SISTEM DETEKSI JUDUL VIDEO CLICKBAIT PADA PLATFORM YOUTUBE MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE

diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana pada program studi Teknologi Informasi

SKRIPSI

Oleh

Pruis Shinta Reza Elbas
212410102001

KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
UNIVERSITAS JEMBER
FAKULTAS ILMU KOMPUTER
TEKNOLOGI INFORMASI
JEMBER

2025

PERSEMBAHAN

Skripsi ini penulis persembahkan untuk:

1. Allah SWT, atas limpahan rahmat dan hidayahnya yang senantiasa menyertai, sehingga penulis dapat menyelesaikan skripsi ini dengan baik;
2. Orang tua penulis, Yeni Yuliana selaku Ibu yang selalu mendukung dan mengusahakan yang terbaik untuk penulis serta tanpa berhenti mendoakan penulis. Muhammad Subahanal Malik selaku Ayah yang secara diam-diam selalu memperhatikan penulis dan memastikan penulis dalam keadaan baik-baik saja setiap harinya;
3. Sintia Anggi Nurhaliza, Uzda Rafifatu Rifda, dan Mafaza Nur Hidayah selaku adik-adik penulis yang selalu memberi senyuman hangat kepada penulis sehingga menjadi penyemangat yang berharga bagi penulis;
4. Almarhum nenek dan tante di surga, yang telah begitu banyak memberikan bantuan dan kasih sayang yang belum sempat terbalaskan;
5. Almamater Program Studi Teknologi Informasi, Fakultas Ilmu Komputer, Universitas Jember.

MOTTO

“Allah tidak mengatakan hidup ini mudah. Tetapi Allah berjanji, bahwa sesungguhnya bersama kesulitan ada kemudahan.”

- (QS. Al-Insyirah 5-6)

“Aku membahayakan nyawa ibuku untuk lahir ke dunia, jadi tidak mungkin aku tidak ada artinya.”

- Moonraeyy

“Jika jalannya mudah, itu bukan jalan menuju mimpiku.”

- Luffy

PERNYATAAN ORISINALITAS

Saya yang bertanda tangan di bawah ini :

Nama : Pruis Shinta Reza Elbas

NIM : 212410102001

Menyatakan dengan sesungguhnya bahwa skripsi yang berjudul: "*Sistem Deteksi Judul Video Clickbait pada Platform Youtube Menggunakan Algoritma Support Vector Machine"* adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya, dan belum pernah diajukan pada institusi manapun, serta bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 28 Mei 2025

Yang menyatakan,

Pruis Shinta Reza Elbas

NIM 212410102001

HALAMAN PERSETUJUAN

Skripsi berjudul *Sistem Deteksi Judul Video Clickbait pada Platform Youtube Menggunakan Algoritma Support Vector Machine* telah diuji dan disetujui oleh Fakultas Ilmu Komputer Universitas Jember pada:

Hari : Rabu

Tanggal : 2 Juli 2025

Tempat : Fakultas Ilmu Komputer Universitas Jember

Pembimbing — Tanda Tangan

1. Pembimbing Utama

   Nama : Achmad Maududie ST, M.Sc. (.....)

   NIP : 197004221995121001

2. Pembimbing Anggota

   Nama : Priza Pandunata S.Kom., M.Sc. (.....)

   NIP : 198301312015041001

Penguji — Tanda Tangan

1. Penguji Utama

   Nama : Muhamad Arief Hidayat S.Kom., M.Kom. (.....)

   NIP : 198101232010121003

2. Penguji Anggota

   Nama : Harry Soepandi S.Kom., M.Kom. (.....)

   NIP : 197604252023211002

ABSTRACT

The widespread use of clickbait titles on YouTube has become a concern due to their misleading nature, which often leads viewers to content that does not match their expectations. This study aims to develop a classification system to detect clickbait in YouTube trending video titles using the Support Vector Machine (SVM) algorithm. A total of 2,062 video titles were collected through web scraping and manually labeled into clickbait and non-clickbait categories. To ensure class balance, the data collection process was repeated three times. The methodology involved several preprocessing steps, including case folding, cleaning using Regular Expressions (RegEx), normalization with a slang word dictionary, tokenization, filtering, and stemming. The processed text was transformed into numerical vectors using Term Frequency–Inverse Document Frequency (TF-IDF), followed by training and evaluation of the SVM model. The model achieved an accuracy of 86.44%, indicating good performance in identifying clickbait and non-clickbait titles. However, a small number of misclassifications were observed, where some clickbait titles were incorrectly labeled as non-clickbait. These errors are likely due to the limited coverage and variation within the training dataset, which may not have captured all patterns of clickbait expressions. Despite this limitation, the model demonstrated reliable results and was successfully implemented into a simple web-based application, allowing users to test titles interactively. The results show that the SVM model has promising potential in supporting automated clickbait detection.

Keywords: *Clickbait Detection, YouTube Trending, Support Vector Machine, TF-IDF, Confusion Matrix*

RINGKASAN

Perkembangan konten digital, khususnya di platform *YouTube,* memicu meningkatnya penggunaan judul video yang bersifat *clickbait*. Judul seperti ini dibuat semenarik mungkin untuk menarik perhatian pengguna, namun sering kali menyesatkan karena tidak sesuai dengan isi video. Fenomena ini menjadi masalah karena dapat menurunkan kepercayaan pengguna terhadap platform dan memengaruhi pengalaman menonton. Dengan demikian, dibutuhkan sebuah sistem yang mampu melakukan pendeteksian terhadap judul video *clickbait*.

Tujuan dari penelitian ini adalah merancang sebuah sistem klasifikasi untuk mendeteksi judul *clickbait* pada *trending* video *YouTube* dalam aplikasi berbasis web. Algoritma yang digunakan dalam penelitian ini adalah *Support Vector Machine* (SVM), yang telah terbukti efektif dalam berbagai permasalahan klasifikasi teks. Dataset yang digunakan diperoleh melalui proses *crawling* dari data judul video *YouTube* dan dilabeli secara manual menjadi dua kelas, yaitu *clickbait* dan *non-clickbait*. Total judul video yang digunakan sebanyak 2.062 judul.

Sebelum dilakukan pemodelan, data diproses terlebih dahulu melalui serangkaian tahapan *preprocessing* meliputi *case folding, cleaning data, normalization, tokenization, filtering*, dan *stemming*. Setelah teks dibersihkan dan diproses, dilakukan pembobotan kata menggunakan TF-IDF.

Pembagian data terbaik yang digunakan adalah pembagian data dengan rasio 80:20. Kemudian, Proses evaluasi dilakukan menggunakan metrik akurasi, yang menunjukkan bahwa model mampu memperoleh akurasi senilai 86,44%. Kemudian model diimplementasikan ke aplikasi berbasis web untuk mendeteksi judul video trending *YouTube* secara *realtime*. Hasil ini menunjukkan bahwa model mampu mengenali pola-pola yang membedakan antara judul *clickbait* dan *non-clickbait* dengan cukup baik. Namun, ditemukan sedikit kesalahan klasifikasi, sepert judul *clickbait* yang terdeteksi sebagai *non-clickbait*. Hal ini kemungkinan disebabkan oleh keterbatasan variasi dalam dataset pelatihan.

PRAKATA

Segala puji dan rasa syukur penulis haturkan kepada Allah SWT atas segala anugerah, rahmat, serta petunjuk-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul *"Sistem Deteksi Judul Video Clickbait pada Platform Youtube Menggunakan Algoritma Support Vector Machine"* dengan baik. Dengan penuh rasa hormat, penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Orang tua tercinta, Muhammad Subahanal Malik selaku Ayah dan Yeni Yuliana selaku Ibu penulis yang tak henti-hentinya memberikan doa dan dukungan kepada putri sulungnya di sepanjang hidupnya;
2. Dekan Fakultas Ilmu Komputer Universitas Jember beserta jajarannya, Prof. Drs. Antonius Cahya Prihandoko, M.App.Sc, Ph,D;
3. Ketua Program Studi Teknologi Informasi, Bapak Priza Pandunata, S.Kom., M.Sc;
4. Dosen Pembimbing Akademik, Bapak Anang Andrianto, S.T., M.T yang selalu membimbing serta memberikan semangat selama proses perkuliahan;
5. Dosen pembimbing, Bapak Achmad Maududie ST, M.Sc. dan Bapak Priza Pandunata, S.Kom., M.Sc. yang telah memberikan bimbingan, ilmu, dan motivasi kepada penulis selama proses pengerjaan skripsi;
6. Dosen penguji, Bapak Muhamad Arief Hidayat S.Kom., M.Kom. dan Bapak Harry Soepandi S.Kom., M.Kom. yang telah menyampaikan berbagai masukan dan saran selama proses pengerjaan skripsi;
7. Seluruh dosen dan segenap civitas akademika Fakultas Ilmu Komputer Universitas Jember;
8. Presiden Republik Indonesia ke-7, Bapak Ir. H. Joko Widodo, atas inisiatif beliau dalam mencetuskan program KIP Kuliah yang telah membuka jalan bagi penulis untuk menempuh pendidikan tinggi tanpa beban biaya;
9. Saudara kandung penulis, Anggi, Rifda, dan Faza yang selalu mendoakan dan memberi semangat kepada penulis selama proses pengerjaan skripsi;
10. Sahabat penulis, Balqis, Hikmi, dan Sherly yang menjadi tempat berbagi cerita, mendengarkan keluh kesah, bahkan ikut pusing tanpa diminta;

11. Seseorang yang membersamai penulis, atas segala dukungan, tanpa keluh terus direpotkan, memberi semangat tanpa henti, serta doa tulus yang senantiasa diberikan kepada penulis;
12. Kucing kesayangan penulis, Kuchi, yang telah menemani proses penyusunan skripsi ini. Kehadirannya memberikan ketenangan, kenyamanan, dan semangat hingga akhir hayatnya.
13. Dan seluruh pihak yang turut membantu namun tidak dapat disebutkan satu per satu.

Penulis menyadari bahwa skripsi ini belum sepenuhnya sempurna dan masih memiliki keterbatasan. Namun, penulis telah berupaya semaksimal dan sebaik mungkin dalam penyusunannya. Oleh karena itu, penulis sangat mengharapkan masukan dan kritik yang membangun demi penyempurnaan di penelitian mendatang. Semoga skripsi ini dapat memberi kontribusi yang bermanfaat bagi siapapun yang membacanya.

Jember, 28 Mei 2025

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

Bab pendahuluan ini akan membahas mengenai latar belakang, perumusan masalah, tujuan, manfaat, dan batasan dari penelitian yang dilakukan.

1.1 Latar Belakang

Internet telah menjadi kebutuhan sehari-hari masyarakat dari berbagai kalangan pada era digital ini. Jumlah pengguna internet di Indonesia sudah menembus angka 221 juta orang pada tahun 2024 (Mufti Prasetiyo et al., 2024). Salah satu platform yang populer di internet yaitu *YouTube*. Pada *Youtube,* judul memiliki daya tarik tersendiri untuk menarik pengguna. Pentingnya kesesuaian judul dengan isi konten tidak selalu menjadi prioritas bagi para konten kreator. Banyak ditemukan video dengan judul yang terkesan bombastik namun rupanya konten di dalamnya tidak sesuai dengan judul yang ada (Sagita et al., 2020). Judul yang bersifat *clickbait* memancing rasa penasaran para pengguna *YouTube* sehingga dapat meningkatkan jumlah penonton (Dewi Trustyanda et al., 2021).

