuk memposting video, tetapi biasanya film dengan konten pornografi tersedia untuk dilihat oleh anak di bawah umur karena tidak ada batasan usia pada aplikasi Tiktok ini. Untuk membantu orang tua memilih aplikasi untuk anak-anak mereka, penulis melakukan analisis sentimen pada ulasan aplikasi TikTok. Berdasarkan permasalahan tersebut, penulis menggunakan metode Naive Bayes dan Categorical Proportional Difference untuk melakukan analisis sentimen. Menggunakan 5-Cross Validation dan kata yang berbeda, penelitian ini diuji. Hasil yang diperoleh dengan menggunakan 100% yang digunakan untuk pengujian memiliki nilai f-measure 82% dan akurasi, presisi, recall, dan f1 score 74% dari

72%. Untuk mendapatkan akurasi yang tinggi, membutuhkan 100% data karena dalam penelitian ini, penggunaan istilah besar dan 5 k-fold merusak akurasi.

Hasil.Pada penelitian yang dilakukan oleh (Al-shufi & Erfina, 2021) Sentimen analisis mengenai aplikasi streaming film menggunakan algoritma *Support Vector Machine* di play store. Selama pandemi ini, streaming film menjadi hobi yang populer. Sangat mudah untuk menonton film online kapan pun dan di mana pun, berkat berbagai perangkat lunak streaming film dan kemudahan akses internet. Bergantung pada layanan yang ditawarkan oleh pengembang aplikasi, ulasan pengguna mungkin berdampak pada reputasi aplikasi. Hingga saat ini, hanya peringkat dan jumlah unduhan yang digunakan untuk menentukan aplikasi mana yang terbaik. namun, ulasan pengguna juga harus dipertimbangkan, karena ada begitu banyak data yang harus dipilah, tidak mungkin mengklasifikasikan ulasan pengguna secara manual. Oleh karena itu pada penelitian ini bertujuan untuk menganalisis ulasan dari para pengguna aplikasi *streaming* film di Playstore, dengan menggunakan algoritma *Support Vector Machine*. hasil dari proses analisis data menunjukan bahwa tingkat keakurasian paling tinggi adalah Disney hotstar 69.33%, Wetv 64.67%, dan vidio 62.00%.

Penelitian yang dilakukan oleh (Sahara & Permana, 2019) tentang metode K-NN untuk menganalisis tinjauan sentimen aplikasi anak. Aplikasi game merupakan aplikasi yang sering digunakan oleh anak-anak, remaja maupun dewasa. Oleh karena itu, pengembang aplikasi meningkatkan kualitas performa game untuk memaksimalkan keuntungan, terutama bagi pengguna game di pojok kanan bawah layar, yang menyebabkan orang dewasa menjadi bingung tentang apa yang dikonsumsi oleh anak-anak dan meminta pemilihan melalui komentar pada kategori game aplikasi untuk anak-anak. Berdasarkan umpan balik atau komentar pengguna yang diposting di situs web aplikasi oleh pengguna yang telah menggunakannya, aplikasi tersebut. Dari sampel komentar yang disampaikan pembaca dan dievaluasi menggunakan berbagai variabel, antara lain teks positif dan negatif, dengan menggunakan metode k-Nearest Neighbors (k-NN). Nilai akurasi sebesar 78,50% diperoleh pada kumpulan data yang diuji, yang menunjukkan bahwa kumpulan data tersebut masih dalam batas yang dapat diterima ketika menggunakan pendekatan k-NN. Namun, agar temuan klasifikasi sentimen dalam penelitian ini akurat, penting untuk memasukkan fase case folding, Remove Punctuation, dan Spelling Correction dalam preprocessing.

Dalam Penelitian (Darmawan & Amini, 2022) Yaitu perbandingan hasil analisis

sentimen menggunakan algoritma naïve bayes dan *K-Nearest Neighbor* di Twitter. Maraknya pengguna media sosial, khususnya Twitter, mengakibatkan terjadinya persebarluasan data, dan data yang disebarluaskan juga sangat mudah diperoleh. Akibatnya, diperlukan pengolahan data untuk mengubah data menjadi sumber informasi yang bermanfaat, seperti mengklasifikasikan opini publik di Twitter ke dalam kategori emosional. Pertanyaan metode mana yang terbaik untuk melakukan analisis sentimen muncul karena ada banyak pendekatan yang dapat digunakan untuk melakukannya. Peneliti melakukan tiga pengujian dengan memanfaatkan data dari Twitter untuk menilai kinerja Naive Bayes dan K-Nearest Neighbor, dua algoritma yang digunakan untuk melakukan analisis sentimen. Tingkat akurasi akhir untuk algoritma Naive Bayes adalah 65%, 40%, 80%, sedangkan untuk pendekatan K-Nearest Neighbor adalah 55%, 45%, 75%. Namun akurasinya masih rendah sehingga diperlukan perbandingan metode *machine learning* lainya dan menambahkan pembobotan lain seperti TF-IDF.

Penelitian yang dilakukan oleh (Saputra, 2022) Analisis sentiment pada media sosial Twitter menggunakan metode *K-Nearest Neighbors.*Twitter. Pengguna satu situs microblogging dapat menulis tentang berbagai topik dan mendiskusikan peristiwa terkini. Metode klasifikasi K-Nearest Neighbor digunakan dalam penelitian ini. K-Nearest Neighbor akan langsung mengklasifikasikan data pembelajaran yang akan dibentuk sebagai model untuk menentukan kategori data baru yang ingin ditentukan yaitu kelas sentimen positif, negatif, dan netral. Temuan penelitian ini menghasilkan sebuah sistem yang secara otomatis dapat mengklasifikasikan sentimen berdasarkan hasil pengujian nilai k. Nilai optimal yang diperoleh dari hasil penelitian adalah k = 1 dari total dataset sebanyak 450 tweet, dengan 315 data latih dan 135 data uji, mencapai 51,11%. Namun pada penelitian ini nilai k kecil maka dapat mempengaruhi nilai akurasi yang dihasilkan. Hal ini dikarenakan, data yang masuk pada k tetangga terdekat terlalu sedikit dan belum banyak tetangga yang digunakan untuk proses klasifikasi.

Berdasarkan penelitian sebelumnya, penelitian ini mengusulkan penggunaan metode perbandingan *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentimen review aplikasi TikTok di Google Play Store dengan klasifikasi positif, negatif, dan netral.

1.2. Rumusan Masalah

Berdasarkan latar belakang permasalahan yang telah dikemukakan, maka dapat dirumuskan masalahnya adalah ;

1. Bagaimana perbandingan metode untuk analisis sentimen ulasan pengguna aplikasi Tiktok pada google play store menggunakan metode *Support Vector Machine* dan *K-Nearest Neighbor* berdasarkan tingkat akurasi, presisi, recall dan f1 score.

1.3. Batasan Masalah

Agar masalah yang dibahas menjadi fokus dan lebih jelas dalam mencapai sasaran, maka dibuat batasan dari perumusan masalah di atas, diantaranya adalah ;

1. Penelitian ini hanya menggunakan perbandingan antara dua metode yang telah ditentukan untuk mendapatkan analisis sentimen ulasan pengguna aplikasi pada google play store
2. Kriteria yang digunakan hanya tiga yaitu positif, negatif dan netral
3. Data yang diambil dari google play store yaitu 3000 data berdasarkan data terbaru pada bulan Juli tahun 2022. Data dibagi menjadi data training sebesar 2.400 dan data testing sebesar 600.

1.4. Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diuraikan di atas, maka tujuan dari penelitian ini adalah perbandingan metode untuk analisis sentiment ulasan pengguna aplikasi Tiktok pada google play store menggunakan metode *Support Vector Machine* dan *K-Nearest Neighbor*. sehingga dapat diketahui metode mana yang lebih baik dan tepat berdasarkan tingkat *accuracy*, *precision* dan *recall* dan *F1 Score.*

1.5. Manfaat Penelitian

Manfaat yang diberikan dari penelitian ini yaitu ;

1. Membandingkan metode *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentiment ulasan pengguna aplikasi tiktok pada google play store, sehingga dapat diketahui metode mana yang lebih baik dan tepat berdasarkan tingkat akurasi, precision dan recall.
2. Dapat membantu pengklasifikasian ulasan pada google play store positif, negatif dan netral.
3. Meningkatkan kegunaan dan manfaat dari aplikasi tiktok.

1.6. Metodologi Penelitian dan Metode Pengembangan Sistem

Metodologi penelitian berisi langkah - langkah yang akan dilakukan pada penelitian ini agar penelitian berjalan dengan runtut dan terarah. Metode pengembangan sistem merupakan alur dikembangkannya sistem yang akan di buat. Metode penelitian dan metode pengembangan sistem pada penelitian ini adalah sebagai berikut,

1.6.1 Metodologi Penelitian

Langkah yang dilakukan dalam penelitian ini sebagai berikut;

1. Studi Literatur

   Penelitian ini diawali dengan pengumpulan referensi untuk memecahkan permasalahan yang ada dan memperkuat teori-teori yang akan digunakan dalam penelitian.

2. *Scraping* Data

   Dataset dikumpulkan setelah tahap studi literatur selesai. Aplikasi Tiktok di Google Play Store menggunakan teknik web scraping (penambang data), yaitu teknik yang digunakan untuk mengekstrak data dalam jumlah besar dari sebuah situs web, dengan data yang diekstraksi disimpan dalam file lokal di komputer atau database dalam tabel (spreadsheet).

3. *Labelling* Data

   Pelabelan data dilakukan dengan mengelola data mentah yang diperoleh melalui pengikisan data. Pelabelan data akan diklasifikasikan sebagai positif, negatif, atau netral.

4. *Text Preprocessing*

   Text preprocessing adalah proses pemilihan data teks dengan mengubah data mentah menjadi data terstruktur.

5. TF-IDF

   Term Frequency Inverse Document Frequency (TF-IDF) adalah metode pembobotan kata yang mengintegrasikan Term Frequency dan Inverse Document Frequency. Pada penelitian ini, metode TF-IDF digunakan untuk menyeleksi fitur sebagai hasil rangkuman, dengan penerapannya pada pemilihan fitur bobot kata.

6. Pembuatan Model

Pemodelan dilakukan pada proses ini menggunakan dua model yaitu Support Vector Machine dan K-Nearest Neighbor. Dengan membandingkan kedua pendekatan tersebut.

7. Pengujian

   Pada proses pengujian ini dilakukan dengan menggunakan *Confusion Matrix*.

8. Analisis dan Kesimpulan

   Kesimpulam diambil dari hasil pengujiam yang sudah dilakukan setelah analisis.

1.6.2 Metodologi Pengembangan

Untuk pengembangan sistem, akan digunakan metode pengembangan sistem prototype. Metode prototyping digunakan karena meningkatkan kemungkinan bahwa sistem yang dibuat akan memenuhi harapan pengembang dan pengguna, dengan penekanan pada komunikasi antara keduanya. Adapun tahapan pengembangan sistem menggunakan **metode *prototype* (Pressman, 2014) yaitu sebagai berikut ;**

Adapun penjelasan pada diagram diatas :

a. *Communication*
b. *Quick Plan dan Modeling Quick Design*
c. *Construction of Prototype*
d. *Deployment Delivery & Feedback*

1.7. Sistematika Penulisan

Sistematika penulisan dari penelitian ini disusun kedalam lima bab yang terdiri dari beberapa bagian utama sebagai berikut.

BAB I Pendahuluan

Bab ini menjelaskan konteks permasalahan yang dibahas, khususnya Perbandingan *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentimen ulasan pada aplikasi Tiktok di Google Play Store, rumusan masalah, definisi masalah, tujuan masalah, manfaat penelitian, dan metodologi penelitian serta sistematika penulisan.

BAB II Tinjuan Pustaka

Bab ini menjelaskan teori-teori yang mendasari metode perbandingan Support Vector Machine dan K-Nearest Neighbor untuk analisis sentimen ulasan pengguna terhadap aplikasi pemecahan masalah Tiktok di Google Play Store.

Bab III Analisis Dan Perancangan Sistem

Pembahasan pada bab ini adalah metodologi penelitian, analisis, perancangan dan perancangan metode perbandingan *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentimen review pengguna aplikasi Tiktok di Google Play Store..

Bab IV Hasil Dan Pembahasan

Bab ini menyajikan temuan penelitian berupa perbandingan *Support Vector Machine* dan *K-Nearest Neighbor* untuk analisis sentimen review aplikasi Tiktok di Google Play Store. Selanjutnya menentukan signifikansi dari hasil yang diperoleh.

Bab V Kesimpulan Dan Saran

Bagian ini membahas kesimpulan dan rekomendasi yang diperoleh dari hasil perbandingan Metode *Support Vector Machine* dan *K-Nearest Neighbor* untuk Analisis Sentimen Review Aplikasi Tiktok di Google Play Store. Selain itu untuk menentukan rangking dari hasil tangkapan yaitu pengklasifikasian kata atau kategori sebagai positif, negatif, atau netral.

BAB II
TINJAUAN PUSTAKA

2.1. Tiktok

Aplikasi Tiktok digunakan untuk membuat dan membagikan berbagai video pendek dalam format vertikal, yang dapat dilihat hanya dengan menggulir layar ke atas atau ke bawah. Aplikasi Tiktok juga sudah banyak mendapatkan update, dan masih banyak fitur tambahan yang tersedia seperti toko tiktok, live streaming dan filter. Tiktok melayani banyak konten, antara lain olahraga, menampilkan kreativitas, ekspresi diri, meningkatkan mood dengan menonton video lucu di TikTok, kampanye, hingga mempromosikan barang-barang yang dijual (Sola et al., 2021). Aplikasi ini dapat diunduh dari Google Play Store untuk pengguna Android dan App Store untuk pengguna iOS. Anda juga dapat mengaksesnya melalui PC. Banyak pengguna akan memberikan ulasan atau komentar tentang kepuasan mereka terhadap aplikasi Tik Tok jika sudah diunduh dari Google Playstore. Selain ulasan di Google Play Store di mana pengguna memberikan peringkat rendah (satu atau dua bintang) tetapi teks ulasannya positif, atau di mana pengguna memberikan peringkat tinggi tetapi teks ulasannya negatif (Daryfayi Edyt & Asror, 2020).

2.2. Perbandingan Metode

Perbandingan metode adalah metode yang membandingkan kinerja dua atau lebih metode dengan kinerja klasifikasi metode lain yang diketahui sehingga menghasilkan perbandingan kinerja metode tersebut (Riyanto, 2018). Metode ini akan membandingkan dua metode yang dipilih, dan akan menghasilkan nilai akurasi,presisi,recall dan f1 score. Selain membandingkan, metode perbandingan juga mengkategorikan teks ulasan kepuasan pengguna sebagai positif, negatif, atau netral (Ilmawan & Mude, 2020). Support vector Machine dan K nearest Neighbors sama merupakan model Supervised Learning, yang mana sama sama membutuhkan dataset berlabel.

2.3. Analisis Sentimen

Analisis sentimen adalah proses yang membuat kumpulan data berbasis teks yang terdiri dari kelas positif, negatif, dan netral yang ditentukan. Dokumen teks adalah fokus analisis sentimen. Dokumen teks dapat berisi emoji nonteks yang dapat digunakan untuk mengungkapkan emosi yang muncul saat menulis opini (Amalia et al., 2021). Analisis

sentimen, juga dikenal sebagai opini manning, yaitu proses menganalisis emosi di balik kata-kata pelanggan. Dengan berkomunikasi melalui platform media online seperti media sosial, e-commerce, dan website (Rahman et al., 2021). Tujuan analisis sentimen adalah untuk mempelajari tentang suka dan tidak suka pendapat yang ada, yang biasanya positif, negatif, dan netral sebagai parameter pengambilan keputusan (Kurniawan et al., 2019). Selain itu, tujuan lain dari analisis sentimen adalah untuk mengidentifikasi opini yang kemudian diklasifikasikan menjadi parameter positif, negatif, dan netral.

2.4. Text Preprocessing

Text preprocessing adalah proses memilih data teks dan menyusunnya sebelum memodelkannya. Dimana data biasanya diambil dalam bentuk teks yang bersumber dari dokumen dengan tujuan menemukan kata kunci yang mewakili sekelompok dokumen sehingga nantinya dapat dilakukan analisis hubungan antar dokumen tersebut (Maarif, 2016). *Text preprocessing* mengubah teks yang tidak terstruktur menjadi data yang baik yang dapat diproses dalam berbagai tahap seperti, *case folding, tokenizing, filtering, dan stemming.*

2.4.1 *Case Folding*

Case folding adalah proses mengubah teks dari huruf besar menjadi huruf kecil. Tujuan dari *case folding* kasus ini adalah untuk menyediakan bentuk teks standar (Rohanah et al., 2021). Data yang ada biasanya tidak selalu terstruktur dan konsisten dalam penggunaan huruf kapital. Alhasil, fungsi *case folding* adalah untuk mendeskripsikan penggunaan huruf kapital (Rokhman et al., 2021). Misalnya setelah proses case folding, data teks berupa tulisan "Saya SUKA sEkali Aplikasi INI" menjadi “saya suka sekali aplikasi ini” artinya semua huruf kita ubah menjadi huruf kecil. Contoh lain dari *case folding* dapat dilihat pada **Tabel 2.1**,

Tabel 2.1 Contoh *Case Folding*

Sebelum *Case Folding*	Sesudah *Case Folding*
“Aplikasi Ini Sangat Bagus.”	“aplikasi ini sangat bagus.”

2.4.2 *Removal Punctuation*

Remove Punctuation adalah fungsi pemrosesan awal teks yang menghapus tanda baca, karakter, dan simbol dari dokumen atau teks (Sola et al., 2021). Ini karena tanda baca dan simbol tidak penting dan dapat memengaruhi klasifikasi teks. Misalnya pada sebuah dokumen, setelah menghilangkan tanda baca “Aplikasi ini sangat bagus!!!!!!(:” setelah di

remove punctuation “ menjadi Aplikasi ini bagus sekali”. Contoh lain dapat dilihat pada **Tabel 2.2**,

Tabel 2.2 Contoh *Removal Punctuation*

Sebelum *Removal Punctuation*	Sesudah *Removal Punctuation*
“Saya sangat suka dengan aplikasi ini sangat membantu??!!”	“Saya sangat suka dengan aplikasi ini sangat membantu”

2.4.3 *Tokenizing*

Tokenizing adalah proses membagi teks menjadi token/bagian, yang dapat berupa kalimat, paragraf, atau dokumen. Tokenizing adalah proses pemisahan kata menjadi token. Tokenizing kalimat "saya akan pergi", misalnya, menghasilkan enam token., yaitu: "Saya", "akan", "pergi" (Rahman et al., 2021). Contoh lain dari *Tokenizing* dapat dilihat pada **Tabel 2.3**,

Tabel 2. 3 Contoh *Tokenizing*

Sebelum *Tokenizing*	Sesudah *Tokenizing*
“jalan di pagi hari sejuk”	[‘jalan’], [‘di’], [‘pagi’,[‘hari’],[‘sejuk’]

2.4.4 *Spelling Correction*

Spelling Correction Pada tahap ini, kata-kata yang tidak baku dan kata-kata yang disingkat dengan kata-kata dari kamus buatan dinormalisasi. Tujuan dari proses ini adalah mengembalikan bentuk tulisan asli kata tersebut agar sesuai dengan kata yang terdaftar dalam Kamus Besar Bahasa Indonesia (KBBI). Contoh lain dapat dilihat pada **Tabel 2.4**,

Tabel 2.4 Contoh *Stemming*

Sebelum *Spelling Correction*	Sesudah *Spelling Correction*
“gw suka sih bagus ”	“gue suka sih bagus”

2.4.5 *Stopword Removal*

stopword adalah proses pemfilteran pemrosesan teks di mana kata-kata penting dipilih dari hasil token dan digunakan untuk mewakili dokumen (Sola et al., 2021). Stopword menghapus kata-kata yang tidak berarti atau mengubah arti asli dokumen. Untuk lebih fokus pada klasifikasi teks, perlu untuk mengucapkan kata-kata yang tidak berarti. (Fanny & Suroyo, 2022). Misalnya “Apakah Hari Ini Sangat Dingin” setelah di *Stopword Removal* menjadi “ Apa Hari Ini Dingin”. Contoh lain dapat dilihat pada **Tabel 2.5**,

Tabel 2.5 Contoh *Stopword Removal*

Sebelum *Stopword Removal*	Sesudah *Stopword Removal*
“Pergi bekerja dengan menggunakan motor”	“Pergi kerja dengan motor”

2.4.6 *Stemming*

Stemming adalah proses mereduksi infleksi suatu kata menjadi bentuk dasarnya, tetapi bentuk dasarnya tidak berarti sama dengan kata dasarnya. Stemming juga bertujuan untuk mempermudah klasifikasi kata dan pengelompokan kata (Amalia et al., 2021). Misalnya, kata "mendengarkan", "dengarkan", dan "didengarkan" akan diubah menjadi "dengarkan". Proses stemming dalam penelitian ini memanfaatkan pustaka sastra yang berisi kamus kata dasar bahasa Indonesia.Penggunaan perpustakaan sastra berlaku untuk algoritma nazief dan Adriana. (Indriani, 2020). Berikut langkah langkah proses algoritma nazief dan Adriana.

9. Langkah pertama yang digunakan adalah mencari kata yang akan disistematisasikan oleh kamus kata dasar. Jika pencarian kata adalah kata dasar, pencarian akan berhenti.
10. Pada langkah kedua, Sufiks Infleksi (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dihilangkan. Jika berupa partikel (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghilangkan *Possesive Pronouns* (“-ku”, “-mu”, atau “- nya") Jika ada.
11. Langkah selanjutnya Hapus Akhiran Derivasi (“-i”, “-an” atau “-kan”). Jika kata tersebut ada dalam kamus, maka algoritma berhenti. Jika tidak maka lanjutkan ke langkah 3a.
    a. sebuah. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga dihapus. Jika kata tersebut ditemukan dalam kamus maka berhentilah. Jika tidak ditemukan maka lakukan langkah-langkahnya
    b. Sufiks yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lalu lanjutkan ke langkah 4.
12. Hapus awalan derivasi DP {“di-”,“ke-”,“se-”,“me-”,“be-”,“pe”, “te-”} dengan maksimal 3 iterasi:Pada langkah ke 4 berhenti jika : terjadi sebuah kombinasi awalan dan akhiran yang terlarang atau tidak diperbolehkan.
    a. Awalan yang terdeteksi saat itu sama dengan awalan yang dihapus sebelumnya.
13. Pada langkah 5 jika pada langkah 4 kata dasar masih belum ditemukan, maka dilakukan proses perekaman dengan mengacu pada tabel 2.5. Perekaman akan dilakukan dengan menambahkan karakter perekaman yang telah dipenggal atau dihilangkan sebelumnya.

14. Pada langkah ini semuanya selesai jika tidak berhasil maka kata awal tetapi akan berakhir sebagai kata dasar.

    Contoh lain dapat dilihat pada **Tabel 2.6**,

Tabel 2.6 Contoh *Stemming*

Dokumen Sebelum Dilakukan Stemming	Dokumen Sebelum Dilakukan Stemming
"Jalananya sangatlah jauh"	" Jalan sangat jauh"

2.5. *Term Frequency Inverse Document Frequency* (TF-IDF)

Algoritma *Term Frequency Inverse Document Frequency*(TF-IDF) adalah algoritma pembobotan yang mengintegrasikan *term frequency dan inverted document frequency*. Pada penelitian ini, metode TF-IDF digunakan untuk menyeleksi fitur sebagai hasil rangkuman, dengan penerapannya pada pemilihan fitur bobot kata (Maarif, 2016). *Inverse Document Frequency* adalah pengurangan term yang sering muncul pada berbagai dokumen, dengan menghitung frekuensi kebalikan dari dokumen yang mengandung kata TF yang dibutuhkan dalam text preprocessing untuk mereduksi nilai term. *Term Frequency* adalah fungsi yang menunjukkan kemunculan suatu term dalam dokumen invers. Inverse *Document Frequency* (IDF) adalah metode penyaringan term yang sering muncul pada dokumen berbeda dengan menghitung frekuensi kebalikan dari dokumen yang mengandung kata TF yang diperlukan dalam preprocessing teks untuk mengurangi nilai *term* (Nurrun Muchammad Shiddieqy et al., 2016). Perhitungan *Term Frequency* (TF-IDF) dapat dilihat pada persamaan 2.1,

$$\boldsymbol{TF} = \{ \mathbf{1} + \boldsymbol{log}\mathbf{10}\ \boldsymbol{tft,d}, \textbf{ IF } \boldsymbol{tft,d} > \mathbf{0}$$
$$\mathbf{0}, \quad \textbf{IF } \boldsymbol{tft,d} = \mathbf{0} \qquad (2.1)$$

Keterangan :

TF : Frekuensi kemunculan *term* (t)

$log10$: Logaritma dengan basis 10

tft,d : Frekuensi kemunculan *term*

(t) : dalam dokumen (d)

Menghitung *Invers Document Frequency* (IDF) Perhitungan IDF dapat dilambangkan dalam persamaan 2.2,

$$\boldsymbol{idft} = \boldsymbol{log}\left(\frac{N}{dft}\right) + 1 \qquad (2.2)$$

Keterangan :

$idft$: Jumlah dokumen yang mengandung *term* (t)

log : Logaritma

N : Jumlah dokumen

dft : Frekuensi dokumen yang mengandung *term* (t)

Langkah terakhir adalah menskor setiap *term* dengan melakukan pembobotan menggunakan rumus TF-IDF yang menggabungkan rumus perhitungan frekuensi suku mentah (tf) dengan rumus frekuensi dokumen terbalik (IDF), sehingga akan ditemukan bobot (W) masing-masing *term* memiliki persamaan yang dilambangkan pada 2.3,

$$\boldsymbol{Wt,d} = \boldsymbol{tft,d} \times \boldsymbol{idf} \quad \text{(2.3)}$$

Keterangan :

Wt,d : Bobot term (t) pada dokumen (d)

tft,d : Frekuensi kemunculan term (t) dalam dokumen (d)

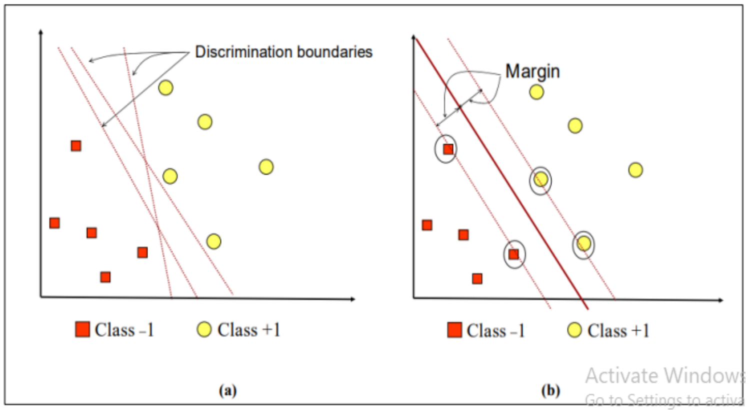
$idft$: Jumlah dokumen yang mengandung *term* (t)

log : Logaritma N = Jumlah dokumen

dft : Frekuensi dokumen yang mengandung *term* (t)

2.6. Support Vector Machine

Metode *Support Vector Machine* dan *K-Nearest Neighbor* adalah metode klasifikasi dan regresi untuk masalah linier dan nonlinier. Memiliki keuntungan dapat menerapkan pemodelan linier pada data input non-linier berdimensi tinggi, yang diperoleh dengan menggunakan fungsi kernel yang diperlukan(Wahyudi & Kusumawardana, 2021). Jenis fungsi kernel yang dipilih dan diimplementasikan berdasarkan karakteristik data berdampak signifikan terhadap efektivitas *Support Vector Machine*. Menurut banyak penelitian, *Support Vector Machine* adalah metode yang paling akurat untuk klasifikasi teks (Herlinawati et al., 2020). Metode *Support Vector Machine* didukung. Algoritma SVM mencari fungsi pemisah pengklasifikasi terbaik *(hyperplane)* dalam jumlah ruang fitur yang tak terbatas. Yang bertindak sebagai garis pemisah antara dua objek atau titik. Berikut contoh *support vector machine* untuk mencari hyperplane terbaik dalam memisahkan kelas -1 dan +1 dapat dilihat pada **Gambar 2.1** di bawah ini :



Gambar 2.1 *Hyperplane* SVM

Ada objek data eksternal terdekat, yang disebut hyperplane, dalam Algoritma SVM yang akan mendukung vektor. Dengan bantuan vektor ini, SVM akan digunakan untuk mencari hyperplane yang paling optimal, sedangkan objek data lainnya akan diabaikan seluruhnya. Pada **Gambar 2.1** menggambarkan pola yang termasuk dalam dua kelas, yaitu -1 dan +1. Pola dengan bentuk persegi dan warna merah termasuk kelas - 1, sedangkan pola bulat kuning termasuk kelas + 1. Menemukan hyperplane (garis pemisah) antara dua kelas memungkinkan untuk klasifikasi. Gambar 1-a menggambarkan berbagai batas diferensiasi. Karena menghubungkan dua kelas, garis pada Gambar 1-b adalah hyperplane terbaik. Vektor pendukung adalah lingkaran hitam dengan titik merah dan kuning di dalamnya.

