DETEKSI *HATE SPEECH* PADA KOLOM KOMENTAR TIKTOK DENGAN MENGGUNAKAN ALGORITMA *NAÏVE BAYES* DENGAN *SMOOTHING*

(Skripsi)

Oleh

ROY RAFLES MATORANG PASARIBU
NPM 2117051058

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

DETEKSI *HATE SPEECH* PADA KOLOM KOMENTAR TIKTOK DENGAN MENGGUNAKAN ALGORITMA *NAÏVE BAYES* DENGAN *SMOOTHING*

Oleh

ROY RAFLES MATORANG PASARIBU

Skripsi

Sebagai Salah Satu Syarat Untuk Mencapai Gelar
SARJANA KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Lampung

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

DETEKSI *HATE SPEECH* PADA KOLOM KOMENTAR TIKTOK DENGAN MENGGUNAKAN ALGORITMA *NAÏVE BAYES* DENGAN *SMOOTHING*

Oleh

ROY RAFLES MATORANG PASARIBU

Hate speech adalah ekspresi yang bersifat bias, bermusuhan, dan diskriminatif yang ditujukan kepada individu atau kelompok berdasarkan karakteristik mereka. Di era digital, *platform* media sosial seperti TikTok sangat rentan terhadap penyebaran *hate speech*, terutama melalui kolom komentar. Metode tradisional untuk mendeteksi *hate speech* sering kali tidak efektif dalam menangani volume data yang besar dan beragamnya ekspresi pengguna. Oleh karena itu, penelitian ini mengembangkan model deteksi *hate speech* berbasis *Multinomial Naïve Bayes* dengan *smoothing*. Penerapan *smoothing* bertujuan untuk mengatasi permasalahan *zero probability* pada kata-kata yang jarang muncul, sehingga meningkatkan akurasi prediksi model. *Dataset* dibagi dengan rasio 80:20, di mana 80% digunakan untuk pelatihan dan 20% untuk pengujian. Model yang dikembangkan mencapai akurasi sebesar 88,41%, dengan nilai presisi, *recall*, dan *F1-score* yang serupa. Evaluasi lebih lanjut melalui pengujian pengguna dengan 35 partisipan yang menganalisis 7.415 komentar menunjukkan akurasi deteksi sebesar 68,6%. Sebagai implementasi, model ini diintegrasikan ke dalam *plugin* Google Chrome yang mendeteksi *hate speech* pada komentar TikTok secara *real time*, dilengkapi dengan visualisasi probabilitas dan mekanisme validasi pengguna.

Kata kunci : Deteksi *hate speech*, *Naïve Bayes*, *smoothing*, TikTok, analisis sentimen, *machine learning*, *plugin* Google Chrome.

ABSTRACT

HATE SPEECH DETECTION IN TIKTOK COMMENT COLUMN USING NAÏVE BAYES ALGORITHM WITH SMOOTHING

By

ROY RAFLES MATORANG PASARIBU

Hate speech is a biased, hostile, and discriminatory expression directed at individuals or groups based on their inherent characteristics. In the digital era, social media platforms such as TikTok have become highly susceptible to the spread of hate speech, particularly through comment sections. Traditional methods for detecting hate speech are often ineffective in handling large volumes of data and diverse user expressions. Therefore, this study develops a hate speech detection model based on Multinomial Naïve Bayes with smoothing. The application of smoothing aims to address the issue of zero probability for rarely occurring words, thereby enhancing the model's predictive accuracy. The dataset is split into an 80:20 ratio, with 80% used for training and 20% for testing. The developed model achieves an accuracy of 88.41%, with precision, recall, and F1-score at similar levels. Further evaluation through user testing with 35 participants analyzing 7,415 comments indicates a detection accuracy of 68.6%. As an implementation, this model is integrated into a Google Chrome plugin that detects hate speech in TikTok comments in real time, featuring probability visualization and user validation mechanisms.

Keywords : *Hate speech detection, Naïve Bayes, TikTok, sentiment analysis, machine learning, Google Chrome plugin.*

Judul Skripsi	:	**DETEKSI *HATE SPEECH* PADA KOLOM KOMENTAR TIKTOK DENGAN MENGGUNAKAN ALGORITMA *NAÏVE BAYES* DENGAN *SMOOTHING***
Nama Mahasiswa	:	Roy Rafles Matorang Pasaribu
Nomor Pokok Mahasiswa	:	2117051058
Jurusan	:	Ilmu Komputer
Fakultas	:	Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing

Didik Kurniawan, S.Si., M.T.
NIP. 198004192005011004

Muhaqiqin, S.Kom., M.T.I.
NIP. 199305252022031009

2. Ketua Jurusan Ilmu Komputer

Dwi Sakethi, S.Si., M.Kom.
NIP. 196806111998021001

MENGESAHKAN

1. Tim Penguji

Ketua : **Didik Kurniawan, S.Si., M.T.** .....

Sekertaris : **Muhaqiqin, S.Kom., M.T.I.** .....

Penguji Utama : **Dr. rer. nat. Akmal Junaidi, M.Sc.** .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.
NIP. 197110012005011002

Tanggal Lulus Ujian Skripsi: **21 April 2025**

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Roy Rafles Matorang Pasaribu

NPM : 2117051058

Menyatakan bahwa skripsi saya yang berjudul **"Deteksi *Hate Speech* Pada Kolom Komentar Tiktok Dengan Menggunakan Algoritma *Naïve Bayes* Dengan *Smoothing*"** merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang di skripsi ini telah mengikuti kaidah penulisan karya tulis ilmiah Universitas Lampung. Apabila dikemudian hari terbukti skripsi saya merupakan hasil jiplakan atau dibuat orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar yang saya terima.

Bandar Lampung, 04 Juni 2025

Penulis,

Roy Rafles Matorang Pasaribu

NPM. 2117051058

RIWAYAT HIDUP

Penulis lahir di Mulyakencana pada tanggal 05 Mei 2003 sebagai anak ketiga dari pasangan Ibu Mimin Aminah (Alm) dan Bapak Arif Pasaribu. Penulis telah menyelesaikan Pendidikan formal di SDN 1 Mulya Kencana yang ada di kecamatan Tulang Bawang Tengah Kabupaten Tulang Bawang Barat Provinsi Lampung pada tahun 2015. Kemudian SMPN 1 Tumijajar yang ada di kecamatan Tumijajar Kabupaten Tulang Bawang Barat Provinsi Lampung pada tahun 2018 dan SMAN 9 Bandar Lampung pada tahun 2021. Di tahun 2021 juga, penulis terdaftar sebagai mahasiswa program studi Ilmu Komputer di Universitas Lampung melalui jalur *test* atau SBMPTN.

Selama menjadi mahasiswa di Ilmu Komputer, penulis aktif di dalam berbagai kegiatan baik didalam maupun diluar Universitas Lampung. Kegiatan yang di lakukan adalah sebagai berikut,

1. Menjuarai lomba Olimpiade Sains Akbar Nasional Perguruan Tinggi yaitu sebagai peraih medali perunggu pada bidang Matematika Tingkat Nasional.
2. Menjadi bagian dari anggota Himpunan Mahasiswa Ilmu Komputer Universitas Lampung 2022.
3. Menjadi Asisten Dosen untuk mata kuliah Dasar-dasar Pemorgraman di Jurusan Ilmu Komputer tahun 2022.
4. Menjadi Asisten Dosen untuk mata kuliah Sistem Operasi di Jurusan Ilmu Komputer tahun 2023.
5. Menjadi Asisten Dosen untuk mata kuliah Pemrorgaman Berorientasi Objek di Jurusan Ilmu Komputer tahun 2023.
6. Menjadi Asisten Dosen untuk mata kuliah Teknologi Aplikasi Mobile di Jurusan Ilmu Komputer tahun 2024.

7. Melakukan Kuliah Kerja Nyata di Desa Gebang, Kecamatan Teluk Pandan Kabupaten Pesawaran, Provinsi Lampung pada tahun 2024 Periode 1.
8. Mengikuti program Magang Kampus Merdeka di Tunas Honda (Tunas Dwipa Matra) sebagai *System Developer* pada tahun 2024.

MOTTO

"Serahkanlah hidupmu kepada TUHAN dan percayalah kepada-Nya, dan Ia akan bertindak"

(Mazmur 37:5)

"Karena masa depan sungguh ada, dan harapanmu tidak akan hilang"

(Amsal 23:18)

"Selama kamu tetap menjadi dirimu sendiri, kamu tidak akan pernah salah."

(Patrick Star)

"Berhentilah hidup di masa lalu, itu hanya akan menyakitimu."

(Patrick Star)

PERSEMBAHAN

Kupersembahkan karya ini kepada :

Tuhan Yesus Kristus,

Yang telah memberikan berkat dan karunianya senantiasa kepada penulis.

Bapak, Mama, Kakak Mery, Abang Ronal Tercinta

Yang telah menjadi penyemangat penulis untuk menyelesaikan pendidikan di Jurusan Ilmu Komputer ini. Terima kasih karena telah mendukung penulis dalam mencapai tujuan penulis.

SANWACANA

Shallom.

Puji syukur kepada Allah Bapa Tuhan Yesus Kristus atas penyertaanNya, skripsi yang berjudul "Deteksi *Hate Speech* Pada Kolom Komentar Tiktok Dengan Menggunakan Algoritma *Naïve Bayes* Dengan *Smoothing*" dapat diselesaikan penulis dengan baik. Skripsi ini merupakan salah satu syarat guna menyelesaikan proses perkuliahan dan mendapat gelar Sarjana Komputer di Jurusan Ilmu Komputer Fakulitas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

Dalam hal ini, penulis berterima kasih kepada setiap pihak yang berpartisipasi dalam membantu proses penyelesaian skripsi ini. Dalam kesempatan ini, penulis mengucapkan terima kasih kepada:

1. Bapak, Mama, Kakak Mery, Abang Ronal yang selalu mendoakan dan mendukung penulis, setiap semangat di saat lelah, dan setiap pelukan di tengah putus asa. Kalian adalah alasan terkuat penulis bertahan.
2. Bapak Dr. Eng. Heri Satria, S.Si., M.Si, selaku Dekan Fakultas MIPA Univeristas Lampung.
3. Bapak Dwi Sakethi selaku Ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.
4. Bapak Tristiyanto, S.Kom., M.I.S., Ph.D., selaku dosen Pembimbing Akademik yang telah senantiasa membimbing dan memberikan saran masukan selama berkuliah di jurusan Ilmu Komputer, Universitas Lampung.
5. Bapak Didik Kurniawan, S.Si., M.T., selaku dosen Pembimbing Utama atas kebaikannya dalam memberikan bimbingan, dukungan dan saran kritik dalam penyelesaian skripsi.

6. Bapak Muhaqiqin, S.Kom., M.T.I., selaku dosen Pembimbing Kedua atas keikhlasannya dalam membimbing penulis dalam proses penyusunan skripsi serta tempat penulis berkeluh kesal selama menyelesaikan skripsi ini.
7. Bapak Dr. rer. nat. Akmal Junaidi, M.Sc., selaku dosen penguji yang telah memberikan saran dan kritik guna penyempurnaan skripsi.
8. Ibu Yunda selaku Sekretaris Jurusan Ilmu Komputer FMIPA Universitas Lampung terima kasih atas segala dukungan dalam urusan akademik maupun administrasi.
9. Ibu Nora, selaku admin Jurusan yang senantisa membantu menyusun adminstrasi dan dukungan dalam urusan administrasi selama di Ilmu Komputer.
10. Seluruh dosen Jurusan Ilmu Komputer FMIPA Universitas Lampung yang telah memberikan pengajaran dalam perkuliahan.
11. Seluruh keluarga dan saudaraku yang mendukung proses perkuliahan hingga penyelesaian skripsi yang tak dapat disebutkan satu persatu.
12. Ainun Farihah yang selalu hadir di setiap fase dari pencarian ide, pengajuan judul, revisi tanpa henti, hingga akhir perjuangan ini dan yang selalu memberikan semangat dan waktunya serta membantu kepada penulis mulai dari pengajuan judul hingga selesai melakukan tahap skirpsi.
13. Sobat HARTA TAHTA NILAI A: Ikhsan, Cindy, Jihan, Abiyi, Vidya dan Ayuni yang menjadi sobat dari semester 1 hingga tamat, kalian bukan hanya teman kuliah, tapi juga bagian dari keluarga kecil yang penuh warna.
14. Sobat Ilmu Komputer Nicholas Elvis, Salsabila Juandira, Shafa Aulia, yang telah menjadi teman penulis.
15. Sahabatku Gumay, Hersan, dan Zaky yang selalu mendukung penulis dan menjadi pendengar keluh kesal penulis dan tetap tinggal ketika semua terasa berat.
16. Ojun Gengs: Arkan, Radit, dan Alfa yang menjadi support, tempat bercerita keluh kesal skripsi dan membantu saya dalam menyelesaikan skirpsi ini.
17. Disi Gibing: Restu, Ainun, Agis, Annisa, Rahma, Sabina, dan Yesha sebagai teman teman yang bersama saya melakukan Kuliah Kerja Nyata di Desa Gebang Kecamatan Teluk Pandan, Kabupaten Pesawaran, Provinsi Lampung pada tahun 2024. Terima kasih atas momen-momen yang tak akan terlupa.

18. Bang Alif, Alfiadi Lim, M. Faiz Arippudin, Bang Riki dan teman-teman kos di De'Nayu yang setia menemani dari senja hingga fajar, dari revisi ke revisi, dari kopi pertama hingga kopi terakhir. Sampai berjumpa di waktu dan tempat yang berbeda ygy.
19. Teman-teman Jurusan Ilmu Komputer FMIPA Universitas Lampung angkatan 2021 yang sudah menjadi bagian dari cerita hidup penulis dalam suka, duka, tugas, presentasi, hingga sidang akhir.
20. Dan semua pihak yang tidak bisa disebutkan satu per satu, namun kehadiran dan kontribusinya begitu berarti dalam menyelesaikan skripsi ini terima kasih dari lubuk hati yang paling dalam.

Penulis menyadari bahwa dalam penulisan skripsi ini masih terdapat banyak kekurangan karena keterbatasan kemampuan, pengalaman serta pengetahuan penulis. Oleh karena itu, saran dan kritik yang membangun sangat diharapkan sebagai bahan evaluasi untuk kedepannya. Semoga skripsi ini dapat bermanfaat bagi semua pihak.

Bandar Lampung, 04 Juni 2025

Penulis,

Roy Rafles Matorang Pasaribu
NPM. 2117051058

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR GAMBAR

Gambar Halaman

I. PENDAHULUAN

1.1 Latar Belakang dan Masalah

Hate speech (ujaran kebencian) didefinisikan sebagai ujaran yang bermotif bias, bersifat permusuhan, dan jahat, yang ditujukan kepada seseorang atau kelompok orang karena beberapa karakteristik bawaan mereka yang sebenarnya. Ujaran ini mengungkapkan sikap diskriminatif, intimidatif, tidak setuju, antagonistik, dan atau prasangka terhadap karakteristik tersebut, yang meliputi jenis kelamin, ras, agama, etnis, warna kulit, asal negara, disabilitas, atau orientasi seksual. *Hate speech* bertujuan untuk melukai, mendiskreditkan, melecehkan, mengintimidasi, merendahkan, menghina, dan mengorbankan kelompok yang menjadi sasarannya serta memicu ketidakpekaan dan kekerasan terhadap mereka (Elliott *et al*., 2016).