Strategi *clickbait* ini tentunya memberikan dampak negatif pada kualitas informasi serta membuat kepercayaan para pengguna terhadap platform menjadi turun. Ketika pengguna berharap sesuatu yang besar dari judul yang mereka baca, namun isi konten di dalamnya ternyata berbeda dari apa yang diharapkan maka dapat menciptakan rasa kecewa dan pengguna merasa tertipu. Hal tersebut dapat menurunkan kredibilitas *YouTube* di mata publik. Maka, perlu diciptakannya sebuah sistem yang dapat melakukan klasifikasi terhadap judul yang *clickbait* dan *non-clickbait.*

Klasifikasi adalah suatu teknik dari *text mining* yang digunakan untuk membagi dokumen teks ke dalam kelompok-kelompok yang berbeda berdasarkan tema atau kategori yang relevan (Srivastava & Sahami, 2009). Terdapat beberapa tahapan yang perlu dilakukan untuk melakukan klasifikasi antara lain *preprocessing*, pembobotan kata, pemodelan menggunakan algoritma *text mining*, dan evaluasi model. Banyak algoritma yang dapat digunakan untuk klasifikasi teks seperti *Naive Bayes Classifier, Random Forest, K-Nearest Neighbour,* dan *Support Vector Machines* (Saputra & Kusnadi, 2021). Pada beberapa penelitian, *Support*

Vector Machine (SVM) dianggap paling baik digunakan untuk melakukan klasifikasi teks dengan nilai presisi sebesar 0.95 mengungguli *Decission Tree* dengan nilai presisi 0.92 dan *Random Forest* dengan nilai presisi 0.94 (Pujahari & Singh Sisodia, 2019). Pada penelitian lain, SVM dengan TF-IDF menghasilkan nilai presisi sebesar 98,53% mengungguli model *Naive Bayes* dan LSTM (Winarto et al., 2023).

Penelitian ini akan menggunakan algoritma SVM karena terbukti paling efektif dibandingkan algoritma yang lain dalam melakukan klasifikasi teks khususnya pada data yang memiliki 2 class (Pujahari & Singh Sisodia, 2019). Diharapkan penelitian ini dapat memberikan manfaat dalam mengidentifikasikan apakah suatu judul termasuk dalam kategori *clickbait* atau bukan, sehingga para pengguna menjadi lebih mengerti dan dapat membedakan antara judul *clickbait* dan *non-clickbait*.

1.2 Rumusan Masalah

Permasalahan yang menjadi fokus dalam penelitian ini dirumuskan sebagai berikut:

1. Bagaimana menerapkan algoritma *Support Vector Machine* (SVM) untuk menghasilkan model klasifikasi *clickbait* pada judul video *YouTube* dalam Bahasa Indonesia?
2. Bagaimana merancang dan menghasilkan sistem untuk mendeteksi *clickbait* pada judul video *trending* di *YouTube* secara otomatis menggunakan algoritma *Support Vector Machine* (SVM)?

1.3 Batasan Penelitian

Penelitian ini menetapkan sejumlah batasan untuk mencegah terjadinya penyimpangan dari tujuan utama. Adapun batasan tersebut adalah:

1. Dataset yang digunakan untuk pemodelan diambil dari data judul video *YouTube* berbahasa Indonesia yang dilabeli manual dengan 0 sebagai *non-clickbait* dan 1 sebagai *clickbait.*

2. Pelabelan dilakukan dengan mengacu pada kategori judul *clickbait* dalam jurnal yang digunakan sebagai landasan (Kemm, 2022). Kemudian data dilabeli oleh 5 orang dan ditentukan berdasarkan voting terbanyak.
3. Menggunakan tahapan *normalization* dalam *preprocessing* untuk mengurangi variasi kata dan menjaga konsistensi data dengan memanfaatkan kamus *new_kamusalay.csv* (Ibrohim & Budi, 2019).
4. Evaluasi model dilakukan dengan menghitung nilai akurasi. Pemilihan metrik akurasi didasari dengan asumsi dataset yang seimbang.
5. Penelitian ini menggunakan 3 rasio pembagian data yang digunakan dan rasio pembagian data dengan hasil akurasi terbaik akan dipilih untuk diimplementasikan ke dalam aplikasi.
6. Model akan diimplementasikan ke aplikasi apabila akurasi yang didapatkan mencapai ≥ 0.80.
7. Sistem dirancang menggunakan *framework flask* dan berbasis web sederhana.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini antara lain:

1. Mengimplementasikan algoritma *Support Vector Machine* (SVM) dalam membangun model klasifikasi *clickbait* pada judul video YouTube dalam Bahasa Indonesia.
2. Untuk merancang dan menghasilkan sistem berupa aplikasi berbasis web yang dapat mendeteksi *clickbait* pada judul video trending di YouTube secara otomatis menggunakan model klasifikasi berbasis algoritma *Support Vector Machine* (SVM).

1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini antara lain:

1. Bagi Pengguna

   Dapat membantu pengguna platform YouTube untuk memilah video supaya tidak terjebak judul *clickbait*.

2. Bagi Pembaca

   Menambah pengetahuan mengenai klasifikasi *text mining* khususnya penggunaan algoritma *Support Vector Machine* yang dapat digunakan dalam klasifikasi judul *clickbait*. Penelitian ini diharapkan dapat menjadi pijakan dalam penelitian atau pengembangan aplikasi yang lebih optimal di masa mendatang.

3. Bagi Peneliti

   Sebagai sarana untuk mengimplementasikan ilmu yang didapat dan meningkatkan pemahaman peneliti mengenai klasifikasi *text mining* dengan memanfaatkan algoritma *Support Vector Machine* (SVM).

BAB 2. TINJAUAN TEORI

Bab ini menguraikan penjelasan mengenai referensi yang digunakan dan kajian teori yang digunakan dalam pelaksanaan penelitian.

2.1 Penelitian Terdahulu

Sejumlah penelitian menunjukkan bahwa algoritma *Support Vector Machine* (SVM) memiliki performa lebih baik dalam mendeteksi *clickbait* dibandingkan algoritma lain seperti *Naïve Bayes, LSTM, Random Forest* dan *Decision Tree*. Dalam penelitian oleh Winarto, SVM menghasilkan performa terbaik dengan akurasi 98,53%, mengalahkan *Naïve Bayes* dan LSTM (Winarto et al., 2023). Goel juga menyatakan bahwa SVM mengungguli M*ultinomial Naïve Bayes* dan *Random Forest* dengan akurasi mencapai 95% (Goel, 2023). Pujahari dan Singh Sisodia juga membandingkan tiga algoritma dan menyimpulkan bahwa SVM menghasilkan performa terbaik dengan akurasi tertinggi yang didapatkan yaitu 97% (Pujahari & Singh Sisodia, 2019). Penelitian Putra juga memperkuat SVM dengan hasil akurasi sebesar 93,06% (Rama Bena Putra & Setya Perdana, 2023). Implementasi SVM juga dapat digunakan untuk melakukan klasifikasi *clickbait* pada Bahasa China dengan hasil yang cukup baik dan memiliki peluang untuk ditingkatkan lagi (Zhang & Clough, 2020).

Pada tahap *preprocessing*, sebagian besar studi mengimplementasikan langkah-langkah umum seperti *case folding, tokenizing*, dan *stopword removal*. Penambahan *preprocessing* yang lebih lengkap seperti *cleaning, tokenizing, stopword removal*, dan *stemming* dapat mempengaruhi hasil akhir klasifikasi dan menghasilkan akurasi mencapai 93.06% (Putra and Perdana 2023).

Ekstraksi fitur juga dapat mempengaruhi hasil akhir klasifikasi, SVM dengan TF-IDF terbukti lebih unggul dibandingkan kombinasi SVM dengan BoW, dengan peningkatan akurasi dari 98,32% menjadi 98,53% (Winarto et al., 2023). Goel dan Putra juga menggunakan TF-IDF sebagai teknik utama dalam pembobotan kata (Rama Bena Putra & Setya Perdana, 2023).

Terkait pembagian data, rasio yang digunakan cukup bervariasi. Winarto menggunakan 80% datanya untuk pelatihan (Winarto et al., 2023). Goel juga

menggunakan proporsi 80:20 untuk *training* dan *testing* (Goel, 2023). Sedangkan Putra membagi data menjadi 70:30 serta menyeimbangkan distribusi antar label (Rama Bena Putra & Setya Perdana, 2023).

Berdasarkan keseluruhan hasil studi, dapat disimpulkan bahwa SVM merupakan algoritma yang sangat kompetitif dalam tugas klasifikasi *clickbait* berbasis teks. Performa optimal SVM juga dipengaruhi oleh implementasi preprocessing yang komprehensif, pemilihan metode ekstraksi fitur, serta proporsi pembagian data.

2.2 Clickbait

Clickbait merupakan teknik yang banyak digunakan untuk membuat sebuah judul agar terlihat menarik di mata para pengguna. Banyak konten kreator yang beramai-ramai menggunakan teknik *clickbait* ini untuk menarik pengguna/netizen supaya kontennya mendapatkan lebih banyak klik dari para pengguna (Putri & Eriend, 2024). Banyak judul *clickbait* yang dengan sengaja dibuat sensasional agar mendapatkan lebih banyak klik. Hal tersebut menyebabkan para pengguna merasa penasaran yang berlebihan (Hadiyat, 2019). Namun, ketika mereka mengakses judul tersebut dan ternyata isi konten tidak sesuai dengan ekspektasi mereka. Maka, dapat menimbulkan hilangnya rasa percaya mereka terhadap platform tersebut karena merasa tertipu. Hal ini memberikan dampak negatif kepada para pengguna yang merasa tertipu dan juga pemilik platform yang kehilangan reputasi.

Judul yang *clickbait* biasanya menggunakan formulasi penulisan dan teknik linguistik yang menjebak pembaca dan menciptakan rasa ingin tahu (Kemm, 2022). Dalam artikelnya, Kemm menjelaskan bahwa tidak semua fitur linguistik dan tipologis dalam judul video *YouTube* memiliki peran yang sama dalam menciptakan efek *clickbait*. Kemm mengelompokkan fitur-fitur tersebut ke dalam empat kategori, yaitu: *indicative* (kuat), *partially indicative*, *facilitative*, dan *not indicative.* Klasifikasi tersebut dapat disajikan secara ringkas dalam tabel 2.1 berikut:

Tabel 2.1 Kategori Judul *Clickbait (Kemm, 2022)*

Kategori	Fitur *Clickbait*
Indicative (kuat)	*Personal pronouns, superlatif, forward-reference, hyperbole*
Partially Indicative	*Emotional appeal, numerals, modals, familiar vocab*
Facilitative	Huruf kapital, simbol/tanda baca
Not indicative	Bentuk pendek, bentuk tanya

Klasifikasi fitur *clickbait* dari Kemm (2022) menjadi acuan penting dalam proses pelabelan data. Proses pelabelan dilakukan secara manual oleh lima orang anotator, yang mengklasifikasikan judul video ke dalam dua kelas yaitu *clickbait* dan *non-clickbait*. Penilaian didasarkan pada keberadaan fitur-fitur dalam kategori *indicative* dan *partially indicative*. Dengan pendekatan ini, proses pelabelan dilakukan secara sistematis dan berbasis teori, sehingga hasilnya lebih konsisten dan dapat dipertanggungjawabkan.