Untuk *hyperplane* klasifikasi linier dapat dilihat pada persamaan yang dilambangkan 2.4,

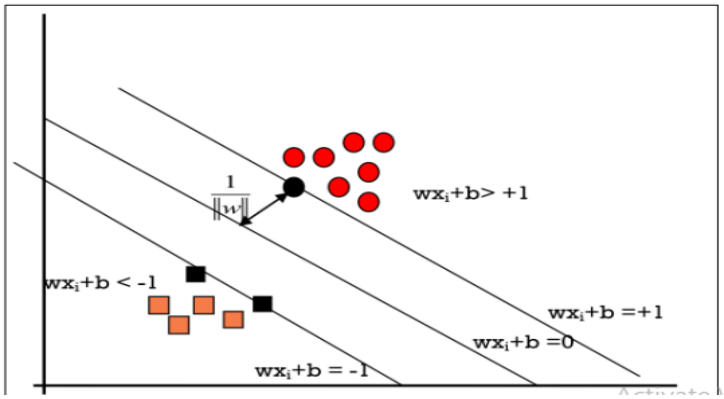
$f(x) = w, x + b = 0$……………………………………………..…...(2.4)

Dari persamaan diatas dapat pertidaksamaan kelas +1 (negatif), dapat dilihat pada pertidaksamaan dilambangkan 2.5,

$w. x + b \leq +1$…………………………………………………………(2.5**)**

Kemudian, pertidaksamaan untuk kelas -1. Persamaan dilambkan 2.6,

$w. x + b \geq -1$………………………………………………………...(2.6)



Gambar 2.2 Fungsi Pemisah Optimal Untuk Objek yang Dapat dipisah Secara Linear

Bidang normal dinyatakan dengan w, dan posisi bidang relatif terhadap koordinat pusat dinyatakan dengan b. Nilai margin terbesar diperoleh dengan mengoptimalkan jarak antara hyperplane dengan titik berikutnya yaitu 1/ ||w||. Hal tersebut dapat dirumuskan sebagai masalah QP(pemrograman kuadratik) yang dimana titik minimum persamaan , dan mengingat kendala dari persamaan dilambangkan 2.7,

$\min\frac{1}{2}\,||\,w||2\ \min\frac{1}{2}\,(\,w_1^2\ +\ w_2^2\,)$ ……..…….……(2.7)

Persamaaan selanjutnya akan dilambangkan pada 2.8,

$\boldsymbol{yi}\ (\boldsymbol{wxi}\ +\ b) \geq \mathbf{1},\ \mathbf{i} = \mathbf{1},\ \mathbf{2},\ \mathbf{3}\ \ldots\ ,\mathbf{N}$……...……..….(2.8)

Dimana Dalam mengambil keputusan dengan klasifikasi *support verctor machine*, pada penelitian ini akan menggunakan fungsi kernel K(xi,xd), dengan persamaan polinomial dilambangkan pada 2.9,

$K(xi,\ xd) = (x_j^T\ \ XJ{+}1\ )\ d\ ,\ Y > 0$……..….(2.9)

langkah pelatihan data *support vector machine* menggunakan algoritma *sequential learning*;

1. Melakukan inisialisasi pada paramter ai , γ, C, dan ε.

   Keterangan :

   ai : alfa, digunakan untuk mencari *support vector*

   γ : gamma / learning rate, digunakan untuk mengontrol kecepatan

   C : merupakan variabel slack

   ε : epsilon, digunakan untuk menemukan nilai error

2. Menghitung matriks Hessian, dengan persamaan dilambangkan pada 2.10,

   $Dij = yiyj(K(xixj)\ \lambda\ 2)$…… (2.10)

Dengan memiliki nilai i dan j = 1,2,3,4...n.

Keterangan :

xi : data ke-i

xj : data ke-j

yi : kelas data ke-i

yj : kelas data ke-j

λ : Lamda

d : degree atau derajat kernel polinomial

$K(xixj)$: fungsi kernel

3. Melakukan perhitungan hingga iterasi data i hingga j dengan persamaan 2.11 dilanjutkan dengan persamaan 2.12 dan persamaan 2.13,

   a. $Ei = \Sigma j\ iajDij$..... (2.11)

   b. $\delta\alpha i$ = min (max[γ(1 − Ei), αi], $C - \alpha i$.....(2.12)

   c. $\alpha i = \alpha i + \delta\ \alpha i$.....(2.13)

Keterangan :

αi : alfa ke-i

γ : konstanta gamma

Ei : error rate

C : variabel slack

$\delta\alpha i$: delta alfa ke-

4. Melakukan langkah 1 sampai 3 hingga memenuhi batas maksimum iterasi Pada proses sequential learning 1 sampai 4 akan menghasilkan nilai dari *support vector*, dengan nilai SV = ($\alpha i > thresholdSV$ yang juga digunakan untuk melakukan perhitungan pada nilai bias b persamaan dilambangkan pada 2.20,

$$b = -\ 1\ 2\ (\Sigma i{=}0\ N\ \alpha iyiK(xi\ ,\ x-) + \Sigma i{=}0\ N\ \alpha iyiK(xi\ ,\ x+) \ldots\ldots (2.14)$$

Keterangan :

ai : alfa ke-i.

yi : kelas data ke-i

$K(xi\ ,\ x-)$: fungsi kernel data negatif

$K(xi\ ,\ x+)$: fungsi kernel data positif

5. Menghitung fungsi $f(x)$, dengan persamaan dilambangkan pada 2.15,

$$f(x) = \Sigma i=0\ m\ \alpha i yi K(xi\ ,\ x) + b \quad \text{(2.15)}$$

Keterangan :

b : Bias

αi : alfa ke-i

yi : kelas data ke-i

$K(xi\ ,\ x)$: fungsi kernel

Pada umunya terdapat empat jenis kernel yang dapat digunakan dalam SVM tetapi pada penelitian ini hanya menggunakan kernel linier sebagai berikut,

1. Kernel linier

kernel linier yaitu kernel yang digunakan ketika data yang diklasifikasikan dipisahkan oleh sebuah garis atau *hyperline*. Dilambangkan pada 2.16,

$$K(xi\ ,\ xj) = xi\ T\ xi \quad \text{(2.16)}$$

Keterangan :

$K(xi\ ,\ xj)$: Fungsi Kernel

x_i^T : Nilai x pada baris ke-i dengan x *transpose*

xj : Nilai x pada kolom ke-j

2.7.K-Nearset Neighbor

Metode K-Nearset Neighbor (K-NN) adalah algoritma K-NN yang mengklasifikasikan sekumpulan data atau supervised learning. Termasuk dalam supervised learning adalah klasifikasi contoh hasil kueri baru berdasarkan sebagian besar kedekatan kategori dalam K-NN. Ide dasar di balik K-NN adalah menemukan jarak terpendek antara data yang akan dievaluasi dan K tetangga terdekat dari data pelatihan. Rumus jarak Euclidean umumnya digunakan untuk mencari tetangga terdekat. Berikut ini adalah beberapa rumus yang digunakan dalam algoritma knn (Darmawan & Amini, 2022) yang dimana persamaannya dilambangkan pada 2.20,

$$d = \sqrt{(x1\ -\ x2)2\ -(\mathrm{y1}\ -\mathrm{y2})2} \quad \text{(2.20)}$$

Keterangan :

d : Jarak Kedekatan

x : Rumus jarak Euclidean umumnya digunakan untuk mencari

tetangga terdekat. Berikut ini adalah beberapa rumus yang digunakan dalam algoritma knn.Data Training

y : Data Testing

2.8. Validasi dan Pengujian

Validasi dan pengujian penelitian ini tentang kebenaran data yang diteliti, serta valid atau tidaknya instrumen tersebut. Instrumen penelitian adalah alat yang digunakan untuk mengumpulkan data atau mengukur objek penelitian guna mendapatkan data untuk validasi. Selain itu, mengukur akurasi metode *Support Vector Machine* dan *K-Nearest Neighbor* Pengujian dilakukan dengan menggunakan metode *Confution Matrix* (Hendra & Fitriyani, 2021)

2.8.1. Confusion Matrix

Confusion Matrix adalah analisis yang menilai kinerjanya untuk mengukur tingkat akurasi prediksi klasifikasi yang dihasilkan. Dengan menampilkan perbandingan antara prediksi dan data aktual, guna mengetahui perbedaan hasil klasifikasi antara prediksi dan data aktual pada Tiktok Review. Tabel Confusion Matrix dapat dilihat pada **Tabel 2.7** di bawah ini ;

Tabel 2.7 Confusion matrix

		Prediksi			**Jumlah**
		Negatif	Netral	Positif	
Aktual	Negatif	TNeg	FNet2	FP	N
	Netral	FNeg2	TNet	FP2	L
	Positif	FNeg	FNet	TP	P
Jumlah		N’	L’	P’	

Keterangan Tabel 2.7;

True Negatif (TN) : Dokumen diprediksi bernilai negatif dan diklasifikasikan sebagaikelas negatif (*true*)

True Positif (TP) : Dokumen diprediksi bernilai positif dan diklasifikasikan sebagaikelas positif (*true*)

True Netral (TL) : Dokumen diprediksi bernilai netral dan diklasifikasikan sebagai kelasnetral (*true*)

False Positif (FP) : Dokumen diprediksi bernilai positif namun diklasifikasikan

sebagaikelas negatif (*false*)

False Negatif (FNeg) : Dokumen diprediksi bernilai negatif namun diklasifikasikansebagai kelas positif (*false*)

False Netral (FNet) : Dokumen diprediksi bernilai netral namun diklasifikasikan sebagaikelas positif (*false*)

False Positif 2 (FP2) : Dokumen diprediksi bernilai positif namun diklasifikasikansebagai kelas netral (*false*)

False Negatif 2 (FNeg2) : Dokumen diprediksi bernilai negatif namun diklasifikasikansebagai kelas netral (*false*)

False Netral 2 (FNet2) : Dokumen diprediksi bernilai netral dan diklasifikasikan sebagaikelas positif (*false*)

Berdasarkan tabel *confusion matrix* tersebut dapat menghitung performance matrix untuk mengukur model yang telah dibuat. Performance matrix yang paling sering digunakan yaitu akurasi, presisi dan *recall*.

2.8.2. Akurasi

Accuracy adalah nilai yang menggambarkan seberapa akurat suatu model dalam melakukan klasifikasi. Ini mewakili tingkat kesamaan antara nilai prediksi dan aktual. Nilai akurasi dihitung dengan membagi jumlah total data dalam dataset dengan jumlah data positif prediksi positif dan data negatif prediksi negatif. Nilai persamaan akurasi dilambangkan pada 2.21,

$$accuracy = \frac{\mathrm{TP}+\mathrm{T}Neg+\mathrm{TNet}}{\mathrm{Total\ data}} \times \mathbf{100}\% \quad \text{.....(2.21)}$$

2.8.3. Presisi

Precision adalah nilai prediksi positif dalam kategori positif yang akan dibandingkan dengan hasil prediksi positif secara global. nilai persamaan presisi dilambangkan pada 2.22,

$$precision = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}+\mathrm{FP2}} \times \mathbf{100}\% \quad \text{.....(2.22)}$$

2.8.4. Recall

Recall yaitu membandingkan prediksi positif yang sebenarnya dengan semua data positif yang sebenarnya.Nilai persamaan recall dilambangkan pada 2.23,

$$recall = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FNeg}+\mathrm{FNet}} \times \mathbf{100}\% \quad \text{..... (2.23)}$$

2.8.5. F-1 Score

F1 Score merupakan hasil perbandingan presisi dan perolehan, yang akan dijumlahkan setelah kedua perhitungan tersebut.. Nilai persamaan $F1\ Score$ dilambangkan pada 2,24,

$$F1\ Score = \frac{2 \text{ x Presisi recall}}{\text{Presisi} + \text{recall}} \times \mathbf{100}\% \quad (2.23)$$

2.9. Penelitian Terdahulu

Penelitian terkait analisis sentiment dilakukan oleh (Sola et al., 2021)Analisis sentimen ulasan aplikasi Tiktok di google play store menggunakan metode Support Vector Machine (SVM) dan Asosiasi. Tiktok adalah aplikasi media sosial. TikTok adalah platform media sosial populer di Indonesia. Kepopuleran TikTok tak lepas dari kehadiran perusahaan tersebut di Indonesia yang sempat ditutup oleh pemerintah karena berbagai pelanggaran. Untuk mengetahui sentimen pengguna TikTok, analisis sentimen digunakan untuk mengkategorikan ulasan sebagai positif atau negatif. Sebuah Support Vector Machine (SVM) dengan metode kernel Radial Basis Function (RBF) digunakan untuk klasifikasi. Data yang digunakan dalam penelitian ini adalah 3.200 review aplikasi TikTok berbahasa Indonesia di Google Play Store dari September 2020 hingga Februari 2021. Tinjauan klasifikasi memiliki tingkat akurasi sebesar 90,62% dan kappa sebesar 81,24%, yang menunjukkan hasil klasifikasi yang hampir sempurna. Berdasarkan data, ulasan positif diberikan karena pengguna menyukai dan merasa nyaman dengan versi TikTok saat ini yang berisi video lucu di fyp. Sedangkan review negatif diberikan karena pengguna gagal melakukan registrasi dan akun diblokir, sehingga pengguna meminta TikTok untuk terus melakukan perbaikan.

Pada penelitian yang dilakukan oleh (Diki Hendriyanto et al., 2022) Analisis sentiment ulasan aplikasi Mola pada google play store menggunakan algoritma Support Vector Machine (SVM). Aplikasi Mola merupakan aplikasi streaming video yang dapat diunduh dari Google Playstore. Ada banyak ulasan tidak terstruktur yang berisi pendapat pengguna tentang kepuasan mereka terhadap aplikasi tersebut. Alhasil, seringkali menjadi pertimbangan bagi calon pengguna saat memilih aplikasi yang digunakan. Aplikasi MOLA meninjau 520 titik data yang digunakan, dengan 312 ulasan positif dan 208 ulasan negatif. Kernel RBF (Radial Basis Function) menghasilkan akurasi 92,31%, presisi 96,3%, recall 89,66%, dan skor f1 92,86% untuk hasil terbaik yang diminta pada skenario 1 (90:10).

Penelitian yang dilakukan oleh (Pamuji, 2021) Performance of the K-Nearest Neighbors method on analysis of social media sentiment. Penelitian ini dilakukan untuk mengetahui kinerja Algoritma KNN dan untuk mengetahui berbagai jenis komentar yang dilakukan pada aplikasi YouTube sehingga menjadi tantangan tersendiri bagi para pengelola. Konsep data mining dibutuhkan dalam penelitian ini melalui metode K-Nearest Neighbor sebagai alat untuk mengklasifikasikan komentar yang mungkin menjadi sentimen. Tahap analisis studi ini mengekstrak dataset dari YouTube dan kemudian melakukan pra-proses untuk tahap analisis. Hasil menunjukkan bahwa metode yang diusulkan dengan menggunakan teknik matriks fusi dapat menggambarkan tingkat akurasi hingga 88%.

Penelitian yang dilakukan oleh (Saputra, 2022) Analisis sentimen media sosial Twitter menggunakan algoritma *K-Nearest Neighbours.*Twitter adalah situs microblogging yang memungkinkan pengguna untuk menulis tentang berbagai topik dan mendiskusikan peristiwa terkini. Metode klasifikasi *K-Nearest Neighbor* digunakan dalam penelitian ini. K-Nearest Neighbor akan langsung mengklasifikasikan data pembelajaran yang akan dibentuk sebagai model untuk menentukan kategori data baru yang ingin ditentukan yaitu kelas sentimen positif, negatif, dan netral. Temuan penelitian ini menghasilkan sebuah sistem yang secara otomatis dapat mengklasifikasikan sentimen berdasarkan hasil pengujian nilai k. Nilai paling optimal yang diperoleh dari hasil penelitian adalah k = 1 dari total dataset sebanyak 450 tweet, dengan tingkat keberhasilan 51,11% dengan 315 data pelatihan 135 data uji.

Dalam Penelitian yang dilakukan Oleh (Sahara & Rizqi, 2022) Metode KNN digunakan untuk menganalisis sentimen produk game Android. Pada penelitian ini komentar review pengguna diberikan data berupa teks positif dan negatif dan diklasifikasikan menggunakan metode KNN. Menurut temuan penelitian, ulasan ini dapat mencakup umpan balik positif dan negatif. Beberapa peneliti juga menggunakan k-NN dalam klasifikasi teks karena kinerjanya yang baik. Berdasarkan pengolahan data yang telah selesai. Data review yang peneliti olah dapat dibagi menjadi dua kategori yaitu positif dan negatif. Pada Game Appstore For Android, akurasi k-NN data review produk sebesar 75,50%, dengan nilai AUC sebesar 0,825.

Pada Penelitian yang dilakukan (Trisnadi et al., 2021) Sentiment analysis pada movie review menggunakan *Feature Selection Mutual Information* dan *K-Nearest Neighbour* Classifier. Ulasan film juga bisa mendapatkan keuntungan dari analisis sentimen. Penonton

dapat mengetahui baik atau buruknya kualitas sebuah film dengan membaca review film. Pemirsa harus berusaha keras untuk mendapatkan informasi tentang film dengan membaca banyak ulasan film. Metode K-Nearest Neighbor dan pemilihan fitur mutual information digunakan dalam penelitian ini untuk menerapkan analisis sentimen pada ulasan film berbahasa Inggris. Polarity v2.0 dari dataset ulasan film Cornell digunakan. Penelitian ini menunjukkan bahwa dengan menggunakan metode K-Nearest Neighbor dan pemilihan fitur Mutual Information menghasilkan nilai akurasi sebesar 73,32% dengan nilai K sebesar 32 dan batas threshold 0,04 dan tidak menggunakan pemilihan fitur Mutual Information menghasilkan nilai akurasi sebesar 73,32. % dengan nilai K 32 dan ambang batas 0,04.mendapatkan akurasi sebesar 77,06%.

Penelitian yang dilakukan oleh (Nurhafida et al., 2022) Algoritma Support Vector Machine (SVM) digunakan untuk analisis sentimen pada aplikasi novel online di Google Play Store. Aplikasi Watppad dan Dreame yang biasanya dibaca online untuk mengisi waktu dan sebagai hiburan. Penelitian ini dilakukan untuk mengetahui kelebihan dan kekurangan aplikasi tersebut. Data tersebut diulas dan diperoleh dari Google Playstore, dimana data aplikasi Wattpad berjumlah 4.137. Akurasi hasil penelitian ini adalah 88,60%. Pengguna Wattpad mengungkapkan bahwa 33,12% sangat tidak menyukainya, 17,04% tidak menyukainya, 15,71% netral, 11,31% menyukainya, dan 22,82% sangat menyukainya.

Berikut Beberapa penelitian di atas telah dirangkum dalam **Tabel 2.8** merangkum nama peneliti, tahun penelitian, sentimen dan hasil akhir yang diperoleh pada setiap penelitian.

Tabel 2.8 State of The Art

No	Penulis	Judul	Data	Metode	Hasil
1.	(Sola Fide dkk., 2021)	*Analisis Sentimen ulasan Aplikasi Tiktok di google Playstore menggunakan metode Support Vector Machine (SVM) dan Asosiasi*	data ulasan aplikasi TikTok di Google Play Store sebanyak 3200 ulasan. 2560 data training dan 640 data testing.	*Support Vector Machine (SVM)* dan Asosiasi	Accuracy sebesar 90,62%.
2.	(Diki Hendriyanto dkk., 2022)	*Analisis Sentimen ulasan aplikasi Mola pada Google Playstore menggunakan Algoritma Support Vector Machine (SVM)*	Data diambil di google playstore sebanyak 520 data. 468 data training dan 52 data testing.	*Metode Support Vector Machine (SVM)*	Hasil accuracy 92,31%, precision 96,3%, recall 89,66%, dan f1-score 92,86%.

Tabel 2.9 State of The Art Lanjutan

No	Penulis	Judul	Data	Metode	Hasil
3.	(Pamuji, 2021)	*Performance of the K-Nearest Neighbors method on analysis of social media sentiment*	Data diambil dari ulasan pengguna youtube yaitu 2000 data. 1600 data training dan 400 data testing.	*K-Nearest Neighbor (K-NN)*	Hasil akurasi sebesar 88%
4.	(Sahara& Rizqi, 2022)	*Metode KNN pada sentiment analisis produk game android*	Dataset diambil dari review game. 100 data training dan 100 data testing.	*K-Nearest Neighbor (K-NN)*	Hasil akurasi sebesar 75.50%
5.	(Saputra, 2022)	*Analisis Sentimen Pada Media Sosial Menggunakan Metode K-Nearest Neighbor (K-NN)*	Dataset diambil dari review pengguna sebesar 450 data. 315 data training dan 135 data testing.	*K-Nearest Neighbor (K-NN*	Hasil akurasi sebesar 51,11%.
6.	(Nurhafida dkk., 2022)	*Analisis Sentimen Aplikasi Novel Online Di Google Play Store Menggunakan Algoritma Support Vector Machine (SVM)*	Data diambil dari ulasan google Playstore yaitu aplikasi Wattpad Sebanyak 4.137 data. 3309 data yang training dan 827 data testing.	*Support Vector Machine (SVM)*	Hasil penelitian ini mendapatkan akurasi sebesar 88,60%.
7.	(Trisnadi et al., 2021)	*Sentiment Analysis pada Movie Review Menggunakan Feature Selection Mutual Information dan K-Nearest Neighbour Classifier.*	dataset movie review berbahsa inggris dengan jumlah 2000. 1600 data training dan 400 data testing.	*Feature SelectionMutual Information dan K-Nearest Neighbour Classifier*	Hasil akurasi sebesar 73,32% dengan nilai K yaitu 32 dengan batas threshold 0,04 dan tanpa feature selection Mutual Information mendapatkan akurasi sebesar 77,06%.
8.	Zeti Ika Anggraini (2022)	*Perbandingan Metode Support Vector Machine dan K-Nearest Neighbor untuk analisis sentiment ulasan Pengguna aplikasi Tiktok Pada Google Playstore*	Data diambil dari ulasan Google Play Store Sebanyak 3000 ulasan. 2700 data training dan 300 data testing.	*Support Vector* Machine dan *K-Nearest Neighbor*	

Berdasarkan penelitian terdahulu yang memiliki topik bersangkutan, penelitian ini akan mengembangkan beberapa aspek, baik dari segi metode penelitian, objek, dan hasil penelitian. Berikut perbandingan penelitian terdahulu dengan penelitian ini:

1. Research gap mencari nilai akurasi kedua algoritma dengan membandingkan dua algoritma berdasrkan tingkat akurasi, presisi, recal dan f1 score.

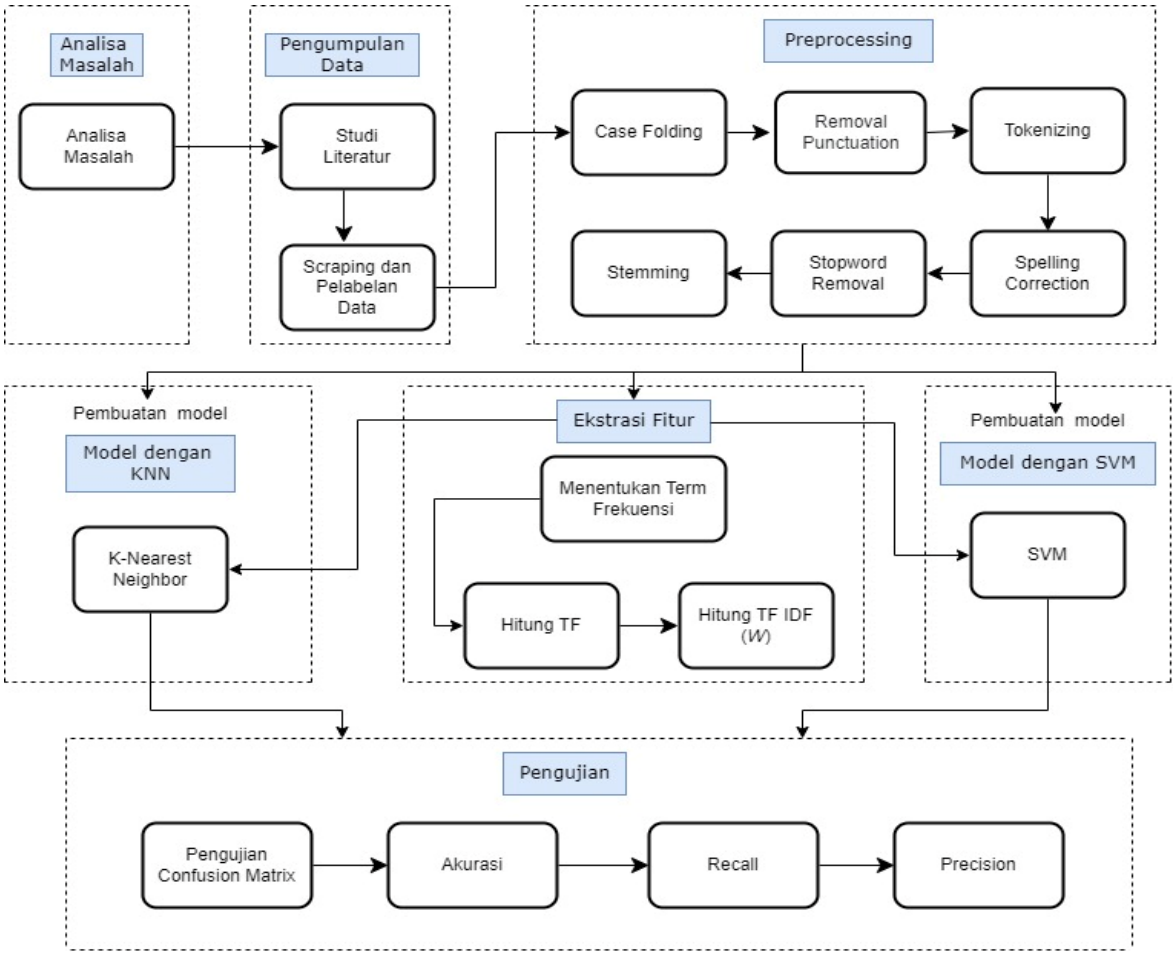
2. Penelitian yang dilakukan (Sola Fide dkk., 2021; Diki Hendriyanto dkk., 2022) menggunakan algoritma *Support Vector Machine* dengan kernel RBF sedangkan pada penelitian yang dilakukan menggunakan kernel linier.

3. Penelitian (Saputra, 2022) dan (Sahara & Rizqi, 2022) menggunakan algoritma *K-Nearest Neighbor* dengan nilai K yang berbeda dengan penelitian ini

BAB III
METODOLOGI PENELITIAN

3.1. Metodologi Penelitian

Metodologi penelitian merupakan sub bab yang membahas alur kerja dan tahapan yang akan dilalui dalam alur yang sistematis, sehingga dapat dijadikan pedoman dalam melakukan penelitian ini. Dalam penelitian ini, data dikumpulkan melalui scraping kemudian diberi label secara manual. Jumlah total data yang dikumpulkan adalah 3.000, dengan 2.400 data latih dan 600 data uji. Metodologi Penelitian yang digunakan dapat dilihat pada **Gambar 3.1,**



Gambar 3.1 Metodologi Penelitian

3.1.1. Analisa Masalah

Pada tahap ini dilakukan analisis masalah, serta solusi dari masalah yang ada. Analisis ini dilakukan dengan membaca jurnal komparatif atau tugas akhir dan meninjau pengguna

menggunakan metode *Support Vector Machine* dan *K-Nearest Neighbor*. Permasalahan penelitian ini adalah membandingkan performansi kedua metode untuk mendapatkan nilai Accuracy, Precision, Recall dan F1 Score. Dua masalah yang dipilih adalah bagaimana meninjau pengguna aplikasi Tiktok sebagai objek penelitian dan membandingkan keakuratannya. Kemudian akan dibagi menjadi tiga kategori: positif, negatif dan netral. Dengan dua permasalahan tersebut, peneliti berharap dapat mengembangkan sistem yang menunjukkan bahwa Metode Support Vector Machine dan K-Nearest Neighbor Comparison dapat digunakan untuk mengevaluasi kepuasan pengguna.menggunakan aplikasi Tiktok.

3.1.2. Studi Literatur

Studi literatur merupakan tahapan yang dilakukan untuk mendapatkan informasi yang paling mutakhir tentang penelitian yang dilakukan. Informasi yang dikumpulkan meliputi masalah penelitian, ide yang digunakan, perbedaan penelitian yang akan dilakukan serta kelebihan dan kekurangan dari penelitian yang telah dilakukan. Tujuan penelitian kepustakaan adalah untuk memperkuat argumentasi dalam penelitian ini dan mempermudah penelitian bagi penulis. Penelitian ini didasarkan pada makalah, jurnal ilmiah dan sumber terpercaya lainnya.

3.1.3. Pengumpulan Data

Pengumpulan data merupakan tahapan yang harus dilakukan secara cermat agar data yang terkumpul dapat menentukan kebutuhan dan tujuan suatu sistem, sehingga terhindar dari resiko kesalahan pengumpulan data. Data yang digunakan diambil dari Google Play Store.

https://play.google.com/store/apps/details?id=com.ss.android.ugc.trill.