Deteksi *hate speech* menjadi penting karena metode tradisional berbasis aturan sering kali tidak mampu menangani jumlah konten yang sangat besar yang dihasilkan pengguna di *platform* media sosial. Metode ini juga kurang fleksibel dan sulit beradaptasi dengan cepat terhadap berbagai jenis bahasa dan cara ekspresi yang terus berkembang. Sebaliknya, teknik *machine learnin*g telah menunjukkan hasil yang menjanjikan dalam mengotomatisasi proses identifikasi bahasa kebencian dan analisis sentimen yang terkandung dalam data teks (Subramanian *et al*., 2023).

Memasuki era globalisasi, teknologi informasi menjadi salah satu media penyampaian data atau pengolahan informasi yang sangat cepat dan dapat menjangkau ke berbagai tempat, salah satu contohnya adalah media sosial berbasiskan internet seperti Tiktok. TikTok telah dikonseptualisasikan sebagai aplikasi media sosial berbasis video yang sangat unik dengan struktur teknis yang khas dan tingkat adopsi pengguna yang tak tertandingi dibandingkan platform lain (Murphy *and* McCashin, 2023). Hal ini menjadikannya sebuah jaringan *online* yang khusus di mana fitur imitasi dan memetik semakin mempercepat interaktivitas pengguna yang beragam (Zulli dan Zulli, 2022). Play Store, TikTok tercatat telah diunduh lebih dari 100 juta kali, dengan rata-rata penilaian pengguna sebesar 4,4 dari skala 5. Saat ini, Indonesia menempati peringkat keempat dalam hal jumlah pengguna TikTok terbanyak di dunia. Hal tersebut menurut keterangan resmi dari perusahaan yang di sampaikan oleh Julia Chan, *Mobile Insights Analyst* (Mahardhika *et al*., 2021). TikTok menawarkan berbagai fitur, termasuk kolom komentar untuk memungkinkan interaksi antar pengguna. Pengguna dapat saling bertukar pendapat secara terbuka melalui komentar, namun semakin banyaknya interaksi dan pertukaran pendapat di TikTok, penggunaan ujaran kebencian masih sering terjadi, baik secara sadar maupun tidak.

Hate speech sangat bertentangan dengan konsep kesantunan bahasa, yang merupakan indikator kecerdasan berbahasa, serta etika dalam komunikasi. Etika adalah kesadaran dan pengetahuan tentang perilaku yang baik atau buruk yang dilakukan oleh manusia. Etika juga bisa dilihat dari cara *netizen* (pengguna yang aktif di media sosial) memberikan komentar. Penilaian terhadap nilai baik dan buruk dari *hate speech* menjadi awal dari penyalahgunaan media sosial di era ini. Saat ini, banyak kasus *hate speech* yang terjadi, seperti penghinaan, pencemaran nama baik, penistaan, provokasi, bahkan penyebaran berita bohong (*hoaks*) di berbagai aplikasi media sosial, termasuk di TikTok. Hal ini terjadi karena *netizen* diberikan kebebasan pribadi dalam menjelajahi media sosial, sehingga mereka merasa bebas berbicara tanpa memikirkan dampak yang mungkin terjadi, apalagi kebencian adalah sifat manusia yang alami (Ria dan Setiawan, 2023).

Metode klasifikasi seperti *support vector machine* (SVM), *deep learning* (DL), dan *naïve bayes* (NB) telah digunakan oleh beberapa peneliti terdahulu untuk pendeteksi *hate speech*. SVM memiliki keunggulan dalam memberikan hasil yang akurat dengan waktu pelatihan yang relatif singkat. SVM sangat efisien dalam menangani masalah *non-linier* yang melibatkan batas-batas yang kompleks, dan ini membuatnya sering digunakan dalam berbagai aplikasi seperti klasifikasi, prediksi, dan pemisahan data yang rumit. Dengan tingkat akurasi yang tinggi dan kemampuan untuk mengatasi masalah *non-linier*, SVM menjadi pilihan yang sangat baik untuk menyelesaikan permasalahan dunia nyata yang seringkali rumit dan membutuhkan hasil yang dapat diandalkan. Namun seperti *machine learning* lainnya SVM memiliki kelemahan yaitu rentan terhadap *overfitting*, terutama jika tidak ada penyesuaian yang tepat (Pisner dan Schnyer, 2019).

Deep learning memiliki keunggulan signifikan dalam analisis data karena kemampuannya untuk secara otomatis belajar dari data mentah dan mengekstraksi fitur yang relevan tanpa intervensi manual. Struktur berlapis-lapis pada lapisan tersembunyinya memungkinkan model ini untuk mengidentifikasi pola yang lebih kompleks dibandingkan dengan teknik *machine learning* tradisional. Selain itu, *deep learning* telah terbukti memberikan akurasi yang lebih baik dan performa yang unggul dalam berbagai studi terkait deteksi *hate speech* dan analisis sentimen, karena mampu menangani data teks dengan lebih efektif dan adaptif (Subramanian *et al*., 2023). Dengan lebih banyak data, model *deep learning* bisa lebih efektif dalam memahami pola dan variasi yang ada, sehingga mengurangi kemungkinan *overfitting*, dan hal ini juga yang membuat kelemahan dari algoritma ini, harus memerlukan data yang besar untuk melakukan klasifikasi yang efektif (Putri *et al*., 2023).

Naïve bayes memiliki kompatibel untuk *dataset* yang sangat besar maupun kecil. Model ini adalah metode klasifikasi yang sangat sederhana, canggih, dan berkinerja dengan baik bahkan dalam penerapan yang rumit sekalipun (Jackins *et al*., 2021). sehingga NB akan menjawab kelemahan dari *deep learning*. Selain itu, NB dapat mengatasi *overfitting* yang terjadi pada *support vector machine* dengan cara mengestimasi parameter-parameternya menggunakan seluruh data *training* (Tan

dan Shenoy, 2020). Akan tetapi NB juga memiliki kelemahan pada probabilitasnya, yaitu memiliki masalah probabilitas nol pada saat pengujian di kelas tertentu yang tidak ada dalam data *training* yang memungkinkan akan berakhir dengan probabilitas *zero frequency* yaitu jika terdapat kategori dalam variabel kategori yang tidak muncul dalam data pelatihan, maka model *naïve bayes* akan mengatributkan probabilitas nol ke kategori tersebut, sehingga tidak dapat digunakan untuk melakukan prediksi, meskipun demikian *zero frequency* dapat diatasi dengan teknik *smoothing* (Noto dan Saputro, 2022).

Smoothing, atau proses pelancaran, adalah teknik yang digunakan untuk menciptakan fungsi pendekatan guna menangkap pola-pola penting dalam data (Pan *et al*., 2020). Penelitian Prasetyo *et al*., (2024), menunjukkan bahwa metode *naïve bayes* dengan *smoothing* berhasil mengklasifikasikan data penerima bantuan langsung dengan kombinasi variabel numerik dan kategorikal, mencapai akurasi tinggi sebesar 95,9% dalam membedakan kategori layak dan tidak layak. Keberhasilan ini membawa potensi besar dalam pengembangan pendeteksi *hate speech* di Indonesia, yang dapat diterapkan dalam konteks yang lebih luas. Dalam kontinuitas penelitian ini, penulis akan mengeksplorasi lebih lanjut penggunaan kombinasi algoritma NB dan metode *smoothing* dalam konteks deteksi *hate speech*, dengan harapan bahwa hal ini akan memberikan hasil akurasi yang tinggi dan bermanfaat bagi individu dalam mendapatkan informasi yang benar. Menurut Paul, (2024) dalam konteks *machine learning*, standar akurasi tinggi merujuk pada seberapa dekat prediksi model terhadap nilai yang benar atau hasil yang diharapkan. Akurasi tinggi berarti model *machine learning* mampu menghasilkan prediksi yang sangat sesuai dengan data latih atau uji yang diberikan Dengan demikian, penelitian ini memiliki potensi untuk memberikan kontribusi yang berharga dalam konteks menciptakan komunikasi bermedia sosial yang lebih baik.

1.2 Rumusan Masalah

Berdasarkan permasalahan yang telah dijelaskan pada latar belakang masalah maka rumusan masalah dari penelitian ini adalah:

1. Bagaimana penerapan algoritma *naïve bayes* dengan *smoothing* dalam mendeteksi *hate speech* pada kolom komentar Tiktok.
2. Bagaimana mengukur akurasi penerapan algoritma *naïve bayes* dengan *smoothing* dalam mendeteksi *hate speech* pada kolom komentar Tiktok.

1.3 Batasan Masalah

Batasan masalah dari penelitian ini adalah:

1. Penelitian ini menggunakan *dataset* publik yang terdapat pada kolom komentar di suatu konten Tiktok.
2. Komentar yang digunakan adalah komentar yang diizinkan TikTok untuk dapat diambil.
3. Komentar yang dideteksi adalah komentar yang menggunakan bahasa Indonesia saja untuk menghindari kompleksitas terkait variasi bahasa dan konteks lokal.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Membuat sebuah perangkat lunak dengan menerapkan algoritma *naïve bayes* dengan *smoothing* untuk mendeteksi *hate speech* pada kolom komentar Tiktok.
2. Mengukur hasil akurasi penerapan algoritma *naïve bayes* dengan *smoothing* dalam mendeteksi *hate speech* pada kolom komentar Tiktok.

1.5 Manfaat Penelitian

Manfaat penelitian ini bertujuan untuk menghasilkan perangkat lunak yang mampu mendeteksi *hate speech* pada Tiktok dengan memanfaatkan algoritma *naïve bayes* dengan *smoothing*. Alat ini akan berperan penting dalam mengidentifikasi dan memfilter informasi *hate speech* di media *online* khususnya Tiktok, yang semakin relevan di era digital saat ini. Kombinasi antara *naïve bayes*, yang unggul dalam menangani dataset kompleks, dan metode *smoothing*, yang efektif dalam

memahami konteks bahasa, diharapkan dapat meningkatkan akurasi deteksi *hate speech* secara signifikan. Dengan pendekatan ini, hasil yang diperoleh diharapkan mendapatkan hasil yang baik.

Selain itu, penerapan *naïve bayes* dengan *smoothing* dalam penelitian ini dapat menjadi contoh inovatif dari penggunaan teknologi *modern* dalam pengolahan bahasa alami dan analisis teks. Ini juga memberikan kontribusi penting dalam pengembangan teknologi informasi di Indonesia, khususnya dalam bidang pengolahan data teks dan deteksi *hate speech*. Manfaat yang dihasilkan dari penelitian ini tidak hanya terbatas pada aspek teknis, tetapi juga memiliki dampak positif bagi masyarakat luas. Dengan menyediakan alat yang dapat diimplementasikan untuk memerangi penyebaran *hate speech*, penelitian ini membantu meningkatkan ketenangan dan kenyamanan terhadap informasi yang beredar di media sosial.

II. TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian terkait dalam penelitian ini digunakan sebagai bahan acuan dan perbandingan untuk deteksi *hate speech* pada kolom komentar Tiktok dengan algoritma *naïve bayes*. Topik penelitian yang menjadi pembanding ialah nilai akurasi dari sebuah sistem deteksi *hate speech*. Secara umum, gambaran mengenai beberapa penelitian yang digunakan dalam penelitian ini terangkum pada Tabel 1.

Tabel 1. Penelitian Terdahulu

Penulis	Judul	Metode	Hasil
(Ariska dan Kamayani, 2024)	Deteksi *Hate speech* pada Kolom Komentar Tiktok dengan menggunakan SVM	TF-IDF, *Support Vector Machines* (SVM)	Hasil evaluasi menunjukkan bahwa deteksi *hate speech* menggunakan VADER Sentiment lebih unggul, dengan akurasi 96,21%, presisi 92,23%, *recall* 99%, dan *f1-score* 95,50%, dibandingkan fitur TF-IDF dan algoritma SVM.

Penulis	Judul	Metode	Hasil
(Fatahillah *et al*., 2017)	*Implementation Of Naive Bayes Classifier Algorithm On Social Media (Twitter) To The Teaching Of Indonesian Hate speech*	*Naïve Bayes*	Penelitian menunjukkan bahwa sistem berhasil mengumpulkan tweet berdasarkan *hashtag* dan mengklasifikasikannya menggunakan *Naïve Bayes* dengan akurasi 93%. Pemilihan data latih sangat memengaruhi hasil klasifikasi, dan pengembangan selanjutnya disarankan agar klasifikasi dilakukan secara real-time dengan mempertimbangkan makna kata untuk meningkatkan akurasi.
(Putri *et al*., 2020)	*A comparison of classification algorithms for hate speech detection*	*Naïve Bayes* dan *SMOTE*	Penelitian ini menghasilkan dataset baru untuk *hate speech* dengan 2776 contoh *hate speech* dan 1226 non-*hate speech*. *Multinomial Naïve Bayes* (MNB) dengan unigram tanpa *SMOTE* terbukti paling efektif, mencapai *recall* 93.2%, dan direkomendasikan sebagai model terbaik.

Penulis	Judul	Metode	Hasil
(Prasetyo *et al.*, 2024)	*Classification of Cash Direct Recipients Using the Naive Bayes with Smoothing*	*Naïve Bayes* dan Metode *Smoothing*	Penelitian ini menghasilkan model untuk memprediksi penerima bantuan BLT dengan penerapan *Naïve Bayes* dan *smoothing yang*, mencapai akurasi 95,9%.

Penelitian mengenai deteksi *hate speech* pada kolom komentar TikTok pernah dilakukan oleh Amelia dan Mia, (2023), dengan judul "Deteksi *Hate speech* pada kolom komentar TikTok dengan menggunakan SVM". Penelitian ini memanfaatkan kombinasi fitur *Term Frequency-Inverse Document Frequency* (TF-IDF) untuk memberikan bobot pada kata-kata yang terdapat dalam komentar, serta algoritma *Support Vector Machines* (SVM) untuk melakukan klasifikasi. TF-IDF digunakan untuk mengekstraksi fitur teks dari komentar-komentar yang diduga mengandung *hate speech*, sementara SVM bertugas untuk mengelompokkan komentar tersebut ke dalam kategori yang relevan. Sebagai bagian dari evaluasi, hasil dari algoritma SVM dibandingkan dengan metode lain seperti *Vader Sentiment* dalam mendeteksi *hate speech*. Dalam penelitian ini, didapati bahwa penggunaan *Vader Sentiment* memberikan hasil yang lebih unggul dengan akurasi sebesar 96,21%, presisi 92,23%, *recall* 99%, dan *f1-score* 95,50%.

Penelitian oleh Fatahillah *et al*., (2017), berjudul "Implementation of Naive Bayes Classifier Algorithm on Social Media (Twitter) to the Teaching of Indonesian *Hate speech*" mengeksplorasi penerapan algoritma *naïve bayes* untuk deteksi *hate speech* di Twitter. Penelitian ini bertujuan mengklasifikasikan tweet yang menggunakan hashtag terkait *hate speech* di Indonesia. Algoritma *naïve bayes* memproses data tweet yang dikumpulkan dan mengelompokkan tweet ke dalam kategori *hate speech* atau non-*hate speech*. Hasilnya menunjukkan akurasi sebesar 93%. Penelitian menekankan pentingnya pemilihan dan penyusunan data latih dalam

mempengaruhi hasil klasifikasi. Untuk pengembangan selanjutnya, disarankan agar sistem dapat melakukan klasifikasi secara *real-time* dan mempertimbangkan makna kata atau frasa untuk meningkatkan akurasi hasil.