2.3 Data Trending Youtube

Data trending YouTube adalah kumpulan data informasi mengenai video-video *YouTube* yang sedang populer dan banyak di tonton pada waktu tertentu. Terdapat beberapa cara untuk mengumpulkan data, salah satunya dengan crawling data. *Crawling* adalah teknik yang digunakan untuk mengumpulkan data pada sebuah website dengan cara memanfaatkan *Uniform Resource Locator* (URL) sebagai referensi untuk mengidentifikasi *hyperlink* di dalamnya (Saputra & Kusnadi, 2021). Dengan *crawling* pengambilan data dapat dilakukan dengan menggunakan API yang disediakan oleh platform yang datanya akan digunakan.

2.4 Klasifikasi

Klasifikasi teks merupakan pengelompokan dokumen berdasarkan kategori atau topik yang menjadi komponen penting dalam sistem pemrosesan teks . Banyak penelitian yang memanfaatkan kata-kata yang muncul dalam dokumen untuk mendapatkan klasifikasi yang akurat (Srivastava & Sahami, 2009).

Terdapat beberapa algoritma untuk melakukan pemodelan pada klasifikasi teks, satu diantaranya adalah *Support Vector Machines* (SVM). Sebelum memasuki tahap pemodelan, data harus melalui tahap *preprocessing*. *Data preprocessing*

dilakukan dengan mengubah data ke format yang lebih terstruktur dan mudah dianalisis. *Preprocessing* juga dilakukan untuk membersihkan data dari *noise* dan menyusunnya agar lebih terstruktur sehingga dapat diproses ke tahap selanjutnya. (Swastika et al., 2023). Berikut merupakan tahapan-tahapan *preprocessing*:

2.4.1 Case Folding

Case folding merupakan tahap dari *preprocessing text mining* yang dilakukan dengan melakukan standarisasi teks ke format yang sama. Hal tersebut bertujuan untuk mengurangi variasi data dan membuatnya menjadi lebih konsisten. Salah satu caranya yaitu dengan mengubah ke huruf kecil (Jurafsky & Martin, n.d.). *Case folding* ke *lower case* dilakukan dengan melakukan transformasi pada semua huruf yang ada dalam dokumen ke huruf kecil.

Kalimat "Mengejutkan Anak Kecil Ini Bisa Berbicara dlm Banyak Bahasa", setelah dilakukan *case folding* akan menjadi "mengejutkan anak kecil ini bisa berbicara dlm banyak bahasa".

2.4.2 Data Cleaning

Pada *Data Cleaning* karakter yang bukan huruf akan dihilangkan atau dibersihkan. Seperti tanda baca, *username (@username*), URL dan simbol-simbol lain yang tidak digunakan (Swastika et al., 2023)

Kalimat "Mengejutkan!!! Anak Kecil Ini Bisa Berbicara dlm Banyak Bahasa", setelah dilakukan *case folding* akan menjadi "Mengejutkan Anak Kecil Ini Bisa Berbicara dlm Banyak Bahasa".

2.4.3 Normalization

Normalization dalam konteks ini berarti mentransformasikan kata tidak baku (slang) dan kata yang disingkat menjadi bentuk formal atau baku. N*ormalization* dilakukan untuk mengurangi variasi kata dan menjaga konsistensi data (Saputra & Kusnadi, 2021). Karena kata "gak" "ga" "nggak" dan "tidak" yang sebenarnya memiliki makna yang sama akan dinilai berbeda apabila normalisasi tidak dilakukan.

Kalimat “mengejutkan anak kecil ini bisa berbicara dlm banyak bahasa”, akan diubah menjadi “mengejutkan anak kecil ini bisa berbicara dalam banyak bahasa”.

2.4.4 Tokenizing

Tokenizing dilakukan dengan memecah teks ke bagian-bagian kecil atau yang dikenal dengan istilah token (Jurafsky & Martin, n.d.). Salah satu cara melakukan *tokenizing* yaitu dengan memecah kalimat menjadi kata. *Tokenizing* penting dilakukan karena dapat mempermudah analisis teks. Karena nantinya algoritma akan menghitung frekuensi kemunculan kata dalam dokumen.

Kalimat “mengejutkan anak kecil ini bisa berbicara dalam banyak bahasa” setelah dilakukan tokenizing akan menjadi "mengejutkan," "anak," "kecil," "ini," "bisa," "berbicara," "dalam," "banyak," dan "bahasa."

2.4.5 Filtering

Filtering dilakukan dengan menyaring kata-kata yang tidak diperlukan (*stop words*) atau kata dengan frekuensi kemunculan yang tinggi seperti kata “ini”, “itu”, “yang”, dan lain-lain. Kata-kata tersebut perlu dihapus karena tidak membawa makna signifikan sehingga tidak memberikan informasi yang penting (Saputra & Kusnadi, 2021).

Setelah dilakukan *filtering*, kata “ini” dan “dalam” akan dihapus sehingga kata yang tersisa adalah “mengejutkan anak kecil bisa berbicara banyak bahasa”.

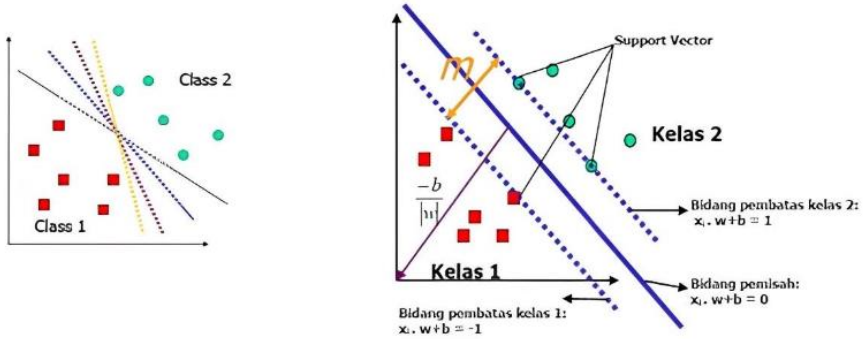
2.4.6 Stemming

Proses *stemming* dilakukan dengan mentransformasikan sebuah kata ke dalam bentuk dasarnya (Swastika et al., 2023). *Stemming* dilakukan untuk mengurangi variasi kata. Kata “memakan”, “dimakan”, “termakan” akan dianggap sebagai satu kata yaitu “makan”.

Setelah dilakukan proses *stemming* kalimat yang awalnya “Mengejutkan!!! Anak Kecil Ini Bisa Berbicara dlm Banyak Bahasa” menjadi “kejut anak kecil bisa bicara banyak bahasa”.

2.5 Support Vector Machine

Algoritma *Support Vector Machine* (SVM) pertama kali diperkenalkan oleh Aronszajn pada tahun 1950 sebagai metode klasifikasi dalam *supervised learning* yang menggunakan konsep *hyperplane*, *kernel* dan konsep pendukung lainnya. SVM dapat menyelesaikan masalah klasifikasi baik linear maupun non-linear. Tujuan dari algoritma ini adalah untuk menentukan *hyperplane* terbaik yang dapat memisahkan dua kelas dalam ruang input (Agustian & Nazir, 2024).



Gambar 2.1 Grafik *Support Vector Machine* (Agustian & Nazir, 2024)

Hyperplane pada SVM dapat dicari dengan menggunakan persamaan (1).

$$(w.x_i) + b = 0 \quad (1)$$

Data x_i yang termasuk dalam kelas -1 dapat dijelaskan dengan menggunakan rumus pada persamaan (2) seperti berikut.

$$(w.x_i + b) = 0 \leq 1, y_1 = -1 \quad (2)$$

Data x_i yang termasuk dalam kelas +1 dapat dijelaskan dengan menggunakan rumus pada persamaan (3).

$$(w.x_i + b) = 0 \geq 1, y_i = 1 \quad (3)$$

Keterangan:

w : *weight vector* atau vektor bobot

b : bias atau *intercept*

x_i : vektor fitur dari titik data i

y_i : label kelas dari titik data i

2.6 Pembobotan Kata

Pembobotan kata dapat dilakukan dengan berbagai metode, satu diantaranya yaitu menggunakan *Term Frequency-Inverse Document Frequency*

(TF-IDF). Metode TF-IDF dilakukan melalui menggabungkan 2 pendekatan, yaitu mengukur seberapa sering kemunculan kata dan melakukan invers berdasarkan jumlah dokumen yang memuat kata tersebut. TF-IDF dapat dihitung dengan menggunakan persamaan (1) dan (2) sebagai berikut.

$$IDF(t) = \log\left(\frac{N_d}{df(t)}\right) \quad (1)$$

$$TF - IDF(d,t) = f(d,t)\ x\ IDF(t) \quad (2)$$

Keterangan:

$f(d,t)$: Jumlah kemunculan term t pada dokumen d

$IDF(t)$: Nilai inverse frekuensi dokumen untuk term t

Nd : Jumlah total dokumen

$df(t)$: Jumlah dokumen yang mengandung term t

Term frequency (TF) dilakukan untuk menentukan bobot kata dalam sebuah dokumen berdasarkan seberapa sering kemunculan kata. Semakin sering kata tersebut muncul dalam dokumen, maka bobotnya akan semakin tinggi. Sehingga, dapat meningkatkan nilai kesesuaiannya. (Ramadhan et al., 2023).

Sementara itu, *Inverse document frequency* (IDF) bertujuan mengurangi bobot kata yang terlalu sering muncul dalam banyak dokumen. Semakin sering kemunculan suatu kata pada banyak dokumen, maka bobot yang diberikan akan semakin rendah (Ramadhan et al., 2023).

2.7 Evaluasi Model

Confusion matrix merupakan representasi data berbentuk tabel yang digunakan untuk mengevaluasi performa dari sebuah model. Evaluasi model adalah tahapan untuk mengetahui hasil performa suatu model (Risnasari, 2022). Hasil klasifikasi dicatat dengan memperhatikan kelas/label yang sesungguhnya dengan hasil percobaan yang dilakukan. *Confusion matrix* menggunakan parameter TP, FP, TN dan FN yang dalam bentuk tabel direpresentasikan sebagai tabel 2.2 berikut.