Gambar Dataset Dapat dilihat pada **Gambar 3.2** sebagai Berikut ;

	A	B	C	D
1	USERNAME	SCORE	AT	komentar
2	Jihan Azalia	5	2022-07-22 10:22:25	Bagus banget
3	Nur Aisyah	5	2022-07-22 10:22:14	Bagus bangettt.emang bagus
4	Sar Mila	4	2022-07-22 10:19:38	Aplikasi ini bagus tapi boros kuota
5	Dewi Murni	5	2022-07-22 10:18:52	Aku suka tiktok soalnya seru banget
6	Martabak populer	5	2022-07-22 10:16:11	keren
7	bathie channel	5	2022-07-22 10:14:30	Mantul pokoknya tiktok
8	Alfiansyah Alfairusy	5	2022-07-22 10:13:48	Saya puasss
9	Tunjang Telaga	3	2022-07-22 10:11:49	Bintang tiga ajah
10	Rimbi Kalianda	5	2022-07-22 10:11:17	bagus dan keren
11	Bayu Aldino	5	2022-07-22 10:09:43	Bagus
12	Nana moey	5	2022-07-22 10:09:33	Uh bagus amett
13	Ilham Ninuk	5	2022-07-22 10:09:33	Bagus sangatÂ² bagus
14	alvian zulfikar	5	2022-07-22 10:08:48	Apk yang sangat membagong kan
15	Fadil Afdal	5	2022-07-22 10:07:20	di jamin aman ga memboroskan pulsa
16	kimyerikim	5	2022-07-22 10:06:35	good
17	Muhammad Taufik urahman	5	2022-07-22 10:06:11	Sangat berkesan
18	Mangg Naimm	1	2022-07-22 10:05:58	Orang yang kita sayang dan perhatian kami akan bantu kita kembali semula
19	Wadon Aja	1	2022-07-22 10:04:55	Tiktok gmna si susah di instal anjy
20	Walls Ogankss	5	2022-07-22 10:04:40	Sy suka tiktokan
21	Denok Dewi Maryanti	5	2022-07-22 10:03:08	Tik tok nya bagus banget Tik tok paling ngetop deh.
22	Raya Azahra	5	2022-07-22 10:00:45	Oke lah
23	Agus Ruswandi	5	2022-07-22 09:58:20	Bagus
24	Tasya Saera	1	2022-07-22 09:57:24	I don't like your aplicassion ,because very lag
25	PIC0	5	2022-07-22 09:56:31	keren dan sangat menarik
26	MUFLIH GAMERS	1	2022-07-22 09:55:58	Mohon segera di perbaiki tanya jawab nya saya ingin bertanya serta menjawab p
27	Istiyono Istiyono	5	2022-07-22 09:55:43	Ok
28	Intan Tarigan	2	2022-07-22 09:55:17	Kok saya tidak bissa download tiktok lagi ya. Pliss kasi tau caranya dong

Gambar 3.2 Dataset Penelitian

3.1.4. *Scraping* dan *Labelling* Data

Bagian ini akan mengumpulkan data untuk penelitian dengan mengorek ulasan dari pengguna aplikasi Tiktok di Google Playstore. Pengikisan data diberi label setelah dikumpulkan. Data review pengguna aplikasi Tiktok merupakan data primer, dengan 3.000 opini review pengguna aplikasi pada Juli 2022. Data dibagi menjadi 80:20 yang dimana 2.400 data training dan 600 data testing. Bahasa pemrograman Python digunakan untuk data scraping, dan pelabelan data dilakukan secara manual. Proses pelabelan berusaha mengkategorikan semua kelas sebagai positif, negatif, atau netral. Setelah dilakukan pelabelan, akan dihasilkan tipe data dengan sentimen yang jelas yang dapat ditarik dari benang merah semua sentimen, menghasilkan data balanced atau unbalanced data.

3.1.5. Text Preprocessing

Text preprocessing adalah sekumpulan prosedur yang digunakan untuk mengelola data teks. Preprocessing merupakan fungsi pembersihan dan penyiapan data sebelum proses klasifikasi dilakukan. Terdiri dari beberapa tahapan yang saling berhubungan yang dapat digunakan tergantung dari kebutuhan sistem yang akan dibangun. beberapa proses yang dilakukan. *case folding, remove punctuation, remove number, stopword removal, spelling correction dan stemming.*

a. Case Folding

Case Folding merupakan gambaran alur perubahan teks data yang bertujuan untuk mengubah karakter pada data sebelumnya, khususnya yang sebelumnya menggunakan huruf kapital, menjadi huruf kecil menggunakan *lower case().* Contoh kasus keluaran proses pelipatan dapat dilihat pada **Tabel 3.1**,

Tabel 3.1 Contoh Hasil *Case Folding*

Sebelum Case Folding	Sesudah Case Folding
Bagus banget	bagus banget

b. Remove Punctuation

Remove Punctuation adalah proses perbaikan data dari simbol-simbol dari data yang sebelumnya dihilangkan dari angka. Tanda koma (,), titik (.), tanda tanya (?), dan tanda seru (!) adalah contoh dari simbol-simbol tersebut. Contoh hasil dari proses *Remove Punctuation* dapat dapat dilihat pada **Tabel 3.2**,

Tabel 3.2 Contoh Hasil *Remove Punctuation*

Sebelum *Remove Punctuation*	Sesudah *Remove Punctuation*
Kenapa update mulu ya??	Kenapa update mulu ya

c. Tokenizing

Tokenizing adalah proses membuat kata-kata dengan menetapkan token untuk masing-masing kata. Contoh hasil dari proses *Tokenizing* dapat dapat dilihat pada **Tabel 3.3**,

Tabel 3.3 Contoh Hasil *Tokenzing*

Sebelum *Tokenizing*	Sesudah *Tokenizing*
Aplikasi ini keren	[‘aplikasi’],[‘ini’],[‘keren’]

d. Spelling Correction

Spelling correction digunakan untuk mengembalikan bentuk istilah atau kata yang tidak baku ke bentuk aslinya. Data yang akan diolah berasal dari hasil proses *tokenizing*. Contoh hasil dari proses *Spelling Correction* dapat dapat dilihat pada **Tabel 3.4**,

Tabel 3.4 Contoh Hasil *Spelling Correction*

Sebelum *Spelling Correction*	Sesudah *Spelling Correction*
“gw suka aplikasi ini bagus banget”	“gue suka aplikasi ini bagus banget”

e. Stopword Removal

Stopword Removal adalah proses menghilangkan kata-kata yang dianggap tidak perlu, seperti kata ganti orang, kata sambung, atau kata-kata yang tidak memiliki arti tertentu. Proses Stopword Removal dimulai dengan mengecek setiap kata dari hasil koreksi ejaan, kemudian mencocokkan kata tersebut dengan stopword kamus; jika kata itu ada dalam kamus, kata itu dihapus; jika kata tersebut tidak ada dalam kamus Stopword Removal, kata tersebut akan dihapus. Hasil stopword disimpan sebagai output. Contoh hasil dari proses *Stopword Removal* dapat dapat dilihat pada **Tabel 3.5**,

Tabel 3.5 Contoh Hasil *Stopword Removal*

Sebelum *Stopword Removal*	Sesudah *Stopword Removal*
di jamin aman ga memboroskan pulsa	jamin aman ga boros pulsa

f. Stemming

Proses *Stemming* Dengan menghilangkan akhiran, kata tersebut dikembalikan ke bentuk dasarnya. Data yang digunakan diperoleh melalui proses stop word removal. Menghilangkan imbuhan pada kata yang diawali dengan akhiran infleksi (-lah, -kah, -ku, dst), diikuti dengan penghilangan akhiran turunan (-i, -kan, -an) dan awalan turunan (-be, -di, -me , -pe, -se, dan -te). Contoh hasil dari proses *Stemming* dapat dapat dilihat pada **Tabel 3.5**,

Tabel 3.6 Contoh Hasil *Stemming*

Sebelum *Stemming*	Sesudah *Stemming*
Mohon segera di perbaiki tanya jawab nya saya ingin bertanya serta menjawab pertanyaan kepada kreator "ada masalah" tolong segera di perbaikii jamin aman ga memboroskan pulsa	mohon baik nya kreator yg muncul tulis tolong baik

3.1.6. Pembobotan TF-IDF (Term Frequency-Inverse)

Term Frequency (TF-IDF) Inverse document weighting merupakan metode pengolahan data yang telah melalui proses preprocessing dan query expansion feature selection. Pembobotan TF-IDF digunakan untuk memberi bobot pada term dalam dokumen teks. Term frequency digunakan untuk menghitung kemunculan term dalam satu dokumen, sedangkan reverse document frequency digunakan untuk menghitung kemunculan term dalam banyak dokumen. Berikut adalah contoh dokumen yang telah diproses sebelumnya, dapat dilahat pada **Tabel 3.7**,

Tabel 3.7 Dokumen preprocessing

ID	Dokumen	Sentimen	Label
Dokumen 1	['aplikasi'], ['ini'], ['bagus']	Positif	1
Dokumen 2	['bagus'], ['banget']	Positif	1
Dokumen 3	['tiktok'], ['pelit'], ['fyp']	Netral	0
Dokumen 4	['Jelek']	Negatif	-1
Dokumen 5	['sangat'], ['menghibur']	Positif	1

1. Perhitungan TF (*Term Frequency*)

Term Frequency adalah konsep yang dibobotkan dengan mencari (frekuensi) term yang sering muncul dalam suatu dokumen (Nurrun Muchammad Shiddieqy et al., 2016) Berikut adalah perhitungan manual dari TF;

Dokumen 1 : ['aplikasi'], ['ini'], ['bagus']

Maka :

$$\textbf{TF} = \frac{\textit{Jumlah Frekuensi Kata yang terpilih}}{\textit{Jumlah Kata}}$$

Diketahui :

Aplikasi = 1

Jumlah Kata = 3

Jadi $TF_{Aplikasi}$ = $\frac{1}{3}$ = 0,3333

Diketahui :

ini = 1

Jumlah Kata = 3

Jadi TF_{ini} = $\frac{1}{3}$ = 0,3333

Diketahui :

bagus = 1

Jumlah Kata = 3

Jadi TF_{bagus} = $\frac{1}{3}$ = 0,3333….dst hingga per kata dokumen 1 habis.

Dokumen 2 : ['bagus'], ['banget']

Diketahui :

bagus = 1

Jumlah Kata = 2

Jadi TF_{bagus} = $\frac{1}{2}$ = 0,5

Diketahui :

banget = 1

Jumlah Kata = 2

Jadi TF_{banget} = $\frac{1}{2}$ = 0,5

Berikut **Tabel 3.8** adalah hasil TF dan DF dari D1, D2, D3, D4.

Tabel 3.8 Hasil TF dan DF

Indeks	***Term***	**TF**					**DF**
		D1	**D2**	**D3**	**D4**	**D5**	
0	Aplikasi	0,3333	0	0	0	0	1
1	Ini	0,3333	0	0	0	0	1
2	Bagus	0,3333	0,5	0	0	0	2
3	Bagus	0,3333	0,5	0	0	0	2
4	Banget	0	0	0	0	0	1
5	Tiktok	0	0	0,3333	0	0	1
6	Pelit	0	0	0,3333	0	0	1
7	Fyp	0	0	0,3333	0	0	1
8	Jelek	0	0	0	1	0	1
9	Sangat	0	0	0	0	0,5	1
10	Menghibur	0	0	0	0	0,5	1

2. Perhitungan IDF (*Inverse Document Frequency*)

IDF (Inverse Document Frequency) adalah perhitungan yang menerapkan formula yang ada untuk mendistribusikan term secara luas pada kumpulan dokumen yang bersangkutan. IDF juga berfungsi untuk mengurangi bobot suatu istilah jika kemunculannya tersebar luas di seluruh koleksi dokumen kita. mengikuti IDF (Sola et al., 2021) .Berikut adalah contoh perhitungan manual pada dokumen 1, dokumen 2, dokumen 3 dan dokumen 4 sehingga total keseluruhan adalah 4 dokumen :

Dokumen 1 : ['aplikasi'], ['ini'], ['bagus']

$$\boldsymbol{IDF_t = \log \frac{N}{dft} + 1}$$

Diketahui :

Perhitungan pada term Aplikasi

Aplikasi = 1

Jadi $IDF_{Aplikasi}$ = $\log \frac{4}{3} + 1 = 1{,}3333$

Perhitungan pada term Aplikasi

Ini = 1

Jadi IDF_{Ini} = $\log \frac{4}{3} + 1 = 1{,}3333$

Hasil perhitungan dari IDF pada D1, D2, D3 dan D4 dapat dilihat pada **Tabel 3.9**,

Tabel 3.9. Hasil IDF

Indeks	***Term***	**TF**					**IDF**
		D1	**D2**	**D3**	**D4**	**D5**	
0	Aplikasi	1	0	0	0	0	1,3333
1	Ini	1	0	0	0	0	1,3333
2	Bagus	2	2	0	0	0	1,3333
3	Bagus	2	2	0	0	0	2
4	Banget	0	1	0	0	0	2
5	Tiktok	0	0	1	0	0	1,3333
6	Pelit	0	0	1	0	0	1,3333
7	Fyp	0	0	1	0	0	1,3333
8	Jelek	0	0	0	1	0	1
9	Sangat	0	0	0	0	1	2

10	Menghibur	0	0	0	0	1	2

3. Perhitungan TF IDF

Setelah perhitungan dari TF dan IDF sudah di dapatkan, masuk pada tahap yang selanjutnya yaitu perhitungan TF-IDF yang keseluruhan. Dengan menggunakan persamaaan yang sudah ada. Berikut Contoh perhitungan manual yang akan dilakukan pada Dokumen 1, dokumen 2, dokumen 3 dan dokumen 4 :

$$Wdt = TFdt \times idft$$

Diketahui :

Dokumen 1 : ['aplikasi'], ['ini'], ['bagus']

$TF_{Aplikasi}$ = 1

$IDF_{Aplikasi}$ = 3

Jadi Wdt = 0,3333 x 1,3333 = 0

Berikut merupakan **Tabel 3.10** hasil dari perhitungan TF – IDF,

Tabel 3.10 Perhitungan TF – IDF

Indeks	*Term*	DF	IDF	TF – IDF				
				D1	D2	D3	D4	D5
0	Aplikasi	1	1,3333	0,4443	0000	0000	0000	0000
1	Ini	1	1,3333	0,4443	0000	0000	0000	0000
2	Bagus	2	1,3333	0,4443	1	0000	0000	0000
3	Bagus	2	2	0,4443	1	0000	0000	0000
4	Banget	1	2	0000	1	0000	0000	0000
5	Tiktok	1	1,3333	0000	0000	0,4443	0000	0000
6	Pelit	1	1,3333	0000	0000	0,4443	0000	0000
7	Fyp	1	1,3333	0000	0000	0,4443	0000	0000
8	Jelek	1	1	0000	0000	0000	1	0000
9	Sangat	1	2	0000	0000	0000	0000	1
10	Menghibur	1	2	0000	0000	0000	0000	1

Untuk mendapatkan nilai TF-IDF dengan melakukan perhitungan sebagai berikut :

$Wt,d = Wtft,d \times idft$

Wt,d = 1 × 1,204

Wt,d = 1,204

perhitungan TF-IDF ini dituliskan dalam bentuk notasi vektor yang akan digunakan untuk melakukan training dengan menggunakan metode *Support Vector Machine*. Dapat dilihat pada **Tabel 3.11,**

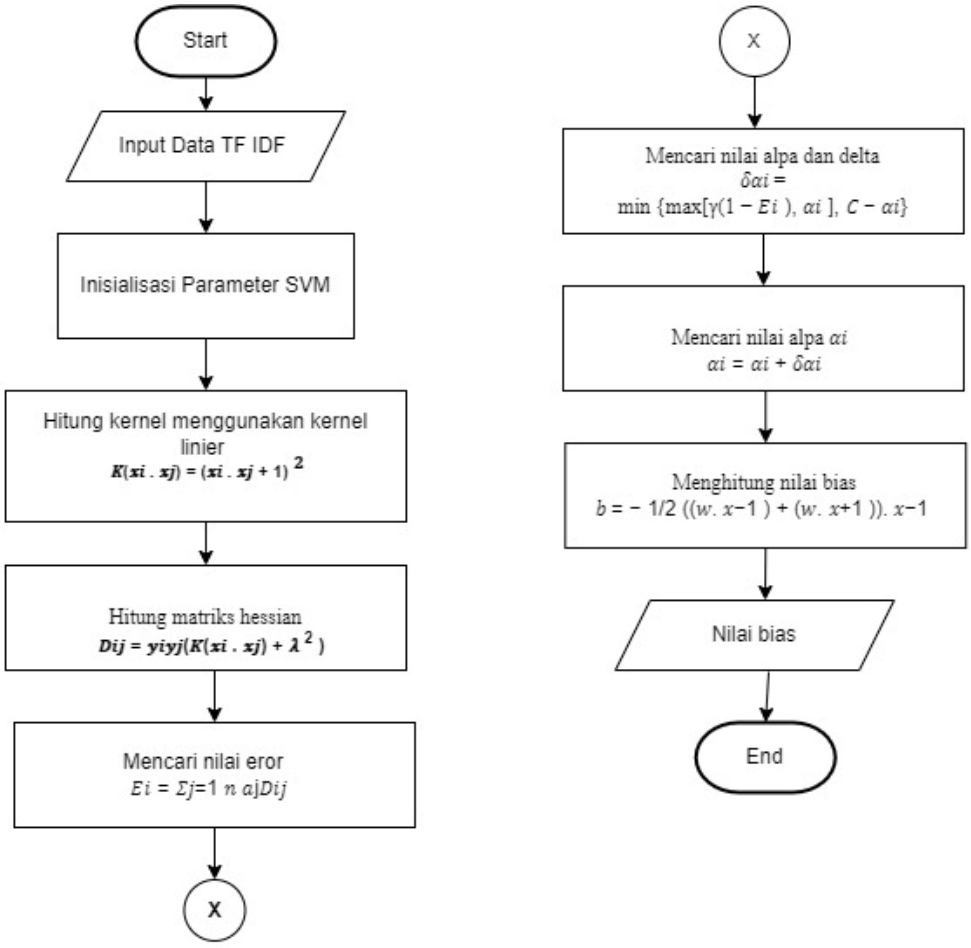
Tabel 3.11 Hasil Perhitungan Akhir TF-IDF

No	X1	X2	X3	X4	X5
1.	1.3333	0	0	0	0
2.	1.3333	0	0	0	0
3.	2.6666	2	0	0	0
4.	0	2	0	0	0
5.	0	1	0	0	0
6.	0	0	0.4443	0	0
7.	0	0	0.4443	0	0
8.	0	0	0	1	0
9.	0	0	0	0	0.5
10.	0	0	0	0	0.5

3.1.7. Pembuatan Model

a. Model Menggunakan *Support Vector Machine*

Untuk mendapatkan hasil akhir dalam melakukan penelitian yaitu nilai perbandingan dan sentimen dari data komentar ulasan aplikasi tiktok pada google playstore, dilakukan perhitungan dengan menggunakan metode klasifikasi *support vector machine*. Metode perhitungan yang akan digunakan yaitu *sequential support vector machine* dengan kernel linear. Beberapa *Flowchart* dari proses klasifikasi menggunakan *support vector machine* dapat dilihat pada **Gambar 3.3**,



Gambar 3.3 Flowchart Proses Support Vector Machine

Berikut Perhitungan manual dari metode *support vector machine*. Langkah-langkah yang akan dilakukan dengan metode *sequential support vector machine* yaitu sebagai berikut.

1. Melakukan inisialisasi pada parameter :
   a. Maksimum iterasi = 2
   b. λ = 0,5
   c. γ = 0,001
   d. α = 0
   e. C = 1
   f. ε = 0,0001
2. Setelah melakukan inisialisasi parameter, selanjutnya yaitu mencari nilai

kernelisasi matriks K dan matriks hessian berukuran n x n sejumlah banyaknya data training. Berikut contoh perhitungannya.

$K(xi\ .\ xj) = (xi\ .\ xj + 1)^{2}$

$K(xi\ .\ xj) = (x1.\ x2 + 1)^{2}$

$K(x1.\ x2) = (((1{,}3333 \times 1{,}3333) + (1{,}3333 \times 1{,}3333) + (2{,}6666 \times 2{,}6666) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + 1)^{2}$

$K(x1.\ x2) = 11{,}66613$

Lakukan hal yang sama untuk dokumen lainnya sehingga memperoleh matriks K sepertipada **Tabel 3.12**,

Tabel 3.12 Hasil Perhitungan Matriks K

Dokumen	D1	D2	D3	D4	D5
D1	11, 666	1	1	1	1
D2	1	10	1	1	1
D3	1	1	1,394	1	1
D4	1	1	1	2	1
D5	1	1	1	1	1,5

Untuk mendapatkan nilai dari matriks hessian, dapat dilakukan perhitungan seperti contoh berikut :

$Dij = yiyj(K(xi\ .\ xj) + \lambda^{2})$

$D11 = (1)(1)(\ 11{,}196 + 0{,}5\ 2\ )$

$Dij = 11{,}196$

Lakukan hal yang sama untuk dokumen lainnya sehingga memperoleh matriks hessian seperti pada **Tabel 3.13**,

Tabel 3.13. Hasil Perhitungan Matriks Hessian

Dokumen	D1	D2	D3	D4	D5
D1	11,196	1.250	1.250	1.250	1.250
D2	1.250	10.250	1.250	1.250	1.250
D3	1.250	1.250	0.3485	1.250	1.250
D4	-0.750	1.250	1.250	2.250	1.250
D5	1.250	1.250	1.250	-0.750	0.375

3. Setelah menghitung matriks hessian, langkah selanjutnya yaitu mencari nilai Ei . Persamaan yang digunakan yaitu $Ei = \Sigma j{=}1\ n\ aiDij$.

Ei = (0x11,196) + (0x1,250) + (0x(0,150)) + (0x(1,250)) + (0x-0,750) +(0x1,250)

= 0

Lakukan perhitungan yang sama untuk dokumen lainnya sehingga memperoleh nilai Ei seperti pada **Tabel 3.14**,

Tabel 3.14 Hasil Perhitungan Ei iterasi pertama

Dokumen	Ei
D1	0
D2	0
D3	0
D4	0
D5	0

4. Kemudian mencari nilai $\delta\alpha i$ dengan menggunakan persamaan $\delta\alpha i$ = min {max[γ(1 − Ei), αi], $C - \alpha i$}. Berikut perhitungannya. $\delta\alpha i$ = min {max[0,001(1 − 0), 0], 1 − 0} = 0,001

   Lakukan perhitungan yang sama untuk dokumen lainnya sehingga memperoleh nilai $\delta\alpha i$ seperti pada **Tabel 3.15**,

Tabel 3.15. Hasil Perhitungan $\delta\alpha i$ iterasi Pertama

Dokumen	$\delta\alpha i$
D1	0,001
D2	0,001
D3	0,001
D4	0,001
D5	0,001

5. Setelah nilai $\delta\alpha i$ ditemukan, maka selanjutnya mencari nilai αi baru dengan persamaan $\alpha i = \alpha i + \delta\alpha i$. Berikut perhitungannya.

   $\alpha i = \alpha i + \delta\alpha i$

   = 0 + 0,001

   = 0,001

   Lakukan perhitungan yang sama untuk dokumen lainnya sehingga memperoleh nilai αi seperti pada **Tabel 3.16**,

Tabel 3.16 Hasil Perhitungan αi iterasi Pertama

Dokumen	αi
D1	0,001
D2	0,001
D3	0,001
D4	0,001
D5	0,001

6. Langkah 3 hingga 5 dilakukan terus menerus hingga mencapai batas

maksimum iterasi. Berikut merupakan perhitungan iterasi ke-2.

Ei = (0,001x11,196) + (0,001x1,250) + (0,001x1,250) +(0,001x−0,750)) + (0,001x1,150)

= −1,2639

Hasil Perhitungan proses iterasi ke-2 dapat dilihat pada **Tabel 3.17,**

Tabel 3.17 Hasil Perhitungan Ei iterasi kedua

Dokumen	Ei
D1	−1,2639
D2	0.0014
D3	0,0143
D4	0,0015
D5	0,0145

$\delta\alpha i$ = min {max[γ(1 − Ei), αi], $C - \alpha i$}

$\delta\alpha i$ = min {max[0,001(1 − 1,26397), 0,001], 1 − 0,001}

= 2,2639

Lakukan perhitungan yang sama untuk dokumen lainnya sehingga memperoleh nilai $\delta\alpha i$ seperti pada **Tabel 3.18**,

Tabel 3.18 Hasil Perhitungan $\delta\alpha i$ iterasi kedua

Dokumen	$\delta\alpha i$
D1	2,2639
D2	0.0986
D3	−0.0143
D4	−1.0015
D5	−1.0145

$\alpha i = \alpha i + \delta\alpha i$

= 0,001 + 2,2639

= 0,0020

Lakukan perhitungan yang sama untuk dokumen lainnya sehingga memperoleh nilai αi seperti pada **Tabel 3.19**,

Tabel 3.19 Hasil Perhitungan αi iterasi kedua

Dokumen	αi
D1	−0,2640
D2	0,0987
D3	41,807
D4	0,0016
D5	0,0155

7. Selanjutnya yaitu melakukan perhitungan untuk nilai bias dengan

menggunakan persamaan

$b = -1\ 2\ ((w.\ x{-}1\ ) + (w.\ x{+}1\ ))$. $x{-}1$ merupakan kelas negatif dengan nilai alpha terbesar dan $x{+}1$ merupakan kelas positif dengan nilai alpha terbesar.

$w.\ x{+}1 = (1x0{,}0987x1) + (1x0{,}0987x10) + (0x0{,}0987x1) + (-1x0{,}0987x1) + (1x0{,}0987x1)$

$= 1{,}0857$

$w.\ x{-}1 = = (1x{-}0{,}2640x11{,}196) + (1x{-}0{,}2640x1) + (0x{-}0{,}2640x1) + (-1x{-}0{,}2640x1) + (1x{-}0{,}2640x1)$

$= -32{,}197$

Dengan demikian, nilai bias dapat diketahui dengan :

$b = -\frac{1}{2}(\ 1{,}0857 - 32{,}197)$

$= -31{,}111$

8. Selanjutnya jika nilai bias sudah ditemukan, maka proses analisis akan dilanjutkan pengujian untuk mendapatkan klasifikasi kata. Data uji yang ingin dilakukan proses klasifikasi sebelumnya telah dilakukan tahap preprocessing kemudia dilakukan pembobotan TF-IDF.

   TF-IDF = [0,4443, 0000, 0000, 0,4443]

   Uji data = [0,4443] yaitu kata (Bagus)

   Kernelisasi :

   a. $K(x\ .\ xj) = (xi\ .\ xj + 1)^{2}$

$= (0{,}4333 \times 0{,}4333) + (0{,}4333 \times 0{,}4333) + (0{,}4333 \times 0{,}4333) + (0{,}4333 \times 0{,}4333) + (0 \times 0{,}4333) + 1)^{2}$

$= ((0,\ 1974) + (0,\ 1974) + (0,\ 1974) + (0,\ 1974) + 0 + 1)^{2}$

$= (0{,}7896 + 1)^{2}$

$= (1{,}7896\ )^{2}$

$= 3{,}202$

$K(x.\ xj) = ((0000 \times 0{,}4333) + (1 \times 0{,}4333) + (1 \times 0{,}4333) + (1 \times 0{,}4333 + 1)^{2}$

$= (01{,}3329\ ) + 1)^{2}$

$= (2{,}3329)^{2}$

$= 5{,}442$

x^3 = (0,1794 + 0,1794) + 0,1794 + 1) ^{2}

= 2,535

Hasil Kernelisasi yang di dapatkan pada perhitungan dapat dilihat sebagai berikut ;

$K(x\,.\,x1) = 3{,}202$

$K(x\,.\,x2) = 5{,}442$

$K(x\,.\,x3) = 2{,}535$

$K(x\,.\,x4) = 2{,}086$

$K(x\,.\,x5) = 3{,}566$

9. Setelah proses kernelisasi dilakukan pada semua data, selanjutnya adalah menentuka proses klasifikasi kelas data.$f(x) = \Sigma_{im=0}\alpha_i y_i K(x_i, x) + b$. Jika nilai yang dihasilkan 1, maka termasuk dalam kelas positif. Jika nilai yang dihasilkan -1, maka termasuk dalam kelas negatif. Jika nilai yang dihasilkan 0, maka termasuk dalam kelas netral.