Penelitian oleh Putri *et al*., (2020), membandingkan algoritma klasifikasi untuk mendeteksi *hate speech* di media sosial menggunakan Multinomial Naïve Bayes (MNB) dan Synthetic Minority Over-sampling Technique (SMOTE). Mereka membuat dataset baru dengan 2776 contoh *hate speech* dan 1226 contoh non-*hate speech* untuk mengatasi masalah ketidakseimbangan data. Hasil penelitian menunjukkan bahwa Multilayer Perceptron (MLP) dengan SMOTE memberikan akurasi tertinggi, namun MNB tanpa SMOTE unggul dalam *recall*, mencapai 93,2%. Karena *recall* penting untuk mendeteksi sebanyak mungkin *hate speech*, MNB direkomendasikan sebagai model yang lebih efektif untuk tugas ini. Penelitian ini menekankan pentingnya dataset seimbang dan fitur yang tepat, serta pemilihan model berdasarkan prioritas metrik seperti akurasi atau *recall*.

Penelitian oleh Prasetyo *et al*., (2024), meneliti prediksi penerima bantuan BLT dengan menggunakan algoritma klasifikasi *naïve bayes* yang dipadukan dengan *smoothing*. Bantuan Langsung Tunai (BLT) adalah program sosial yang diberikan kepada warga yang memenuhi kriteria tertentu. Pemerintah desa menentukan penerima bantuan menggunakan sistem konvensional melalui musyawarah desa. Setelah melalui serangkaian proses pelatihan dan pengujian model, hasil yang diperoleh sangat memuaskan dengan tingkat akurasi mencapai 95,9%. Dengan akurasi yang tinggi, diharapkan model ini dapat menjadi alat bantu bagi pemerintah desa dalam mendukung pengambilan keputusan yang lebih objektif, efektif, dan efisien dalam distribusi BLT.

Penelitian ini dibandingkan penelitian yang dilakukan oleh (Ariska dan Kamayani, 2024), terletak pada pembaruan metode yang digunakan. Jika penelitian sebelumnya menggunakan metode *Support Vector Machine* (SVM), penelitian ini akan menggunakan metode *Naïve Bayes*. Selain itu, pembaruan dibandingkan penelitian (Fatahillah *et al*., 2017), terdapat pada objek media sosial yang dianalisis, di mana penelitian mereka menggunakan Twitter, sedangkan penelitian ini

menggunakan TikTok. Selanjutnya, pembaruan penelitian dari (Putri *et al*., 2020), adalah pada metode tambahan yang diterapkan. Penelitian mereka menggunakan *Naïve Bayes* dengan SMOTE, sedangkan penelitian ini akan menggunakan smoothing. Terakhir, perbedaan dengan penelitian (Prasetyo *et al*., 2024), terletak pada objeknya. Jika penelitian sebelumnya berfokus pada prediksi penerima BLT, maka penelitian ini difokuskan untuk mendeteksi *hate speech*. Selain itu, penelitian ini juga mengembangkan perangkat lunak berupa *plugin* di Google Chrome yang mampu mendeteksi *hate speech* secara *real-time* pada web TikTok. Hal ini menjadi pembeda karena TikTok, sebagai *platform* media sosial, masih tergolong baru berkembang di Indonesia. Oleh karena itu, objek penelitian ini memberikan konteks yang berbeda dibandingkan dengan penelitian-penelitian sebelumnya.

2.2 *Hate speech*

Hate speech adalah bentuk komunikasi yang dilakukan oleh individu atau kelompok dengan tujuan untuk memicu provokasi, memancing kebencian, atau memicu permusuhan terhadap orang atau kelompok tertentu berdasarkan faktor-faktor seperti ras, agama, etnis, orientasi seksual, atau gender (Poletto *et al*., 2021). *Hate speech* sering kali menyebar melalui media sosial dan *platform* daring lainnya, di mana kontennya dapat dengan cepat tersebar luas, menyebabkan ketegangan dan konflik antar kelompok di masyarakat. Fenomena ini juga menimbulkan tantangan serius dalam menjaga keseimbangan antara hak kebebasan berpendapat dan perlindungan martabat manusia, dan menjadi topik penting dalam penelitian serta pengembangan teknologi, termasuk penggunaan kecerdasan buatan untuk mendeteksi dan menangani konten berbahaya tersebut.

2.3 Tiktok

TikTok adalah *platform* media sosial yang berfokus pada pembuatan dan berbagi video pendek hingga 60 detik, dengan fitur pengeditan dalam aplikasi serta berbagai efek dan suara yang mendukung kreativitas pengguna. Sering dibandingkan dengan

Vine, TikTok menonjol karena lebih mengutamakan interaksi melalui konten viral daripada hubungan antar pengguna, di mana imitasi dan replikasi tren menjadi inti dari interaksi sosial. Meskipun menawarkan fitur sosial seperti profil dan pesan, TikTok lebih mengedepankan partisipasi kolektif dalam tren budaya pop, menciptakan jaringan sosial berdasarkan mimesis. TikTok juga diwarnai kontroversi, termasuk kekhawatiran privasi data, seperti dalam investigasi keamanan oleh pemerintah AS terhadap perusahaan induknya, ByteDance. Terlepas dari tantangan ini, TikTok memiliki dampak budaya yang besar, memungkinkan pengguna untuk berinteraksi secara kreatif dan membentuk pola jaringan sosial yang berbeda dari platform lain, dengan partisipasi dalam tren viral menjadi pusatnya (Zulli dan Zulli, 2022).

2.4 *Preprocessing*

Preprocessing merupakan langkah penting dalam analisis teks, terutama untuk analisis sentimen, karena informasi teks sering kali mengandung banyak data yang tidak terstruktur dan berisik. Proses *preprocessing* membantu membersihkan dan menyiapkan data untuk analisis, mengurangi proses komputasi, dan mengoptimalkan kinerja dan akurasi klasifik (Pradana dan Hayaty, 2019).

Berikut merupakan tahapan *preprocessing* yang umum digunakan:

1. *Cleaning* Teks

   Cleaning merupakan proses pembersihan teks dari karakter atau simbol yang tidak relevan seperti tanda baca, angka, link URL, emoji, atau spasi ganda. Tujuan dari *cleaning* adalah untuk menghilangkan *noise* agar teks lebih bersih dan siap diproses lebih lanjut.
2. *Case Folding*

   Case folding merupakan tahapan untuk mengubah semua huruf dalam dokumen teks menjadi huruf kecil untuk menyamakan format teks dan mengurangi variasi yang tidak diperlukan.

3. *Tokenization*

   Tokenzation merupakan tahapan memisahkan teks menjadi kata-kata atau token individual untuk memudahkan analisis lebih lanjut.

4. *Stopwords Removal*

   Stopwords removal merupakan tahapan menghapus kata-kata yang sering muncul tetapi tidak memiliki makna signifikan, seperti "yang", "di", "untuk", dan "dari" dalam bahasa Indonesia.

5. *Normalization*

   Normalization adalah proses menyamakan bentuk kata-kata dengan cara memperbaiki kata yang salah eja atau mengonversi kata tidak baku menjadi bentuk standar. Dalam konteks bahasa Indonesia, ini bisa meliputi konversi kata slang seperti “gpp” menjadi “tidak apa-apa” atau "aku" menjadi "saya".

6. *Word Stemming*

   Word stemming merupakan tahapan yang dilakukan untuk mengubah kata menjadi bentuk dasarnya dengan menghilangkan prefiks dan sufiks.

2.5 TF-IDF

TF-IDF (*Term Frequency-Inverse Document Frequency*) adalah ukuran statistik yang digunakan dalam pemrosesan teks dan pengambilan informasi untuk menilai pentingnya suatu istilah dalam sebuah dokumen relatif terhadap koleksi dokumen lainnya atau korpus. Metode ini terdiri dari dua komponen utama. Pertama, *Term Frequency* (TF), yang mengukur seberapa sering suatu istilah muncul dalam dokumen tertentu. Semakin sering istilah tersebut muncul, semakin tinggi nilai TF-nya. Kedua, *Inverse Document Frequency* (IDF), yang mengukur seberapa penting istilah tersebut di seluruh korpus dokumen. Semakin banyak dokumen yang mengandung istilah tersebut, semakin rendah nilai IDF-nya karena istilah tersebut dianggap kurang unik. Skor akhir TF-IDF diperoleh dengan mengalikan nilai TF dan IDF, yang membantu menyeimbangkan frekuensi istilah dalam dokumen dengan kelangkaannya di seluruh korpus (Zhu *et al*., 2019).

Rumus perhitungan untuk TF-IDF dapat dinyatakan menggunakan Persamaan sebagai berikut:

$$TF(i,x) = \frac{jumlah\ kali\ kata(i)\ muncul\ dalam\ dokumen\ (x)}{Total\ jumlah\ kata\ dalam\ dokumen\ (x)} \quad (1)$$

$$IDF(i) = \log\left(\frac{Total\ jumlah\ dokumen\ koleksi\ dokumen}{Jumlah\ dokumen\ yang\ mengandung\ kata\ i}\right) \quad (2)$$

IDF dengan *smoothing*

$$IDF(i) = \log\left(\frac{Total\ jumlah\ dokumen\ koleksi\ dokumen+1}{Jumlah\ dokumen\ yang\ mengandung\ kata\ i+1}\right) + 1 \quad (3)$$

$$TF - IDF(i,x) = TF(i,x)\ x\ IDF(i) \quad (4)$$

2.6 *InSet Lexicon*

Penelitian oleh (Koto dan Rahmaningtyas, 2017), menyatakan bahwa *InSet Lexicon* adalah leksikon sentimen yang dikembangkan khusus untuk analisis sentimen dalam bahasa Indonesia. Leksikon ini digunakan untuk mengidentifikasi opini tertulis dan mengkategorikan sentimen tersebut ke dalam dua kelompok utama, yaitu opini positif dan negatif. *InSet Lexicon* memuat 3.609 kata positif dan 6.609 kata negatif, sehingga sangat berguna dalam analisis sentimen berbasis leksikon, terutama untuk bahasa Indonesia. Salah satu keunggulan utama dari *InSet Lexicon* adalah kemampuannya dalam menangani bahasa informal yang umum ditemukan di platform mikroblogging dan media sosial. Bahasa informal ini sering kali memerlukan penyesuaian khusus untuk mendapatkan hasil klasifikasi yang lebih akurat. Penelitian ini berfokus pada evaluasi dan peningkatan kemampuan klasifikasi teks berbasis sentimen, yang membantu memahami persepsi pengguna terhadap suatu topik di ranah digital.

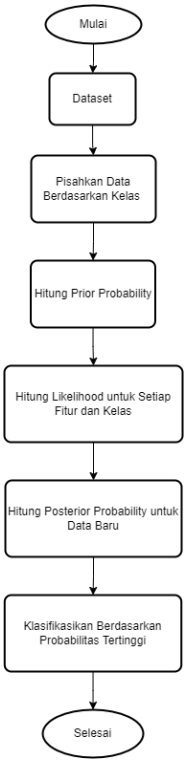
Pembobotan kata dalam penelitian ini dilakukan secara manual oleh dua penutur asli bahasa Indonesia. Proses pembobotan dilakukan dengan memberikan skor polaritas pada masing-masing kata berdasarkan valensinya, yaitu dalam rentang -5 (sangat negatif) hingga +5 (sangat positif). Pembobotan ini tidak mempertimbangkan aspek subjektivitas atau objektivitas, melainkan hanya berfokus pada polaritas (positif atau negatif). Kedua penutur asli tersebut diberikan instruksi yang sama sebelum melakukan penilaian, dan hasil dari kedua penutur dirata-ratakan untuk menghasilkan skor akhir, yang kemudian dibulatkan ke angka bulat terdekat. Validasi pembobotan dilakukan dengan menghitung tingkat

kesepakatan antara dua penilai. Untuk kata-kata negatif, tingkat kesepakatan rata-rata adalah 0,45, dan meningkat menjadi 0,97 jika kata-kata dengan skor 0 dikecualikan. Ini menunjukkan bahwa kesepakatan antara kedua penilai cukup baik, yang memperkuat validitas hasil pembobotan.

2.7 Algoritma *Naïve Bayes*

2.7.1 Teorema *Naïve Bayes*

Pada penelitian (Saritas dan Mücahid Mustafa, 2019), Algoritma *naïve bayes* adalah pengklasifikasi probabilitas sederhana yang menghitung sekumpulan probabilitas dengan menghitung frekuensi dan kombinasi nilai dalam kumpulan data yang diberikan. Algoritma ini menggunakan teorema *bayes* dan mengasumsikan bahwa semua variabel *independen* dengan mempertimbangkan nilai variabel kelas. Asumsi independensi bersyarat ini jarang berlaku dalam aplikasi dunia nyata, sehingga dikategorikan sebagai *naive*, tetapi algoritma ini cenderung belajar dengan cepat dalam berbagai masalah klasifikasi yang terkendali.



Gambar 1. Alur Algoritma Naïve Bayes

Penelitian Chen *et al*., (2021), menyatakan algoritma klasifikasi *naive bayesian* banyak digunakan dalam analisis data besar dan lainnya karena struktur algoritmanya yang sederhana dan cepat. Bertujuan untuk mengatasi kekurangan dari algoritma klasifikasi I, penelitian ini menggunakan pembobotan fitur dan kalibrasi *laplace* untuk memperbaikinya, dan mendapatkan klasifikasi *naïve bayes* yang lebih baik algoritma yang lebih baik. Melalui simulasi *numerik*, ditemukan bahwa ketika ukuran sampel besar, akurasi algoritma klasifikasi *naive bayes* yang ditingkatkan lebih dari 99%, dan sangat stabil, ketika atribut sampel kurang dari 400 dan jumlah kategori kurang dari 24, akurasi *Bayes* yang ditingkatkan algoritma klasifikasi *naïve bayes* yang ditingkatkan lebih dari 95%. Melalui penelitian empiris, ditemukan bahwa algoritma klasifikasi *naïve bayes* yang ditingkatkan dapat sangat meningkatkan akurasi tingkat analisis diskriminasi dari 49,5 menjadi 92%. Melalui analisis kekokohan, algoritma-algoritma klasifikasi *naïve bayes* yang ditingkatkan memiliki akurasi yang lebih tinggi.