Tabel 2.2 *Confusion Matrix*

		Nilai Sesungguhnya	
		Positif	**Negatif**
Hasil Percobaan	**Positif**	TP	FP
	Negatif	FN	TN

Keterangan:

True Positive (TP) : Jumlah data berlabel positif yang diprediksi dengan benar sebagai positif

False Positive (FP) : Jumlah data berlabel positif yang diprediksi dengan salah sebagai negatif

True Negative (TN) : Jumlah data berlabel negatif yang diprediksi dengan benar sebagai negatif

False Negative (FN) : Jumlah data berlabel negatif yang diprediksi dengan salah sebagai positif

Melalui *confusion matrix* maka nilai akurasi, presisi, *recall* dan *F1-Score* bisa ditentukan. Pada data seimbang, *accuracy* baik untuk digunakan karena metrik ini menghitung proporsi prediksi yang benar pada kedua kelas terhadap seluruh data tanpa pengecualian. *Accuracy* menunjukkan sejauh mana model dapat melakukan klasifikasi pada data dengan tepat atau akurat (Saputra & Kusnadi, 2021). Berikut rumus untuk menghitung *accuracy*:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

BAB 3. METODOLOGI PENELITIAN

Bab metodologi penelitian ini berisi mengenai lokasi dan waktu penelitian, jenis penelitian, data dan objek penelitian, serta prosedur penelitian.

3.1 Lokasi dan Waktu Penelitian

Penelitian ini berlangsung di lingkungan Fakultas Ilmu Komputer Universitas Jember, dengan kurun waktu sekitar empat bulan, terhitung dari bulan Maret 2025 hingga Juni 2025.

3.2 Jenis Penelitian

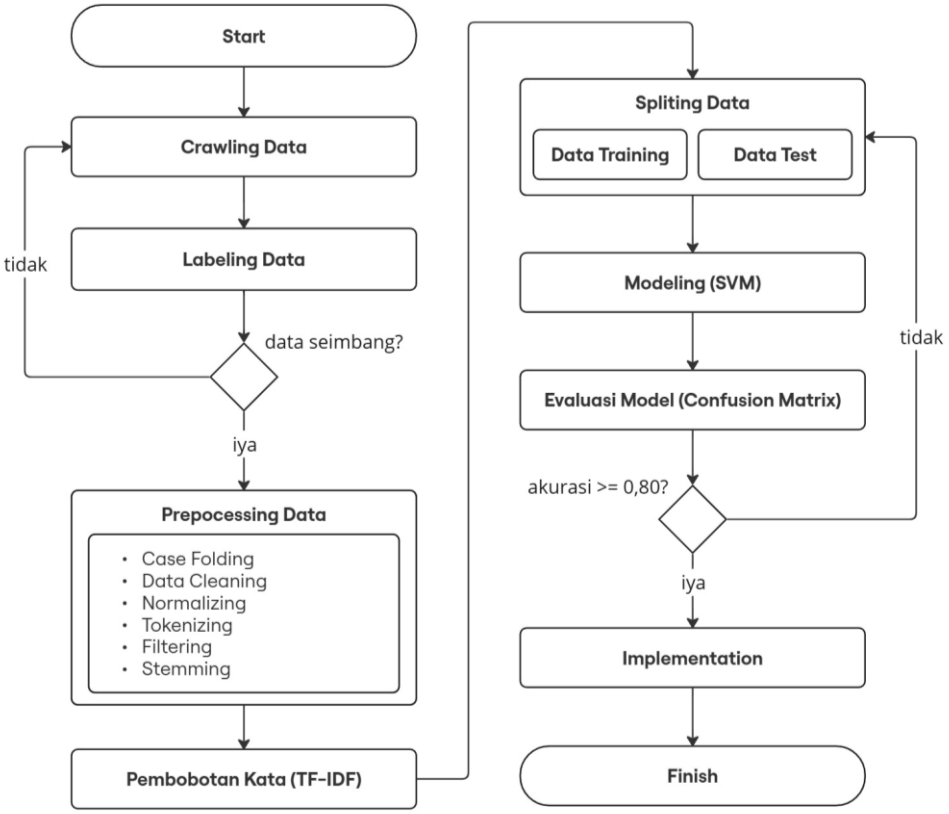
Jenis penelitian yang digunakan dalam penelitian ini adalah implementatif dengan pendekatan kuantitatif, dimana penggunaan data dapat diukur dan dianalisis secara statistik. Metode yang digunakan yaitu algoritma *Support Vector Machine* (SVM) untuk menghasilkan model klasifikasi *clickbait*, yang hasilnya diimplementasikan ke sebuah aplikasi berbasi web yang secara otomatis dapat mendeteksi judul *clickbait*.

3.3 Data dan Objek Penelitian

Sumber data yang digunakan untuk objek penelitian pada pemodelan klasifikasi *clickbait* didapatkan dengan melakukan crawling data judul video *YouTube* yang dilabeli secara manual dengan 0 sebagai *non-clickbait* dan 1 sebagai *clickbait*. Pelabelan dilakukan oleh lima orang dengan menggunakan sistem voting, dimana label ditentukan berdasarkan voting terbanyak.

3.4 Prosedur Penelitian

Gambar 3.1 berikut menampilkan ilustrasi prosedur yang dilakukan pada penelitian ini:



Gambar 3.1 Prosedur Penelitian

3.4.1 Pengumpulan Data

Proses pengambilan data dilakukan dengan *crawling data* pada *YouTube* menggunakan API *YouTube*. Pada pengumpulan data, kemungkinan akan terjadi perulangan apabila hasil pelabelan data yang didapatkan belum seimbang antar kelas *clickbait* dan *non-clickbait*. Perulangan akan terus dilakukan hingga data yang didapatkan sudah cukup seimbang.

3.4.2 Pelabelan Data

Pelabelan dilakukan oleh lima orang dimana semua orang mendapatkan seluruh data dan masing-masing orang melabeli seluruh data yang didapat. Pelabelan didasarkan pada keberadaan fitur-fitur dalam kategori *indicative* dan *partially indicative*. Berikut beberapa fitur-fitur yang dapat dikategorikan sebagai *clickbait* berdasarkan artikel (Kemm, 2022) disajikan dalam tabel 3.1 berikut:

Tabel 3.1 Fitur-Fitur Judul *Clickbait (Kemm, 2022)*

No	Fitur *Clickbait*	Contoh	Penjelasan
1	Kata Ganti Personal & Demonstratif	*She Surprises Me For Valentines Day!!!*	Menimbulkan rasa relevansi dan rasa ingin tahu melalui kata ganti personal.
2	Superlatif & Intensifier	*Zenfone 8 – The most boring INCREDIBLE phone ever*	Menekankan klaim berlebihan dan memperkuat kesan dramatis.
3	*Forward-reference*	*What happens next WILL SHOCK you*	Menyiratkan informasi penting yang menciptakan rasa penasaran.
4	*Hyperbolic Words & Strong Lexical Units*	*Tik Toks That Will Melt your Brain*	Menggunakan kata-kata yang berlebihan atau sensasional.
5	*Emotional Appeal*	*Why Everyone In China Hates Me…*	Menyentuh emosi pembaca (marah, sedih, penasaran).
6	*Numerals* (Angka)	*Top 10 MOST INSANE Sandcastles EVER BUIL*	Menggunakan format list dan angka yang menciptakan ekspektasi.
7	Modals (Kata Kerja Modal)	*Inventions That Will CHANGE YOUR LIFE!*	Menambahkan unsur kepastian atau urgensi dalam klaim.
8	Kosakata Familiar / Trending	*Last To Stop Playing AMONG US Wins $10,000 – Challenge*	Menarik minat pembaca melalui istilah yang sedang populer.

Kemudian seluruh label yang telah tuliskan akan ditetapkan berdasarkan voting terbanyak menjadi label final. Tabel 3.2 berikut merupakan contoh tampilan hasil pelabelan:

Tabel 3.2 Contoh Tampilan Hasil Pelabelan

no	*title*	label					final label
		p1	p2	p3	p4	p5	
1	Spongebob | Momen-momen Paling DRAMATIS Pearl | Nickelodeon Bahasa	1	0	0	0	1	0
2	NGAKAK BANGET! Semua Capek Ngeladenin Haji Bolot & Limbad - Family 100 Kilau Konser	1	0	1	1	1	1
3	TEREKAM CCTV!!! Nampak Hantu KAYANG, Paling Berbahaya di Kampungku!!!	1	1	1	1	1	1
4	Rebutan ayunan sampe nyangkut di atap rumah	1	0	1	0	0	0
5	YUKA-CHAN DIGANGGU HANTU WAKTU LAGI TIDUR! TEREKAM CCTV! | *a day in our life*	1	1	1	1	0	1

3.4.3 Keseimbangan Data

Hasil pelabelan kemudian diperiksa untuk memastikan keseimbangan jumlah data pada setiap kelas. Apabila jumlah data antara kelas masih belum seimbang, maka akan dilakukan pengumpulan data kembali (tahapan 3.4.1), yang kemudian dilanjutkan dengan proses pelabelan ulang (tahapan 3.4.2). Dalam penelitian ini, target jumlah judul untuk setiap kelas labelnya sekitar 1000 judul.

3.4.4 Preprocessing

Pada tahap *preprocessing*, data akan dibersihkan terlebih dahulu untuk menghilangkan komponen yang mengganggu atau tidak relevan, misalnya karakter khusus, angka, dan tanda baca yang tidak diperlukan. *Preprocessing data* juga dilakukan agar data yang digunakan menjadi lebih bersih dan terstuktur. Pada tahap ini peneliti melakukan *Case Folding*, *Data Cleaning*, *Normalization,* Tokenisasi, *Filtering,* dan *Stemming.*

a. Case Folding

Proses *case folding* dilakukan penyamaan format teks dengan cara mengubah seluruh huruf kapital menjadi huruf kecil. Proses ini bertujuan untuk menyederhanakan data dan memastikan konsistensi kata.

Meskipun dalam teori *clickbait* huruf kapital kerap dianggap sebagai salah satu ciri, pada praktiknya judul video di platform seperti *YouTube* umumnya ditulis menggunakan huruf kapital secara keseluruhan atau sebagian besar. Hal ini menyebabkan kapitalisasi tidak lagi dapat dijadikan acuan dalam membedakan judul *clickbait* dan *non-clickbait.* Oleh karena itu, proses *case folding* tetap dilakukan menggunakan fungsi *lower().*

b. Cleaning Data

Pada tahap ini dilakukan penghapusan bagian yang tidak dibutuhkan seperti karakter khusus, angka, tanda baca, dan emoji dari teks. Proses ini dilakukan dengan memanfaatkan pola *RegEx* seperti re.sub(r'[^\w\s]', ' ', text) untuk menghapus tanda baca dan karakter khusus, serta re.sub(r'\d+', '', text) untuk menghapus angka.