$f(x) = sign((1 \times \text{-}0{,}2640 \times 3{,}202) + (1 \times 0{,}0987 \times 5{,}442\ ) + (1 \times 41{,}807 \times 2{,}535) + (1 \times 0{,}0016 \times 2{,}086\ ) + (1 \times\ 0{,}0155\ \times 3{,}566) + (-31{,}111))$
$= (-0{,}845) + (0, 537) + (105{,}9) + (0{,}003) + (0, 055) - (31{,}111)$
$= sign\ (74, 536)$
$= 0{,}963$
$= 1$ (kelas positif)

10. Jadi hasil klasifikasi yang dilakukan kata (Bagus) masuk pada kelas positif

b. Model Menggunakan *K -Nearest Neighbor*

Dengan menggunakan metode *K-Nearest Neighbor* yang sering digunakan dalam algoritma pembelajaran Beberapa *Flowchart* dari proses klasifikasi menggunakan *K-Nearest Neighbor* dapat dilihat pada **Gambar 3.12**,

Gambar 3.4 *Flowchart* Proses *K-Nearest Neighbor*

Berikut Perhitungan manual dari metode *K-Nearest Neighbor*. Langkah-langkah yang akan dilakukan dengan metode *K-Nearest Neighbor* yaitu sebagai berikut menghitung nilai K dengan menggunakan persamaan,

$$d = \sqrt{(x1 - x2)2 - (y1 - y2)2}$$

Berikut merupakan contoh kasus dalam pengklasifikasian *headline* menggunakan *K-Nearest Neighbor headline* atau dokumen yang akan diklasifikasikan sudah melalui tahap *preprocessing*. Contoh Kasus Dapat dilihat pada **Tabel 3.20,**

Tabel 3.20 Contoh kasus

Indeks	Kata	D1	D2	D3	Kategori
1.	Aplikasi	1	0	0	Positif
2.	Fyp	1	0	1	Netral
3.	Jelek	0	0	0	Negatif
4.	Bagus	-1	2	0	????

Dibentuk sebuah model menghitung data , dengan mengacu pada persamaan;

1) Perhitungan data dengan kata

   = Aplikasi

$d=$

$$\sqrt{(Aplikasi\ -\ Bagus)2\ +\ (\text{Aplikasi}\ -\text{Bagus})2\ +\ (\text{Aplikasi}\ -\text{Bagus})2}$$

$$d = \sqrt{(1\ -\ 2)2\ +\ (1\ -2)2\ +\ (1\ -0)2}$$

$$d = \sqrt{1\ +1\ +\ 1}$$

$$d = 1\ -\ 22\ +1\ -22\ +\ d = \sqrt{3}$$

$$= \sqrt{1{,}7}$$

2) Perhitungan data dengan kata

= Fyp

$$d = \sqrt{(Fyp\ -\ Bagus)2\ +\ (\text{Fyp}\ -\text{Bagus})2\ +\ (\text{Fyp}\ -\text{Bagus})2}$$

$$d = \sqrt{(0\ -\ 2)2\ +\ (0\ -2)2\ +\ (0\ -0)2}$$

$$d = \sqrt{-2\ +-2\ +\ 0}$$

$$d = 1\ -\ 22\ +1\ -22\ +\ d = \sqrt{-4}$$

$$= \sqrt{2}$$

3) Perhitungan data dengan kata

= Jelek

$$d = \sqrt{(Jelek\ -\ Bagus)2\ +\ (\text{Jelek}\ -\text{Bagus})2\ +\ (\text{Jelek}-\text{Bagus})2}$$

$$d = \sqrt{(0\ -\ 2)2\ +\ (0\ -2)2\ +\ (0\ -0)2}$$

$$d = \sqrt{-2\ +-2\ +\ 0}$$

$$d = 1\ -\ 22\ +1\ -22\ +\ d = \sqrt{-4}$$

$$= \sqrt{2}$$

Nilai Hasil Akhir yang di dapatkan pada kata “Bagus” dengan nilai K terdekat yaitu Positif. Tabel Hasil Akhir dapat dilihat pada **3.21** sebagai berikut,

Tabel 3.21 Hasil Akhir Perhitungan K terdekat

Indeks	Kata	Sentimen	D1	D2	D3	Kategori
1.	Aplikasi	Positif	2	2	0	Positif
2.	Fyp	Netral	2	2	0	Netral
3.	Jelek	Negatif	2	2	0	Negatif
4.	Bagus	**Positif**	**2**	**2**	**0**	**Positif**

3.1.8. Pengujian

Pengujian model dilakukan dengan menggunakan *confution matrix*. Pengujian dilakukan pada perbandingan model *Support Vector Machine* dan *K Nearest Neighbor* untuk analisis Sentimen kepuasan pengguna aplikasi.

3.1.9. Analisis Dan Hasil

Analisis dan kesimpulan dilakukan dengan membandingkan hasil pengujian model *Support Vector Machine* dan *K Nearest Neighbor* berdasarkan akurasi, presisi, dan *recall*.

3.2. Pengembangan Sistem

Pada penelitian ini metode pengembangan sistem yang digunakan adalah metode *prototyping*. dengan tahapan *communication, quick plan, modeling quick design, construction of prototype,* dan *deployment delivery-feedback*

3.2.1. Communication

Tahap *communication* dilakukan dengan melakukan anilisis kebutuhan sistem mengidentifikasi kebutuhan fungsional dan non fungsional sistem yang akan dibuat.

1. Kebutuhan Fungsional

   Kebutuhan fungsional adalah segala proses yang dilakukan oleh sistem. Adapun kebutuhan fungsional sistem yang akan dibuat adalah sebagai berikut:

a. Sistem dapat menampilkan halaman awal.
b. Sistem dapat menampilkan presentase grafik perbandingan metode yang sudah dilakukan.
a. Sistem dapat melakukan proses *preprocessing* terhadap dataset.
b. Sistem menginputkan teks baru.
c. Sistem dapat mengklasifikasikan Ulasan atau *Review* dengan tiga kelas yaitu positif, negatif dan netral.
d. Sistem dapat menampilkan keseluruhan data hasil *scrapping*.

2. Kebutuhan Non Funsional

   Kebutuhan perangkat keras merupakan perangkat keras yang diperlukan dalam mengembangkan sistem perbandingan metode analisis sentimen kepuasan pada aplikasi. Kebutuhan perangkat keras pada penelitian ini dapat dilihat pada **Tabel 3.22**,

Tabel 3.22 Kebutuhan Perangkat Keras

No	Perangkat Keras	Keterangan
1	Processor	Intel Core N3160 (1. 60 – 2.24 GHz, 2MB L2)
2	RAM	4 GB
3	ROM	500

3. Proses Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak merupakan perangkat lunak yang diperlukan dalam mengembangkan sistem perbandingan metode analisis sentimen kepuasan pada aplikasi. Kebutuhan perangkat lunak pada penelitian ini dapat dilihat pada **Tabel 3.23,**

Tabel 3.23 Kebutuhan Perangkat Lunak

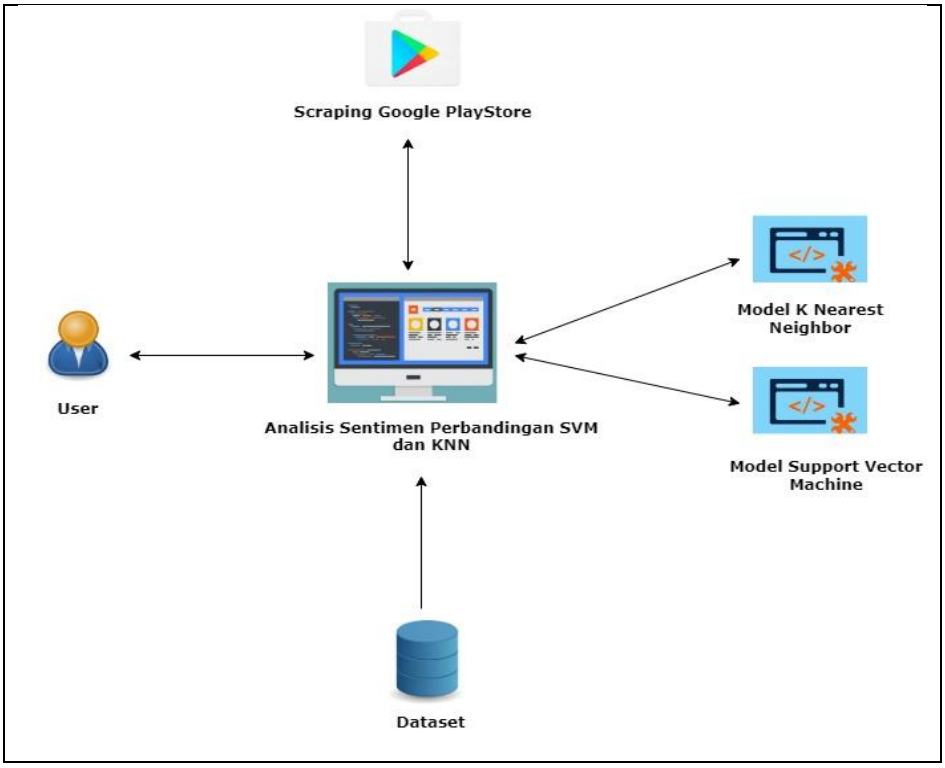
No	Perangkat Lunak	Keterangan
1	Windows 10	Intel Core N3160 (1. 60 – 2.24 GHz, 2MB L2)
2	Google Colab	4 GB
3	IDE	Jupyter Notebook, Sublime Text
4	Chrome	Web Browser
5	Bahasa Pemograman	Python 3.10
4	Chrome	Web Browser
5	Bahasa Pemograman	Python 3.10
6	Mysqli	Database

3.2.2. *Quick Plan* and *Modelling quick design*

Quick plan and modeling quick design dilakukan untuk mendapatkan gambaran umum sistem yang akan dikembangkan. Tahap *quick plan and modeling quick design* terdiri dari perancangan arsitektur sistem, perancangan proses sistem, dan perancangan antarmuka sistem. Berikut gambaran dari sistem perbandingan metode dari analisis sentimen untuk kepuasan pelanggan pada aplikasi;

3.2.3. Perancangan Arsitektur

a. Arsitektur Sistem



Gambar 3.5 Perancangan Arsitektur

Komponen perancangan arsitektur sistem adalah perbandingan sistem dengan metode *Support Vecctor Machine* dan *K-Nearest Neighbor*, semua proses akan berlangsung pada sistem. Tahap pertama dari arsitektur sistem adalah *scraping* data yang diambil dari Google Playstore. kemudian hasil dari data *scraping* tersebut dimasukkan ke dalam sistem sebagai perantara agar dapat masuk ke model *Support Vector Machine* dan *K-Nearest Neighbor*. Pada model *Support Vector Machime,* data akan melalui tahap preprocessing, mengubah ekstraksi fitur dan pembobotan dengan TF-IDF kemudian mengklasifikasikan menggunakan model yang telah dibuat sebelumnya yaitu metode *Support Vector Machine* dengan kernel linier. Begitu juga dengan model *K-Nearest Neighbor* data akan melalui tahap *Preprocessing,* pengubahan fitur ekstraksi dan pembobotan dengan TF-IDF kemudian dilakukan klasifikasi sebelumnya dengan menggunakan model yang telah dibuat yaitu metode *K-Nearest Neighbor* setelah itu sistem akan memberikan hasil klasifikasi kepada pengguna.

Komponen kedua dari perancangan sistem arsitektur adalah pengguna, yaitu pengguna Salah satu aktor yang memiliki hak penuh dalam sistem ini adalah melakukan perbandingan metode yang dilakukan dengan memberikan masukan ke sistem berupa

halaman awal kemudian halaman yang merupakan kumpulan data yang berisi daftar data yang dilakukan dalam pelatihan dan pengujian serta perbandingan antar label sebenarnya dengan hasil prediksi sistem, halaman selanjutnya adalah grafik halaman ini berisi informasi yang berkaitan dengan kinerja akurasi sistem yang terdiri dari presisi, *recall*, f1-skor dan akurasi

3.2.4. Perancangan Proses

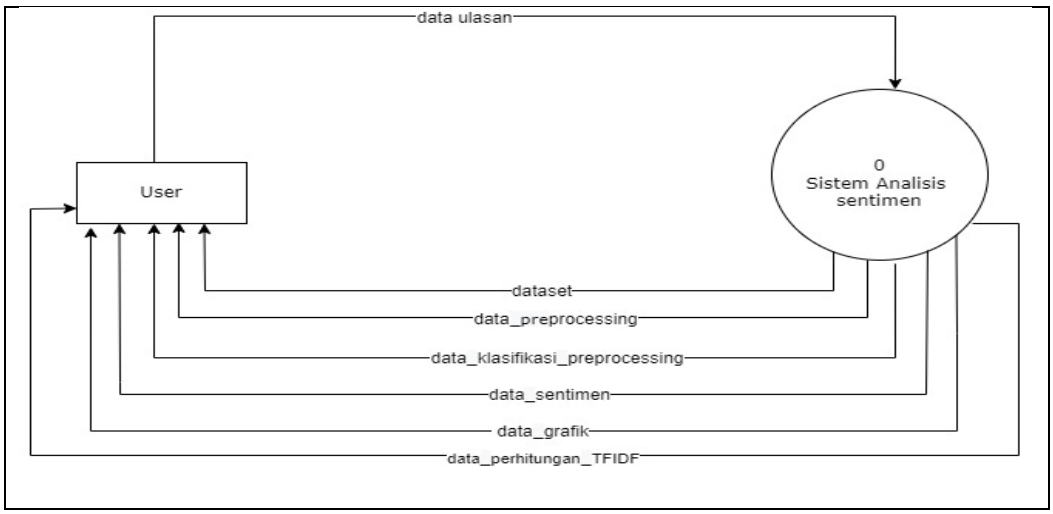
Perancangan proses adalah cara untuk menggambarkan aliran proses yang yang ada pada sistem, perancangan ini digambarkan dalam bentuk *Data Flow Diagram* (DFD) dari level 0 sampai level 3 yang dilengkapi dengan flowchart dari setiap proses.

a. *Data Flow Diagram* (DFD)

Data Flow diagram merupakan aliran data yang membantu Untuk memahami aliran suatu sistem, DFD banyak digunakan karena mampu menggambarkan aliran proses ke bentuk yang paling detail dan memiliki struktur dan jernih. DFD yang diterapkan pada sistem ini terdiri dari 2 level yaitu DFD level 0 hingga level 1, berikut rincian tiap levelnya:

1. DFD Level 0

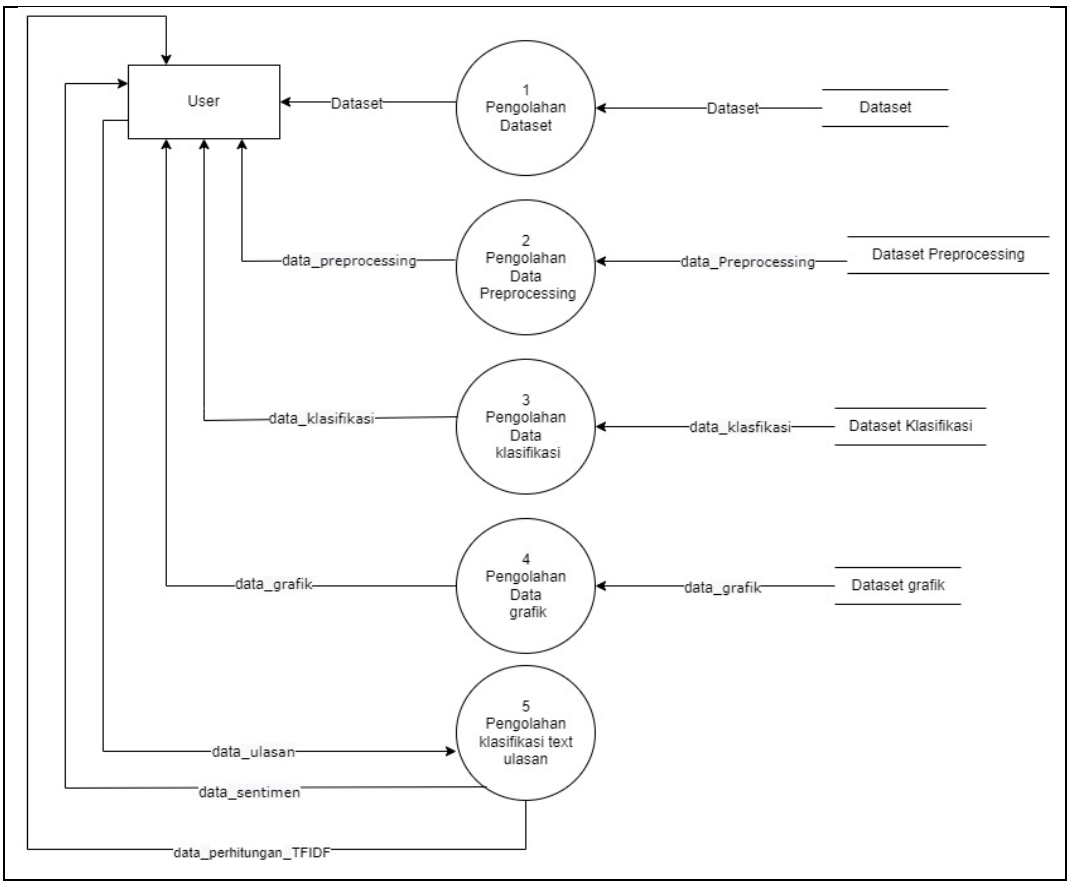
Peracangan DFD level 0 merupakan gambaran umum data keluar dan data masuk pada sistem Perbandingan metode *Support Vector Machine* dan *K-Nearest Neighbor*. Yang dimana pada DFD level 0 terdapat gambaran umum yang telah dirancang, dimana sistem memiliki proses utama yang akan dilakukan. Proses yang dimana akan menampilkan gambaran halaman awal dari sistem, menampilkan data awal yang telah di *scrapping*, dan dilanjutkan dengan menampilkan hasil *Preprocessing* terdiri dari *hasil case folding, remove punctuation, tokenization, Spelling Correction, stopword removal*, dan *stemming*. Selain itu sistem menampilkan pengujian dan grafik perbandingan yang telah dibuat, DFD level 0 dapat dilihat pada **Gambar 3.6**,



Gambar 3.6 DFD Level 0

2. DFD Level 1

Peracangan DFD level 1 merupakan gambaran data hasil *Text Preprocessing* dan pengujian, dan menampilkan grafik perbandingan metode *Support Vector Machine* dan *K-Nearest Neighbor*. DFD level 1 dapat dilihat pada **Gambar 3.16**,



Gambar 3.7 DFD Level 1

DFD Level 1 Proses Perbandingan metode ulasan pengguna aplikasi Tiktok

DFD level 1 proses merupakan dekomposisi dari DFD level 0 yang menjelaskan alur proses perbandingan metode ulasan Tiktok secara lebih spesifik.

a. DFD Level 1 Proses Pengolahan Dataset/ Data hasil *scraping*

   DFD level 1 proses 1 merupakan alur menampilkan dataset hasil scrapping sistem yang telah dibuat. Dimana dataset/ data hasil *scraping* yang ditampilkan dimasukkan dalam file CSV dan diberi nama dataset.

b. DFD Level 1 Proses Pengolahan data *Preprocessing*

   DFD level 1 Proses *Preprocessing* merupakan alur untuk menampilakan dataset awal hasil *Scraping* dan menampilkan dataset hasil dari *Preprocessing* yang telah dilakukan. *User* bisa memilih menu dataset awal hasil *Scraping* dan juga hasil preprocessing kemudian akan tampil setiap menu data hasil *case folding, remove punctuation, tokenization,spelling correction, stopword removal, stemming.* . Dimana dataset yang ditampilkan dimasukkan dalam file CSV dan diberi nama file sesuai preprocessing diatas tersebut.

c. DFD Level 1 Proses Pengolahan data klasifikasi

   DFD level 1 Proses data klasifikasi merupakan alur untuk menampilakan halaman klasifikasi. Dimana *user* dapat melihat hasil klasifikasi *Text Preprocessing* dan hasil perhitungan serta kelas sentimen.

d. DFD Level 1 Proses Pengolahan data Grafik

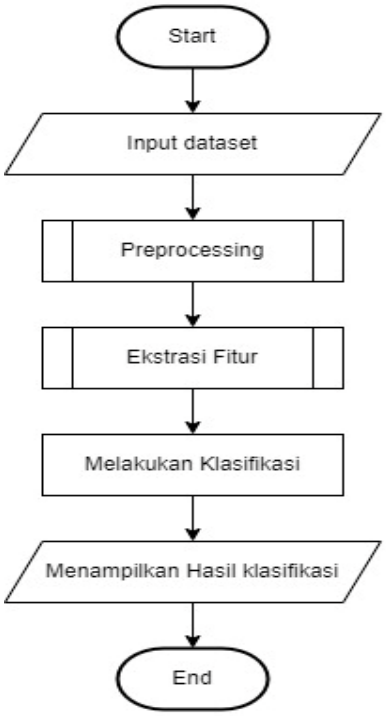
   DFD level 1 Proses data klasifikasi merupakan alur untuk menampilkan halaman data hasil *accuracy, precision, f1 score dan recall*. Pada halaman ini juga menampilkan grafik akurasi yang telah di dapatkan. Dimana dataset yang ditampilkan dimasukkan dalam file CSV.

e. DFD Level 1 Proses Pengolahan klasifikasi text ulasan

   DFD level 1 Proses data klasifikasi merupakan alur untuk menampilkan halaman yang dimana *User* dapat menginput kata atau ulasan aplikasi tiktok pada halaman ini. Kemudian akan memproses kata tersebut dan menghasilkan proses klasifikasi *Text Preprocessing*, perhitungan TF-IDF dan kelas sentiment.

   3. Flowchart Klasifikasi File

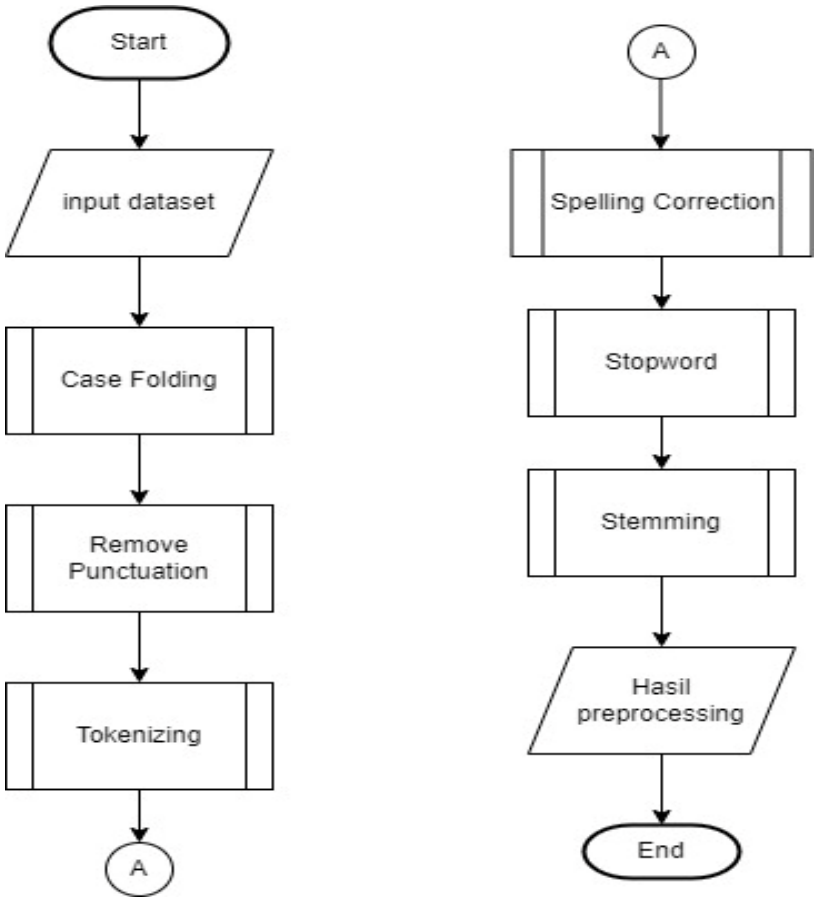
Adapun Peracangan Flowchart Klasifikasi Ulasan yang merupakan gambaran untuk klasifikasi ulasan pada sistem perbandingan metode SVM dan KNN. Pada *Flowchart* Klasifikasi Ulasan dimulai dengan input dataset kemudian tahap preprocessing, jika sudah masuk proses presprocessing selanjutnya proses ekstrasi fitur, melakukan klasifikaso lalu akan menampilkan hasil klasfikasi dan selesai. Perancangan flowchart klasifikasi ulasan dapat dilihat pada **Gambar 3.8,**



Gambar 3.8 *Flowchart* klasifikasi ulasan

4. Flowchart Preprocessing

Adapun Flowchart proses Preprocessing yang dilakukan yaitu *case folding, remove punctuation, remove number, stopword removal, spelling correction dan stemming*. Berikut Alur *preprocessing* teks yang dilakukan dapat dilihat pada *flowchart* **Gambar 3.9,**



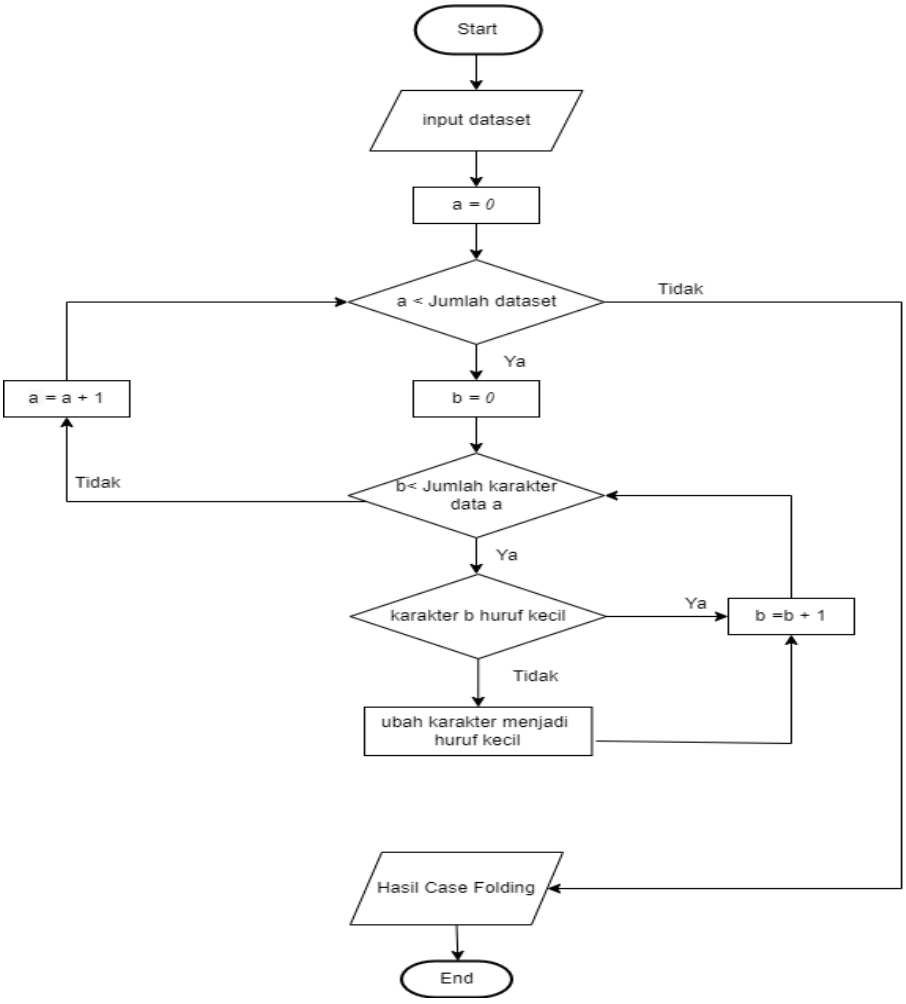
Gambar 3.9 *Flowchart Preprocessing*

Pada Preprocessing ada beberapa tahapan yang akan dijelaskan dalam alur Flowchart yaitu *case folding, remove punctuation, remove number, stopword removal, spelling correction dan stemming.* Alur Flowchart dapat dilihat sebagai berikut ;

a. Flowchart *Case Folding*

Pada tahapan alur Flowchart *Case Folding* dimulai dengan menginput dataset atau data hasil scraping awal. Kemudian proses akan dimulai dengan a=0, jika jumlah dataset lebih kecil dari a maka akan langsung ke proses hasil *case folding.* jika nilai jumlah dataset lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya

lebih dari b. proses akan terus berulang samai jumlah karakternya lebih besar dari b, jika karakter datanya sudah benar maka akan mengecek apakah karakter b itu huruf kecil. Jika karakter huruf b ada huruf besar maka akan melakukan proses ubah karakter menjadi huruf kecil. Jika semua sudah makan proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar makan akan ke hasil akhir *case folding*. *Flowchart* dari proses *case folding* dapat dilihat pada **Gambar 3.10,**