Teorema *bayes* dapat dirumuskan sebagai berikut (Bustami, 2013).

$$P(H|X) = \frac{P(X|H)\,P(H)}{P(X)} \qquad (5)$$

Keterangan:

X : data dengan kelas yang belum di ketahui

H : hipotesa data X merupakan suatu kelas spesifik

P(H|X) : probabilitas hipotesis H berdasarkan kondisi X (*posterior probability*)

P(H) : probabilitas hipotesis H (prior probability)

P(X|H) : probabilitas X berdasarkan kondisi hipotesis H

P(X) : probabilitas X

Teorema *bayes* sering dikembangkan karena adanya hukum probabilitas total yang ditunjukkan oleh Persamaan 6.

$$P(H|X) = \frac{P(X|H)P(H)}{\sum_{i=1}^{n} P(H_i|X)} \qquad (6)$$

2.7.2 Penerapan Teorema Bayes dalam Klasifikasi

Untuk memahami Teorema *Naïve Bayes*, perlu dipahami bahwa proses klasifikasi membutuhkan sejumlah indikator untuk menentukan kelas yang paling sesuai dengan sampel yang dianalisis. Oleh karena itu, Teorema Bayes disesuaikan dengan kebutuhan tersebut sebagai berikut.

$$P(C|F_1, \dots, F_n) = \frac{P(F_{1,\dots,}F_n|C)\ P(C)}{P(F_{1,\dots,}F_n)} \quad (7)$$

Dimana variabel C merepresentasikan kelas, sedangkan variabel F_1 ... F_n menggambarkan karakteristik atau fitur yang digunakan untuk proses klasifikasi. Rumus tersebut menjelaskan bahwa probabilitas suatu sampel dengan karakteristik tertentu termasuk dalam kelas C (*posterior*) ditentukan oleh hasil perkalian antara probabilitas kemunculan kelas C (sebelum sampel dianalisis, dikenal sebagai *prior*) dengan probabilitas kemunculan karakteristik sampel dalam kelas C (disebut *likelihood*), kemudian dibagi dengan probabilitas kemunculan karakteristik tersebut secara keseluruhan (dikenal sebagai *evidence*). Oleh karena itu, rumus tersebut dapat ditulis secara lebih ringkas pada Persamaan 8 dan 9.

$$Posterior = \frac{prior\ x\ likehood}{evidence} \quad (8)$$

$$P(C|F_1, F_2, \dots, F_n) = \frac{P(C)x\ P(F_1, F_2, \dots, F_n|C)}{P(F_1, F_2, \dots, F_n)} \quad (9)$$

Evidence pada setiap kelas akan tetap konstan untuk suatu sampel. Selanjutnya, nilai posterior dari masing-masing kelas akan dibandingkan untuk menentukan kelas yang paling sesuai dengan sampel tersebut. Teorema Bayes kemudian diuraikan lebih lanjut menggunakan aturan perkalian pada Persamaan 10.

$$P(C|F_1, \dots, F_n) =$$
$$P(C).P(F_1|C).P(F_2|C, F_1).P(F_3|C, F_1, F_2) \dots P(F_n|C, F_1, F_2, F_3, \dots, F_{n-1}) \quad (10)$$

Dari penjabaran tersebut, diketahui bahwa semakin banyak dan kompleks faktor yang memengaruhi nilai probabilitas, sehingga tidak memungkinkan untuk dianalisis satu per satu dan membuat perhitungannya menjadi rumit. Untuk menyederhanakan perhitungan, diterapkan asumsi independensi kuat di mana setiap fitur (F_1, F_2, ..., F_n) dianggap bebas satu sama lain. Namun, untuk menangani rangkaian kata atau variabel kompleks, asumsi Markov diterapkan dengan memperhitungkan hanya variabel sebelumnya secara langsung, bukan keseluruhan rangkaian variabel. Sebagai hasil dari penerapan prinsip Markov, n-gram digunakan sebagai teknik untuk menangkap ketergantungan lokal antar kata yang berurutan dalam teks, yang sejalan dengan ide dasar Markov mengenai ketergantungan pada keadaan sebelumnya. Oleh karena itu, n-gram dalam penelitian ini dipilih sebagai representasi praktis dari model Markov yang lebih sederhana dan dapat digunakan untuk ekstraksi fitur, meskipun tidak sepenuhnya membangun model Markov probabilistik dalam arti matematis. Maka, Persamaan 10 dapat dirumuskan ulang dengan asumsi Markov sebagai:

$$P(C|F_1, \dots, F_n) = P(C).P(F_1|C).P(F_2|C, F_1).P(F_3|C, F_1, F_2) \dots P(F_n|C, \dots, F_{n-1}) \quad (11)$$

Dengan demikian, dengan mengaplikasikan asumsi independensi dan Markov, rumus akhir dari model Naïve Bayes dalam proses klasifikasi menjadi lebih sederhana dan efisien:

$$P(C|F) = P(F_1|C).P(F_2|C).P(F_3|C) \dots P(F_n|C).P(C) \quad (12)$$

Persamaan di atas merupakan model dari teorema *naïve bayes* yang digunakan dalam proses klasifikasi.

2.7.3 Pendekatan Logaritmik

Untuk menghindari masalah *underflow* yang sering terjadi akibat perkalian probabilitas yang sangat kecil, serta untuk menyederhanakan perhitungan, digunakan pendekatan *log* terhadap *posterior*. Dengan asumsi bahwa fitur-fitur

bersifat independen (sesuai asumsi *Naive Bayes*), rumus *log posterior* dapat dituliskan sebagai berikut:

$$\log P(C|X) = \log P(C) + \sum_{i=1}^{n} \log P(X_i|C) \quad (13)$$

Fitur X_i bisa diwakili oleh nilai bobot dari representasi TF-IDF (Term Frequency-Inverse Document Frequency), yang memperhitungkan pentingnya suatu kata dalam dokumen terhadap keseluruhan korpus. Oleh karena itu, kontribusi setiap fitur dalam menghitung *log posterior* akan dikalikan dengan nilai TF-IDF dari fitur tersebut. Rumus ini kemudian menjadi:

$$\log P(C|X) = \log P(C) + \sum_{i=1}^{n} \log P(X_i|C)\ x\ TF - IDF(X_i) \quad (14)$$

Setelah menghitung *log posterior* untuk masing-masing kelas, langkah selanjutnya adalah mengonversi kembali nilai tersebut ke dalam bentuk probabilitas normal. Proses ini dilakukan dengan menggunakan fungsi eksponensial (*e*). Misalkan *log posterior* untuk dua kelas, yaitu C dan D, masing-masing adala

$$logP(C|X) = \alpha \quad (15)$$

$$logP(D|X) = \beta \quad (16)$$

Untuk mendapatkan nilai probabilitas normal dari hasil *log posterior*, digunakan rumus berikut:

$$P(C|X) = \frac{e^{\alpha}}{e^{\alpha} + e^{b}} \quad (17)$$

$$P(D|X) = \frac{e^{\beta}}{e^{\alpha} + e^{\beta}} \quad (18)$$

Di mana α dan β adalah log posterior untuk kelas C dan D. Normalisasi ini memastikan total probabilitas adalah 1. Nilai eksponensial dari *log posterior* akan dibandingkan satu sama lain untuk mendapatkan probabilitas akhir. Dengan menggunakan pendekatan ini, meskipun nilai *log posterior* awalnya sangat kecil (negatif besar), hasil akhirnya akan berbentuk probabilitas yang mudah diinterpretasikan.

2.8 *Zero Frequency*

Probabilitas *zero frequency* merujuk pada situasi di mana terdapat kategori dalam variabel kategori yang tidak pernah muncul dalam data pelatihan. Dalam model *naïve bayes*, ini mengakibatkan pemberian probabilitas nol kepada kategori tersebut, sehingga kategori tersebut tidak dapat digunakan untuk melakukan prediksi. Namun, *zero frequency* dapat diatasi dengan menggunakan teknik *smoothing* (Noto dan Saputro, 2022). Teknik *smoothing* membantu mengatasi masalah ini dengan memberikan probabilitas kecil namun tidak nol kepada kategori yang jarang muncul, sehingga memungkinkan model *naïve bayes* untuk menghasilkan prediksi yang lebih baik.

2.9 Metode *Smoothing*

Dalam kumpulan data yang besar, pemilihan data pelatihan secara acak akan mengarah pada kemungkinan nilai nol dalam model probabilitas. Nilai nol ini akan menyebabkan *naïve bayes classifier* tidak dapat mengklasifikasikan input data. Oleh karena itu, diperlukan suatu metode penghalusan yang dapat menghindari nilai nol dalam model probabilitas. *Smoothing* adalah metode yang umum digunakan dalam dalam *naïve bayes classifier*. *Laplacian smoothing* adalah biasa dikenal dengan istilah *add one smoothing*, karena dalam perhitungannya, setiap variabel pada setiap parameter ditambahkan dengan 1 (Ali *et al*., 2021). Persamaan 19 adalah Persamaan *Laplacian Smoothing.*

$$P(X_c|C) = \frac{P(X_k|C)+1}{P(C)+a.|V|} \quad (19)$$

Keterangan:

$P(X_k|C)$: probabilitas suatu fitur atau kata X_k muncul dalam kelas C.

$P(X_k|C)+1$: *Laplace smoothing* menambahkan nilai 1 untuk setiap fitur atau kata X_k agar tidak ada probabilitas nol, meskipun fitur tersebut tidak muncul dalam data pelatihan.

P(C) : probabilitas kelas C atau jumlah total contoh dalam kelas C.

|V| : ukuran atau jumlah total fitur (*vocabulary*) dalam dataset. Penambahan ini adalah bagian dari normalisasi untuk menjaga probabilitas total tetap konsisten.

a : alpha = 1

Smoothing, atau proses pelancaran, adalah teknik yang digunakan untuk menciptakan fungsi pendekatan dengan tujuan menangkap pola-pola penting dalam data. Tujuan utama dari penggunaan teknik *smoothing* adalah untuk menghilangkan *noise* atau struktur berkecilan lainnya yang dapat mengaburkan pola atau tren yang signifikan dalam data. Dengan menerapkan teknik *smoothing*, data yang awalnya kasar atau memiliki fluktuasi cepat dapat dihaluskan, sehingga memudahkan identifikasi pola utama atau tren yang terkandung dalam data tersebut. Penerapan *smoothing* memiliki berbagai aplikasi dalam berbagai bidang seperti analisis data, statistik, dan pengolahan sinyal. Teknik ini membantu peneliti dan analis data untuk memahami data dengan lebih baik dan mengungkapkan pola-pola yang mungkin tersembunyi di tengah kebisingan data. *Smoothing* merupakan alat yang berguna dalam mendukung pengambilan keputusan yang lebih baik dan menghasilkan analisis yang baik untuk menghasilkan akurasi menjadi lebih akurat (Pan *et al*., 2020).

2.10 *Plugin* Atau *Extensions*

Plugin dan ekstensi adalah modul perangkat lunak yang dirancang untuk menambah atau memodifikasi fungsionalitas program utama tanpa mengubah kode sumbernya. Plugin biasanya digunakan untuk menampilkan konten khusus, seperti video atau dokumen, langsung di dalam aplikasi atau peramban web. Sementara itu, ekstensi memungkinkan modifikasi yang lebih luas pada antarmuka dan perilaku peramban, seperti penambahan toolbar, pengelolaan *cookie*, atau pemblokiran iklan. Ekstensi umumnya dikembangkan menggunakan teknologi web seperti HTML, CSS, dan JavaScript, dan didistribusikan sebagai kode sumber. Peramban modern seperti Google Chrome, Mozilla Firefox, dan Safari mendukung

penggunaan ekstensi untuk menyesuaikan pengalaman pengguna sesuai kebutuhan mereka (Teguh dkk., 2012).

2.11 *Black Box Testing*

Menurut Febiharsa *et al*., (2019), *black box testing* adalah metode pengujian yang berfokus pada fungsionalitas, yaitu menguji bagaimana perangkat lunak merespons input yang diberikan oleh pengguna untuk menghasilkan *output* yang diinginkan, tanpa memperhatikan proses internal atau kode program yang dijalankan. Dengan demikian, dapat disimpulkan bahwa *black box testing* adalah teknik pengujian perangkat lunak di mana proses internalnya tidak diketahui, sehingga pengujian diperlukan.

2.12 *White Box Testing*

Menurut Verma *et al*., (2017), *white box testing* adalah teknik pengujian perangkat lunak yang berfokus pada struktur internal aplikasi, termasuk logika, struktur kode, dan alur kontrol program. Teknik ini memerlukan pemahaman mendalam tentang kode sumber dan keterampilan pemrograman yang kuat dari penguji. *White box testing* sering disebut juga sebagai *clear box testing*, *glass box testing*, atau s*tructural testing*, dan umumnya digunakan untuk memeriksa semua jalur yang mungkin dalam kode sumber guna memastikan jalur-jalur tersebut dilalui setidaknya satu kali. Dalam white box testing terdapat tiga Teknik utama yaitu.

1. *Statement Coverage*

Statement Coverage adalah teknik pengujian yang mengukur *persentase* baris kode (*statements*) yang telah dieksekusi setidaknya satu kali selama pengujian. Tujuannya adalah untuk memastikan bahwa semua baris kode dalam program telah diuji.

Rumus Perhitungan:

$$Statement\ Coverage = \left(\frac{Jumlah\ pernyataan\ yang\ dieksekusi}{Total\ pernyataan\ dalam\ kode}\right) x\ 100\% \qquad (20)$$

2. *Branch Coverage*

Branch Coverage (juga dikenal sebagai *Decision Coverage*) mengukur *persentase* cabang logika (*true*/*false*) yang telah dieksekusi selama pengujian. Ini memastikan bahwa setiap kondisi cabang (seperti dalam pernyataan *if*, *else*, *switch*) diuji baik dalam kondisi benar maupun salah.

$$Branch\ \ Coverage = \left(\frac{Jumlah\ cabang\ yang\ dieksekusi}{Total\ cabang\ logika}\right) x\ 100\% \qquad (21)$$

3. *Path Coverage*

Path Coverage mengukur persentase semua jalur eksekusi yang mungkin diambil dalam program selama pengujian. Ini adalah metode yang lebih komprehensif karena memastikan bahwa setiap kombinasi jalur logika diuji.

$$Path\ Coverage = \left(\frac{Jumlah\ jalur\ yang\ diuji}{Total\ jalur\ eksekusi\ yang\ mungkin}\right) x\ 100\% \qquad (22)$$

2.13 Evaluasi

Untuk menilai kinerja sebuah algoritma, dapat digunakan *confusion matrix* yang memiliki empat istilah untuk merepresentasikan hasil klasifikasi, yaitu *true positive* (TP), *false positive* (FP), *true negative* (TN), dan *false negative* (FN). Nilai TP menunjukkan jumlah data positif yang berhasil diklasifikasikan dengan benar, sementara FP menunjukkan jumlah data positif yang diklasifikasikan secara salah. Di sisi lain, TN mengacu pada jumlah data negatif yang diklasifikasikan dengan benar, sedangkan FN adalah jumlah data negatif yang diklasifikasikan secara keliru. Dalam klasifikasi biner yang hanya memiliki dua kelas keluaran, *confusion matrix* dapat disajikan seperti pada tabel 2.

Tabel 2. *Confusion Matrix*

Predict / ***Actual***	Positif	Negatif
Positif	TP (*True Positive)*	FN (*False Negative)*
Negatif	FP (*False Positive)*	TN (*True Negative)*

Evaluasi dilakukan dengan mengukur performa terbaik berdasarkan akurasi, presisi, *recall*, dan *f1-score* (Ariska dan Kamayani, 2024). Performa ini dapat dihitung menggunakan Persamaan sebagai berikut.