Tanda seru dan emoji memang sering muncul dalam judul *clickbait*. Namun, dalam penelitian ini karakter-karakter tersebut tidak dipertahankan karena ditemukan banyak kasus di mana judul video yang tidak tergolong *clickbait* tetap menggunakan tanda seru atau emoji. Oleh karena itu, elemen tersebut tidak dianggap sebagai indikator yang konsisten dalam membedakan antara *clickbait* dan *non-clickbait*. Contoh judul seperti "Selamat Hari Kemerdekaan ke-78! 🇮🇩" mengandung emoji dan tanda seru, tetapi tidak dapat dikategorikan sebagai *clickbait*.

c. Normalization

Pada tahap *normalization* kata - kata slang akan diubah menjadi kata yang baku. Kata - kata yang disingkat juga akan diubah menjadi kata yang utuh. *Normalization* dilakukan dengan memanfaatkan kamus new_*kamusalay.csv* pada *GitHub* (Ibrohim & Budi, 2019) yang dapat diunduh dari laman https://github.com/okkyibrohim/id-multi-label-hate-speech-and-abusive-language-detection.

d. Tokenizing

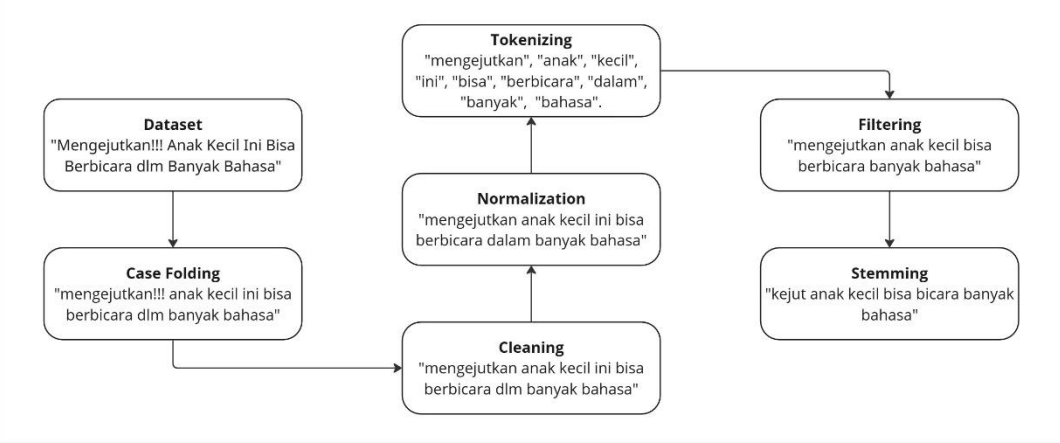
Pada tahap *tokenizing* teks dipecah menjadi kata individu menggunakan *library nltk.tokenize* dengan *function word_tokenize*. Kemudian dilakukan tokenisasi untuk memecah kalimat menjadi unit yang lebih kecil disebut token. Tokenisasi pada penelitian ini dilakukan dengan memecah kalimat menjadi kata.

e. Filtering

Dalam proses *filtering* kata-kata yang tidak penting dihapus menggunakan *library nltk.corpus import stopwords*. Namun, kata "ini" dan "inilah" yang seharusnya dihapus pada tahap *filtering* tetap dipertahankan pada tahap ini karena dianggap sebagai salah satu karakteristik khas dari judul *clickbait*. Contohnya seperti pada judul "Kamu Tidak Akan Percaya Apa yang Terjadi di Video Ini" dan "Inilah Tradisi Pernikahan Terunik di Thailand".

f. Stemming

Tahap terakhir dari *preprocessing data* yaitu *stemming*. Tahap ini dilakukan untuk menghilangkan imbuhan sehingga hanya tersisa kata dasar saja menggunakan *stemmer.factory* pada *library sastrawi*.



Gambar 3.2 Tahapan Preprocessing Data

3.4.5 Pembobotan Kata

Metode yang digunakan untuk pembobotan kata ialah *Term Frequency-Inverse Document Frequency* (TF-IDF). Tahapan ini penting dilakukan untuk mengetahui apakah sebuah kata memiliki pengaruh dalam sebuah klasifikasi teks. TF-IDF dilakukan dengan menggunakan *library scikit-learn* dengan *import TfidfVectorizer* pada *sklearn.feature_extraction.text.*

3.4.6 Spliting Data

Sebelum pembuatan model, perlu dilakukan pembagian data (*split data*). Data dibagi menjadi dua bagian, yaitu *data training* dan *data test* dengan tiga rasio perbandingan untuk melihat perbandingan akurasi dari ketiga rasio perbandingan tadi. Ketiga rasio perbandingan tersebut yakni 70:30, 80:20, dan 90:10. Pembagian data latih dan data uji dilakukan menggunakan fungsi *train_test_split* dari library *sklearn.model_selection.*

3.4.7 Modeling

Kemudian model klasifikasi dibangun menggunakan algoritma SVM dengan kernel linear. Penggunaan kernel linear dipilih karena sesuai untuk data yang dapat dipisahkan secara linear. Kemudian, model terlatih disimpan dalam

format .pkl menggunakan *library joblib* agar dapat diimplementasikan ke sebuah sistem nantinya.

3.4.8 Evaluasi Model

Evaluasi model dilakukan dengan menggunakan *library scikit-learn* untuk mengukur kinerja model. *Confusion matrix* digunakan untuk menampilkan total prediksi yang benar dan prediksi yang salah untuk setiap kelas yaitu *clickbait* dan *non-clickbait* yang nantinya dapat digunakan untuk menghitung nilai akurasi, presisi, *recall, dan F1-Score.* Semua metrik tersebut dapat dihitung dengan memanfaatkan *function* yang disediakan oleh *sklearn.metrics.*

Karena data telah seimbang, langkah berikutnya adalah melakukan evaluasi dengan menghitung nilai akurasi dari model yang telah dibuat. Dari ketiga rasio pembagian data sebelumnya, akan dipilih model dengan rasio yang memberikan nilai akurasi terbaik dengan nilai ≥ 0,80. Nilai 0,80 dijadikan patokan karena akurasi dengan nilai 0.80 – 0.90 dapat dikategorikan sebagai klasifikasi yang baik (Trishnanti & Al Azies, n.d.).

Dalam jurnalnya, Trishnanti dan Al Aziez (2019) mengkategorikan nilai akurasi model klasifikasi ke dalam beberapa kategori untuk menilai tingkat keberhasilan model. Klasifikasi nilai akurasi tersebut ditampilkan pada Tabel 3.3 berikut:

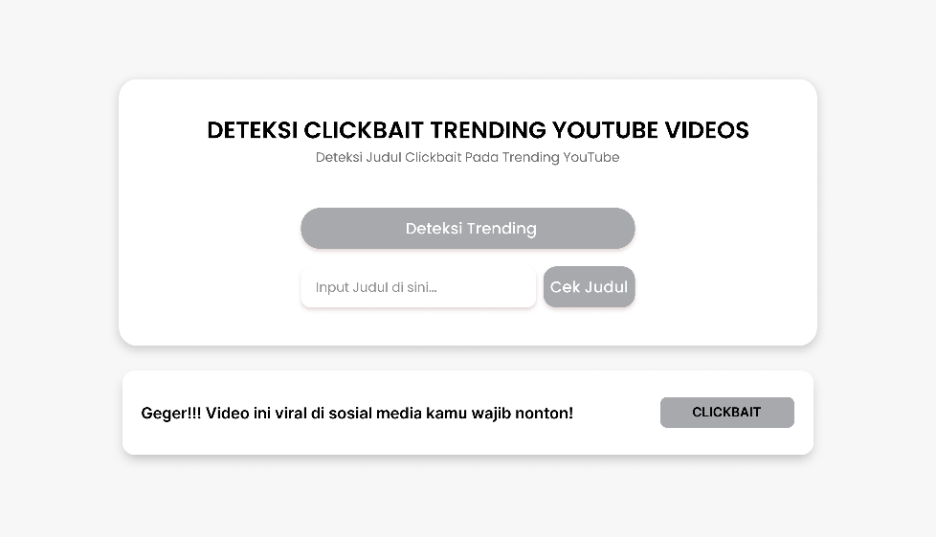
Tabel 3.3 Klasifikasi nilai akurasi *(Trishnanti & Al Azies, n.d.)*

Rentang Akurasi	**Kategori Klasifikasi**
0.90 – 1.00	*Very good classification*
0.80 – 0.90	*Good classification*
0.70 – 0.80	*Enough classification*
0.60 – 0.70	*Bad classification*
0.50 – 0.60	*Wrong classification*

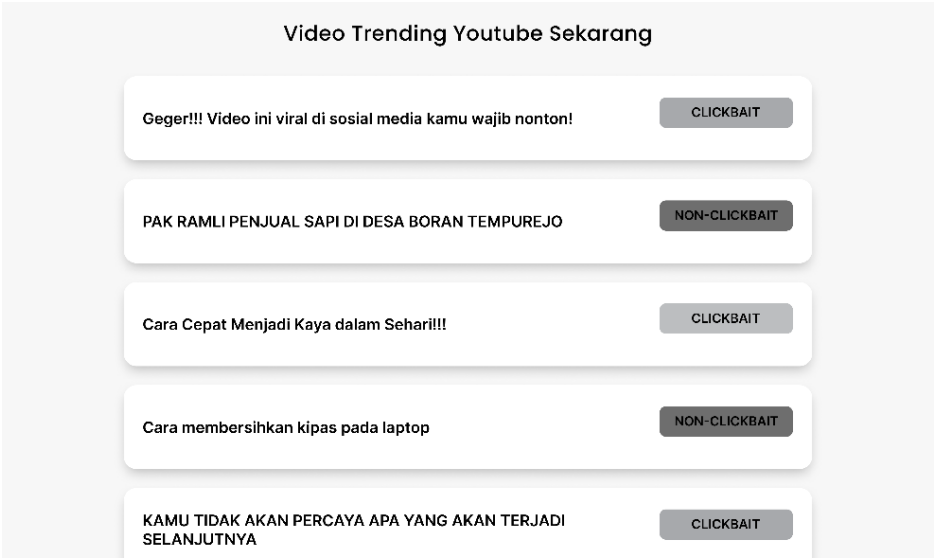
3.4.9 Implementation

Pada fase ini model yang telah dibangun akan diterapkan ke dalam sebuah aplikasi berbasis web, sehingga dapat diketahui apakah hasil yang diperoleh dari aplikasi sesuai dengan tujuan yang diinginkan. Model akan diimplementasikan ke sebuah aplikasi berbasis web yang dikembangkan menggunakan *flask*. Sistem ini

bertujuan untuk mendeteksi judul video pada *trending Youtube* apakah termasuk dalam kategori *clickbait* atau *non-clickbait.*



Gambar 3.3 Gambaran *Home Page*



Gambar 3.4 Gambaran *Detect Page*

Langkah pertama adalah memuat model yang telah disimpan termasuk *model preprocessing*, *TF-IDF vectorizer*, dan model SVM menggunakan *function joblib.load. Pipeline preprocessing* dimuat untuk mengolah teks judul video. Setelah *model* dan *pipeline* dimuat, sistem mengambil data video *YouTube trending* melalui YouTube API dan memproses judulnya dengan *function preprocess_text*. Judul yang telah melewati tahap *preprocessing* akan dikonversi menjadi bentuk

numerik dengan menggunakan *tfidf_vectorizer*. Setelah data diproses dan diubah ke format yang sesuai, model SVM digunakan untuk mendeteksi label setiap judul video apakah termasuk *clickbait* atau *non-clickbait*. Sistem kemudian menampilkan hasil deteksi pada web.