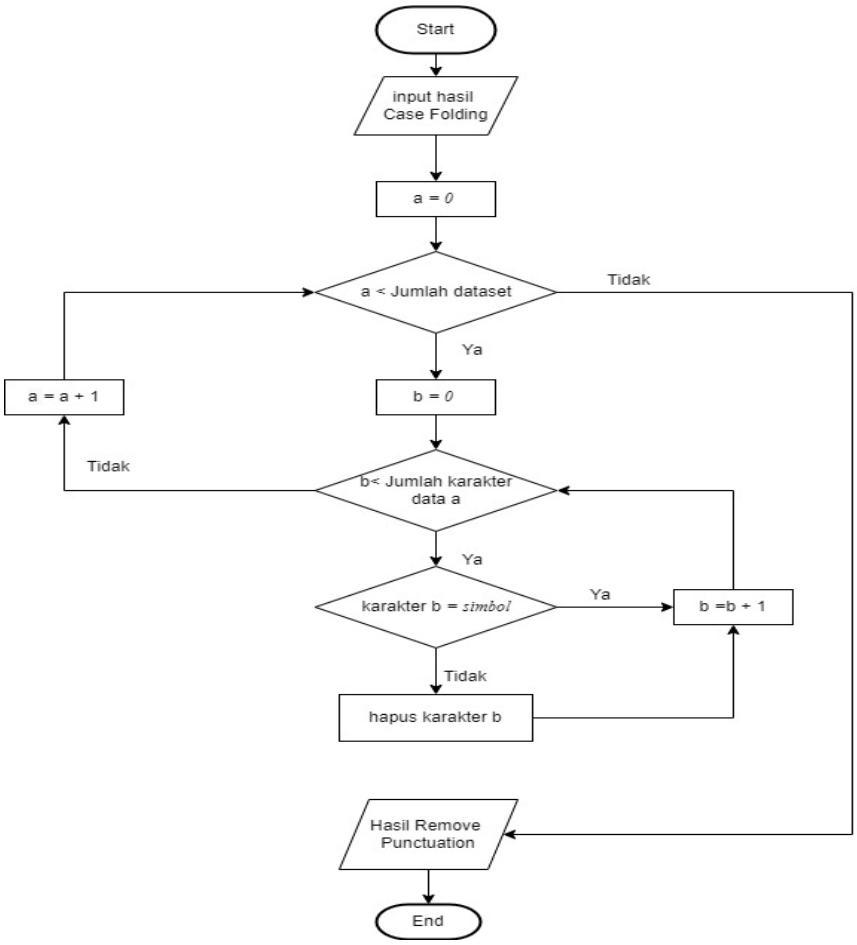


Gambar 3.10 *Flowchart Case Folding*

b. Flowchart *Remove Punctuation*

Pada tahapan alur Flowchart *Remove Puntuation* dimulai dengan menginput data hasil case folding. Kemudian proses akan dimulai dengan a=0, jika jumlah dataset lebih kecil dari a maka akan langsung ke proses hasil *Remove Punctuation.* jika nilai jumlah dataset

lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya lebih dari b. proses akan terus berulang samai jumlah karakternya lebih besar dari b, jika karakter datanya sudah benar maka akan mengecek apakah karakter b itu symbol, Jika karakter huruf b ada simbolnya maka akan melakukan proses ubah simbol. Jika semua sudah makan proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar makan akan ke hasil akhir *Remove Punctuation*. *Flowchart* dari proses *Remove Punctuation* dapat dilihat pada **Gambar 3.11**,

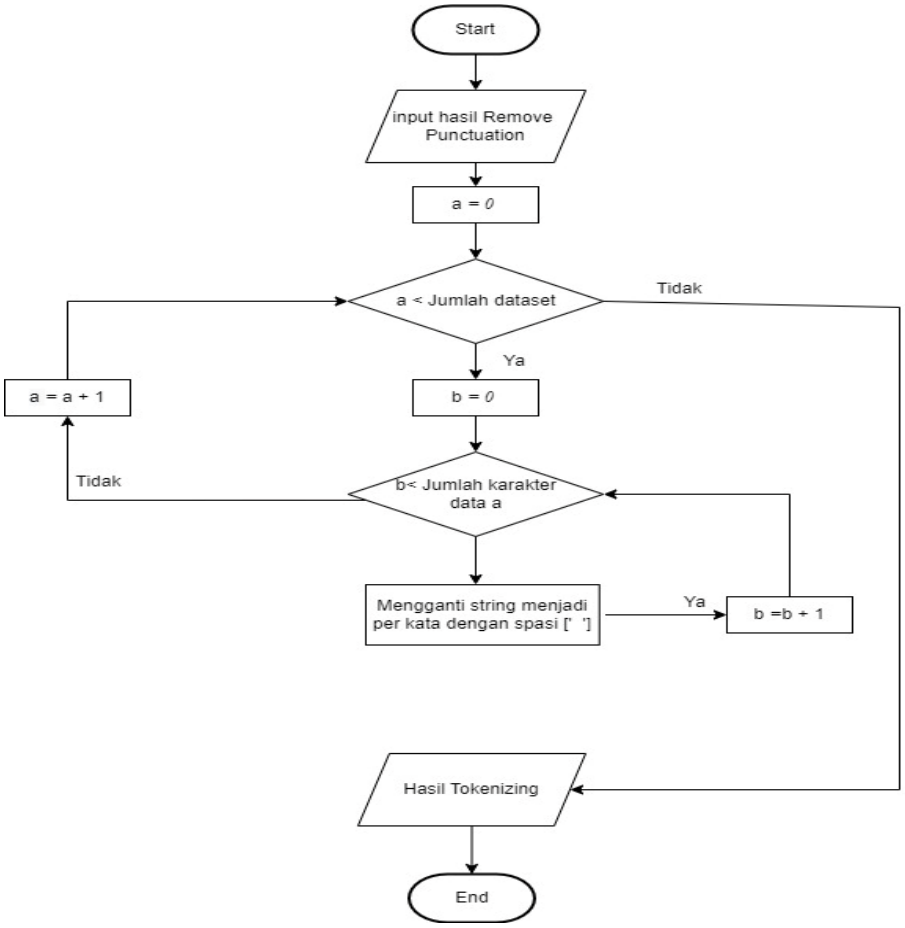


Gambar 3.11 *Flowchart Remove Punctuation*

c. Flowchart *Tokenizing*

Pada tahapan alur Flowchart *Remove Puntuation* dimulai dengan menginput data hasil Remove punctuation. Kemudian proses akan dimulai dengan a=0, jika jumlah dataset lebih kecil dari a maka akan langsung ke proses hasil *Remove Punctuation.* jika nilai jumlah dataset lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya lebih dari b. proses akan terus berulang samai jumlah karakternya lebih besar dari b, jika karakter datanya sudah benar maka akan mengecek apakah karakter b itu

symbol, Jika karakter huruf b ada simbolnya maka akan melakukan proses ubah simbol. Jika semua sudah makan proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar makan akan ke hasil akhir *Remove Punctuation*. *Flowchart* dari *Tokenizing* bisa dilihat pada **Gambar 3.12**,

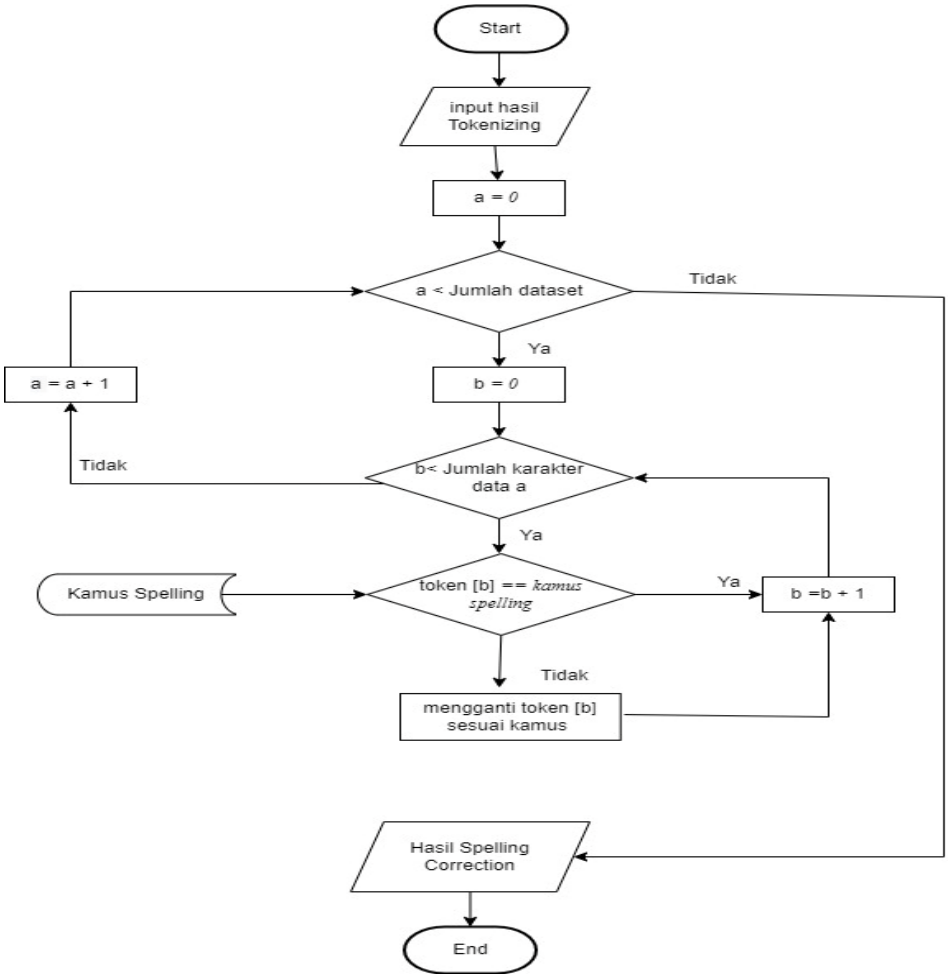


Gambar 3.12 *Flowchart Tokenizing*

d. Flowchart *Spelling Correction*

Pada tahapan alur Flowchart *Spelling correction* dimulai dengan menginput data hasil Tokenizing. Kemudian proses akan dimulai dengan a=0, jika jumlah dataset lebih kecil dari a maka akan langsung ke proses hasil *Spelling Correction.*. jika nilai jumlah dataset lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke

proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya lebih dari b. proses akan terus berulang sampai jumlah karakternya lebih besar dari b, jika karakter datanya sudah benar maka akan mengecek apakah token b sama dengan kamus spelling, Jika karakter huruf b benar akan berlanjut mengganti sesuai kamus. Jika semua sudah maka proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar maka akan ke hasil akhir *Spelling Correction*. Flowchart dari proses *spelling correction* dapat dilihat pada **Gambar 3.13,**

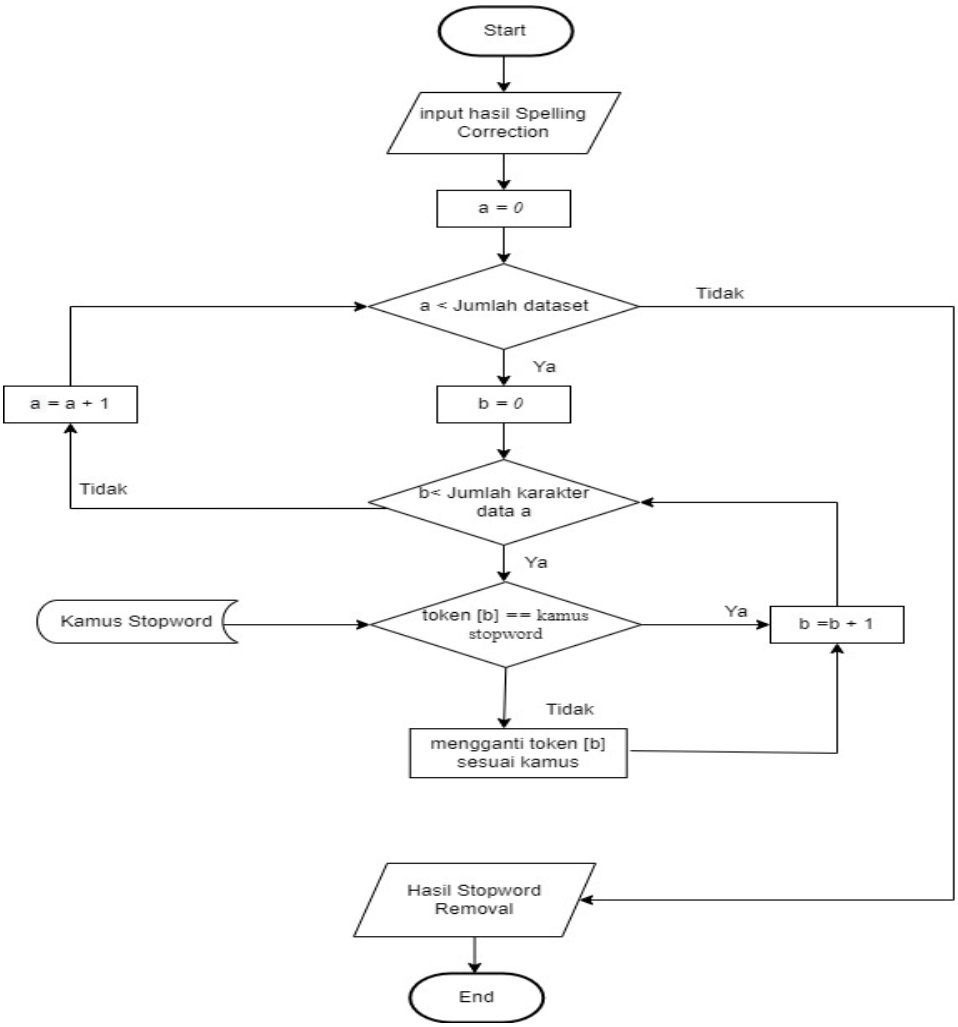


Gambar 3.13 *Flowchart Spelling Correction*

e. Flowchart *Stopword*

Pada tahapan alur Flowchart *Spelling correction* dimulai dengan menginput data hasil *Spelling Correction*. Kemudian proses akan dimulai dengan a=0, jika jumlah

dataset lebih kecil dari a maka akan langsung ke proses hasil *Stopword Removal*.. jika nilai jumlah dataset lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya lebih dari b. proses akan terus berulang sampai jumlah karakternya lebih besar dari b, jika karakter datanya sudah benar maka akan mengecek apakah token b sama dengan kamus *Stopword Renoval*, Jika karakter huruf b benar akan berlanjut mengganti sesuai kamus. Jika semua sudah maka proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar maka akan ke hasil akhir *Stopword*. *Flowchart Stopword Removal* telah disajikan dan dapat dilihat pada **Gambar 3.14**,

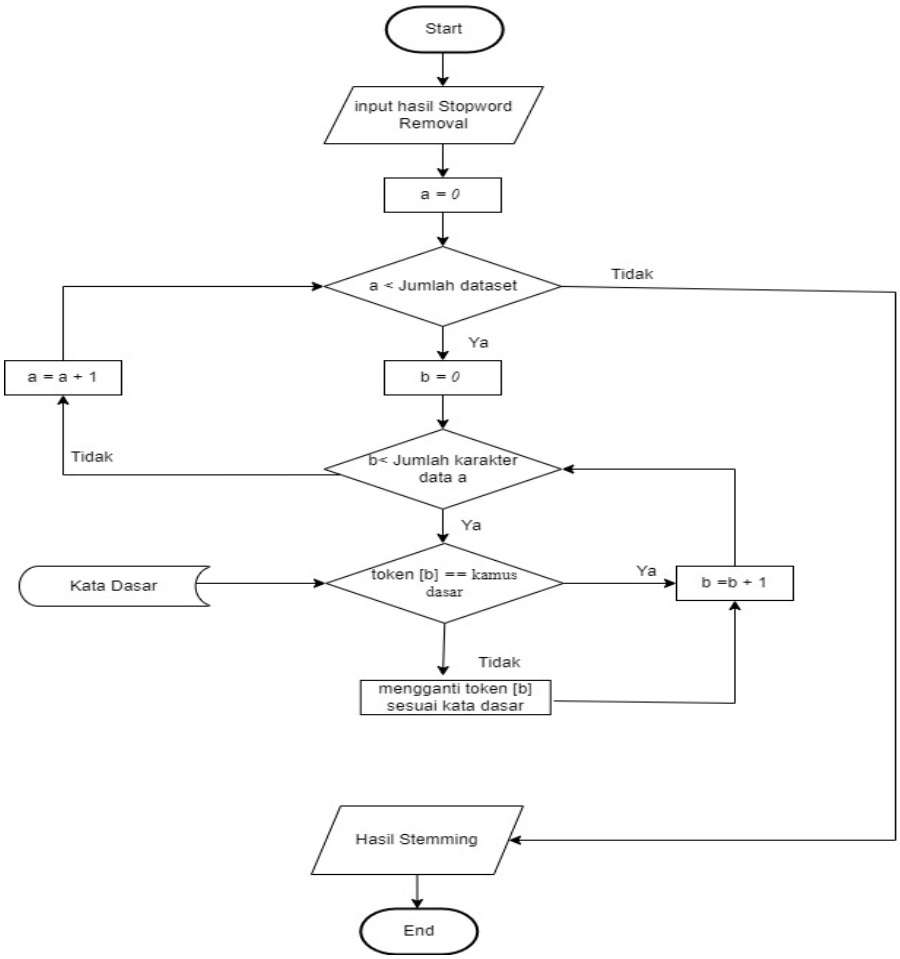


Gambar 3.14. *Flowchart Stopword Removal*

f. Flowchart *Stemming*

Pada tahapan alur Flowchart *Stopword Removal* dimulai dengan menginput data hasil *Stopword Removal*. Kemudian proses akan dimulai dengan a=0, jika jumlah dataset lebih kecil dari a maka akan langsung ke proses hasil *Stemming.* jika nilai jumlah dataset lebih besar dari a makan akan berlanjut membandingkan data a dengan nilai yang sudah di deklarasikan yaitu b=0. Jika memang benar jumlah dataset lebih kecil dari nilai a, maka akan membandingkan jumlah karakter data a dengan nilai yang sudah di deklarasikan dimulai dari 0. Kemudian jika karakter data a lebih kecil akan kembali ke proses membandingkan dataset dengan nilai a yang semula 0 menjadi 1 karena prosesnya a+1 sampai berulang kembali bisa dikatakan sampai jumlah karakter datanya lebih dari b. proses akan terus berulang sampai jumlah karakternya lebih besar dari b,

jika karakter datanya sudah benar maka akan mengecek apakah token b sama dengan kata dasar, Jika karakter huruf b benar akan berlanjut mengganti sesuai kata dasar. Jika semua sudah maka proses akan terus berlanjut ke proses b=b+1. Pada semua proses pengecekan sudah benar maka akan ke hasil akhir *Stemming.* Flowchart dari proses stemming dapat dilihat pada **Gambar 3.15**,

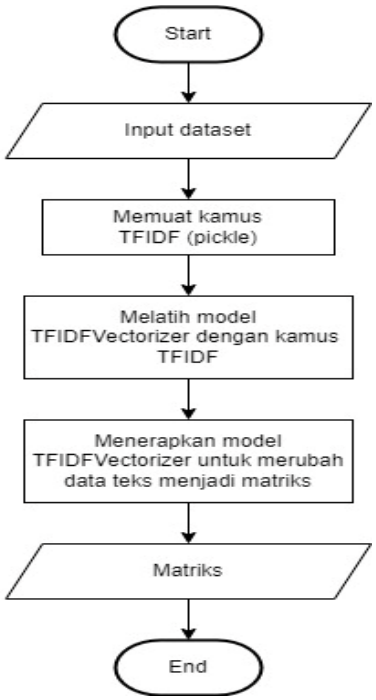


Gambar 3.15 *Flowchart Stemming*

5. Flowchart Ekstrasi Fitur

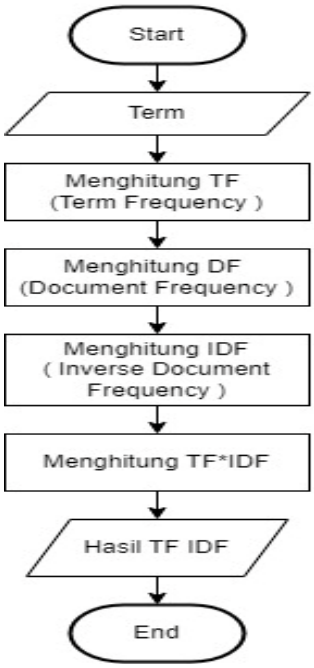
Adapun Peracangan Flowchart Ekstrasi Fitur yang merupakan gambaran untuk Ektrasi Fitur pada sistem perbandingan metode SVM dan KNN. Pada *Flowchart* Perancangan *flowchart* Ektrasi Fitur dimulai dengan input dataset, kemudian memuat kamus TF IDF (Pickle), proses selanjutnya yaitu melatih model TF IDF Vectorizer dengan kamus TF IDF. Jika sudah dilakukan makan proses selanjutnya menerapkan model TF IDF Vectorizer untuk merubah data teks menjadi matriks Flowchart ekstrasi Fitur dapat

dilihat pada **Gambar 3.16,**



Gambar 3.16 *Flowchart* Klasifikasi Fitur

Adapun *Flowchart* dari proses pembobotan TF-IDF proses dimulai dengan memasukkan kata atau term kemduian menghitung TF *(Term Frequency)*, jika sudah masuk menghitung DF *(Document Frequency)*, proses selanjutnya menghitung IDF *(Inverse Document Frequency)*, selanjutnya menghitung TF-IDF kemudian akan mendapatkan hasil akhir TF IDF lalu proses berakhir. Flowchart dari pembobotan TF IDF dapat dilihat pada **Gambar 3.17**,

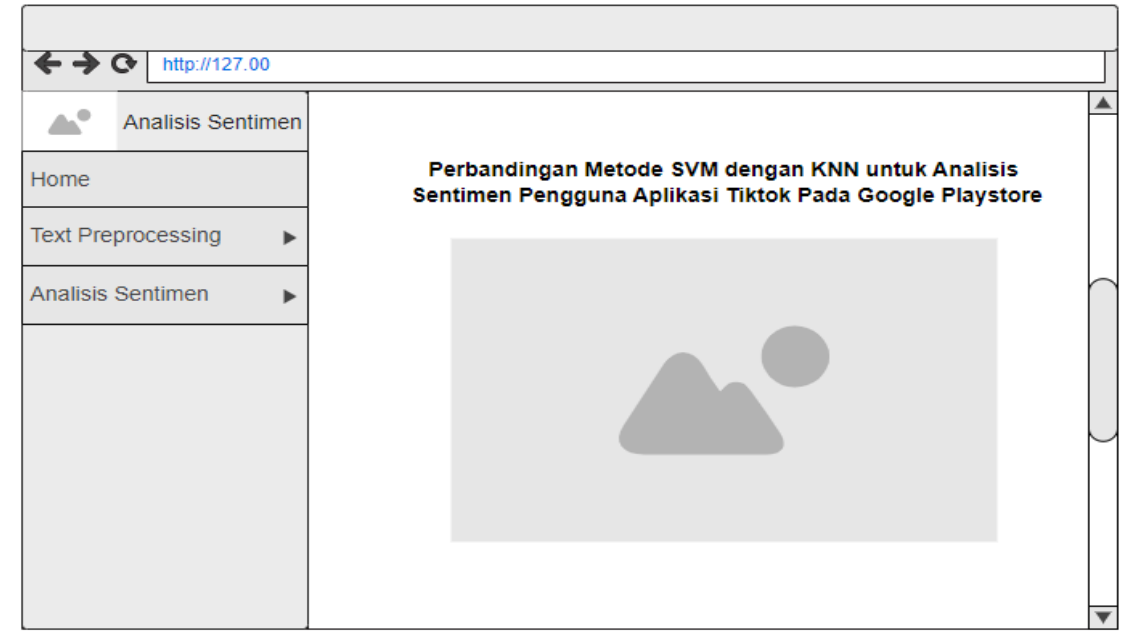


Gambar 3.17 *Flowchart* Pembobotan TF-IDF

1. Perancangan Antarmuka

   a. Desain Home

      Pada Perancangan Antarmuka desain awal sistem dengan beberapa menu pilihan dapat dilihat pada **Gambar 3.18**,

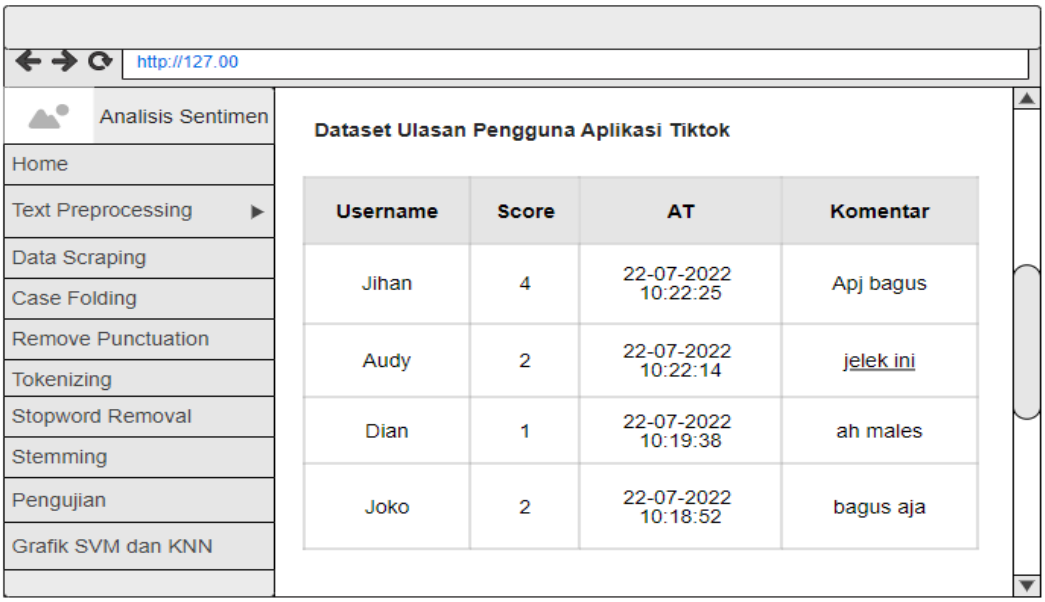


Gambar 3.18 Desain *Home*

b. Desain Text Preprocessing

1. Halaman Data Scraping

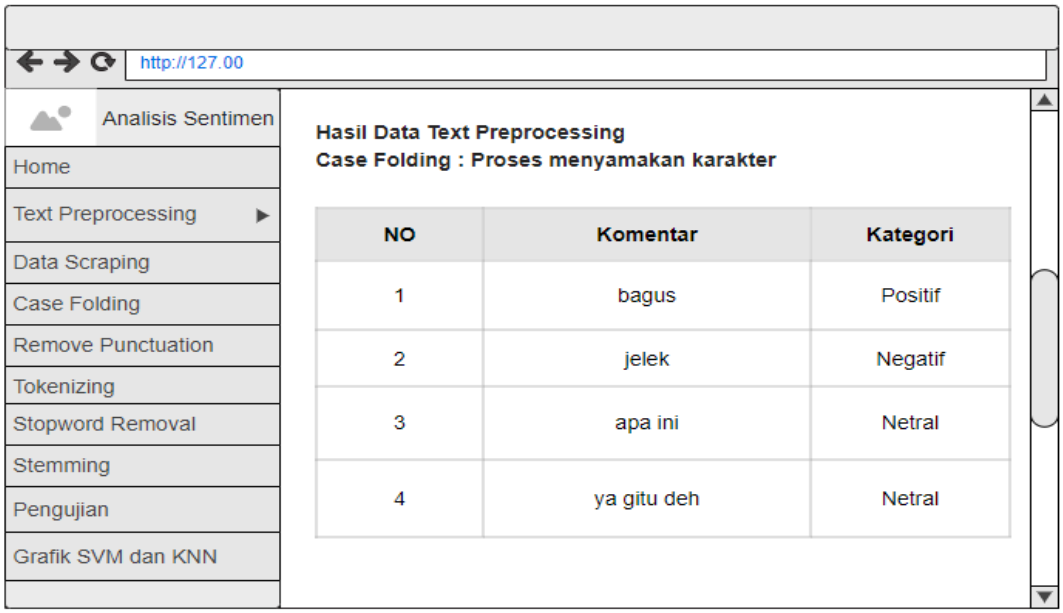
Menampilkan Dataset Awal hasil scraping dapat dilihat pada **Gambar 3.19**,



Gambar 3.19 Desain Halaman Dataset

2. Desain *Case Folding*

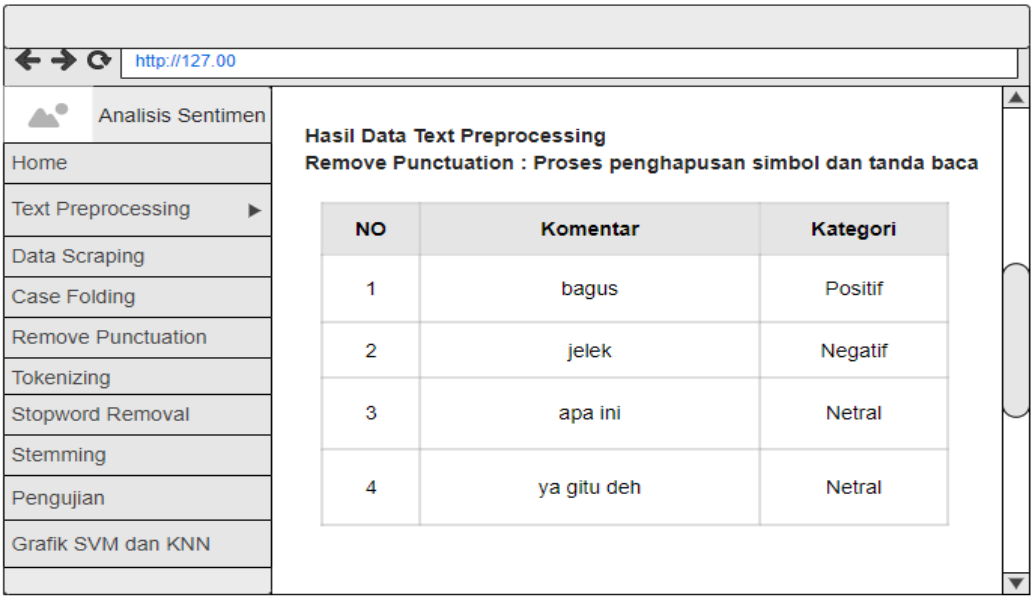
Menampilkan Menu Hasil *Case Folding* dapat dilihat pada **Gambar 3.20**,