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (23)$$

Akurasi adalah rasio prediksi yang benar (baik prediksi benar positif maupun benar negatif) terhadap keseluruhan prediksi. Dengan kata lain, akurasi menghitung persentase prediksi yang benar dari seluruh kasus.

a. TP (*True Positive*): Jumlah data positif yang diprediksi dengan benar sebagai positif.
b. TN (*True Negative*): Jumlah data negatif yang diprediksi dengan benar sebagai negatif.
c. FP (*False Positive*): Jumlah data negatif yang diprediksi secara salah sebagai positif.
d. FN (*False Negative*): Jumlah data positif yang diprediksi secara salah sebagai negatif.

$$Precision = \frac{TP}{(TP+FP)} \quad (24)$$

Presisi mengukur proporsi prediksi positif yang benar-benar positif. Artinya, dari semua prediksi yang dinyatakan positif, seberapa banyak yang benar. Presisi tinggi berarti sedikit prediksi yang salah (*false positives*) dari seluruh prediksi positif.

$$Recall = \frac{TP}{(TP+FN)} \quad (25)$$

Recall mengukur proporsi data positif yang terprediksi dengan benar. Artinya, dari semua kasus positif yang sebenarnya, seberapa banyak yang bisa terprediksi dengan benar. *Recall* tinggi menunjukkan bahwa model mampu menemukan hampir semua instance positif (mengurangi FN).

$$f1 - score = 2x \frac{(recal\ x\ presisi)}{(presisi+recal)} \quad (26)$$

F1-Score adalah rata-rata harmonis antara presisi dan *recall*. *F1-Score* digunakan ketika menginginkan keseimbangan antara presisi dan *recall*, terutama ketika ada ketidakseimbangan antara kelas positif dan negatif. Ini adalah metrik yang baik untuk digunakan jika *false positives* dan *false negatives* memiliki dampak yang berbeda pada masalah yang diselesaikan.

III. METODOLOGI PENELITIAN

3.1 Tempat dan Waktu Penelitian

3.1.1 Tempat Penelitian

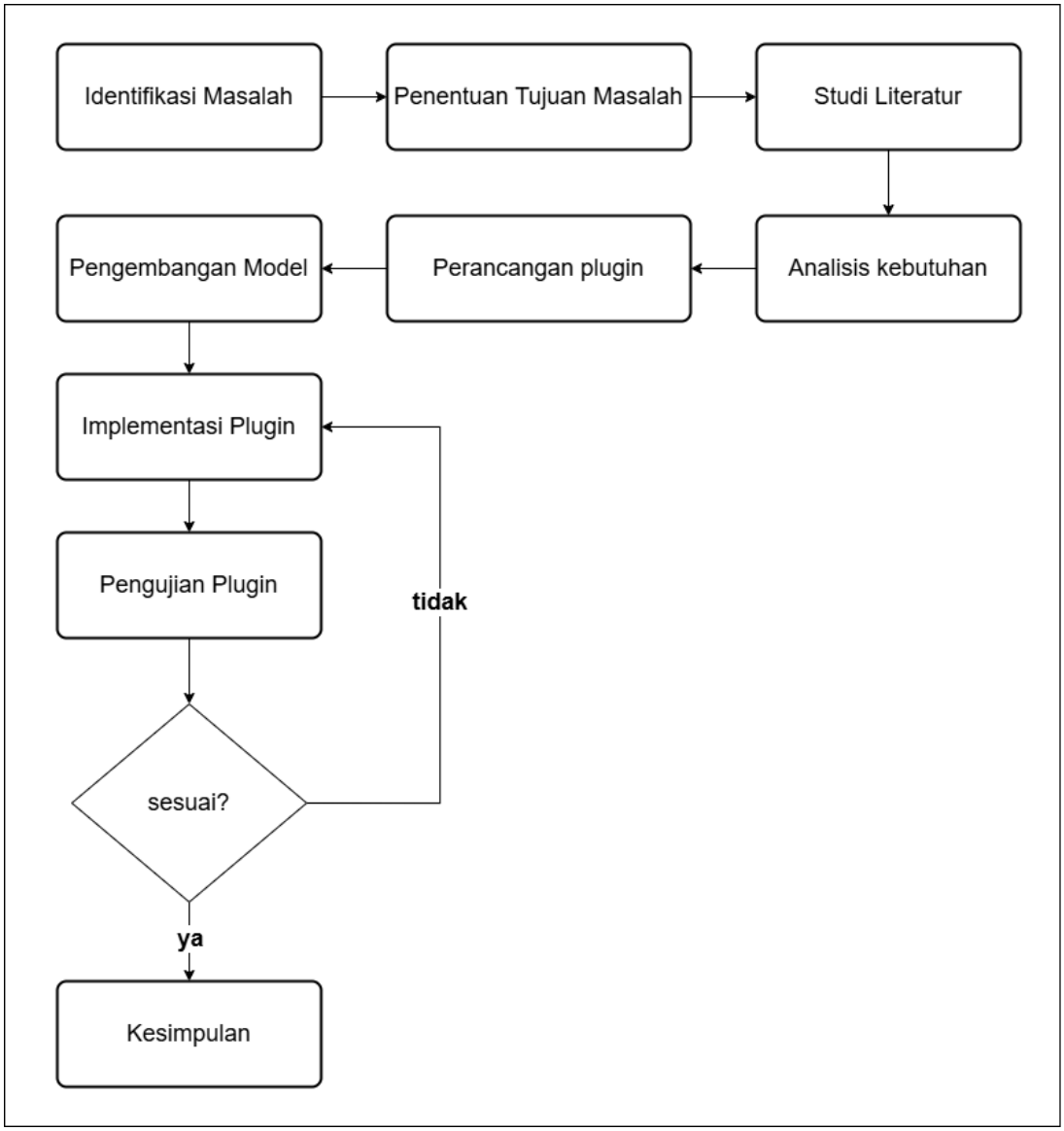
Penelitian ini dilakukan di Laboratorium Rekayasa Perangkat Lunak (RPL) di Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung. Tempat pelaksanaan berada di Jalan Prof. Dr. Soemantri Brojonegoro No. 1, Gedung Meneng, Bandar Lampung.

3.1.2 Waktu Penelitian

Penelitian ini berlangsung pada semester ganjil mulai dari bulan Oktober 2024 hingga penyelesaian pada bulan Januari 2025 dalam tahun ajaran 2024/2025. Proses pengerjaannya dibagi menjadi tiga tahap, dimulai dengan tahap pertama yang fokus pada pengumpulan dan pemahaman terhadap studi literatur. Selanjutnya, dilakukan penyusunan draf dan pengumpulan data terkait pada tahap kedua. Tahap terakhir melibatkan penyusunan hasil pengujian dan analisis program dalam draf laporan serta penyampaian hasil penelitian.

3.2 Alur Penelitian

Secara keseluruhan, rangkaian alur penelitian ini dapat diilustrasikan melalui diagram penelitian yang terlihat dalam Gambar 2.



Gambar 2. Alur Penelitian

Alur penelitian dimulai dengan identifikasi masalah, di mana peneliti menentukan masalah utama yang memerlukan solusi atau pengembangan lebih lanjut. Setelah itu, dilakukan penentuan tujuan masalah untuk menetapkan hasil yang ingin dicapai dari penelitian tersebut. langkah berikutnya adalah studi literatur, di mana peneliti mengumpulkan informasi dari berbagai sumber atau penelitian sebelumnya yang relevan. Setelah memahami konteks dan kebutuhan dari masalah yang sedang diteliti, peneliti melakukan analisis kebutuhan, yaitu menentukan spesifikasi yang dibutuhkan untuk mengembangkan solusi. Tahap berikutnya adalah perancangan *plugin*, di mana desain awal *plugin* dirumuskan untuk memenuhi kebutuhan yang telah dianalisis. Setelah desain dirumuskan, peneliti berlanjut ke pengembangan

model untuk menghasilkan *plugin* berdasarkan desain yang telah dibuat. *Plugin* yang telah dikembangkan kemudian diimplementasikan dalam tahap implementasi *plugin*, di mana *plugin* mulai diterapkan atau diintegrasikan ke dalam sistem.

Selanjutnya, dilakukan pengujian *plugin* untuk memastikan bahwa *plugin* berfungsi sesuai dengan yang diharapkan. Jika hasil pengujian menunjukkan bahwa *plugin* belum sesuai, dilakukan perbaikan dan pengujian ulang hingga mencapai hasil yang diinginkan. Setelah *plugin* dinyatakan sesuai, penelitian diakhiri dengan kesimpulan, yang merangkum hasil akhir serta memberikan rekomendasi untuk penelitian atau pengembangan lebih lanjut.

3.3 Identifikasi Masalah

Deteksi *hate speech* menjadi semakin penting karena metode berbasis aturan tradisional tidak mampu menangani volume data yang sangat besar dan beragam yang dihasilkan pengguna. Oleh karena itu, diperlukan pendekatan yang lebih efisien dan akurat, seperti algoritma *machine learning*, untuk mengotomatisasi proses deteksi *hate speech* di kolom komentar TikTok.

3.4 Penentuan Tujuan Masalah

Tujuan utama dari penelitian ini adalah untuk menerapkan algoritma *naïve bayes* dengan *smoothing* dalam mendeteksi *hate speech* pada kolom komentar di TikTok. Selain itu, penelitian ini bertujuan untuk mengukur akurasi algoritma tersebut dalam mengidentifikasi *hate speech*. Dengan pengembangan perangkat lunak yang menggunakan kombinasi algoritma *naïve bayes* dan metode *smoothing*, diharapkan deteksi *hate speech* dapat dilakukan secara efektif dan akurat, serta memberikan kontribusi dalam menciptakan lingkungan media sosial yang lebih aman dan bersih dari ujaran kebencian.

3.5 Studi Literatur

Untuk memahami lebih lanjut tentang masalah dan konteksnya, peneliti melakukan studi literatur. Ini melibatkan pencarian dan analisis informasi yang relevan dari penelitian-penelitian sebelumnya untuk mendukung pemahaman dan arah penelitian yang akan dilakukan.

3.6 Analisis Kebutuhan *Plugin*

Adanya *plugin* ini sangat dibutuhkan untuk memenuhi kebutuhan fungsional dan *non*-fungsional, serta untuk mendukung pencapaian tujuan utama penelitian, yaitu membuat perangkat lunak yang dapat mendeteksi *hate speech* di kolom komentar TikTok dari penerapan algoritma *naive bayes* dengan *smoothing*. Dan *plugin* yang langsung terintegrasikan di web TikTok ini juga yang membuat penelitian ini berbeda dengan penelitian-penelitian yang sudah ada.

3.6.1 Kebutuhan Fungsional

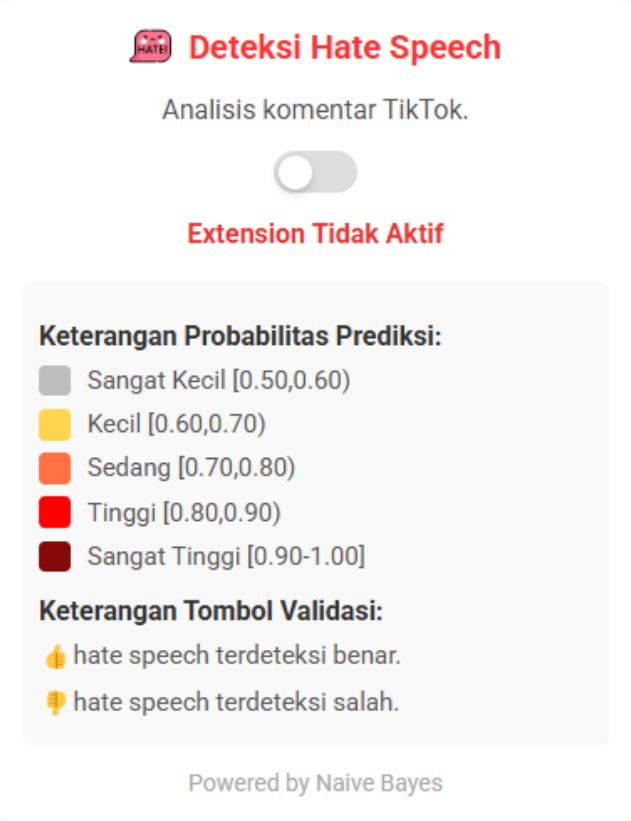
Kebutuhan fungsional, *plugin* ini memungkinkan proses deteksi *hate speech* secara langsung di *platform* TikTok, menjadikannya alat yang efisien dan praktis tanpa perlu proses manual atau aplikasi tambahan. Dengan menerapkan algoritma *naïve bayes* dengan *smoothing*, *plugin* ini dapat meningkatkan ketepatan deteksi, terutama di kolom komentar TikTok yang menggunakan bahasa tidak baku atau penuh variasi. Implementasi ini juga memberi peluang untuk mengukur akurasi algoritma dalam lingkungan nyata, sehingga mendukung tujuan penelitian secara langsung. *Plugin* diharapkan dapat menampilkan hasil deteksi dengan memberikan efek pewarnaan pada setiap komentar apakah termasuk ujaran kebencian atau tidak.

3.6.2 Kebutuhan Non Fungsional

Kebutuhan non-fungsional, *plugin* ini penting agar pengguna dari berbagai latar belakang dapat menggunakannya dengan mudah, tanpa perlu pengetahuan teknis. Hal ini dicapai melalui antarmuka yang sederhana dan intuitif. Selain itu, plugin dirancang agar kompatibel dengan versi terbaru Google Chrome dan tidak mengalami konflik dengan *plugin* lain, sehingga mendukung pengalaman pengguna yang stabil dan nyaman, sekaligus memastikan proses penelitian berjalan sesuai harapan tanpa gangguan.

3.7 Perancangan *Plugin*

Plugin Google Chrome merupakan komponen perangkat lunak yang memperluas fungsionalitas browser Chrome, memungkinkan pengguna menambahkan fitur tambahan sesuai kebutuhan. Dengan menggunakan *plugin* atau ekstensi, pengguna dapat menyesuaikan pengalaman browsing untuk berbagai keperluan pada penelitian ini digunakan untuk mendeteksi *hate speech* pada kolom komentar TikTok. Pada tahap perancangan *plugin*, dilakukan perancangan secara menyeluruh mengenai bagaimana *plugin* akan bekerja, termasuk arsitektur, komponen utama, dan mekanisme deteksi *hate speech* di kolom komentar TikTok.



Gambar 3. UI *Plugin* Deteksi *Hate Speech*

3.7.1 Fungsi Utama *Plugin*

Plugin ini memiliki fungsi utama untuk mendeteksi *hate speech* pada komentar yang muncul di konten TikTok web pada Google Chrome yang sudah dibuka. *Hate speech* ini akan ditandai secara visual, misalnya dengan memberikan efek pewarnaan pada komentar yang terdeteksi sebagai *hate speech*.

Tabel 3. Skenario *Plugin*

Aksi Aktor	Reaksi *Plugin*
Skenario normal	
User menginstal *plugin* deteksi *hate speech*.	
	Plugin terinstal.
User mengaktifkan deteksi *hate speech* di halaman komentar di suatu konten Tiktok.	
	Plugin aktif dan mendeteksi kolom komentar TikTok yang mengandung *hate speech*.
	Plugin memberikan warna dan tombol *like* serta *dislike* pada komentar yang mengandung *hate speech*.
User menonaktifkan *plugin*.	*Plugin* dinonaktifkan dan *merefresh* Web TikTok.