BAB 4. HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil dan pembahasan dari setiap tahapan yang dilakukan dalam penelitian. Langkah pertama mencakup proses pengumpulan dan pelabelan dataset, serta pengecekan keseimbangan kelas antara data *clickbait* dan *non-clickbait*. Tahap berikutnya adalah *preprocessing* data dan penentuan bobot kata yang dilakukan melalui pendekatan TF-IDF. Kemudian, dataset dibagi untuk keperluan pelatihan dan pengujian model. Selanjutnya dilakukan pembuatan model klasifikasi, diikuti dengan evaluasi untuk menilai performa model. Terakhir, model yang telah dievaluasi diimplementasikan ke dalam sebuah aplikasi berbasis web.

4.1 Crawling Data

Pada tahap ini dilakukan pengumpulan data judul video *YouTube* menggunakan *API key* yang sudah dimiliki. Data yang dikumpulkan merupakan judul video berbahasa Indonesia yang diunggah pada platform YouTube dengan jumlah penayangan minimal 10.000 kali. Tabel 4.1 berikut merupakan beberapa contoh data yang didapat setelah dilakukan proses *crawling data*.

Tabel 4.1 Contoh Data *Crawling*

no	*title*
1	TEREKAM CCTV! Penampakan Hantu SADAKO/TEKE TEKE, Paling Sakti dan Ditakuti di Pulau ini!!!
2	TERTANGKAP CCTV!!! Nampak Hantu LAMPOR KERANDA, Paling Berbahaya di Desaku!!!
3	IBU HAMIL BERUNTUNG VS IBU HAMIL SIAL!!ðŸ˜± Drama Parodi | Mikael TubeHD
4	SISI BAIK TARI || KONTRAKAN REMPONG EPISODE 766
5	DRAMA - IBU HAMIL BESAR PINGSAN ðŸ¥´ SAAT MAU MELAHIRKAN | Salsa and Family
6	PART 2 || DRAMA - IBU HAMIL BESAR AKHIRNYA MELAHIRKAN ðŸ‘¶ PAPAH MALAH PINGSAN

Tahapan *crawling* dilakukan dalam tiga kali proses pengambilan data. Hal ini disebabkan karena pada tahap *labeling* setelah *crawling* pertama, diperoleh sebanyak 1.559 judul video dengan komposisi 557 judul *clickbait* dan 1.002 judul *non-clickbait*. Komposisi ini menunjukkan ketidakseimbangan jumlah data antar kelas. Meskipun jumlah judul *non-clickbait* sudah mencapai target (±1.000 data),

tetapi jumlah judul *clickbait* masih belum mencukupi. Oleh karena itu, dilakukan proses *crawling* kedua untuk menambah data.

Pada *crawling* kedua, total data meningkat menjadi 2.005 judul video, dengan rincian 850 judul *clickbait* dan 1.160 judul *non-clickbait*. Karena jumlah judul clickbait masih belum mencapai target, dilakukan *crawling* ketiga yang difokuskan pada pengambilan judul *clickbait* saja. Hasil *crawling* ketiga meningkatkan total data menjadi 2.062 judul video, dengan 902 judul *clickbait* dan 1.160 judul *non-clickbait.* Setelah proses *crawling* ketiga, proporsi data antar kelas akhirnya seimbang dan sesuai dengan jumlah yang ditargetkan.

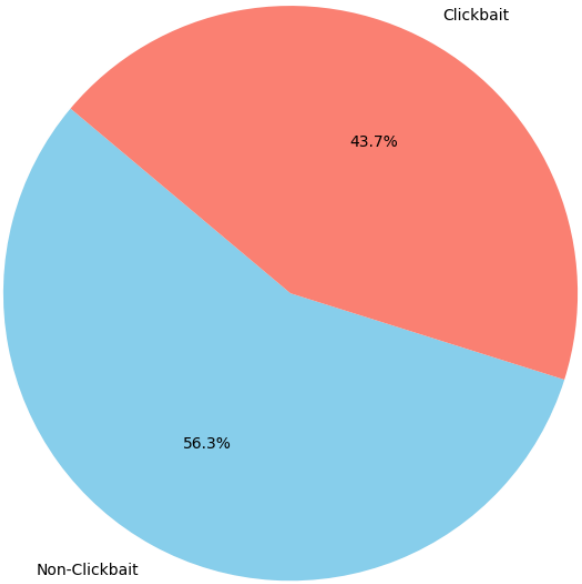
4.2 Labeling Data

Proses pelabelan data yang dilakukan oleh lima orang yang secara bersama memberikan label dari setiap hasil *crawling data*. Tabel 4.2 berikut merupakan contoh hasil pelabelan dan penentuan final label menggunakan teknik voting.

Tabel 4.2 Hasil Voting

no	*title*	label					final label
		p1	p2	p3	p4	p5	
1	Spongebob | Momen-momen Paling DRAMATIS Pearl | Nickelodeon Bahasa	1	0	0	0	1	0
2	NGAKAK BANGET! Semua Capek Ngeladenin Haji Bolot & Limbad - Family 100 Kilau Konser	1	0	1	1	1	1
3	TEREKAM CCTV!!! Nampak Hantu KAYANG, Paling Berbahaya di Kampungku!!!	1	1	1	1	1	1
4	Rebutan ayunan sampe nyangkut di atap rumah	1	0	1	0	0	0
5	YUKA-CHAN DIGANGGU HANTU WAKTU LAGI TIDUR! TEREKAM CCTV! | a day in our life	1	1	1	1	0	1

Berdasarkan hasil voting, diperoleh sebanyak 902 judul video dengan label *clickbait* (1) dan 1160 judul video dengan label *non-clickbait* (0) dari total 2062 judul video. Berikut merupakan diagram lingkaran untuk melihat persentase perbandingan datanya.



Gambar 4.1 Distribusi Persentase Label *Clickbait* dan *Non-Clickbait*

4.3 Preprocessing Data

Tahap ini mencangkup beberapa tahapan, diantaranya adalah *case folding, cleaning data, normalizing, tokenizing, filtering,* dan *stemming.*

4.3.1 *Case Folding*

Pada tahap ini, dilakukan proses *case folding* terhadap data judul video menggunakan fungsi *lower()*. Implementasi fungsi *case folding* ditunjukkan pada potongan kode berikut:

```
def case_folding(text):
        text = text.lower()
        return text
```

Fungsi *case_folding()* akan menerima input berupa teks dan mengembalikannya dalam bentuk huruf kecil. Setelah dilakukan proses ini, semua karakter alfabet dalam teks telah diseragamkan. Tabel 4.3 berikut menyajikan perbandingan antara judul sebelum serta sesudah proses *case folding.*

Tabel 4. 3 Hasil *Case Folding*

***Title* Sebelum**	***Title* Sesudah**
TEREKAM CCTV!!! Nampak Hantu KAYANG, Paling Berbahaya di Kampungku	terekam cctv!!! nampak hantu kayang, paling berbahaya di kampungku
NGAKAK BANGET! Semua Capek Ngeladenin Haji Bolot & Limbad - Family 100 Kilau Konser	ngakak banget! semua capek ngeladenin haji bolot & limbad - family 100 kilau konser

Dalam tabel 4.3 terlihat bahwa seluruh karakter berupa huruf kapital telah berubah menjadi huruf kecil. Sebagai contoh, kata "TEREKAM" menjadi "terekam". Hal ini membuktikan bahwa fungsi *lower()* yang digunakan telah bekerja seperti yang diharapkan.

4.3.2 *Cleaning Data*

Pada *cleaning data*, dilakukan proses pembersihan data judul video dari elemen-elemen yang tidak relevan menggunakan fungsi *Regex*. Elemen yang dihapus meliputi *hashtag*, *mention*, tanda baca, angka, dan spasi yang berlebih. Implementasi fungsi *cleaning data* ditunjukkan pada potongan kode berikut:

```
def clean_text(text):
        text = re.sub(r'#\w+', '', text)
        text = re.sub(r'@\w+', '', text)
        text = re.sub(r'[^\w\s]', ' ', text)
        text = re.sub(r'\d+', '', text)
        text = re.sub(r'\s+', ' ', text).strip()
        return text
```

Fungsi *clean_text()* menerima input berupa teks, kemudian membersihkannya dari berbagai karakter yang tidak diperlukan dengan bantuan pustaka re (*regular expression*), lalu mengembalikannya dalam bentuk teks yang lebih bersih. Tabel 4.4 berikut menyajikan perbandingan antara judul sebelum serta sesudah proses *cleaning data*.

Tabel 4.4 Hasil *Cleaning Data*

***Title* Sebelum**	***Title* Sesudah**
yuta mio panik cctv merekam anomali tung tung tung sahur di depan rumah😱 sakura school simulator	yuta mio panik cctv merekam anomali tung tung tung sahur di depan rumah sakura school simulator
saputra kecelakaan😭 semua sedih lihat kondisi saputra kakinya patah?? | mikael tubehd	saputra kecelakaan semua sedih lihat kondisi saputra kakinya patah mikael tubehd

Dalam tabel 4.4 terlihat bahwa elemen-elemen yang tidak diperlukan seperti tanda baca, angka, *hashtag*, *mention*, dan spasi berlebih telah berhasil dihapus dari teks. Sebagai contoh tanda baca "?" dan emoji "😭" dihapuskan setelah *cleaning data* dilakukan. Proses ini menunjukkan bahwa fungsi *RegEx* yang digunakan telah berhasil menghapus karakter yang tidak relevan dari teks.

4.3.3 *Normalizing*

Pada tahap ini, dilakukan normalisasi terhadap data judul video dengan memanfaatkan kamus *new_kamusalay.csv*. Implementasi fungsi normalisasi ditunjukkan pada potongan kode berikut:

```
def normalize_text(text, alay_dict):
        words = text.split()
        return ' '.join([alay_dict[word] if word in alay_dict else word for
word in words])
```

Fungsi *normalize_text()* akan membandingkan setiap kata pada teks dengan isi kamus *new_kamusalay.csv*. Jika ditemukan padanan dalam kamus, kata tersebut akan diganti dengan bentuk bakunya. Jika tidak, kata akan dibiarkan sebagaimana adanya. Tabel 4.5 berikut menyajikan perbandingan antara judul sebelum serta setelah proses normalisasi.

Tabel 4.5 Hasil *Normalizing*

***Title* Sebelum**	***Title* Sesudah**
terekam cctv nampak hantu kayang, paling berbahaya di kampungku	terekam cctv menampak hantu kayang paling berbahaya di kampungku
ngakak banget semua capek ngeladenin haji bolot limbad family kilau konser	tertawa banget semua cape meladeni haji bolot limbad famili kilau konser

Dalam tabel 4.5 terlihat bahwa kata tidak baku atau kata slang telah berhasil diubah ke dalam bentuk baku. Sebagai contoh, kata “ngakak” berubah menjadi “tertawa” dan “capek” menjadi “cape”. Proses ini menunjukkan bahwa fungsi normalisasi dengan memanfaatkan kamus *new_kamusalay.csv* telah berjalan sesuai dengan yang diharapkan.