Gambar 3.20 Desain *Case Folding*

3. Desain *Remove Punctuation*

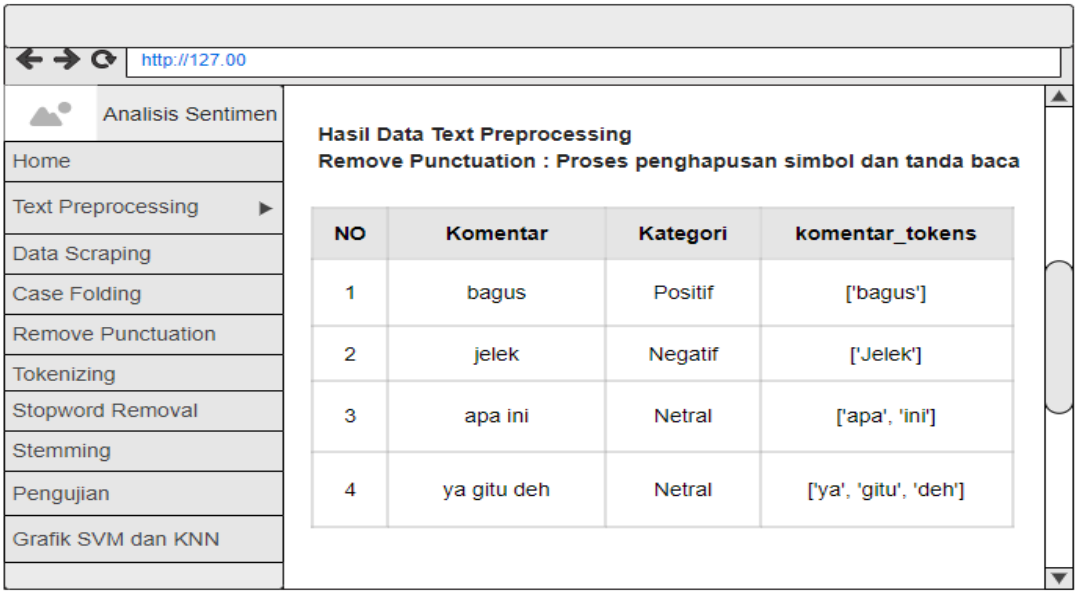
Menampilkan Menu Hasil *Remove Punctuation* dapat dilihat pada **Gambar 3.21**,



Gambar 3.21 Desain *Remove Punctuation*

4. Desain *Tokenizing*

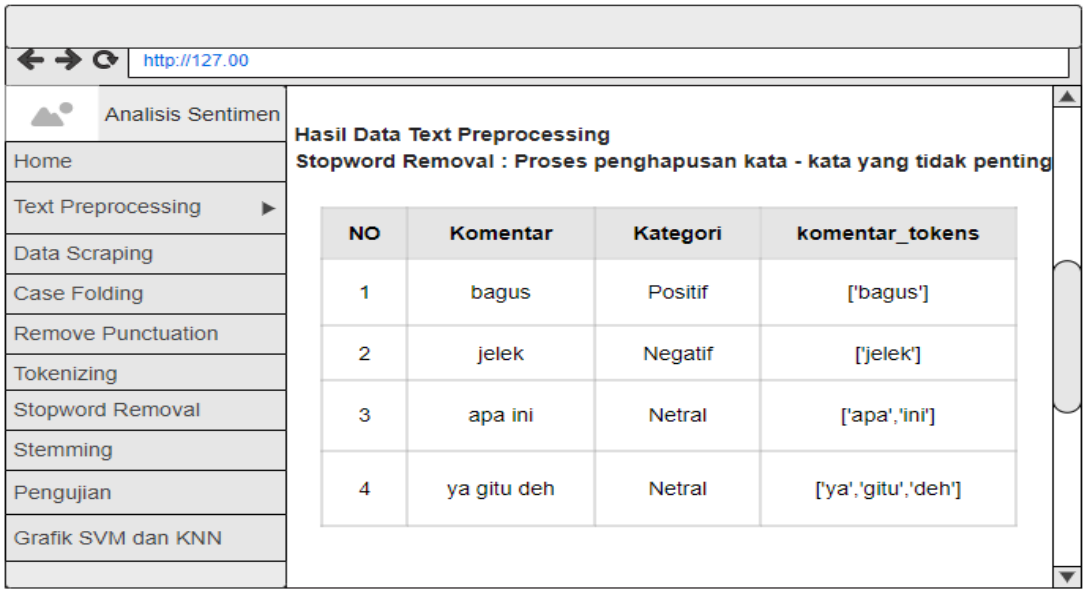
Menampilkan Menu Hasil *Tokenizing* dapat dilihat pada **Gambar 3.22**,



Gambar 3.22 Desain *Tokenizing*

5. Desain *Stopword Removal*

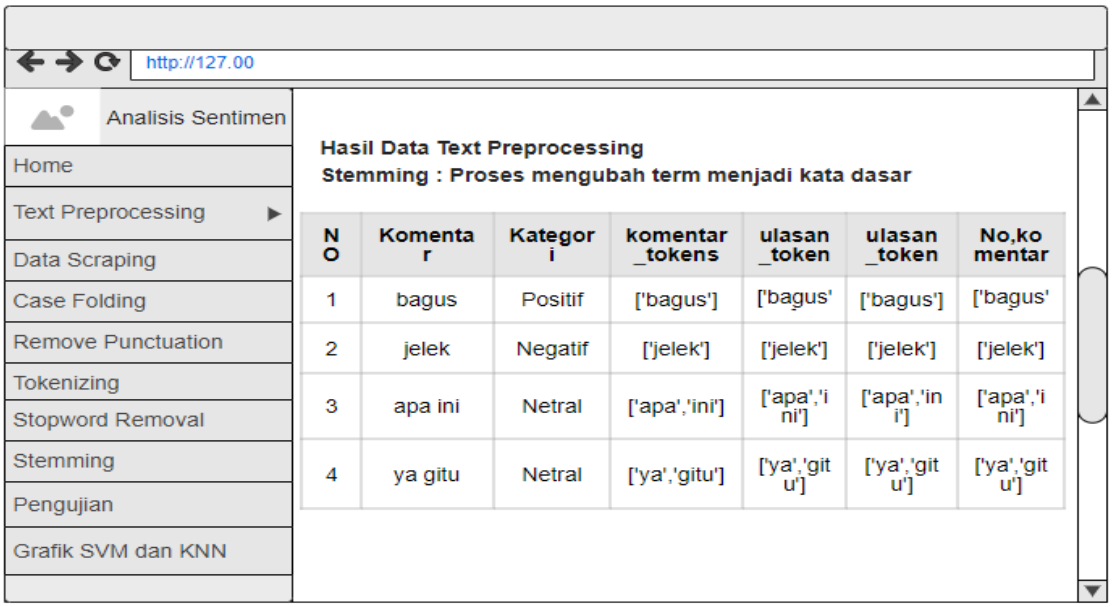
Menampilkan Menu Hasil *Stopword Removal* dapat dilihat pada **Gambar 3.23,**



Gambar 3.23 Desain *Stopword Removal*

6. Desain *Stemming*

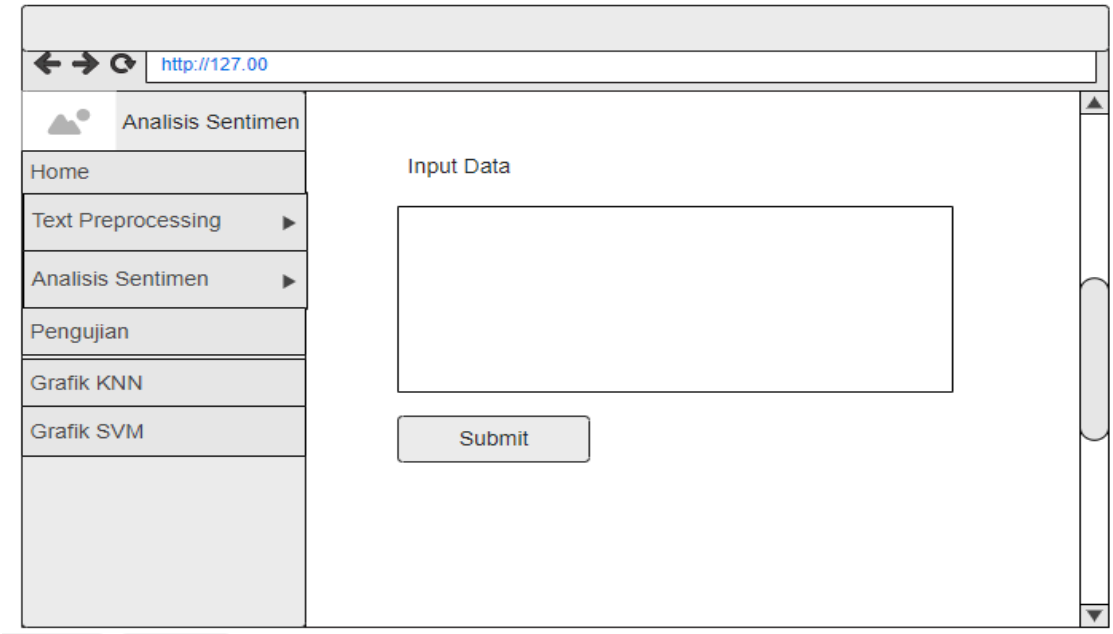
Menampilkan Menu Hasil *Stemming* dapat dilihat pada **Gambar 3.24**,



Gambar 3.24 Desain *Stemming*

7. Desain Analisis Sentiment Pengujian

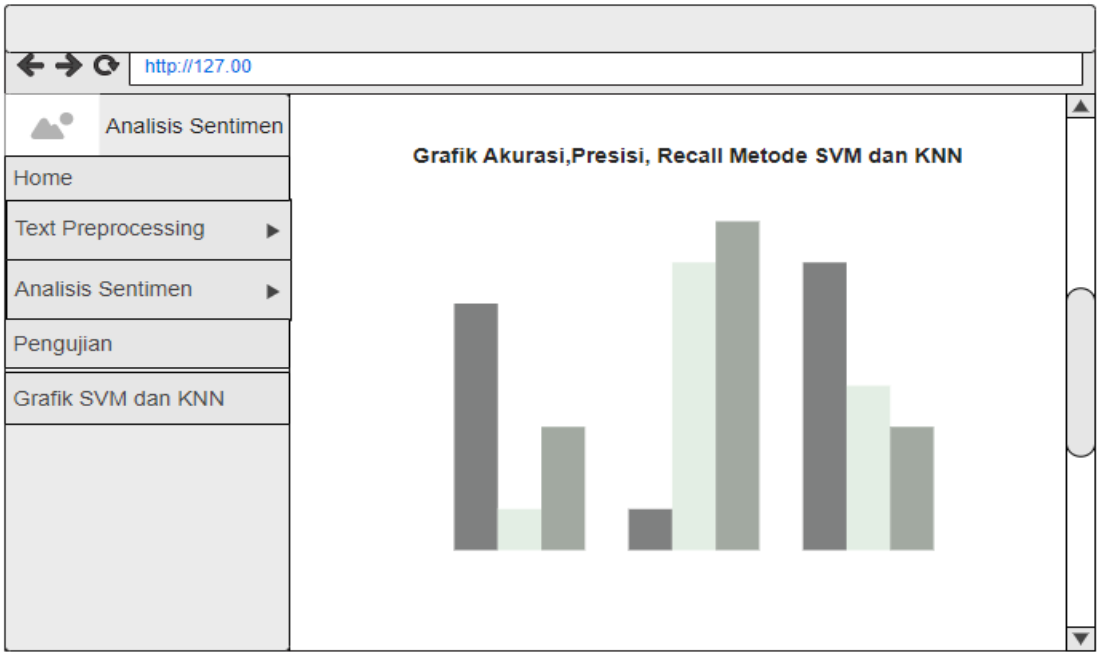
Menampilkan Menu pengujian dapat dilihat pada **Gambar 3.25**,



Gambar 3.25 Desain Pengujian

8. Halaman Analisis Sentimen Grafik Perbandingan Metode

Menampilkan Menu grafik dapat dilihat pada **Gambar 3.26,**



Gambar 3.26 Desain Grafik Perbandingan Metode

3.2.5. Construction of Prototype

Tahap *construction of prototype* dilakukan dengan mengimplementasikan rancangan desain untuk membuat sistem ke dalam program dengan menggunakan bahasa pemrograman.

3.2.6. Deployment delivery and feedback

Deployment Delivery & Feedback dilakukan dengan menerapkan pengujian sistem menggunakan metode blackbox testing untuk menguji fungsional sistem pada perbandingan metode untuk analisis kepuasan pelanggan dalam penggunaan aplikasi. Pengujian dapat dilihat pada **Tabel 3.24**,

Tabel 3.24 Perancangan Pengujian Sistem Perbandingan Metode

No.	Skenario Pengujian	Hasil yang diharapkan	Hasil	Kesimpulan
1	user dapat melihat Halaman Awal tampilan yang telah dibuat	Sistem dapat diakses.		
2	User dapat melihat seluruh data yang telah dilakukan proses Text Preprocessing	Sistem dapat menampilkan halaman data hasil text preprocesing		
3	sistem menampilkan kategori kelas dari teks yang telah dimasukan. Beserta proses preprocessing yang terjadi.	Sistem dapat menampilkan kolom input klasifikasi serta hasil klasifikasi		
4	Sistem menampilkan table hasil akurasi dan juga menampilkan grafik yang telah dibuat.	Sistem dapat menampilkan hasil grafik perbandingan metode SVM dan KNN		

3.2.Rancangan Pengujian

Bagian desain pengujian ini mencari informasi tingkat keberhasilan klasifikasi yang telah dilakukan. Untuk mengetahui tingkat kebenaran proses klasifikasi dibuat tabel *Confusion Matrix* untuk pengujian, serta hasil pengujian akurasi, presisi, dan recall. Untuk pemodelan sentimen, matriks kebingungan digunakan. Rancangan yang akan digunakan untuk pengujian *confusion matrix* dapat dilihat pada Tabel 3.25 dan 3.26,

Tabel 3.25 Confusion Matrix Model klasifikasi sentiment metode SVM

		Prediksi			**Jumlah**
		Negatif	Netral	Positif	
Aktual	Negatif	TNeg	FNet2	FP	N
	Netral	FNeg2	TNet	FP2	L
	Positif	FNeg	FNet	TP	P
Jumlah		N’	L’	P’	

Tabel 3.26 Confusion Matrix Model klasifikasi sentiment metode KNN

		Prediksi			**Jumlah**
		Negatif	Netral	Positif	
Aktual	Negatif	TNeg	FNet2	FP	N
	Netral	FNeg2	TNet	FP2	L
	Positif	FNeg	FNet	TP	P
Jumlah		N’	L’	P’	

BAB IV
HASIL PENGUJIAN DAN PEMBAHASAN

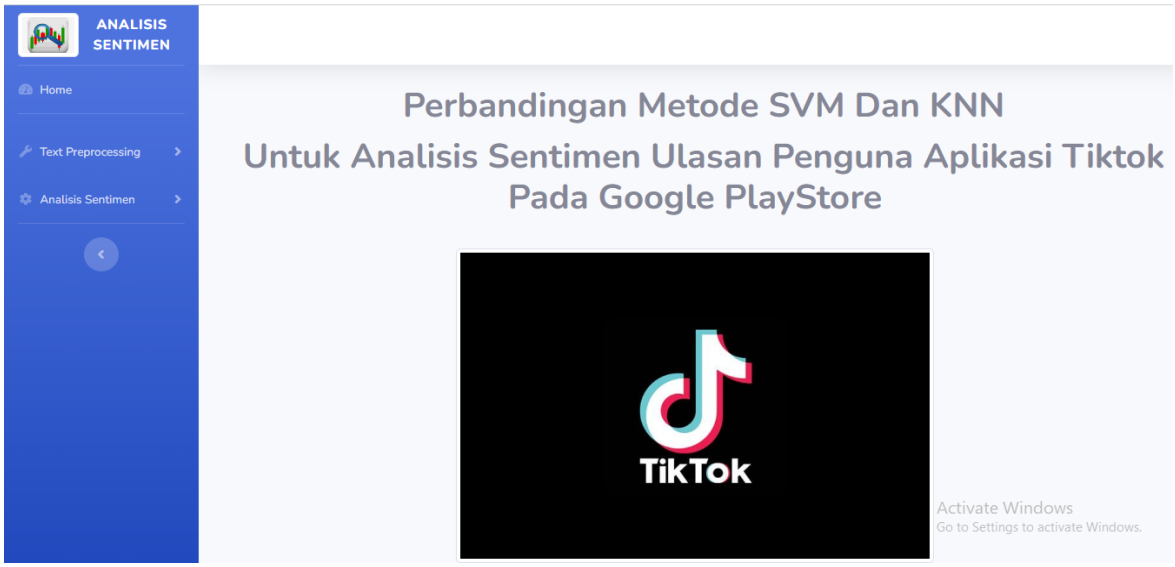
Bab ini berisi implementasi perancangan yaitu aplikasi implementasi yang terdiri dari halaman text preprocessing, halaman grafis, dan halaman pengujian. Dibahas pula implementasi text *preprocessing,* seperti *case folding, remove punctuation, tokeninzing, spelling correction, stopword removal, tokenizing, dan stemming*. Bab ini juga berisi implementasi dari perancangan yang telah dibuat sebelumnya yaitu aplikasi implementasi yang terdiri dari halaman utama, halaman dataset dan data preprocessing, halaman klasifikasi preprocessing, halaman grafik, dan Testing. Setelah itu adalah pengujian untuk menguji, yang menggunakan validasi silang k-fold dan confusion matrix.

4.1 Implementasi Aplikasi

Implementasi aplikasi berisi tampilan dari sistem yang dibuat ketika sedang dijalankan. Pada sub bab berikut merupakan gambar tampilan dari sistem dan *source code.*

4.1.1 Halaman Utama

Halaman ini berisi tampilan awal ketika sistem dijalankan. Halaman ini menampilkan gambar yang merepresentasikan penelitian yang dilakukan. Halaman ini merupakan tampilan yang paling sederhana jika dibandingkan dengan halaman-halaman lain. Tampilan dari halaman utama dapat dilihat pada gambar 4.1,



Gambar 4.1 Halaman Utama

Source code untuk menampilkan halaman utama dapat dilihat pada **modul 4.1 dan 4.2,**

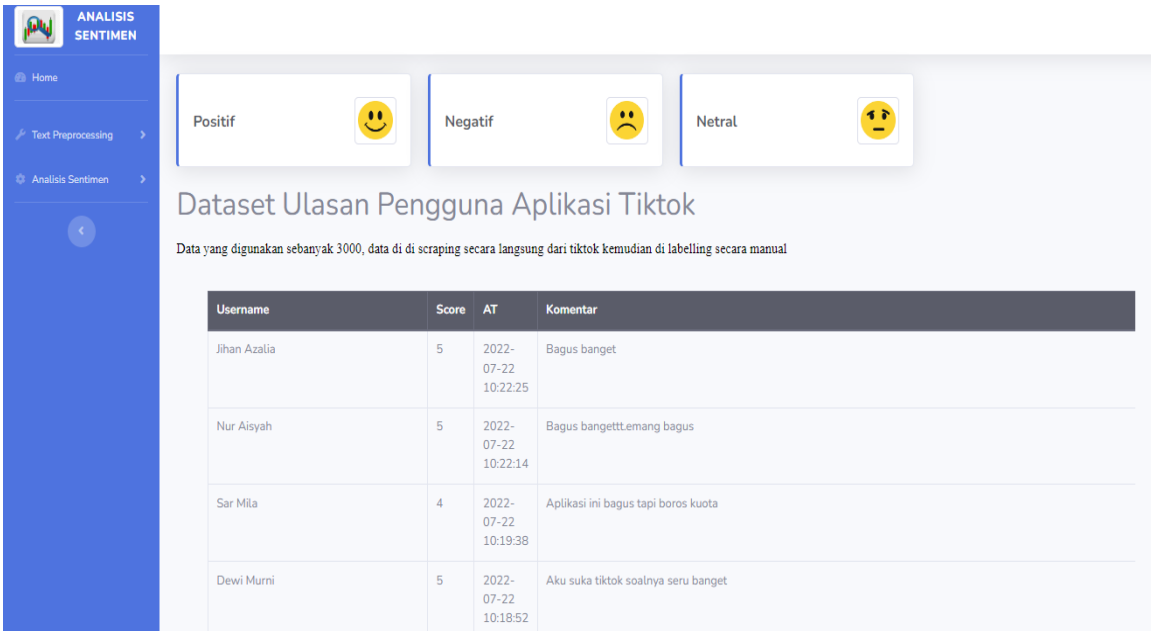
Kode Program 1 : Proses menampilkan halaman utama

```
@app.route("/homee")
        def homee():
                        return render_template('homee.html')
```

Modul Program 4.1 Source Code Menampilkan Halaman Utama

4.1.2 Halaman Halaman Dataset

Halaman *Text Preprocessing* dataset berisi tabel yang menampilkan dataset yang digunakan untuk training dalam melakukan penelitian. Tabel tersebut memiliki kolom yang terdiri dari *username, score, AT, dan* komentar. Pada bagian atas tabel, terdapat 3 sticker yang merepresentasikan jumlah data pada tiap kelas sentimen. Tampilan dari halaman dataset dapat dilihat pada **Gambar 4.2** sebagai berikut,



Gambar 4.2 Tampilan Halaman Dataset

Source code untuk menampilkan halaman dataset dapat dilihat pada **modul 4.2,**

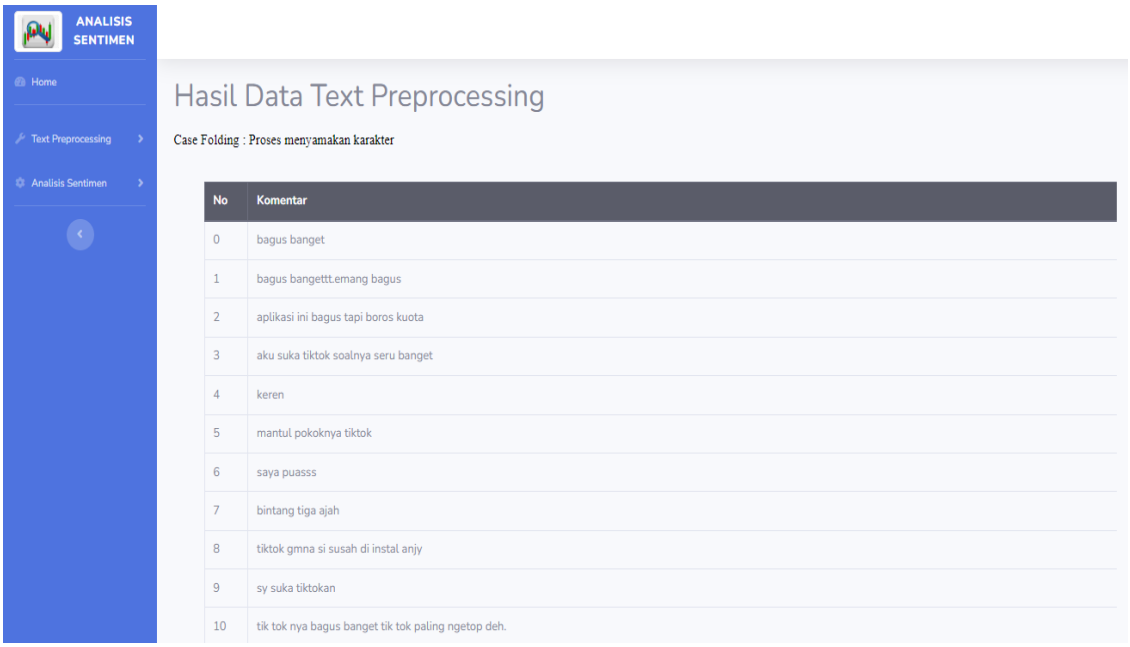
Kode Program 1 : Proses menampilkan halaman Dataset

```
@app.route("/dataset", methods=['GET', 'POST']) #fungsi untuk
def dataset():
    with open('DataTiktokLabel.csv', encoding='latin-1') as csv_file:
        data = csv.reader(csv_file, delimiter=',')
        first_line = True
        dataset = []
        for row in data:
            if not first_line:
                dataset.append({
 #fungsi membuka dataset
                    "Username": row[0],
                    "Score": row[1],
                    "AT": row[2],
                    "KATEGORI": row[3],
                    "komentar": row[4],
                }) else:
                first_line = False
            return render_template('dataset.html', menu='dataset',
         submenu='data', dataset=dataset)
```

Modul Program 4.2 *Source Code* Menampilkan Halaman Dataset

4.1.3 Halaman *Text Preprocessing Case Folding*

Halaman *Text Preprocessing case folding* berisi tabel yang menampilkan hasil *Text preprocessing case folding*. Tabel tersebut memiliki kolom yang terdiri dari nomor, komentar, kategori. Tampilan dari halaman *Text preprocessing case folding* dapat dilihat pada **Gambar 4.3,**



Gambar 4.3 Menampilkan Halaman Text *Preprocessing Case Folding*

Source code untuk menampilkan halaman *Text Preprocessing Case Folding* dapat dilihat pada **modul 4.4,**

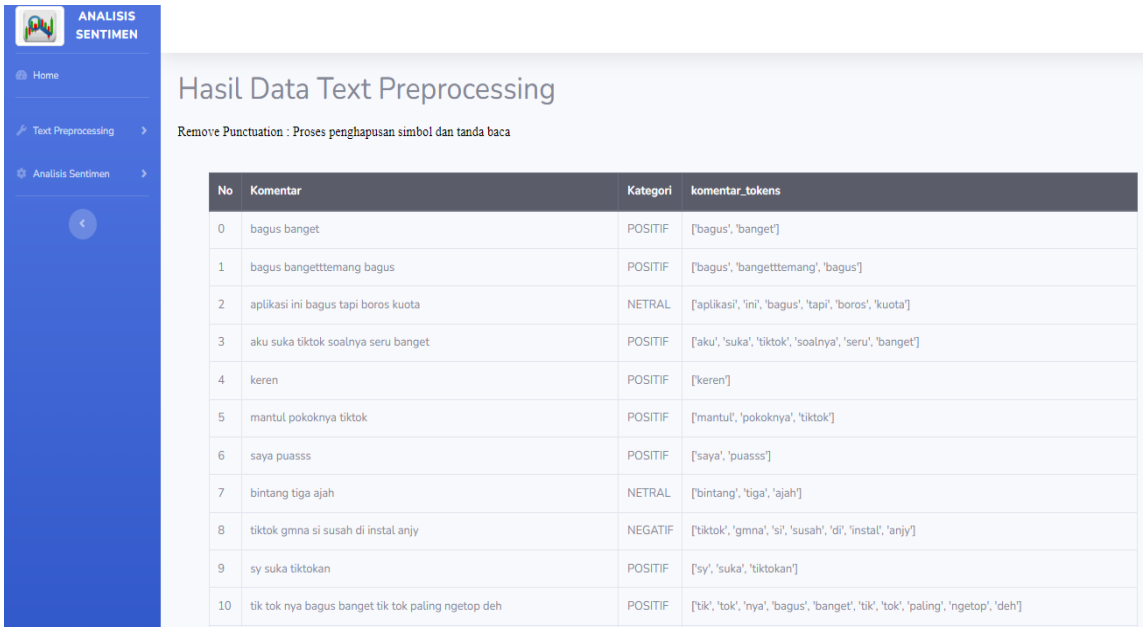
Kode Program 1 : Proses Menampilkan Halaman *Case Folding*

```
@app.route("/casefolding", methods=['GET', 'POST'])
def casefolding():
    with open('casefolding.csv', encoding='latin-1') as csv_file
            if not first_line:
                casefolding.append({
                    "No": row[0],
                    "komentar": row[1],
                    "KATEGORI": row[2],
                })
            else:
                first_line = False
    return render_template('casefolding.html', menu='casefolding',
submenu='data', casefolding=casefolding)
```