3.7.2 Arsitektur Sistem

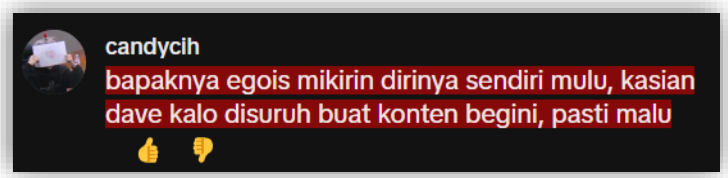
Arsitektur *plugin* ini terdiri dari beberapa komponen utama:

a. *Content Script*: *Script* ini berfungsi untuk berinteraksi langsung dengan halaman TikTok yang terbuka di browser. *Content script* akan mengekstrak semua komentar yang ada di halaman TikTok.
b. *Background Script*: *Script* ini berjalan di belakang dan menangani komunikasi antara *content script* dan model deteksi *hate speech* yang diintegrasikan ke dalam ekstensi. Ekstensi ini adalah sebuah *plugin* atau *add-on* yang berjalan di dalam peramban (browser) Google Chrome, yang dibuat untuk mendeteksi *hate speech* di kolom komentar, dalam hal ini di TikTok. Ekstensi ini bekerja dengan memantau dan memproses komentar, mendeteksi adanya kata atau frasa yang dianggap *hate speech* menggunakan model *naïve bayes* dengan *smoothing* yang telah diimplementasikan.

c. *Popup* (UI): Ini adalah antarmuka pengguna sederhana yang menampilkan status ekstensi, opsi pengaturan sensitivitas, dan laporan singkat mengenai komentar yang telah dianalisis.
d. *Machine learning model*: Model *naïve bayes* dengan *smoothing*, yang sudah dilatih untuk mendeteksi *hate speech*, akan diterapkan di sini. Model ini bisa disimpan dalam bentuk file JavaScript (misalnya dengan TensorFlow.js atau diimplementasikan dengan algoritma *custom* dalam JavaScript).

3.7.3 Mekanisme Deteksi

a. Ekstraksi Komentar: *Plugin* perlu merancang cara untuk menangkap data komentar dari halaman TikTok saat pengguna membuka konten video. Data ini bisa berupa teks komentar yang akan diekstrak oleh *content script*.
b. Proses Deteksi: Setelah komentar diekstrak, *plugin* akan mengirim data tersebut ke model deteksi yang sudah ditanamkan di dalam *extension*. Jika ada komentar yang terdeteksi sebagai *hate speech*, maka *plugin* akan memberikan tanda pada komentar tersebut.
c. Tindakan Visualisasi: Komentar yang terdeteksi mengandung *hate speech* dapat ditandai dengan efek pewarnaan atau cara visual lainnya untuk memberikan peringatan kepada pengguna.



Gambar 4. Visualisasi Efek Pewarnaan Komentar

3.7.4 Perangkat Penelitian

Berdasarkan hasil identifikasi masalah yang memerlukannya *machine learning* untuk pendeteksian *hate speech,* studi literatur mengenai penerapan *machine leraning* dalam deteksi *hate speech*, dan analisis kebutuhan untuk pembuatan *plugin* deteksi *hate speech* serta pengembangan *plugin* untuk mendeteksi *hate speech*

maka penelitian ini membutuhkan perangkat penelitian yang mencakup perangkat lunak untuk pengembangan model deteksi *hate speech* dengan algoritma *naïve bayes* dan *plugin* serta perangkat keras untuk mendukung pengembangan model secara optimal dan *plugin*.

3.7.4.1 Perangkat lunak

Adapun perangkat lunak yang digunakan dalam penelitian ini adalah:

1. Sistem Operasi Windows 11 *Home 64-bit*

   Sistem operasi ini digunakan sebagai platform utama untuk menjalankan semua perangkat lunak dan alat yang dibutuhkan selama penelitian. Windows 11 memberikan lingkungan yang stabil dan kompatibel untuk menjalankan berbagai program seperti *Visual Studio Code*, Python, serta alat-alat lain yang mendukung pengembangan *plugin* dan model machine learning.
2. *Visual Studio Code*

   Visual Studio Code digunakan sebagai *Integrated Development Environment* (IDE) untuk pengembangan *plugin* dan *scrapping* data. *Visual Studio Code* menawarkan fitur seperti *debugging*, integrasi dengan berbagai *plugin*, serta dukungan multi-bahasa yang memudahkan dalam proses pengkodean dan pengujian program. Dalam konteks penelitian ini, *Visual Studio Code* digunakan untuk mengembangkan *plugin* deteksi *hate speech* serta melakukan *web scraping* untuk mengambil data dari TikTok.
3. Google Colabs

   Google Colab digunakan sebagai platform berbasis *cloud* untuk pembuatan dan pelatihan model *machine learning*. Dalam penelitian ini, Google Colab memfasilitasi eksperimen dengan *naïve bayes* dan *smoothing* menggunakan GPU atau TPU yang tersedia di *cloud* tanpa perlu menggunakan sumber daya lokal. Google Colab juga mendukung kolaborasi, memungkinkan berbagi *notebook* dengan peneliti lain.
4. Bahasa Pemrograman *Python*

   Python merupakan bahasa pemrograman yang dipilih untuk implementasi model *Naive Bayes*, pengolahan data, serta pembuatan *plugin* deteksi *hate speech*

digunakan karena kompatibel dengan berbagai pustaka penting untuk *machine learning*, pandas untuk analisis data, dan *libraries* lainnya yang mendukung pengembangan algoritma dan *plugin*.

5. Bahasa Pemrograman Javascript

   Javascript digunakan dalam pengembangan antarmuka pengguna dan komponen interaktif dari *plugin* yang diimplementasikan untuk mendeteksi *hate speech* di platform TikTok. Bahasa ini memungkinkan manipulasi elemen-elemen pada halaman web, seperti menampilkan hasil deteksi *hate speech* secara *real-time*.

6. Bahasa Pemrograman HTML

   HTML (*Hypertext Markup Language*) adalah bahasa yang digunakan untuk membangun struktur dasar dari antarmuka pengguna yang terlihat dalam *plugin*. HTML memastikan bahwa konten dan elemen visual *plugin* seperti form *input*, tombol, dan teks ditampilkan dengan baik di dalam halaman web. Dalam konteks penelitian ini, HTML digunakan untuk menyusun tata letak dan elemen visual dari *plugin* yang akan diintegrasikan dengan *website* TikTok, sehingga pengguna dapat dengan mudah berinteraksi dengan *plugin* tersebut.

7. FastAPI adalah kerangka kerja (*framework*) web modern berbasis Python yang dirancang untuk membangun API dengan performa tinggi dan efisiensi yang tinggi. FastAPI menggunakan tipe anotasi Python untuk validasi data otomatis dan menghasilkan dokumentasi API secara otomatis. Dibangun di atas Starlette untuk performa web dan Pydantic untuk validasi data. FastAPI sangat cocok untuk aplikasi modern seperti machine learning, microservices, atau aplikasi yang membutuhkan API cepat dengan validasi data yang kuat.

3.7.4.2 Perangkat keras

Untuk dapat menjalankan dan menggunakan perangkat lunak dalam pengembangan model deteksi *hate speech* dan pengembangan *plugin* di penelitian ini maka diperlukannya perangkat keras untuk menunjang perangkat lunak yang akan digunakan, dalam penelitian ini digunakannya sebuah laptop dengan detail sebagai berikut.

1. Merk : Asus
2. Tipe : ROG Strix
3. Model : G512LI
4. CPU : Intel® Core ™ i5-10300H
5. GPU : NVIDIA GeForce GTX 1650 Ti
6. Penyimpanan : SSD 512GB VISIPRO M.2 2280 NVME

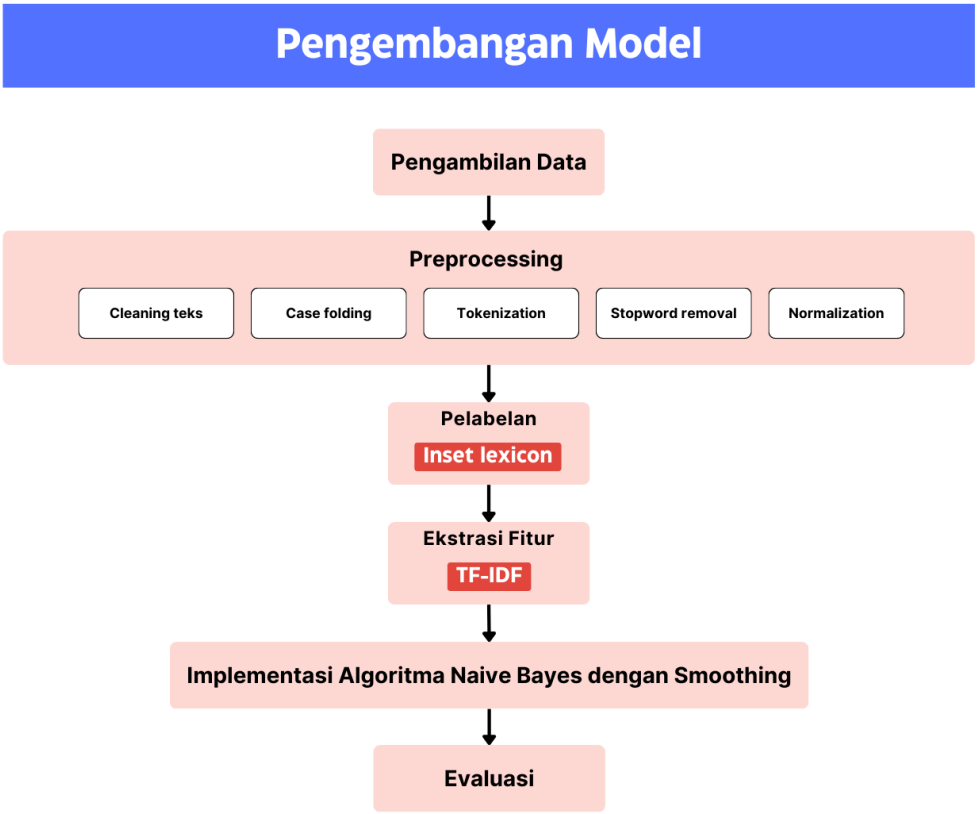
Spesifikasi minimal perangkat keras untuk *plugin* selain perangkat keras yang digunakan dalam pengembangan, *plugin* deteksi *hate speech* pada kolom komentar TikTok memiliki spesifikasi minimal perangkat keras sebagai berikut untuk dapat *diinstal* dan dijalankan pada Google Chrome:

1. Prosesor: Minimal dual-core dengan kecepatan 1.6 GHz atau lebih tinggi.
2. RAM: Minimal 4 GB, direkomendasikan 8 GB untuk performa lebih optimal.
3. Penyimpanan: Minimal 200 MB ruang kosong.
4. Sistem Operasi: Windows 7 atau lebih baru, macOS 10.13 atau lebih baru, Linux distribusi modern, atau ChromeOS.
5. Browser: Google Chrome versi 96 atau lebih baru.

Dengan spesifikasi tersebut, plugin dapat berjalan dengan baik sesuai kebutuhan pengembangan dan implementasi pada penelitian ini.

3.8 Tahap Pengembangan Model

Tahapan yang akan dilakukan pada pengembangan model dipenelitian ini antara lain dimulai dari pengumpulan *dataset*, *preprocessing dataset*, pelabelan, ekstrasi fitur, implementasi algoritma *naïve bayes* dengan *smoothing,* dan evaluasi. Alur dari tahapan yang akan dilakukan pada penelitian ini diilustrsikan pada Gambar 3.



Gambar 5. Alur Pengembangan Model

3.8.1 Pengambilan Data

Dataset yang digunakan dalam penelitian ini berupa kumpulan *comment* pada kolom komentar media sosial Tiktok dengan durasi pengambilan data Oktober hingga November 2024 pada beberapa konten akun yang ada di TikTok. Data komentar di kolom tiktok akan dipisahkan menjadi data latih, dan data pengujian. Data ini dikumpulkan dari konten yang terdapat pada Tiktok.

Pengambilan data atau *scrapping* data komentar pada aplikasi TikTok dilakukan dengan menggunakan *python* mendapatkan data sebanyak 17.710 komentar. Proses untuk mendapatkan *dataset*, dengan cara memanfaatkan TikTok API (*Application Programming Interface*) untuk memperoleh komentar dari video tertentu. Proses ini dimulai dengan mengirimkan request HTTP menggunakan pustaka *requests* ke *endpoint* API TikTok yang ditentukan berdasarkan ID video. Komentar yang berhasil diambil akan difilter agar hanya menyertakan informasi yang relevan.

Seluruh hasil yang diperoleh disimpan dalam format yang terstruktur untuk memudahkan analisis lebih lanjut.

3.8.2 *Preprocessing*

Tahapan ini bertujuan untuk mempermudah mesin dalam membaca dan mengelola *dataset*. Pada tahap ini, *dataset* yang masih mengandung banyak karakter dan tanda baca yang tidak relevan akan diproses. Langkah-langkah *preprocessing* pada penelitian ini adalah sebagai berikut:

3.8.2.1 *Cleaning*

Proses ini meliputi penghapusan tanda baca (tanda titik (.), koma (,), tanda seru (!), tanda tanya (?), kutipan (“ “), tanda kurung (( ) [ ] { }), tanda hubung (- _), tanda atau (/)), angka (seluruh angka), *emotikon* (seluruh *emotikon*), karakter asing (@ # $ % ^ & *), spasi ganda ( ), dan elemen-elemen *noisy* lainnya (mention social media (@username), hashtag). Penghapusan elemen-elemen tersebut dilakukan untuk memfokuskan analisis pada teks utama dan menghindari gangguan dari elemen *non*-teks yang tidak relevan secara linguistik, karena tujuan utama adalah untuk menganalisis teks secara linguistik, yaitu mempelajari aspek-aspek bahasa seperti kata, frasa, kalimat, dan struktur tata bahasa. Selain itu, *emotikon*, tanda baca, dan karakter khusus sering kali menjadi *noise* dalam data dan menambah variasi yang tidak signifikan bagi model *machine learning*. *Noise* adalah elemen-elemen yang tidak relevan yang dapat mengganggu atau merusak pemahaman model terhadap pola-pola penting dalam teks. Ketika elemen-elemen seperti *emotikon* dan tanda baca ini dipertahankan, mereka dapat menambah variasi yang tidak diperlukan pada data, sehingga memperbesar kompleksitas yang tidak memberi informasi penting. Ini dapat membingungkan model saat belajar mengenali pola sebenarnya. Makna *emotikon* juga bisa ambigu dan bervariasi tergantung pada konteks. Ketika mesin melakukan interpretasi, ia mencoba memahami dan mengkategorikan makna teks sesuai dengan pola dan aturan yang telah dipelajari. Namun, karena *emotikon* tidak memiliki aturan yang konsisten dan

bergantung pada persepsi individu, hal ini bisa menimbulkan kesalahan interpretasi oleh mesin. Dengan menghapus elemen-elemen ini, proses tokenisasi dan analisis teks menjadi lebih sederhana, sehingga menghasilkan model yang lebih akurat dan efisien. Tujuan dari langkah ini adalah untuk mempermudah data dalam tahap pemrosesan selanjutnya.