4.3.4 *Tokenizing*

Pada tahap ini, dilakukan proses tokenisasi terhadap data judul video. Implementasi fungsi tokenisasi ditunjukkan pada potongan kode berikut:

```
def tokenizing(text):
        return word_tokenize(text)
```

Fungsi *tokenizing* akan memecah kalimat ke bentuk kata dengan memanfaatkan fungsi *word_tokenize* dari *library nltk*. Tabel 4.6 berikut menyajikan perbandingan antara judul sebelum dan setelah proses *tokenizing*.

Tabel 4.6 Hasil *Tokenizing*

***Title* Sebelum**	***Title* Sesudah**
ratusan jet tempur rafale tiba di lombok indonesia dikirim dengan kapal induk charles deh gaulle	'ratusan', 'jet', 'tempur', 'rafale', 'tiba', 'di', 'lombok', 'indonesia', 'dikirim', 'dengan', 'kapal', 'induk', 'charles', 'deh', 'gaulle'
bacaan al alquran pengantar tidur surat al mulk ayat kursi ar rahman penenang hati dan pikiran cemas	'bacaan', 'al', 'alquran', 'pengantar', 'tidur', 'surat', 'al', 'mulk', 'ayat', 'kursi', 'ar', 'rahman', 'penenang', 'hati', 'dan', 'pikiran', 'cemas'

Dalam tabel 4.6 terlihat bahwa kalimat telah berhasil dipisahkan menjadi potongan-potongan kata (token). Sebagai contoh, kalimat “bacaan al alquran

pengantar tidur surat al mulk ayat kursi ar rahman penenang hati dan pikiran cemas" dipisahkan menjadi token-token: ['bacaan', 'al', 'alquran', 'pengantar', 'tidur', 'surat', 'al', 'mulk', 'ayat', 'kursi', 'ar', 'rahman', 'penenang', 'hati', 'dan', 'pikiran', 'cemas']. Proses ini menunjukkan bahwa fungsi *word_tokenize()* yang digunakan telah berjalan dengan baik untuk memisahkan kata per kata dalam teks.

4.3.5 *Filtering*

Pada tahap ini, dilakukan proses *filtering* untuk menghilangkan *stopwords* data judul video. Implementasi fungsi *filtering* ditunjukkan pada potongan kode berikut:

```
def filtering(tokens, exclude_words=None):
        stop_words = set(stopwords.words('indonesian') +
set(stopwords.words('english'))

        if exclude_words is not None:
                stop_words -= set(exclude_words)

        return [word for word in tokens if word not in stop_words and len(word)
```

Fungsi *filtering* dilakukan dengan memanfaatkan daftar *stopwords* dari *nltk.* Terdapat parameter *exlude_words* yang digunakan untuk mempertahankan istilah atau kata yang relevan serta signifikan dalam konteks *clickbait*. Tabel 4.7 berikut menyajikan perbandingan antara judul sebelum dan sesudah dilakukan *filtering*.

Tabel 4.7 Hasil *Filtering*

***Title* Sebelum**	***Title* Sesudah**
'jangan', 'lakukan', 'ini', 'pada', 'ban', 'motorcross', 'mu'	'lakukan', 'ini', 'ban', 'motorcross', 'mu'
'inilah', 'kapal', 'yang', 'bisa', 'membelah', 'es'	'inilah', 'kapal', 'membelah', 'es'

Dalam tabel 4.7 terlihat bahwa proses filtering telah berhasil menghapus kata-kata umum (*stopwords*). Sebagai contoh kata 'pada', 'yang', 'bisa' yang termasuk *stopwords* berhasil dihapus. Kata 'inilah' dan 'ini' berhasil dipertahankan meskipun termasuk dalam bagian *stopwords*. Proses ini menunjukkan bahwa fungsi *filtering()* berjalan sesuai dengan yang diharapkan.

4.3.6 *Stemming*

Pada *stemming* dilakukan proses penyederhanaan kata ke bentuk dasar terhadap data judul video. Implementasi fungsi *stemming* ditunjukkan pada potongan kode berikut:

```
def stemming(tokens):
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    return [stemmer.stem(token) for token in tokens]
```

Tabel 4.8 berikut menyajikan perbandingan antara judul sebelum dan setelah proses *stemming*.

Tabel 4.8 Hasil *Stemming*

***Title* Sebelum**	***Title* Sesudah**
'gara', 'gara', 'enak', 'banget', 'acara', 'kumpul', 'buatkan', 'kue', 'ini', 'ludes', 'sekejap'	'gara', 'gara', 'enak', 'banget', 'acara', 'kumpul', 'buat', 'kue', 'ini', 'ludes', 'kejap'
'drone', 'berhasil', 'menangkap', 'lampor', 'keranda', 'terbang', 'nyata', 'warga', 'desa', 'panik'	'drone', 'hasil', 'tangkap', 'lampor', 'keranda', 'terbang', 'nyata', 'warga', 'desa', 'panik'

Dalam tabel 4.8 terlihat bahwa proses *stemming* telah berhasil mengubah kata-kata menjadi bentuk dasar. Misalnya, seperti “buatkan”, “sekejap”, “menangkap”, dan “berhasil” diubah menjadi bentuk dasar: “buat”, “kejap”, “tangkap”, dan “hasil”. Proses ini menunjukkan bahwa fungsi *stemming()* yang digunakan telah berjalan dengan baik dalam menyederhanakan kata tanpa menghilangkan maknanya.

Setelah semua tahapan pada *preprocessing* data telah selesai dijalankan, kemudian hasil *preprocessing* disimpan dalam file CSV untuk diproses di tahap TF-IDF. Selanjutnya, *pipeline preprocessing* yang sudah dibangun disimpan dalam file PKL untuk digunakan kembali saat implementasi ke web.

4.4 Pembobotan Kata

Tahap berikutnya yaitu melakukan pembobotan kata menggunakan TF-IDF. Proses ini diimplementasikan menggunakan pustaka *scikit-learn*. Berikut merupakan langkah-langkah yang akan dilaksanakan:

a. Membaca data hasil *preprocessing* dari file *preprocessed.csv*.

```
input_file = os.path.join('..', 'data', 'preprocessed.csv')
df = load_data(input_file)
```

b. Melakukan transformasi data teks pada kolom *title* ke bentuk ruang vektor menggunakan class *TfidfVectorizer*.

```
def transform_tfidf(titles):
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(titles)
    return tfidf_matrix, vectorizer
```

c. Menyimpan hasil pembobotan ke dalam file *tfidf.csv* (untuk pembuatan model klasifikasi) dan obyek model vektorisasi ke dalam file *tfidf_vectorizer.pkl* (untuk diimplementasikan dalam aplikasi).

```
output_file = os.path.join('..', 'data', 'tfidf.csv')
save_tfidf_matrix(tfidf_matrix, tfidf_vectorizer, output_file)

tfidf_model_file = os.path.join('..', 'model', 'tfidf_vectorizer.pkl')
save_vectorizer(tfidf_vectorizer, tfidf_model_file)
```

Setelah data melewati proses pembobotan, maka tiap kata pada baris data akan memiliki nilai bobot. Contoh hasil pembobotan kata menggunakan TF-IDF dapat dilihat pada tabel 4.9 berikut.

Tabel 4.9 Contoh Hasil Pembobotan Kata Menggunakan TF-IDF

	akibat	**bikin**	**coldiac**	**dulunya**	**guitar**	**ini**
D1	0	0	0.408248	0	0.408248	0
D2	0	0	0	0.707107	0	0.707107
D3	0	0	0	0	0	0
D4	0	0.57735	0	0	0	0
D5	0.408248	0	0	0	0	0

Dalam tabel 4.9 terlihat bahwa pembobotan kata dengan TF-IDF telah berhasil dilakukan. Setiap nilai yang muncul di dalam tabel menunjukkan bobot atau tingkat kepentingan dari suatu kata terhadap masing-masing dokumen. Sebagai

contoh, kata “dulunya” dan “ini” memiliki bobot sebesar 0.707107 pada D2, yang artinya kedua kata tersebut dianggap penting hanya pada dokumen dua dan tidak muncul pada dokumen lainnya.

4.5 Spliting Data

Tahap selanjutnya adalah melakukan pembagian dataset menjadi 2 subset yaitu data *training* dan data *test*, dengan memanfaatkan fungsi *train_test_split* dari *library scikit-learn*. Pembagian dilakukan secara acak dengan parameter *random_state* = 42.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Pembagian dataset dilakukan dengan tiga skenario yang ditampilkan pada tabel 4.10 berikut.

Tabel 4.10 Skenario *Spliting Data*

Skenario	Rasio	Jumlah Data Latih	Jumlah Data Uji
1	7:3	1443	619
2	8:2	1649	413
3	9:1	1856	206

Dalam tabel 4.10 terlihat bahwa pembagian data ke dalam tiga skenario nantinya dapat menunjukkan adanya perbedaan tingkat akurasi saat data diuji menggunakan model *Support Vector Machine* dan TF-IDF. Variasi pembagian data ini menjadi dasar dalam mengevaluasi performa masing-masing model untuk menentukan hasil yang paling optimal.

4.6 Modeling

Tahap selanjutnya adalah membangun model klasifikasi dengan menggunakan algoritma SVM. Untuk mengetahui pengaruh rasio pembagian data terhadap performa model, dilakukan tiga skenario pembagian data *training* dan data *test* yakni 70:30, 80:20, dan 90:10. Masing-masing skenario dilakukan pelatihan dan pengujian model secara terpisah.