Modul Program 4.3 *Source Code* Menampilkan *Halaman Case Folding*

4.1.4 Halaman *Text Preprocessing Remove Punctuation*

Halaman Text Preprocessing *Remove punctuation* berisi tabel yang menampilkan hasil *Text Preprocessing* yang sudah dilakukan. Tabel tersebut memiliki kolom yang terdiri dari nomor, komentar, dan kategori. Tampilan dari halaman *Text Preprocessing Remove Punctuation* dapat dilihat pada **Gambar 4.4,**



No	Komentar	Kategori	komentar_tokens
0	bagus banget	POSITIF	['bagus', 'banget']
1	bagus bangetttemang bagus	POSITIF	['bagus', 'bangetttemang', 'bagus']
2	aplikasi ini bagus tapi boros kuota	NETRAL	['aplikasi', 'ini', 'bagus', 'tapi', 'boros', 'kuota']
3	aku suka tiktok soalnya seru banget	POSITIF	['aku', 'suka', 'tiktok', 'soalnya', 'seru', 'banget']
4	keren	POSITIF	['keren']
5	mantul pokoknya tiktok	POSITIF	['mantul', 'pokoknya', 'tiktok']
6	saya puasss	POSITIF	['saya', 'puasss']
7	bintang tiga ajah	NETRAL	['bintang', 'tiga', 'ajah']
8	tiktok gmna si susah di instal anjy	NEGATIF	['tiktok', 'gmna', 'si', 'susah', 'di', 'instal', 'anjy']
9	sy suka tiktokan	POSITIF	['sy', 'suka', 'tiktokan']
10	tik tok nya bagus banget tik tok paling ngetop deh	POSITIF	['tik', 'tok', 'nya', 'bagus', 'banget', 'tik', 'tok', 'paling', 'ngetop', 'deh']

Gambar 4.4 Tampilan Halaman *Remove Punctuation*

Source code untuk menampilkan halaman utama dapat dilihat pada **modul 4.4,**

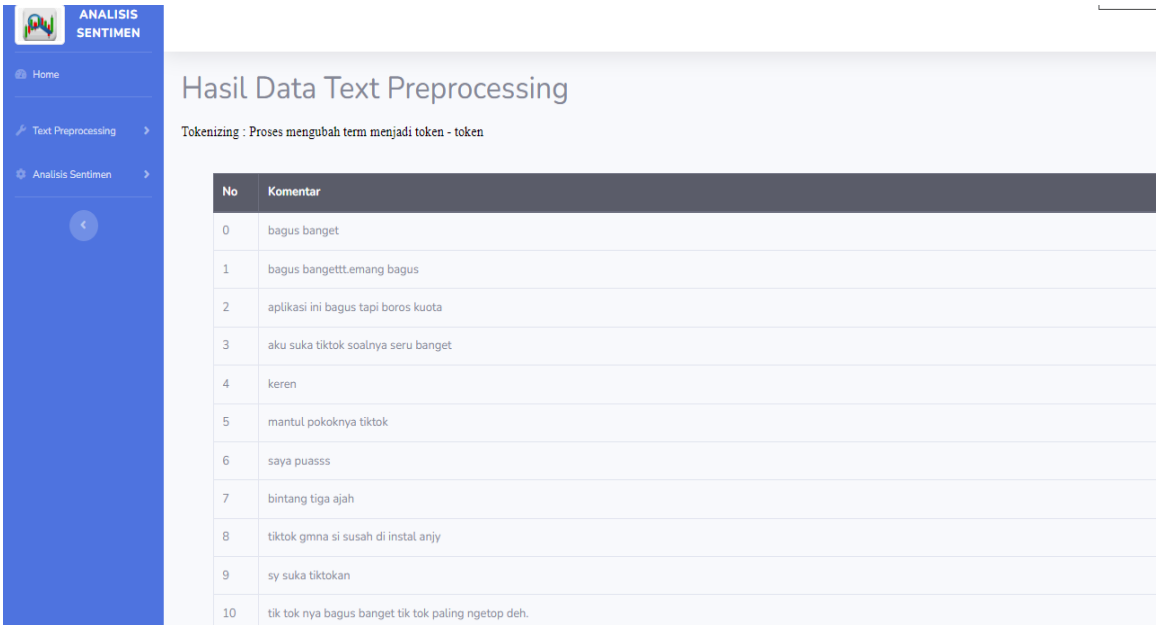
Kode Program 1 : Proses menampilkan halaman *Remove Punctuation*

```
@app.route("/remove", methods=['GET', 'POST'])
def remove():
    with open('removenp.csv', encoding='latin-1') as csv_file:
        data = csv.reader(csv_file, delimiter=',')
            if not first_line:
                remove.append({
                    "No": row[0],
                    "KATEGORI": row[1],
                    "komentar": row[2],
                    "komentar_tokens": row[3],
                })
            else:
                first_line = False
              return render_template('remove.html', menu='remove',
          submenu='data', remove=remove)
```

Modul Program 4.4 *Source Code* Menampilkan *Remove Punctuation*

4.1.5 Halaman *Text Preprocessing Tokenizing*

Halaman *Text Preprocessing Tokenizing* berisi tabel yang menampilkan hasil data *Tokenizing* yang dilakukan pada *Text Preprocessing*. Tabel tersebut memiliki kolom yang terdiri dari nomor, komentar, kategori dan komentar_tokens.. Tampilan dari halaman *Text Preprocessing Tokenizing* dapat dilihat pada **Gambar 4.5,**



Gambar 4.5 Tampilan Halaman Tokenizing

Source code untuk menampilkan halaman *Text Preprocessing* dapat dilihat pada **modul 4.5,**

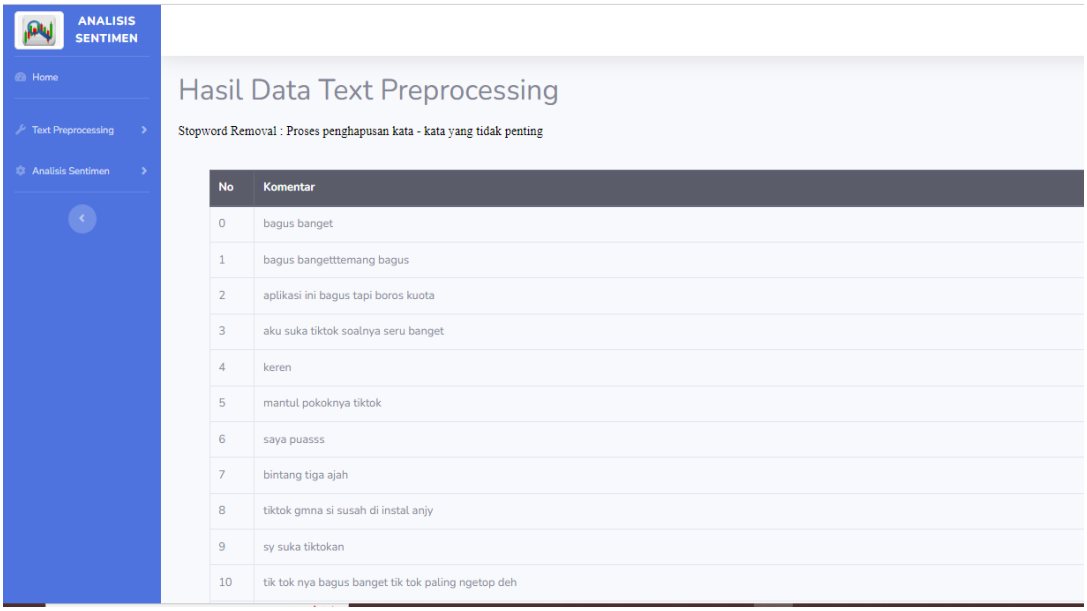
Kode Program 1 : Proses menampilkan halaman *Tokenizing*

```
@app.route("/token", methods=['GET', 'POST'])
def token():
    with open('tokenizing.csv', encoding='latin-1') as csv_file:
        data = csv.reader(csv_file, delimiter=',')
        first_line = True
        token = []
        for row in data:
            if not first_line:
                token.append({
                    "No": row[0],
                    "komentar": row[1],
                    "KATEGORI": row[2],
                })
            else:
```

Modul Program 4.5 *Source Code* Menampilkan Halaman *Tokenizing*

4.1.6 Halaman *Text Preprocessing Stopword Removal*

Halaman *Text Preprocessing Stopword removal* berisi tabel yang menampilkan data hasil *Stopword removal* yang telah di dapatkan dari hasil *Text Preprocessing*. Tabel tersebut memiliki kolom yang terdiri dari nomor, komentar, kategori dan komentar_tokens. Tampilan dari *Text Preprocessing Stopword removal* dapat dilihat pada **Gambar 4.6,**



Gambar 4.6 Tampilan Halaman *Stopword Removal*

Source code untuk menampilkan halaman Text Preprocessing *Stopword Removal* dapat dilihat pada **modul 4.6,**

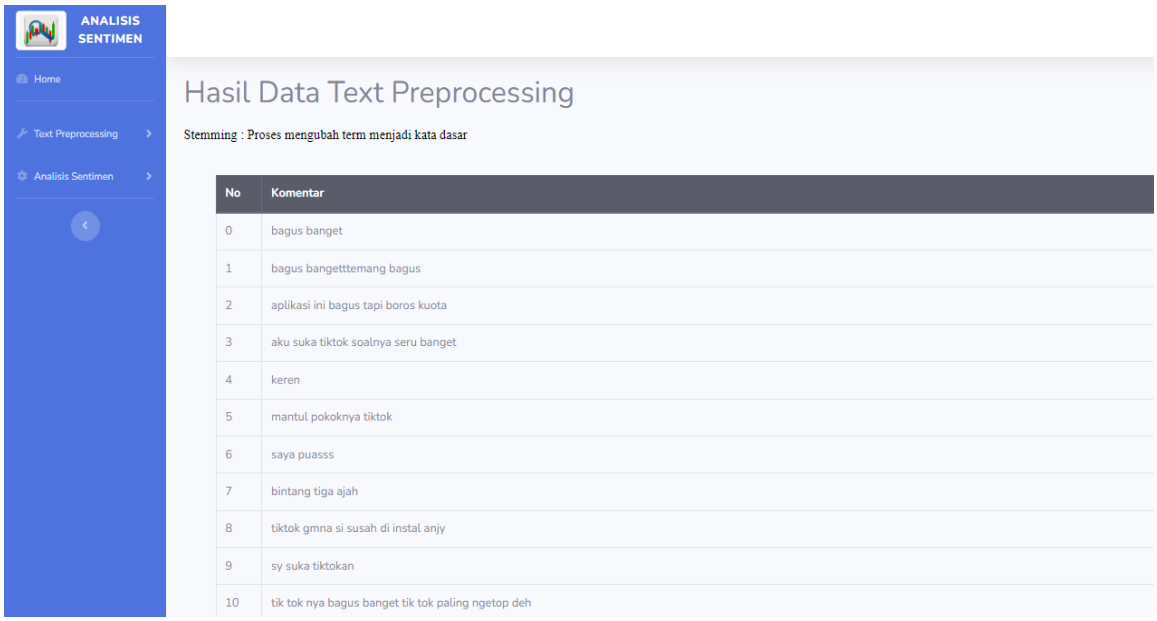
Kode Program 1 : Proses menampilkan halaman *Stopword Removal*

```
@app.route("/stopword", methods=['GET', 'POST'])
def stopword():
    with open('stopword.csv', encoding='latin-1') as csv_file:
        data = csv.reader(csv_file, delimiter=',')
        first_line = True
        stopword = []
                    "No": row[0],
                    "komentar": row[1],
                    "KATEGORI": row[2],
                    "komentar_tokens": row[3],
                    "ulasan_tokens_WSW": row[4],
                })else:
                first_line = False
        return render_template('stopword.html', menu='stopword',
        submenu='data', stopword=stopword)
```

Modul Program 4.6 Source Code Menampilkan Halaman *Stopword Removal*

4.1.7 Halaman *Text Preprocessing Stemming*

Halaman *Text Preprocessing stemming* berisi tabel yang menampilkan hasil *Stemming* yang didaptkan dari hasil *Text Preprocessing*. Tabel tersebut memiliki kolom yang terdiri dari nomor, komentar, kategori,komentar_tokens, ulasan_tokens,ulasan_tokensWSW, komentar_ normalized, dan komentar_tokens_stemmed. Tampilan dari halaman dataset dapat dilihat pada **Gambar 4.7,**



Gambar 4.7 Tampilan Halaman *Stemming*

Source code untuk menampilkan halaman *Text preprocessing Stemming* dapat dilihat pada **modul 4.7,**

Kode Program 1 : Proses menampilkan halaman *Stemming*

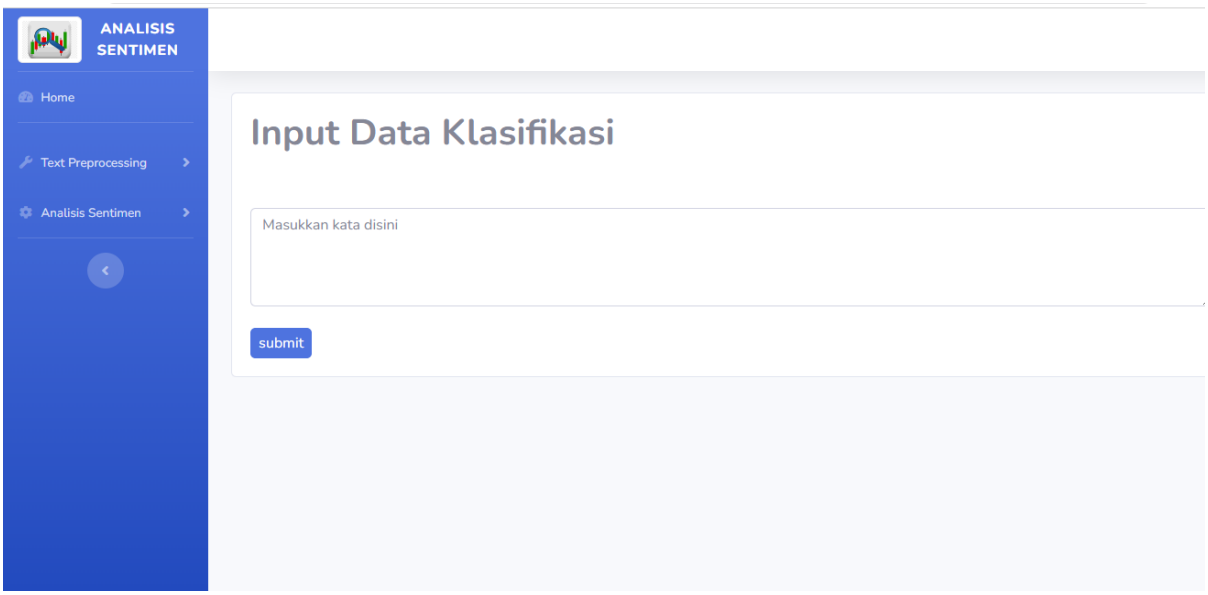
```
@app.route("/stemming", methods=['GET', 'POST'])
def stemming():        stemming = []
        for row in data:
            if not first_line:
                stemming.append({
                    "No": row[0],
                    "komentar": row[1],
                    "KATEGORI": row[2],
                    "komentar_tokens": row[3],
                    "ulasan_tokens_WSW": row[4],
                    "komentar_normalized": row[5],
                    "komentar_tokens_stemmed": row[6],
                })
```

```
            else:
                first_line = False
            return render_template('stemming.html', menu='stemming',
        submenu='data', stemming=stemming)
```

Modul Program 4.7 *Source Code* Menampilkan Halaman *Stemming*

4.1.8 Halaman Analisis Sentimen Pengujian

Halaman Analisis Sentimen pengujian ini menampilkan *Textbox* dan bagian untuk menginput text ulasan. Tampilan dari halaman Analisis sentiment pengujian dapat dilihat pada Gambar 4.8,



Gambar 4.8 Tampilan Halaman Analisis Sentimen Pengujian

Source code untuk menampilkan halaman Pengujian dapat dilihat pada **modul 4.8,**

Kode Program 1 : Proses menampilkan halaman pengujian

```
@app.route("/pengujian")
def pengujian():  # fungsi yang akan dijalankan ketike route dipanggil
            return render_template('pengujian.html')
```

Modul Program 4.8 *Source Code* Menampilkan Halaman Pengujian

Pada **Gambar 4.9** jika sudah menginputkan text yang makan selanjutkan akan menampilkan hasil klasifikasi yang dilakukan sebagai berikut,

Gambar 4.9 Tampilan Halaman Klasifikasi

Setelah data uji diinputkan, maka sistem akan melakukan klasifikasi dengan mengikuti model yang sudah dibuat sebelumnya dengan menggunakan data training. Tampilan dari hasil klasifikasi dapat dilihat pada **Gambar 4.9,**

Kode Program 1 : Proses menampilkan halaman klasifikasi

```
@app.route("/klasifikasi")

                    def klasifikasi():  # fungsi yang akan dijalankan
          ketike route dipanggil
```

Modul Program 4.9 Source Code Tampilan Halaman Klasifikasi

Data uji yang telah diinputkan sebelumnya akan melalui proses *text preprocessing*, dianataranya yaitu *case folding, remove number, remove punctuation, spelling correction, stopword removal, tokenization, dan stemming*. *Source code* untuk semua proses *text preprocessing* tersebut dapat dilihat pada **modul program 4.10,**

Kode Program 1 : Proses menampilkan halaman klasifikasi

```
Subject = request.args.get("sub")
    subject = [subject]

    result = {}

    def case_folding(tokens): #fungsi untuk case folding
        return tokens.lower()

    test_casefolding=[]
```

```
    for i in range(0, len(subject)):
        test_casefolding.append(case_folding(subject[i]))

    result['casefolding'] = ' '.join(list(map(lambda x: str(x),
test_casefolding)))
    casefolding = result['casefolding']

    def remove_num(text):  #fungsi untuk remove number pada data
        text_nonum = ''
        text_nonum = re.sub(r'\d+',' ', text)
        return text_nonum

    test_removenum=[]
    for i in range(0,len(test_casefolding)):
        test_removenum.append(remove_num(test_casefolding[i]))

    result ['remove_num'] = ' '.join(list(map(lambda x: str(x),
test_removenum)))
    removenum = result ['remove_num']

    def remove_punct(text): #fungsi untuk remove punctuation
        text_nopunct = ''
        text_nopunct = re.sub('['+string.punctuation+']', ' ', text)
        return text_nopunct

    def word_tokenize_wrapper(text): #fungsi untu tokenizing
        return word_tokenize(text)

    result['hasil_token'] = word_tokenize_wrapper(removepunct)
    hasil_token = result['hasil_token']
    kamus_slang = open_kamus_prepro('Kamus spelling_word.txt')

    def slangword(text):    #digunakan untuk perbaikan kata pada setiap data
        sentence_list = text.split()
        new_sentence = []

        for word in sentence_list:
            for candidate_replacement in kamus_slang:
                if candidate_replacement == word:
                    word       =       word.replace(candidate_replacement,
kamus_slang[candidate_replacement])
```

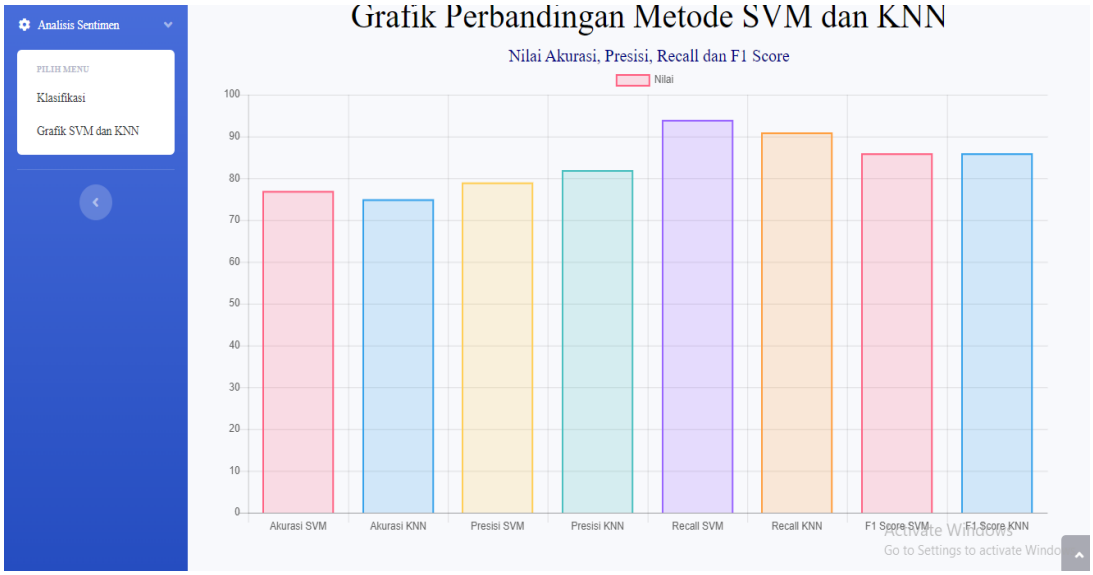
```
            new_sentence.append(word)
        return " ".join(new_sentence)

    test_slangword=[]
    for i in range(0,len(test_removepunct)):
        test_slangword.append(slangword(test_removepunct[i]))

    slangword_ = test_slangword

    list_stopwords = stopwords.words('indonesian') #fungsi untuk stopwords

    list_stopwords.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo',
                           'kalo', 'amp', 'biar', 'bikin', 'bilang',
                           'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                           'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                           'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                      'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                  '&amp', 'yah' ,'ter','bgs','y', 'yg', 'bgs','emg',
                  'knp','wkwk','napa','moga','g','dn','bgsss','sy'
                   'p','sya','ap','dah','gg','apk'])

    list_stopwords = set(list_stopwords)

    def stopwords_removal(text):
        return [word for word in text if word not in list_stopwords]

    result['hasil_stopword'] = stopwords_removal(slangword_)
    hasil_stopword = result['hasil_stopword']

    text_final = stemming_
    #load model naive bayes
    transformer = TfidfTransformer()
    loaded_vec                                                          =
CountVectorizer(decode_error="replace",vocabulary=pickle.load(open("TFIDFse
ntimen.pickle", "rb")))
    tfidf = transformer.fit_transform(loaded_vec.fit_transform(text_final))
    print (tfidf)

    loaded_MNB_model                                                    =
pickle.load(open('MultinomialNBSentimen_model.pickle', 'rb'))
    hasilmnb = loaded_MNB_model.predict(tfidf) #hasil prediksi
```

```
    print (hasilmnb)

    #load model SVM
    transformer = TfidfTransformer()
    loaded_vec                                                          =
CountVectorizer(decode_error="replace",vocabulary=pickle.load(open("TFIDF-
sentimen.pkl", "rb")))
    tfidfsvm                                                            =
transformer.fit_transform(loaded_vec.fit_transform(text_final))
    print (tfidfsvm)

    loaded_SVM_model                                                    =
pickle.load(open('SVMLinierSentimen_model.sav','rb'))
    hasilsvm = loaded_SVM_model.predict(tfidfsvm) #hasil prediksi
    print (hasilsvm)
    return render_template('klasifikasi.html',
                                subject=subject,
                                casefolding = casefolding,
                                remove_num = remove_num,
                                removepunct = removepunct,
                                hasil_token=hasil_token,
                                slangword_ = slangword_,
                                hasil_stopword = hasil_stopword,
                                stemming_= stemming_,
                                grafik=grafik,
                                grafik2=grafik2,
                                text_final= text_final,
                                tfidf=tfidf,
                                hasilmnb= hasilmnb,
                                tfidfsvm=tfidfsvm,
                                hasilsvm=hasilsvm,)
```

Modul Program 4.10 *Source Code* Tampilan Halaman Klasifikasi

4.1.9 Halaman Analisis Sentimen Grafik *Support Vector Machine*

Halaman Analisis Sentimen Grafik *Support Vector Machine* menampilkan grafik dari metode SVM. Grafik tersebut menampilkan beberapa diagram batang perbandingan dari metode *Support Vector Machine* dan *K Nearest Neighbor*. Diagram tersebut berisi hasil *Accuracy*, Presisi dan *Recall* dari kedua metode tersebut. Tampilan dari halaman Grafik *Support Vector Machine* dapat dilihat pada **gambar 4.10,**



Gambar 4.10 Tampilan Halaman Grafik *Support Vector Machine*

Source code untuk menampilkan halaman Grafik *Support Vector Machine* dapat dilihat pada **modul 4.11,**

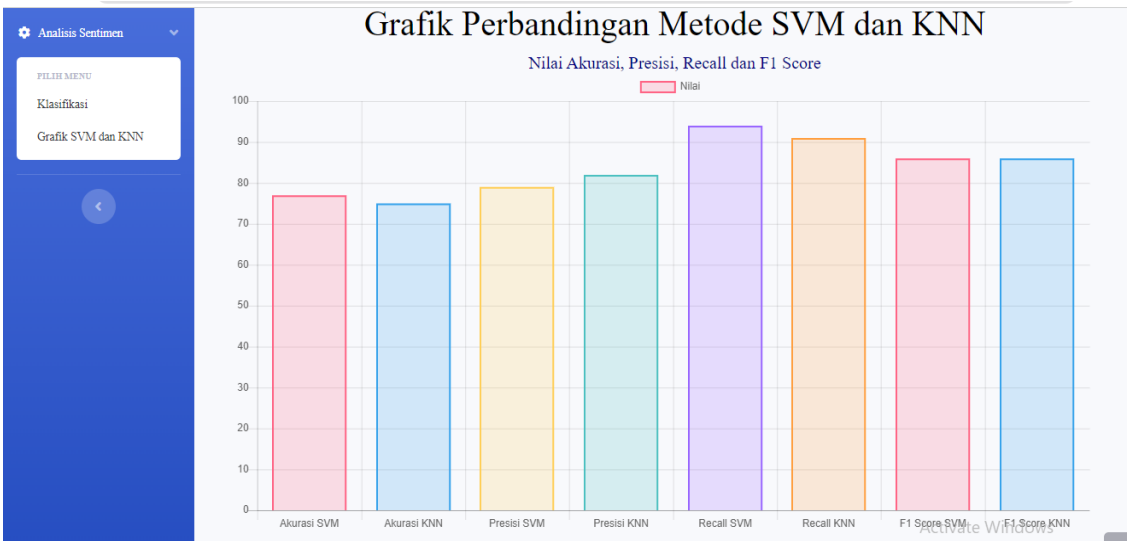
Kode Program 1 : Proses menampilkan halaman *Support Vector Machine*

```
@app.route("/grafik2", methods=['GET', 'POST']) #fungsi untuk
def grafik2():  # fungsi yang akan dijalankan ketike route dipanggil
    with open('sentimendataSVM_report.csv', encoding='latin-1') as csv_file:
            if not first_line:
                grafik2.append({
                    "Name": row[0],
                    "precision": row[1],
                    "recall": row[2],
                    "score": row[3],
                }) else:
                first_line = False
            return render_template('grafik2.html', menu='grafik2',
        submenu='data', grafik2=grafik2)
```

Modul Program 4.11 *Source Code* Menampilkan Grafik *Support Vector Machine*

4.1.10 Halaman Analisis Sentiment Grafik *K-Nearest Neighbor*

Halaman Analisis Sentimen Grafik *K-Nearest Neighbor* menampilkan grafik dari metode KNN. Grafik tersebut menampilkan beberapa diagram batang perbandingan dari metode *Support Vector Machine* dan *K Nearest Neighbor*. Diagram tersebut berisi hasil *Accuracy*, Presisi dan *Recall* dari kedua metode tersebut. Tampilan dari halaman Grafik *K-Nearest Neighbor* dapat dilihat pada gambar 4.11,



Gambar 4.11 Tampilan Halaman Grafik K-Nearest Neighbor

Source code untuk menampilkan halaman Grafik *K-Nearest Neighbor* dapat dilihat pada **modul 4.12,**

Kode Program 1 : Proses menampilkan halaman *K-Nearest Neighbor*

```
@app.route("/grafik", methods=['GET', 'POST']) #fungsi untuk
def grafik():  # fungsi yang akan dijalankan ketike route dipanggil
    with open('knn_report.csv', encoding='latin-1') as csv_file:
        data = csv.reader(csv_file, delimiter=',')
            if not first_line:
                grafik.append({
                    "Name": row[0],
                    "precision": row[1],
                    "recall": row[2],
                    "score": row[3],
                })
            else:
                first_line = False
              return render_template('grafik.html', menu='grafik',
          submenu='data', grafik=grafik)
```

Modul Program 4.12 *Source Code* Menampilkan Halaman Grafik *K-Nearest Neighbor*

4.2 Implementasi *Text Preprocessing*

Data kalimat yang telah diinput untuk mengetahui nilai sentimennya akan dilakukan *preprocessing* terlebih dahulu, prosesnya yaitu *case folding, remove punctuation, stopword removal, spelling correction, stemming, dan tokenizing.*

4.2.1 Implementasi *Case Folding*

Pada Implementasi *Case Folding* dapat dilihat *Source code* pada **4.13** sebagai berikut,