Tabel 4. *Cleaning* Data

Sebelum *Cleaning*	**Sesudah *Cleaning***
'halo koh apa kabar hari ini 😁'	halo koh apa kabar hari ini
'kok?!! di PASANG LAGI?!'	kok di PASANG LAGI
"Wow!!! 😍 Saya sangat suka video ini!!! 👍👍👍 Ini keren banget #awesome #video2024 😎. Apakah Anda setuju? 🧐"	Wow Saya sangat suka video ini Ini keren banget awesome video Apakah Anda setuju

Dalam fungsi clean_text, terdapat dua *library* yang digunakan, yaitu re (*Regular Expressions*), dan *string*. Pada fungsi clean_text beberapa elemen yang tidak relevan dalam teks dihapus untuk membersihkan data agar lebih mudah diproses dalam analisis. Berikut adalah penjelasan elemen-elemen yang dihapus dan bagaimana pemanggilannya:

1. **Menghapus tanda baca**:

a) Pemanggilan:

```
text = re.sub(r'[{}]'.format(string.punctuation), ' ', text)
```

b) Penjelasan:

Fungsi ini menggunakan string.punctuation untuk merepresentasikan semua tanda baca, seperti titik, koma, dan tanda seru. Kemudian, re.sub() digunakan untuk mengganti setiap tanda baca dengan spasi.

2. Menghapus simbol non-alfanumerik:

a) Pemanggilan:

```
text = re.sub(r'[^\w\s]', '', text)
```

b) Penjelasan:

Di sini, \w berarti karakter alfanumerik (huruf, angka, dan underscore), dan \s berarti spasi. Pola ini menghapus simbol atau karakter selain huruf, angka, dan spasi.

3. Menghapus angka:

a) Pemanggilan:

```
text = re.sub(r'[0-9]+', '', text)
```

b) Penjelasan:

Pola ini mencari dan menghapus angka yang ada dalam teks dengan r'[0-9]+', sehingga teks hanya berisi huruf dan spasi.

4. Menghapus emotikon:

a) Pemanggilan:

```
emoticon_pattern = re.compile("["
u"\U0001F600-\U0001F64F"  # emotikon wajah
u"\U0001F300-\U0001F5FF"  # simbol & ikon
u"\U0001F680-\U0001F6FF"  # transportasi & simbol
u"\U0001F1E0-\U0001F1FF"  # bendera
"]+", flags=re.UNICODE)
text = emoticon_pattern.sub(r'', text)
```

b) Penjelasan:

Menggunakan pola *Unicode* untuk mendeteksi emotikon, simbol, ikon, bendera, dan menggantinya dengan string kosong, menghapus elemen-elemen tersebut dari teks.

5. Menghapus karakter asing dan elemen *noisy*:

a) Pemanggilan:

```
text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
```

b) Penjelasan:

Pola ini menghapus semua karakter yang bukan huruf (a-z, A-Z), angka (0-9), atau spasi. Ini berfungsi untuk membuang karakter-karakter yang dianggap "noisy" seperti simbol dan karakter asing yang tidak relevan untuk analisis.

6. Menghapus spasi ganda:

a) Pemanggilan:

```
text = re.sub(r'\s+', ' ', text).strip()
```

b) Penjelasan:

Pola \s+ digunakan untuk mencari spasi ganda atau lebih dan menggantinya dengan satu spasi tunggal. strip() digunakan untuk menghapus spasi yang tersisa di awal atau akhir teks.

3.8.2.2 *Case folding*

Kata atau kalimat yang awalnya menggunakan huruf besar (*uppercase*) akan diubah menjadi huruf kecil (*lowercase*).

Tabel 5. *Case Folding*

Sebelum *Case Folding*	***Case Folding***
'tapi CCTV nya di cabut'	tapi cctv nya di cabut
'DIPASANG LAGI BAH'	dipasang lagi bah
'kok di PASANG LAGI'	kok di pasang lagi

Fungsi case_folding tidak memerlukan library eksternal untuk diimplementasikan. Fungsi ini hanya menggunakan metode bawaan Python, yaitu:

1. Fungsi *Case Folding*:

a) Pemanggilan:

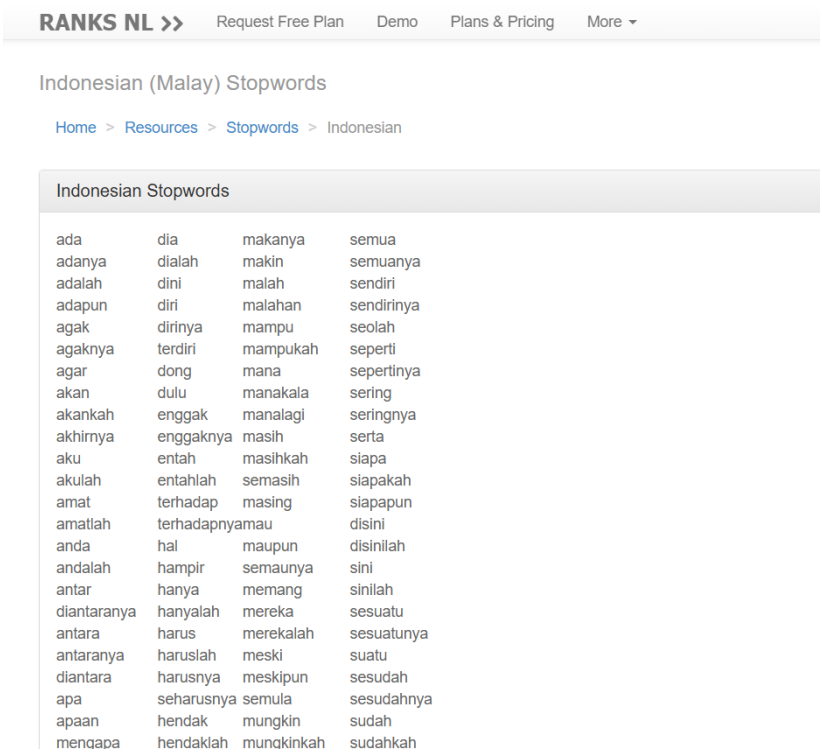
```
return text.lower()
```

b) Penjelasan:

lower():Merupakan metode bawaan dari tipe *data string* di Python. Fungsinya adalah untuk mengonversi semua karakter dalam teks menjadi huruf kecil.

3.8.2.3 *Stopwords removal*

Menghilangkan kata-kata umum yang tidak diperlukan dalam analisis teks, sehingga hasil analisis lebih fokus pada kata-kata penting.



Gambar 6. List Daftar *Stopword Removal*

Untuk melakukan *stopwords removal*, penelitian ini menggunakan *library* dari *https://www.ranks.nl/stopwords/indonesian* yang menyediakan berbagai bahasa, termasuk bahasa Indonesia. Langkah-langkah pemrosesan dalam fungsi *stopwords removal* adalah sebagai berikut:

1. **Fungsi *Stopwords Removal*:**

```
def stopwords_removal(tokens):
    # ranks.nl stopwords indonesian list
    stop_words = set([
        "ada", "adanya", ...,"yang" ])
    # Filter tokens
    filtered_tokens = [word for word in tokens if word not in
stop_words]
    return filtered_tokens
```

2. **Penjelasan:**

a) Fungsi ini menggunakan daftar *stopwords* dalam bahasa Indonesia, yang disimpan dalam bentuk set untuk mempercepat pencarian.

b) Memanfaatkan *list comprehension*, fungsi ini memfilter setiap kata dalam daftar tokens dan hanya mempertahankan kata-kata yang tidak termasuk dalam daftar *stopwords*. Hasil akhirnya adalah daftar kata-kata yang lebih relevan untuk analisis teks.

3.8.2.4 *Normalization*

Tahapan ini bertujuan untuk mengubah kata-kata singkat atau tidak baku kembali ke bentuk kata dasar. Proses ini sering diterapkan pada teks komentar untuk memastikan konsistensi penggunaan kata.

Tabel 6. *Normalization*

Sebelum *Normalization*	**Sesudah *Normalization***
mau marah tapi gmn	mau marah tapi bagaimana
klo jalan knp ga brngkt sndiri aja	kalau jalan kenapa tidak berangkat sendiri saja
gak tau terimaksi	tidak tahu terima kasih

Kamus normalisasi merupakan sebuah *dictionary* (normalization_dict) di Python yang berisi pasangan kata yang sering digunakan dalam bentuk informal/slang (misalnya "gak") dan kata yang lebih formal atau sesuai dengan standar bahasa (misalnya "tidak"). Tujuannya adalah untuk mengganti kata-kata slang atau variasi bahasa informal dengan kata yang lebih formal agar teks lebih konsisten dan mudah dipahami oleh model analitik. Pada penelitian ini menggunakan *dictionary* yang dibuat oleh peneliti yang disimpan pada GitHub.

1. **Fungsi untuk normalisasi :**

```
def normalize_text(text):words = text.split()
    normalized_words = [normalization_dict.get(word, word) for word in
    words]
    return ' '.join(normalized_words)
```

2. Penjelasan Fungsi:

A. Memisahkan kata: text.split():

a) Metode ini digunakan untuk memisahkan teks menjadi kata-kata individual berdasarkan spasi. Setiap kata akan disimpan sebagai elemen dari sebuah list.

b) Misalnya, "saya gak bisa" akan diubah menjadi ["saya", "gak", "bisa"].

B. Proses Normalisasi:

a) Fungsi ini memeriksa setiap kata yang dipisahkan dalam teks. Dengan menggunakan normalization_dict.get(word, word), fungsi akan memeriksa apakah kata tersebut ada di dalam kamus normalisasi yang telah dibuat oleh peneliti secara otomatis dengan fungsi yang sudah dibuat.

b) Jika kata ada dalam kamus, maka kata tersebut akan diganti dengan kata yang lebih formal (misalnya, "gak" menjadi "tidak").

c) Jika kata tidak ada di dalam kamus, maka kata tersebut dibiarkan seperti aslinya.

C. Menggabung kata: ' '.join(normalized_words):

Setelah semua kata dinormalisasi, fungsi ini akan menggabungkan kembali kata kata tersebut menjadi sebuah string, dipisahkan oleh spasi.

3.8.2.5 *Tokenizing*

Proses ini dilakukan dengan memecah kalimat menjadi kata-kata individual. Tujuan dari tokenisasi adalah untuk mempermudah proses pengolahan data pada tahap-tahap berikutnya.

Tabel 7. *Tokenizing*

Sebelum *Tokenizing*	**Sesudah *Tokenizing***
apa di pasang lagi	[apa, di, pasang, lagi]
mungkin ini vidio lama	[mungkin, ini, vidio, lama]
ini teknik marketing	[ini, teknik, marketing]

Salah satu cara paling sederhana untuk melakukan *tokenizing* adalah dengan menggunakan metode split() dari Python.

1. **Implementasi *Tokenizing***

```
def tokenize_text(text):
    return text.split()
```

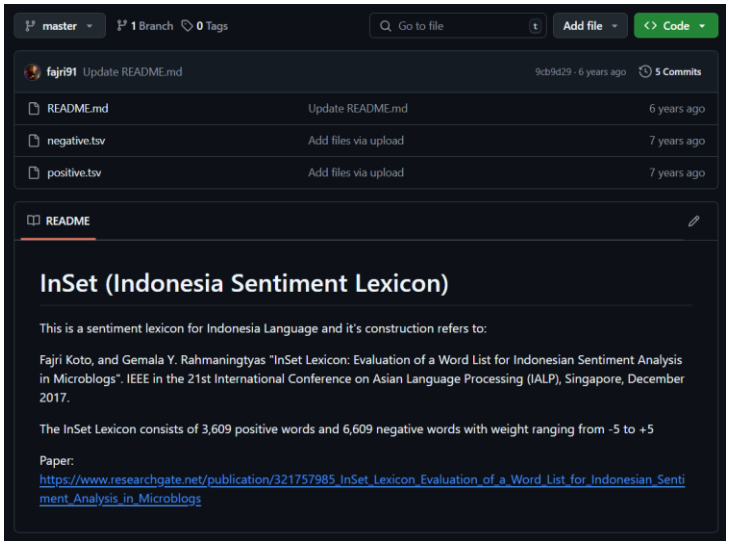
2. **Penjelasan:**

a) Metode ini digunakan untuk memecah teks berdasarkan spasi.

b) Setiap kata dalam teks akan diubah menjadi elemen dari sebuah list, sehingga teks akan terbagi menjadi token-token individual.

3.8.3 Pelabelan

Penelitian ini menerapkan pelabelan dengan pendekatan berbasis leksikon yang diadaptasi dari penelitian (Koto dan Rahmaningtyas, 2017) pada *repositori* [INSET GitHub](https://github.com/fajri91/InSet). Teknik ini menganalisis sentiment menggunakan korpus yang telah dilengkapi dengan bobot atau skor polaritas, yang berfungsi sebagai referensi atau sumber data dalam proses tersebut. *Library* lexicon yang digunakan adalah Indonesian Sentiment (InSet) Lexicon, yang telah dimodifikasi untuk menyesuaikan konteks *hate speech*, sehingga dapat meningkatkan akurasi pelabelan dalam penelitian ini.



Gambar 7. *Repositori Inset Lexicon*

Tabel 8. Pelabelan Berdasarkan Inset Lexicon

Kata Lexicon Positif	**Skor**	**Kata Lexicon Negatif**	**Skor**
hai	3	anjir	-3
merekam	2	bodo	-5
detail	2	gelo	-2

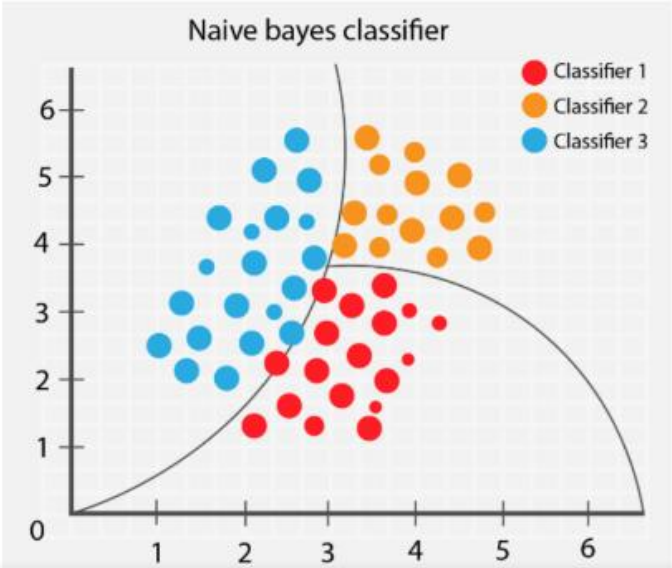
Tabel 9. Perhitungan Skor

No	Komentar	Kata lexicon positif	Skor positif	Kata lexicon negatif	Skor negatif	Total skor	Label
1	"hai, apa kabar"	hai, kabar	+3, +2	-	0	+5	Positif
2	"tidak jelas anjir"	-	0	tidak, anjir	-5, -3	-8	Negatif

3.8.4 Ekstraksi Fitur

Dalam penelitian ini, fitur diekstrak menggunakan metode TF-IDF. Metode ini umum digunakan untuk menentukan bobot kata. Proses ini melibatkan pengukuran frekuensi kemunculan kata dalam sebuah dokumen. TF mengacu pada seberapa sering suatu kata muncul di dokumen tersebut, sementara IDF menilai pentingnya kata dalam konteks keseluruhan dokumen. Nilai TF-IDF merupakan hasil dari perkalian antara skor TF dan skor IDF.

3.8.5 Implementasi Algoritma *Naïve Bayes* Dengan *Smoothing*



Gambar 8. Algoritma *Naïve Bayes*

Dalam *naïve bayes*, probabilitas kondisional digunakan untuk memperkirakan kemungkinan suatu kelas berdasarkan nilai fitur yang tersedia. Fitur adalah variabel yang menggambarkan data, dan dalam algoritma ini, fitur digunakan untuk memprediksi kelas atau label.

Naïve bayes mengasumsikan bahwa semua fitur dalam data saling independen, meskipun asumsi ini sering tidak tepat dalam situasi nyata. Meskipun demikian, algoritma ini tetap efektif dan sering digunakan untuk berbagai tugas, seperti klasifikasi teks, deteksi *spam*, pengenalan tulisan tangan, analisis ekspresi gen, deteksi wajah, serta deteksi anomali. Untuk meningkatkan performa dari *naïve bayes* dan meminimalisir terjdainya probabilitas nol pada hasil klasifikasi, dibutuhkan adanya metode *smoothing*. *Smoothing* yang akan digunakan pada penelitian ini adalah *laplace smoothing.*

3.8.6 Evaluasi

Tahap akhir dalam penelitian ini adalah evaluasi untuk memastikan efektivitas dan kinerja model yang telah dibangun. Evaluasi dilakukan dengan menghitung akurasi,

presisi, *recall*, dan *f1-score*, di mana setiap metrik memiliki fungsi spesifik dalam menggambarkan kemampuan model. Akurasi mengukur persentase prediksi yang benar dari keseluruhan data, presisi menunjukkan sejauh mana model dapat meminimalkan kesalahan dengan memprediksi hanya data yang benar-benar termasuk dalam kategori *hate speech*, *recall* menilai kemampuan model dalam mendeteksi semua data yang relevan sebagai *hate speech*, dan *f1-score* memberikan rata-rata harmonis antara presisi dan recall untuk menggambarkan performa model secara lebih komprehensif. Evaluasi ini tidak hanya bertujuan untuk menentukan performa terbaik dari model, tetapi juga untuk mengetahui bagaimana model beradaptasi dengan data yang digunakan, sehingga dapat memberikan wawasan mendalam tentang kekuatan dan kelemahan model dalam mendeteksi *hate speech* pada komentar TikTok.

Selain metrik-metrik tersebut, ditambahkan juga tahapan verifikasi model dengan melibatkan pengguna. Tahapan ini memungkinkan pengguna untuk berpartisipasi aktif dengan menggunakan tombol *like* dan *dislike* pada *plugin* untuk menyatakan kesesuaian atau ketidaksesuaian mereka dengan hasil klasifikasi *hate speech* yang diberikan oleh model. Umpan balik dari pengguna ini tidak hanya membantu memperbaiki akurasi dan adaptasi model terhadap data yang digunakan, tetapi juga memberikan wawasan lebih mendalam tentang kekuatan dan kelemahan model dalam mendeteksi *hate speech* pada komentar TikTok.

3.9 Implementasi *Plugin*

Plugin yang dikembangkan adalah ekstensi Google Chrome yang dirancang untuk mendeteksi *hate speech* dalam kolom komentar TikTok. Ketika pengguna membuka halaman TikTok, *plugin* ini otomatis aktif dan melakukan pemantauan pada kolom komentar. Komentar yang ditemukan akan dianalisis di latar belakang menggunakan algoritma *naïve bayes* dengan *smoothing*. Hal inilah juga yang menjadikan pembeda dengan penelitian-penelitian yang sudah dilakukan sebelumnya karena TikTok, sebagai *platform* media sosial, masih tergolong baru berkembang di Indonesia.

Antarmuka *plugin* ini sederhana, menampilkan tombol *toggle* "Deteksi Aktif" untuk mengaktifkan atau menonaktifkan pendeteksian *hate speech*. Ketika deteksi aktif, *plugin* akan menyoroti komentar yang terdeteksi sebagai *hate speech*, misalnya dengan efek warna khusus. *Plugin* juga menyediakan tombol "*like*" dan "*dislike*" sebagai fitur verifikasi, memungkinkan pengguna memberi umpan balik terkait hasil klasifikasi *hate speech*. Langkah-langkah implementasi yang dilakukan meliputi:

1. Pembuatan ekstensi Chrome, ekstensi dirancang untuk memantau halaman TikTok, mendeteksi kolom komentar, dan memproses teks komentar untuk mendeteksi *hate speech*. Algoritma *naïve bayes* yang telah dilatih sebelumnya (menggunakan *dataset* berisi contoh *hate speech*) diintegrasikan ke dalam ekstensi ini.
2. Penggunaan algoritma *naïve bayes* dengan *smoothing*, algoritma *naïve bayes* digunakan untuk mengklasifikasikan apakah suatu komentar mengandung *hate speech* atau tidak. *smoothing*, *laplace smoothing*, digunakan untuk menangani kemungkinan kata yang belum pernah muncul dalam data pelatihan, sehingga prediksi tetap bisa dilakukan meski ada variasi baru dalam bahasa komentar.
3. Integrasi ke dalam UI Chrome, *plugin* harus mampu mengambil komentar-komentar dari halaman TikTok yang sedang dibuka oleh pengguna. Komentar-komentar ini diproses di latar belakang oleh algoritma yang berjalan di *plugin*, dan hasil klasifikasi *hate speech* ditampilkan kepada pengguna dalam bentuk visual (misalnya, dengan menandai komentar dengan efek pewarnaan jika terdeteksi sebagai *hate speech*).

3.10 Pengujian *Plugin*

Pengujian dilakukan untuk memastikan *plugin* berfungsi dengan benar dan dapat mendeteksi *hate speech* secara akurat. Pengujian yang dilakukan akan menggunakan dengan pendekatan *black box testing*, dan *white box testing.* Pada pengujian *black box testing*, peneliti akan difokuskan pada pengujian fungsionalitas *plugin* tanpa memperhatikan proses internal yaitu mengaktifkan *plugin* deteksi *hate*

speech dan menonaktifkan deteksi *hate speech*. Dalam pengujian *white*-box, peneliti akan menguji aspek internal dan logika kerja dari *plugin*. Pengujian ini akan menguji fungsi-fungsi yang mewakili semua isi dari baris program *plugin* deteksi *hate speech* pada kolom komentar TikTok.

V. SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan sebagai berikut:

1. Penelitian ini berhasil mengembangkan model deteksi *hate speech* pada kolom komentar TikTok menggunakan algoritma Multinomial Naïve Bayes dengan *smoothing*. Model menunjukkan performa dengan hasil evaluasi pada data uji berupa akurasi 88,410%, *precision* 88,413%, *recall* 88,410%, dan *F1-score* 88,407%. Selain itu, model ini menerapkan prinsip Markov dengan konfigurasi ngram_range=(1, 2) pada fitur yang dihasilkan menggunakan TF-IDF.
2. Penelitian ini juga melakukan evaluasi model melalui *user testing* dengan melibatkan 35 pengguna yang menganalisis 7.415 komentar dari 36 konten TikTok. Berdasarkan hasil evaluasi, tingkat akurasi model dalam mendeteksi *hate speech* di media sosial TikTok mencapai 68,6%, dimana sebanyak 5.084 komentar berhasil terdeteksi dengan benar sebagai *hate speech*, sementara 2.331 komentar tidak terdeteksi dengan benar.
3. Sebagai bagian dari implementasi model, penelitian ini juga berhasil mengembangkan *plugin* Google Chrome yang berfungsi mendeteksi *hate speech* di kolom komentar TikTok secara *real-time*. *Plugin* ini dilengkapi dengan fitur visualisasi probabilitas menggunakan sistem pewarnaan, serta mekanisme validasi oleh pengguna melalui tombol *like* dan *dislike*. Dengan integrasi ini, *plugin* dapat berfungsi secara efisien dan interaktif, memberikan solusi praktis dalam mendeteksi *hate speech* pada *platform* TikTok.

5.2 Saran

Saran untuk pengembangan penelitian di masa mendatang adalah sebagai berikut:

1. Mengembangkan model *naïve bayes* dengan metode *smoothing* menggunakan dataset yang lebih beragam untuk mencakup berbagai jenis komentar.
2. Mengganti metode pelabelan dengan tidak lagi menggunakan *lexicon* Indonesia (INSET), karena perkembangan bahasa yang dinamis membuat sebagian besar kata-kata pada *lexicon* tersebut sudah tidak relevan.
3. Menerapkan validasi silang (*k-fold cross-validation*) pada data pelatihan untuk mengevaluasi kestabilan dan performa model secara lebih akurat. Dengan membagi data pelatihan menjadi beberapa fold, model dilatih dan diuji secara bergantian sehingga hasil evaluasi lebih representatif dan risiko overfitting berkurang. Meskipun akurasi yang dihasilkan bisa sedikit menurun dibandingkan evaluasi biasa, teknik ini memberikan gambaran yang lebih realistis tentang kemampuan generalisasi model terhadap data baru, sehingga menghasilkan model yang lebih andal.

DAFTAR PUSTAKA

Ali, A., Khairan, A., Tempola, F., and Fuad, A. 2021. Application of Naïve Bayes to Predict the Potential of Rain in Ternate City. *E3S Web of Conferences*. 328.

Ariska, A., dan Kamayani, M. 2024. Deteksi Hate Speech pada Kolom Komentar TikTok dengan Menggunakan SVM. *Indonesian Journal of Computer Science*. 13(3) : 284–301.

Bustami. 2013. Penerapan Algoritma Naive Bayes untuk Mengklasifikasi Data Nasabah Asuransi. *TECHSI: Jurnal Peneliti Teknik Informatika.* 127-146.

Chen, H., Hu, S., Hua, R., and Zhao, X. 2021. Improved Naive Bayes Classification Algorithm for Traffic Risk Management. *Eurasip Journal on Advances in Signal Processing*. 2021(1) : 1-12.

Elliott, C., Chuma, W., and Gendi, Y. E. 2016. *Hate Speech, Key Concept Paper*. Media Conflict and Democratisation (MeCoDEM). United Kingdom.

Koto, F and Gemala, Y. R. 2017. InSet Lexicon: Evaluation of a Word List for Indonesian Sentiment Analysis in Microblogs. *International Conference on Asian Language Processing (IALP).* 391–394.

Fatahillah, N. R., Suryati, P., and Haryawan, C. 2017. Implementation of Naive Bayes Classifier Algorithm on Social Media (Twitter) to the Teaching of Indonesian Hate Speech. *Proceedings - 2017 International Conference on Sustainable Information Engineering and Technology.* 128–131.

Febiharsa, D., Sudana, I. M., dan Hudallah, N. 2019. Uji Fungsionalitas (Blackbox Testing) Sistem Informasi Lembaga Sertifikasi Profesi (SILSP) Batik dengan AppPerfect Web Test dan Uji Pengguna. *Joined Journal (Journal of Informatics Education)*. 1(2) : 117.

Jackins, V., Vimal, S., Kaliappan, M., and Lee, M. Y. 2021. AI-based Smart Prediction of Clinical Disease Using Random Forest Classifier and Naive

Bayes. *Journal of Supercomputing*. 77(5) : 5198–5219.

Mahardhika, S. V., Nurjannah, I., Ma'una, I. I., dan Islamiyah, Z. 2021. Faktor-Faktor Penyebab Tingginya Minat Generasi Post-Millenial Di Indonesia Terhadap Penggunaan Aplikasi Tik-Tok. *SOSEARCH : Social Science Educational Research*. 2(1) : 40–53.

Murphy, C. M., and McCashin, D. 2023. Using TikTok for Public and Youthmental Health – a Systematic Review and Content Analysis. *Clinical Child Psychology and Psychiatry*. 28(1) : 279-306.

Noto, A. P., and Saputro, D. R. S. 2022. Classification Data Mining with Laplacian Smoothing on Naïve Bayes Method. *AIP Conference Proceedings*. Solo.

Pan, J., Sun, M., Wang, Y., and Zhang, X. 2020. an Enhanced Spatial Smoothing Technique with ESPRIT Algorithm for Direction of Arrival Estimation in Coherent Scenarios. *IEEE Transactions on Signal Processing*. 68 : 3635–3643.

Pisner, D. A and Schnyer, D. M. 2019. *Support Vector Machine*. In Machine Learning: Methods and Applications to Brain Disorders. Elsevier Inc.

Poletto, F., Basile, V., Sanguinetti, M., Bosco, C., and Patti, V. 2021. Resources and Benchmark Corpora for Hate Speech Detection: a Systematic Review. *Language Resources and Evaluation*. 55(2) : 477–523.

Pradana, A. W and Hayaty, M. 2019. The Effect of Stemming and Removal of Stopwords on the Accuracy of Sentiment Analysis on Indonesian-language Texts. *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*. 4(3) : 375–380.

Prasetyo, E., Al-adni, M. F., dan Tias, R. F. 2024. Classification of Cash Direct Recipients Using the Naive Bayes With Smoothing. *Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer*. 23(3) : 615–626.

Putri, R. K., Athoillah, M., Haqiqiyah, A., dan Lestari, F. W. A. 2023. Deteksi Penggunaan Masker Wajah Dengan Algoritma Deep Learning. *Prosiding Seminar Nasional Hasil Riset dan Pengabdian*. Surabaya.

Putri, T. T. A., Sriadhi, S., Sari, R. D., Rahmadani, R., and Hutahaean, H. D. 2020. a Comparison of Classification Algorithms for Hate Speech Detection. *IOP Conference Series: Materials Science and Engineering*. 830(3).

Ria, R. N., and Setiawan, T. 2023. Forensic Linguistic Analysis of Netizens' Hate Speech Acts in Tik-Tok Comment Section. *Britain International of Linguistics Arts and Education (BIoLAE) Journal*. 5(2) : 141–152.

Saritas, M.M and Yasar, A. 2019. Performance Analysis of ANN and Naive Bayes Classification Algorithm for Data Classification. *IJISAE : International Journal of Intelligent Systems and Applications in Engineering*. 7(2) : 88–91.

Subramanian, M., Easwaramoorthy, S. V., Deepalakshmi, G., Cho, J., and Manikandan, G. 2023. a Survey on Hate Speech Detection and Sentiment Analysis Using Machine Learning and Deep Learning Models. *Alexandria Engineering Journal*. 80 : 110–121.

Tan, Y., and Shenoy, P. P. 2020. a Bias-Variance Based Heuristic for Constructing a Hybrid Logistic Regression-Naïve Bayes Model for Classification. *International Journal of Approximate Reasoning*. 117 : 15–28.

Teguh, K., Kridalukmana., Rinta., dan Martono. 2012. Pembuatan Chrome Extension untuk Akses Website Sistem Komputer. *Proceedings Business Intelligence: Extending Your Business*. 81-92.

Verma, A., Khatana, A., and Chaudhary, S. 2017. a Comparative Study of Black Box Testing and White Box Testing. *International Journal of Computer Sciences and Engineering*. 5(12) : 301–304.

Wester, P., Heiding, F., and Lagerstrom, R. 2021. Anomaly-based Intrusion Detection Using Tree Augmented Naive Bayes. *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW*. 112–121.

Zhu, Z., Liang, J., Li, D., Yu, H., and Liu, G. 2019. Hot Topic Detection Based on a Refined TF-IDF Algorithm. *IEEE Access*. 7 : 26996–27007.

Zulli, D., and Zulli, D. J. 2022. Extending the Internet Meme: Conceptualizing Technological Mimesis and Imitation Publics on the TikTok Platform. *New Media and Society*. 24(8) : 1872–1890.