Kode Program 1 : Proses Implementasi *Case Folding*

```
def case_folding(tokens): #fungsi untuk case folding
        return tokens.lower()
    test_casefolding=[]
    for i in range(0, len(subject)):
```

```
        test_casefolding.append(case_folding(subject[i]))


    result['casefolding']    =    '    '.join(list(map(lambda    x:    str(x),
test_casefolding)))
             casefolding = result['casefolding'].html', menu='grafik',
          submenu='data', grafik=grafik)
```

Modul Program 4.13 Source Code Implementasi *Case Folding*

4.2.2 Implementasi Remove Punctuation

Pada Implementasi *Remove Punctuation* dapat dilihat *Source code* pada **4.14** sebagai berikut,

Kode Program 1 : Proses Implementasi *Remove Punctuation*

```
def case_folding(tokens): #fungsi untuk case folding
        return tokens.lower()
    test_casefolding=[]
    for i in range(0, len(subject)):
        test_casefolding.append(case_folding(subject[i]))

    result['casefolding'] = ' '.join(list(map(lambda x: str(x),
        text_nopunct = ''

        text_nopunct = re.sub('['+string.punctuation+']', ' ', text)

        return text_nopunct

    test_removepunct=[]

    for i in range(0,len(test_removenum)):

           test_removepunct.append(remove_punct(test_removenum[i]))
```

Modul Program 4.14 *Source Code* Implementasi *Remove Punctuation*

4.2.3 Implementasi *Tokenizing*

Pada Implementasi *Tokenizing* dapat dilihat *Source code* pada **4.15** sebagai berikut,

Kode Program 1 : Proses Implementasi *Tokenizing*

```
def word_tokenize_wrapper(text): #fungsi untu tokenizing
        return word_tokenize(text)
    result['hasil_token'] = word_tokenize_wrapper(removepunct)
             hasil_token = result['hasil_token']
```

Modul Program 4.15 *Source Code* Implementasi *Tokenizing*

4.2.4 Implementasi Spelling Corection

Pada Implementasi *Spelling Correction* dapat dilihat *Source code* pada **4.16** sebagai berikut,

Kode Program 1 : Proses Implementasi *Spelling Correction*

```
def open_kamus_prepro(x): #fungsi membuka kamus prepro spelling word (kamus
normalisasi)
        kamus={}
        with open(x,'r') as file :
            for line in file :
                slang=line.replace("'","").split(':')
```

```
                kamus[slang[0].strip()]=slang[1].rstrip('\n').lstrip()
        return kamus

    kamus_slang = open_kamus_prepro('Kamus spelling_word.txt')

    def slangword(text):    #digunakan untuk perbaikan kata pada setiap data
        sentence_list = text.split()
        new_sentence = []
        return " ".join(new_sentence)
    test_slangword=[]
```

Modul Program 4.16 *Source Code* Implementasi *Spelling Correction*

4.2.5 Implementasi Stopword Removal

Pada Implementasi *Stopword Removal* dapat dilihat *Source code* pada **4.17** sebagai berikut,

Kode Program 1 : Proses Implementasi *Stopword Removal*

```
      list_stopwords = stopwords.words('indonesian') #fungsi untuk
stopwords

list_stopwords.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo',
                      'kalo', 'amp', 'biar', 'bikin', 'bilang',
                      'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                       'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                      'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                   'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                    '&amp', 'yah' ,'ter','bgs','y', 'yg',
'bgs','emg',
                         'p','sya','ap','dah','gg','apk'])

          list_stopwords = set(list_stopwords)
      def stopwords_removal(text):
              return [word for word in text if word not in list_stopwords]
          result['hasil_stopword'] = stopwords_removal(slangword_)
             hasil_stopword = result['hasil_stopword']
```

Modul Program 4.17 *Source Code* Implementasi *Stopword Removal*

4.2.6 Implementasi Stemming

Pada Implementasi *Stemming* dapat dilihat *Source code* pada **4.18** sebagai berikut,

Kode Program 1 : Proses Implementasi *Stemming*

```
factory = StemmerFactory()
stemmer = factory.create_stemmer()
    def stemming_(text):  #fungsi untuk stemming
     data_stem =[]
     for i in text:
     kata = stemmer.stem(i)
     data_stem.append(kata)
    return data_stem
   stem=[]
   result['stemming_'] = stemming_(hasil_stopword)
   stemming_ = result['stemming_']
         text_final = stemming_
```

Modul Program 4.18 Source Code Implementasi *Stemming*

4.2.7 Implementasi Model *Support Vector Machine*

Pada Implementasi Model *Support Vector Machine* dapat dilihat *Source code* pada **4.19** sebagai berikut,

Kode Program 1 : Proses Implementasi *Support Vector Machine*

```
#load model SVM
    transformer = TfidfTransformer()
    loaded_vec                                                          =
CountVectorizer(decode_error="replace",vocabulary=pickle.load(open("TFIDF-
sentimen.pkl", "rb")))
    tfidfsvm                                                            =
transformer.fit_transform(loaded_vec.fit_transform(text_final))
    print (tfidfsvm)
    loaded_SVM_model                                                    =
pickle.load(open('SVMLinierSentimen_model.sav','rb'))
    hasilsvm = loaded_SVM_model.predict(tfidfsvm) #hasil prediksi
              print (hasilsvm)
```

Modul Program 4.19 *Source Code* Implementasi Model *Support Vector Machine*

4.2.8 Implementasi Model *K-Nearest Neighbor*

Pada Implementasi Model *K Nearest Neighbor* dapat dilihat *Source code* pada **4.20** sebagai berikut,

Kode Program 1 : Proses Implementasi *K Nearest Neighbor*

```
#load model K Nearest Neighbors
    transformer = TfidfTransformer()
    loaded_vec                                                          =
CountVectorizer(decode_error="replace",vocabulary=pickle.load(open("TFIDFse
ntimen.pickle", "rb")))
    tfidf = transformer.fit_transform(loaded_vec.fit_transform(text_final))
    print (tfidf)
    loaded_MNB_model                                                    =
pickle.load(open('MultinomialKNNSentimen_model.pickle', 'rb'))
    hasilmnb = loaded_KNN_model.predict(tfidf) #hasil prediksi
              Print(hasilKNN)
```

Modul Program 4.20 *Source Code* Implementasi Model *K-Nearest Neighbor*

4.3. Pengujian

Pengujian dilakukan untuk mengetahui tingkat keberhasilan dari model yang telah dibuat untuk melakukan analisis sentimen pada data komentar Pengguna Aplikasi Tiktok. Model yang dibuat berjumlah 4 model, yaitu model TF-IDF menggunakan metode *Support Vector Machine* dan *K-Nearest Neighbor*, model TFIDF menggunakan metode *K-Nearest Neighbor* dan model metode *Support Vector Machine.* Pengujian model ini diuji dengan membuat *confusion matrix.* Jumlah dtaset yang digunakan dalam pengujian yaitu sebanyak

4.3.1 Pengujian Confusion Matrix

Setelah dilakukan klasifikasi, akan dilakukan pengujian menggunakan confusion matrix dari ke dua metode yaitu metode *Support Vector Machine* dan Metode *K-Nearest*

Neighbor.Berikut merupakan hasil pengujian metode *Support Vector Machine* dapat dilihat pada tabel 4.1,

Tabel 4.1 *Confusion Matrix* Pengujian dengan Metode SVM

		Prediksi			**Jumlah**
		Negatif	Netral	Positif	
Aktual	Negatif	66	3	61	130
	Netral	12	4	39	55
	Positif	18	5	392	415
Jumlah		96	13	492	

$$Accuracy = \frac{66 + 4 + 392}{600} \times 100\% = \frac{462}{600} \times 100\%$$
$$= 0{,}77 \times 100\%$$
$$= 77\%$$

$$Presisi\ Negatif = \frac{66}{66 + 30} \times 100\% = \frac{66}{96} \times 100\%$$
$$= 68{,}7 \times 100\%$$
$$= 69\%$$

$$Presisi\ Netral = \frac{4}{4 + 8} \times 100\% = \frac{4}{12} \times 100\%$$
$$= 0{,}33 \times 100\%$$
$$= 33\%$$

$$Presisi\ Positif = \frac{392}{392 + 100} \times 100\% = \frac{392}{492} \times 100\%$$
$$= 0{,}79 \times 100\%$$
$$= 80\%$$

$$Recall\ Negatif = \frac{66}{66 + 64} \times 100\% = \frac{66}{130} \times 100\%$$
$$= 0{,}507 \times 100\%$$
$$= 51\%$$

$$Recall\ Netral = \frac{4}{4 + 51} \times 100\% = \frac{4}{51} \times 100\%$$
$$= 0{,}07 \times 100\%$$
$$= 0{,}07\%$$

$$Recall\ Positif = \frac{392}{392 + 492} \times 100\% = \frac{392}{492} \times 100\%$$

$$= 0{,}94 \times 100\%$$

$$= 94\%$$

$$F1\ Score\ Negatif = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$

$$= 2 \times \frac{51\% \times 69\%}{51 + 69\%} \times 100\%$$

$$= 2 \times \frac{3519\%}{120\%} \times 100\%$$

$$= 2 \times \frac{7{,}038\%}{120\%} \times 100\%$$

$$= 58{,}65\%$$

$$F1\ Score\ Netral = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$

$$= 2 \times \frac{0{,}07\% \times 33\ \%}{0{,}07\% + 33\%} \times 100\%$$

$$= 2 \times \frac{23.1\%}{33.07\%} \times 100\%$$

$$= 2 \times \frac{46{,}2\%}{33.07\%} \times 100\%$$

$$= 0.12\%$$

$$F1\ Score\ Positif = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$

$$= 2 \times \frac{94\% \times 80\%}{94\% + 80\%} \times 100\%$$

$$= 2 \times \frac{7520\%}{174\%} \times 100\%$$

$$= 2 \times \frac{15040\%}{174\%} \times 100\%$$

$$= 86\%$$

adapun hasil pengujian Confusion matrix untuk metode *K Nearest Neighbor*, table pengujian *confution matrix* dapat dilihat pada tabel 4.2,

Tabel 4.2 *Confusion Matrix* Pengujian dengan Metode KNN

		Prediksi			**Jumlah**
		Negatif	Netral	Positif	
Aktual	Negatif	55	7	60	122
	Netral	21	8	21	50
	Negatif	29	8	391	428
Jumlah		105	23	472	

$$Accuracy = \frac{55 + 8 + 391}{600} \times 100\% = \frac{454}{600} \times 100\%$$
$$= 0{,}75 \times 100\%$$
$$= 75\%$$

$$Presisi\ Negatif = \frac{55}{55 + 50} \times 100\% = \frac{55}{105} \times 100\%$$
$$= 0{,}52 \times 100\%$$
$$= 52\%$$

$$Presisi\ Netral = \frac{8}{8 + 15} \times 100\% = \frac{8}{23} \times 100\%$$
$$= 0{,}34 \times 100\%$$
$$= 35\%$$

$$Presisi\ Positif = \frac{391}{391 + 81} \times 100\% = \frac{391}{472} \times 100\%$$
$$= 0{,}82 \times 100\%$$
$$= 83\%$$

$$Recall\ Negatif = \frac{55}{55+67} \times 100\% = \frac{55}{122} \times 100\%$$
$$= 0{,}45 \times 100\%$$
$$= 45\%$$

$$Recall\ Netral = \frac{8}{8 + 42} \times 100\% = \frac{8}{50} \times 100\%$$
$$= 0{,}16 \times 100\%$$
$$= 16\%$$

$$Recall\ Positif = \frac{391}{391 + 37} \times 100\% = \frac{391}{428} \times 100\%$$
$$= 0{,}91 \times 100\%$$
$$= 91\%$$

$$F1\ Score\ Negatif = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$
$$= 2 \times \frac{45\% \times 52\%}{45 + 52\%} \times 100\%$$
$$= 2 \times \frac{2340\%}{97\%} \times 100\%$$
$$= 2 \times \frac{4680\%}{97\%} \times 100\%$$

$$= 48{,}24\% \times 100\%$$

$$= 48\%$$

$$F1\ Score\ Netral = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$

$$= 2 \times \frac{16\% \times 35\%}{16\% + 35\%} \times 100\%$$

$$= 2 \times \frac{560\%}{51\%} \times 100\%$$

$$= 2 \times \frac{10{,}980\%}{51\%} \times 100\%$$

$$= 2 \times \frac{21{,}96\%}{51\%} \times 100\%$$

$$= 0{,}22 \times 100\%$$

$$= 22\%$$

$$F1\ Score\ Positif = 2 \times \frac{Recall \times \text{Presisi}}{Recall \times \text{Presisi}} \times 100\%$$

$$= 2 \times \frac{91\% \times 83\%}{91\% + 83\%} \times 100\%$$

$$= 2 \times \frac{7553\%}{174\%} \times 100\%$$

$$= 2 \times \frac{15{,}10\%}{174\%} \times 100\%$$

$$= 86{,}81\% \times 100\%$$

$$= 86\%$$

Berdasarkan tabel confusion matrix yang dihasilkan pada ke dua metode yang digunakan untuk pengujian tercantum pada **Tabel 4.3, Tabel 4.4, Tabel 4.5, Tabel 4.6** maka dapat dihitung nilai akurasi, presisi, *recall* dan *f-1 score*.

Tabel 4.3 Hasil pengujian

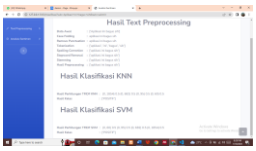
Metode	Accuracy	Precision	Recall	F-1 Score
SVM	77%	80%	94%	86%
KNN	75%	83%	91%	87%

Pada Tabel 4.3 merupakan kumpulan nilai akurasi, presisi, recall dan f-1 score yang dihasilkan dari pengujian menggunakan confusion matrix. Nilai akurasi berkaitan dengan seberapa akurat model dalam melakukan klasifikasi data uji dengan benar. Nilai presisi merupakan perbandingan antara data yang diprediksi benar dibandingkan dengan data yang diprediksi. Nilai recall merupakan perbandingan antara data uang diprediksi benar dengan 81

data yang benar. Sedangkan nilai f-1 score merupakan perbandingan rata-rata presisi dan recall yang dibobotkan. Pada metode *Support vector machine* di dapatkan akurasi 77%, nilai presisi 80%, recall sebesar 94% dan untuk nilai f-1 score sebesar 86%. Sedangkan untuk Algoritma *K-Nearest Neighbor* di dapatkan akurasi sebesar 75%, nilai presisi 83%, recall 91% dan nilai f-1 score sebesar 87%.

Selain pengujian pada model, pada penelitian ini juga dilakukan pengujian pada system. Pengujian system dilakukan menggunakan blackbox testing. Pengujian ini dilakukan untuk memeriksa apakah system berjalan sesuai dengan rancangan atau tidak.

Tabel 4.4 Hasil pengujian sistem

No.	Skenario Pengujian	Hasil yang diharapkan	Hasil	Kesimpulan
1	user dapat melihat Halaman Awal tampilan yang telah dibuat	Sistem dapat diakses.		**Berhasil**
2	User dapat melihat seluruh data yang telah dilakukan proses Text Preprocessing	Sistem dapat menampilkan halaman data hasil text preprocesing		**Berhasil**
3	sistem menampilkan kategori kelas dari teks yang telah dimasukan. Beserta proses preprocessing yang terjadi.	Sistem dapat menampilkan kolom input klasifikasi serta hasil klasifikasi		**Berhasil**
4	Sistem menampilkan table hasil akurasi dan juga menampilkan grafik yang telah dibuat.	Sistem dapat menampilkan hasil grafik perbandingan metode SVM dan KNN		**Berhasil**

4.3.2 Pembahasan

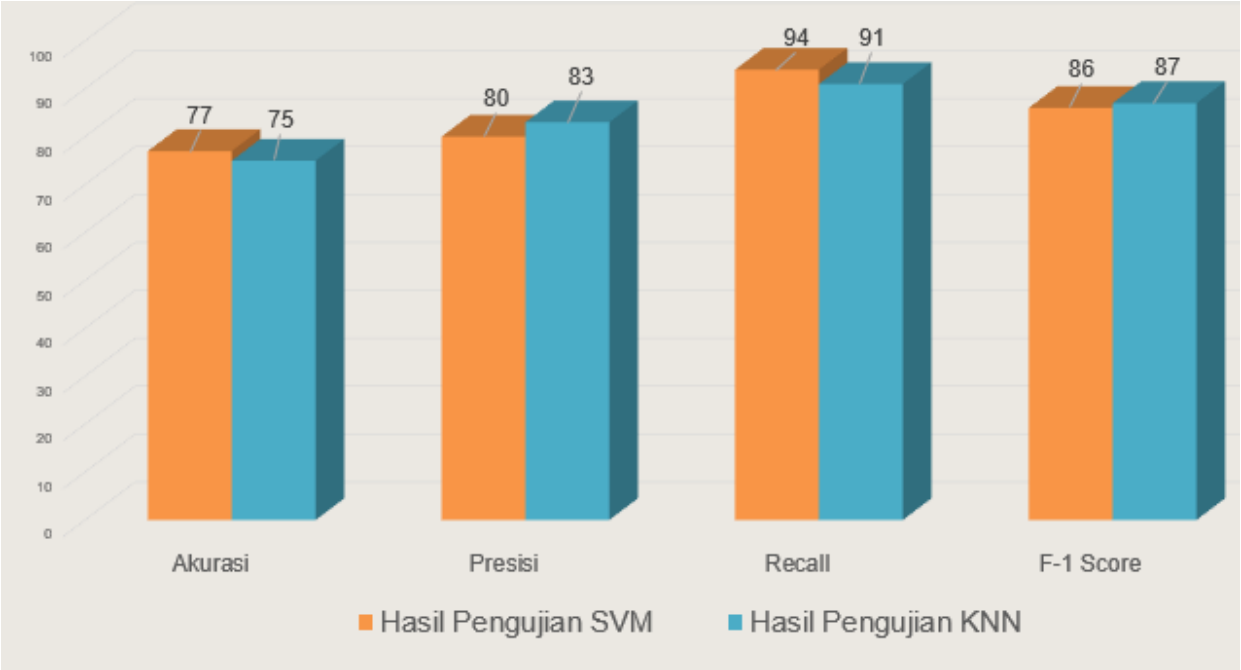
Pada Perbandingan metode *Support Vector machine* dan *K-Nearest Neighbor* yaitu pengujian yang telah dilakukan dengan menggunakan 3000 data hasil scraping yang dibagi menjadi 2400 data training dan 600 data testing dari aplikasi Tiktok. Yang dimana dua metode tersebut dibandingkan dan dilihat dari nilai akurasi, presisi, recall dan f-1 score. Pengujian yang dilakukan dengan menggunakan metode *Support Vector Machine* mendapatkan tingkat akurasi sebesar 77% 2% lebih besar dari KNN, nilai presisi 79% 3% lebih kecil dari KNN, nilai *recall* 94% 3% lebih besar dari KNN, dan nilai f1-*score* 86% 1%

lebih kecil dari KNN.Sedangkan untuk metode *K-Nearest Neighbor* mendapatkan tingkat akurasi sebesar 75% 2% lebih kecil dari SVM, nilai presisi 82% 3% lebih besar dari SVM, nilai *recall* 91% 3% lebih kecil dari SVM, dan nilai f1-*score* 87% 1% lebih besar dari SVM.

selain itu dilakukan proses Preprocessing yaitu *Case folding, Remove Punctuation, Tokenizing, Spelling Correction, Stopword Removal dan Stemming.* Serta digunakan TF IDF sebagai langkah untuk pembobotan setiap kata yang digunakan. Kemudian hasil akhir untuk menentukan hasil kelas, digunakan klasifikasi *Support Vector machine dan K-Nearest Neighbor*.

Setelah dilakukan pengujian terhadap algoritma *Support Vector machine dan K-Nearest Neighbor* dengan dibandingkan. Maka di dapatkan hasil dari perbandingan antara keuda algotitma tersebut, dengan data aplikasi Tiktok. Grafik mengenai perbandingan Akurasi, Presisi, Recall dan F1 Score dari Algotitma *Support Vector machine dan K-Nearest Neighbor* dapat dilihat pada gambar 4.12.

Pada penelitian ini memberikan informasi bahwa dengan dilakukanya perbandingan metode antara *Support vector machine* dan *K-Nearest Neighbor* dapat memberikan pengaruh terhadap akurasi, presisi, recal dan nilai f1 score yang di daptakan. juga mengetahui algoritma mana yang mendaptkan akurasi yang lebih tinggi untuk analisis sentimen aplikasi Tiktok yaitu algoritma *Support Vector Machine.* Selain itu dalam penggunaan *Preprocessing* seperti *case folding, Remove punctuation, tokenizing, spelling correction, stopword removal* dan *stemming* berpengaruh dalam proses klasifikasi. Serta proses pembobotan menggunakan TF-IDF berpengaruh dalam pembobotan kata yang dilakukan dalam klasifikasi kata.



Gambar 4.12 Hasil perbandingan akurasi

BAB V
PENUTUP

5.1. Kesimpulan

Dari hasil pengujian, yang telah dilakukan, pada penelitian ini untuk mengetahui tingkat akurasi dari ke dua metode tersebut, diperoleh perbandingan antara Algoritma *Support Vector Machine* dengan *K-Nearest Neighbor*. Di dapatkan Tingkat akurasi sebesar 77% pada *Support Vector Machine* dan 75% untuk *K-Nearest Neighbor*. Algoritma *Support Vector machine* mendapatkan tingkat akurasi yang cukup tinggi menggunakan kernel linier dan menggunakan pembobotan TF IDF untuk algoritma *K-Nearest Neighbor* dengan menggunakan nilai K=3 agar mendaptakan akurasi lebih baik. Hasil perbandingan dari algoritma *Support Vector Machine* dan *K-Nearest Neighbor* dapat disimpulkan hasil yang tinggi di dapatkan oleh algoritma *Support Vector Machine* yaitu 77%.

5.2. Saran

Saran untuk penelitian selanjutnya yaitu,

1. melakukan perbandingan terhadap algoritma lain, seperti pada metode deep learning.
2. penggunaan metode *Support Vector Machine* tidak hanya menggunakan kernel linier tetapi menggunakan beberapa kernel lainya seperti kernel RBF, Sigmoid dan Polynomial.
3. Pada algoritma *K-Nearst Neighbor* nilai k yang kecil maka dapat mempengaruhi nilai akurasi yang dihasilkan. Hal ini dikarenakan, data yang masuk pada k tetangga terdekat terlalu sedikit dan belum banyak tetangga yang digunakan untuk proses klasifikasi.

DAFTAR PUSTAKA

Al-shufi, M. F., & Erfina, A. (2021). Sentimen Analisis Mengenai Aplikasi Streaming Film Menggunakan Algoritma Support Vector Machine Di Play Store. *Sismatik*, 156–162.

Amalia, B. S., Umaidah, Y., & Mayasari, R. (2021). Analisis Sentimen Review Pelanggan Restoran Menggunakan Algoritma Support Vector Machine Dan K-Nearest Neighbor. *SITEKIN: Jurnal Sains, Teknologi Dan Industri*, *19*(1), 28–34.

Darmawan, R., & Amini, S. (2022). *Perbandingan Hasil Sentimen Analysis Menggunakan Algoritma Naïve Bayes dan K-Nearest Neighbor pada Twitter Comparison of Sentiment Analysis Results Using Naïve Bayes and K- Nearest Neighbor Algorithm on Twitter*. *September*, 78–85.

Daryfayi Edyt, D. P., & Asror, I. (2020). Sentimen Analisis pada Ulasan Google Play Store Menggunakan Metode Naïve Bayes. *Sentimen Analisis Pada Ulasan Google Play Store Menggunakan Metode Naïve Bayes*, *7*(Ulasan Pada Google Play Store), 11.

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).

Fanny, O., & Suroyo, H. (2022). Analisis Sentimen Pengguna Media Sosial Terhadap Omnibus Law Berdasarkan Hashtag di Twitter. *Januari*, *11*(1), 2540–9719. http://sistemasi.ftik.unisi.ac.id

Hendra, A., & Fitriyani, F. (2021). Analisis Sentimen Review Halodoc Menggunakan Nai ̈ve Bayes Classifier. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *6*(2), 78–89. https://doi.org/10.14421/jiska.2021.6.2.78-89

Herlinawati, N., Yuliani, Y., Faizah, S., Gata, W., & Samudi, S. (2020). Analisis Sentimen Zoom Cloud Meetings di Play Store Menggunakan Naïve Bayes dan Support Vector Machine. *CESS (Journal of Computer Engineering, System and Science)*, *5*(2), 293. https://doi.org/10.24114/cess.v5i2.18186

Ilmawan, L. B., & Mude, M. A. (2020). Perbandingan Metode Klasifikasi Support Vector Machine dan Naïve Bayes untuk Analisis Sentimen pada Ulasan Tekstual di Google Play Store. *ILKOM Jurnal Ilmiah*, *12*(2), 154–161. https://doi.org/10.33096/ilkom.v12i2.597.154-161

Indriani, A. (2020). Analisa Perbandingan Metode Naïve Bayes Classifier Dan K-Nearest Neighbor Terhadap Klasifikasi Data. *Sebatik*, *24*(1), 1–7. https://doi.org/10.46984/sebatik.v24i1.909

Kurniawan, S., Gata, W., Puspitawati, D. A., -, N., Tabrani, M., & Novel, K. (2019). Perbandingan Metode Klasifikasi Analisis Sentimen Tokoh Politik Pada Komentar Media Berita Online. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *3*(2), 176–183. https://doi.org/10.29207/resti.v3i2.935

Maarif, M. R. (2016). Perbandingan Naïve Bayes Classifier dan Support Vector Machine untuk Klasifikasi Judul Artikel. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *1*(2), 90–93. https://doi.org/10.14421/jiska.2016.12-05

Nurhafida, S. I., Sembiring, F., Raya, J., & No, C. (2022). *Analisis Sentimen Aplikasi Novel Online Di Google Play Store Menggunakan Algoritma Support Vector Machine (SVM)*. *6*, 317–327.

Nurrun Muchammad Shiddieqy, H., Paulus Insap, S., & Wing Wahyu, W. (2016). Studi Literatur Tentang Perbandingan Metode Untuk Proses Analisis Sentimen Di Twitter. *Seminar Nasional Teknologi Informasi Dan Komunikasi*, *February*, 57–64.

Pamuji, A. (2021). Performance of the K-Nearest Neighbors Method on Analysis of Social Media Sentiment. *Juisi*, *07*(01), 32–37.

Rahman, A., Utami, E., Informatika, M. T., & Yogyakarta, U. A. (2021). *Sentimen Analisis Terhadap Aplikasi pada Google Playstore Menggunakan Algoritma Naive Bayes dan Algoritma Genetika*. *5*(1), 60–71.

Riyanto, U. (2018). *ANALISIS PERBANDINGAN ALGORITMA NAIVE BAYES DAN SUPPORT VECTOR MACHINE DALAM MENGKLASIFIKASIKAN JUMLAH PEMBACA*. 62–72.

Rohanah, A., Rianti, D. L., & Sari, B. N. (2021). Perbandingan Naïve Bayes dan Support Vector Machine untuk Klasifikasi Ulasan Pelanggan Indihome. *STRING (Satuan Tulisan Riset Dan Inovasi Teknologi)*, *6*(1), 23. https://doi.org/10.30998/string.v6i1.9232

Rokhman, K. A., Berlilana, B., & Arsi, P. (2021). Perbandingan Metode Support Vector Machine Dan Decision Tree Untuk Analisis Sentimen Review Komentar Pada Aplikasi Transportasi Online. *Journal of Information System Management (JOISM)*, *3*(1), 1–7. https://doi.org/10.24076/joism.2021v3i1.341

Sahara, & Rizqi. (2022). *METODE KNN PADA SENTIMENT ANALISIS REVIEW PRODUK GAME ANDROID*. *11*(2).

Sahara, S., & Permana, R. A. (2019). Methode K-Nn for Analys Sentiment Review Kids Apps. *JST (Jurnal Sains Dan Teknologi)*, *8*(2), 127–137. https://doi.org/10.23887/jstundiksha.v8i2.21240

Saputra, R. A. (2022). *Analisis Sentimen Pada Media Sosial Menggunakan Metode K-Nearest Neighbor (K-NN)*. *2*(8), 1–8.

Sola, Fide, Suparti, & Sudarno. (2021). *ANALISIS SENTIMEN ULASAN APLIKASI TIKTOK DI GOOGLE PLAY MENGGUNAKAN METODE SUPPORT VECTOR MACHINE (SVM) DAN ASOSIASI*. *10*, 346–358.

Trisnadi, Moch. F., Faraby, S. Al, & Dwifebri, M. (2021). Sentiment Analysis pada Movie Review Menggunakan Feature Selection Mutual Information dan K-Nearest Neighbour Classifier. *E-Proceeding of Engineering*, *8*(5), 1–11.

Wahyudi, R., & Kusumawardana, G. (2021). Analisis Sentimen pada Aplikasi Grab di Google Play Store Menggunakan Support Vector Machine. *Jurnal Informatika*, *8*(2), 200–207. https://doi.org/10.31294/ji.v8i2.9681

Zulqornain, J. A., & Adikara, P. P. (2021). *Analisis Sentimen Tanggapan Masyarakat Aplikasi Tiktok Menggunakan Metode Naïve Bayes dan Categorial Propotional Difference (CPD)*. *5*(7), 2886–2890.