Model SVM dibangun menggunakan kernel linear (kernel='linear') karena cocok untuk klasifikasi biner pada data teks. Proses pelatihan dilakukan dengan

metode *.fit()* pada data latih hasil masing-masing pembagian. Berikut merupakan kode program untuk melakukan pemodelan:

```
def train_svm(X_train, y_train):
    model = SVC(kernel='linear')
    model.fit(X_train, y_train)
    return model
```

Setelah proses pelatihan selesai untuk masing-masing rasio, dilakukan prediksi terhadap data uji. Hasil prediksi inilah yang kemudian digunakan untuk mengevaluasi performa model pada tahap Evaluasi Model. Tabel 4.11 berikut menampilkan contoh hasil prediksi model:

Tabel 4.11 Contoh Hasil Prediksi

Title	**Label Aktual**	**Label Prediksi**
Tali Termahal Di Dunia	1	1
PERSAHABATAN SUSI DAN YUNI !!!	0	0
tradisi pernikahan unik adat Thailand â€ ¼ï¸• â€ ¼ï¸• â€ ¼ï¸•	1	1
WARGA NEGARA UNI SOVIET TERAKHIR YANG BERADA DI LUAR ANGKASA | SERGEI KRIKALEV	1	1
GILGA SAHID FT HAPPY ASMARA â€ “ IKI WEKE SOPO (Official Live Music Video)	0	0

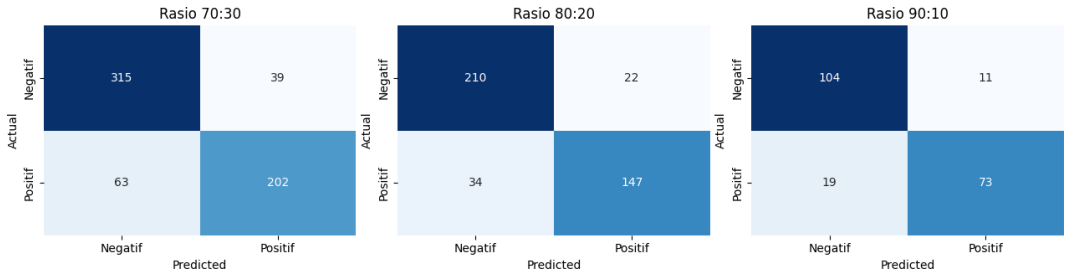
Dalam tabel 4.11 terlihat bahwa model klasifikasi telah berhasil membedakan antara judul *clickbait* dan *non-clickbait* dengan baik. Setiap label prediksi yang ditampilkan sesuai dengan label sebenarnya, yang berarti model mampu mengenali pola-pola *clickbait*. Sebagai contoh, judul “Tali Termahal Di Dunia” diprediksi sebagai *clickbait* (1) dan sesuai dengan label aktualnya.

4.7 Evaluasi Model

Performa model dinilai dengan berdasarkan pada metrik evaluasi berupa akurasi. Metrik ini dipilih karena distribusi label pada dataset tergolong seimbang antara kelas *clickbait* dan *non-clickbait*. Selain akurasi, *confusion matrix* juga ditampilkan untuk memberikan gambaran jumlah prediksi benar dan salah. Berikut merupakan kode program untuk melakukan evaluasi model:

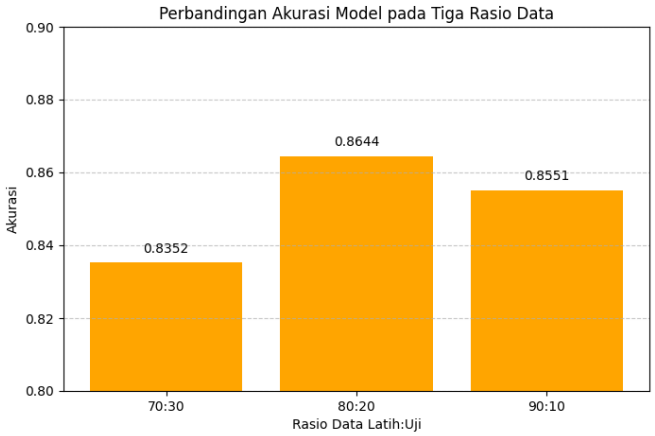
```
def evaluate_model(model, X_test, y_test):
    y_pred = model.predict(X_test)
    conf_matrix = confusion_matrix(y_test, y_pred)

    correct = conf_matrix[0][0] + conf_matrix[1][1]  # TP + TN
    total = conf_matrix.sum()
    accuracy = correct / total

    return y_pred, conf_matrix, accuracy
```

Proses evaluasi model menggunakan fungsi *evaluate_model* dilakukan dengan memanfaatkan *confusion matrix* dari *library sklearn*. *confusion_matrix* memberikan gambaran jumlah prediksi benar dan salah pada masing-masing kelas yang kemudian digunakan untuk menghitung nilai akurasi. Gambar 4.2 berikut adalah visualisasi *confusion matrix* dari ketiga rasio data:



Gambar 4.2 Visualisasi *Confusion Matrix*

Dari *confusion matrix* tersebut dapat diperoleh perbandingan nilai akurasi dari ketiga rasio data sebagaimana ditunjukkan pada gambar 4.3 berikut ini:



Gambar 4.3 Perbandingan Akurasi Model pada Tiga Rasio Data

Dalam gambar 4.3 tersebut terlihat bahwa model paling optimal diperoleh dari rasio 80:20 dengan hasil akurasi 86.44%. Selanjutnya model dengan rasio

tersebut disimpan dalam format berkas .pkl untuk selanjutnya dilakukan implementasi ke web.

4.8 Implementation

Setelah proses pelatihan model deteksi *clickbait* menggunakan algoritma SVM selesai dilakukan, tahapan berikutnya yaitu mengimplementasikan model tersebut ke dalam sebuah aplikasi berbasis web. Tujuan dari implementasi ini adalah untuk menunjukkan bagaimana model deteksi *clickbait* dapat diintegrasikan ke dalam aplikasi web.

4.8.1 Kebutuhan Sistem untuk Implementasi

Kebutuhan sistem yang digunakan untuk implementasi antara lain:

a. *API YouTube*: Digunakan untuk mengambil data video trending dari *YouTube* secara *real-time* pada saat tombol *"Detect Trending"* di tekan.

```
API_KEY = 'AIzaSyDMB88euf37_JmSwCNXTWpbM1-Kk4ySafw'
URL =
f'https://youtube.googleapis.com/youtube/v3/videos?part=snippet&chart=mostP
opular&regionCode=ID&maxResults=50&key={API_KEY}'

def fetch_trending_videos(): response = requests.get(URL) data =
response.json() return data.get('items', [])
```

b. *Pipeline Preprocessing*: Berisi tahapan preprocessing dan transformasi fitur menggunakan TF-IDF.

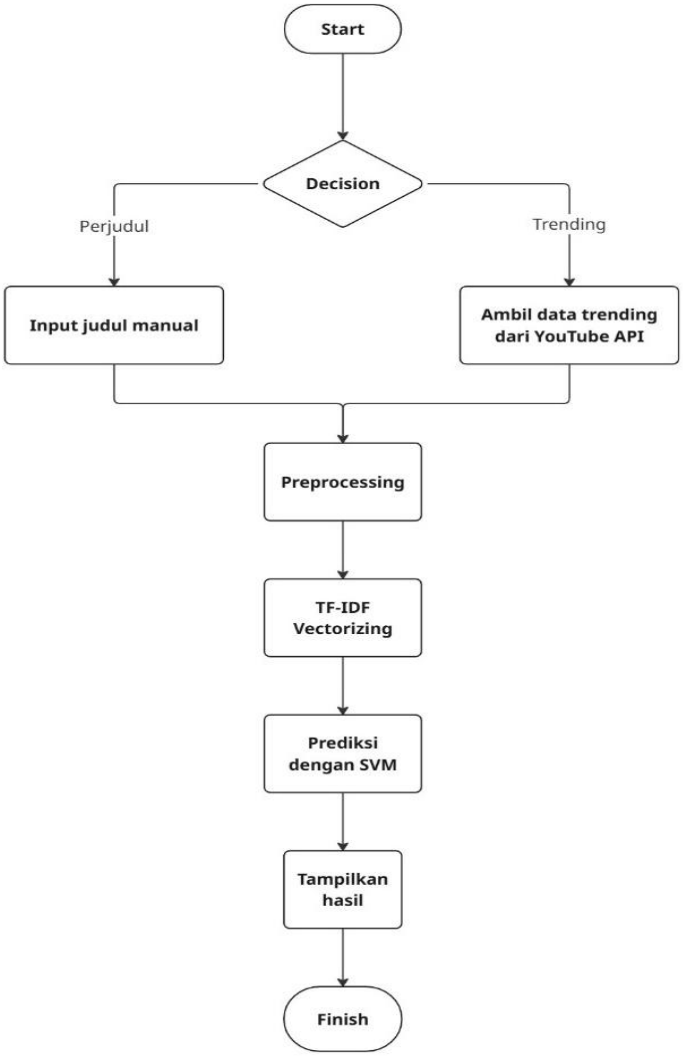
```
pipeline = joblib.load(os.path.join('model', 'preprocessing_pipeline.pkl'))
tfidf_vectorizer = joblib.load(os.path.join('model',
'tfidf_vectorizer.pkl'))
```

c. *Model*: Model klasifikasi *clickbait* yang telah dilatih menggunakan SVM.

```
svm_model = joblib.load(os.path.join('model', 'svm_model.pkl'))
```

4.8.2 Alur Kerja Program

Sistem terdiri dari dua jalur utama, yaitu input manual oleh pengguna dan input otomatis dari API YouTube. Keduanya melalui tahapan preprocessing, transformasi TF-IDF, serta klasifikasi menggunakan model SVM sebelum hasil ditampilkan ke web. Gambar 4.4 berikut menunjukkan flowchart alur kerja sistem deteksi clickbait pada judul video YouTube:



Gambar 4. 4 Alur Kerja Program

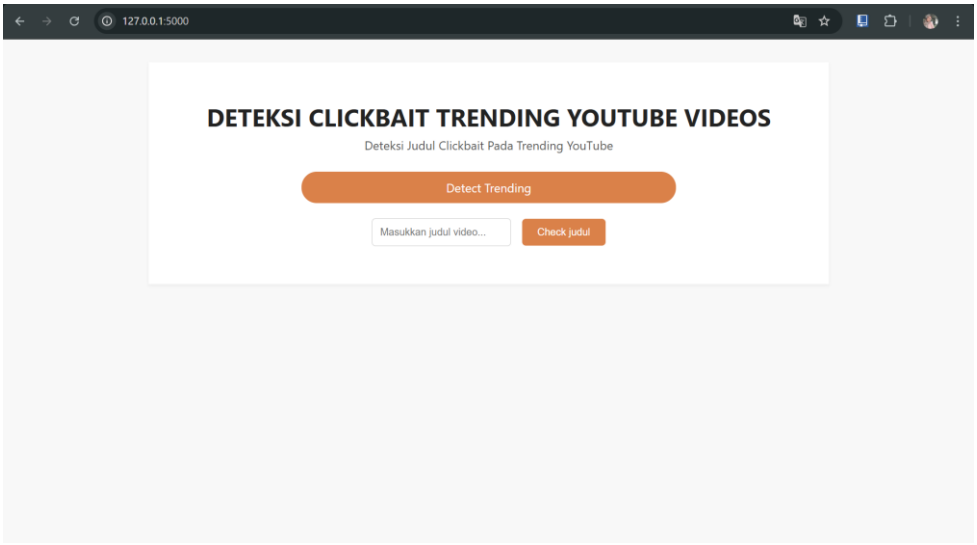
Secara garis besar, sistem bekerja dengan mengambil data video trending dari *YouTube* atau menunggu input manual judul, kemudian memproses judul video tersebut menggunakan *pipeline preprocessing*, mengubahnya menjadi vektor TF-IDF, lalu memprediksi apakah judul tersebut termasuk *clickbait* atau tidak dengan model SVM. Berikut adalah cuplikan kode penting yang menunjukkan bagaimana prediksi dilakukan:

```
for video in videos:
    title = video['snippet']['title']
    preprocessed_title = preprocess_text(title, alay_dict)
    title_vector = tfidf_vectorizer.transform([preprocessed_title]).toarray()
    label = svm_model.predict(title_vector)
```

4.8.3 Hasil Implementasi

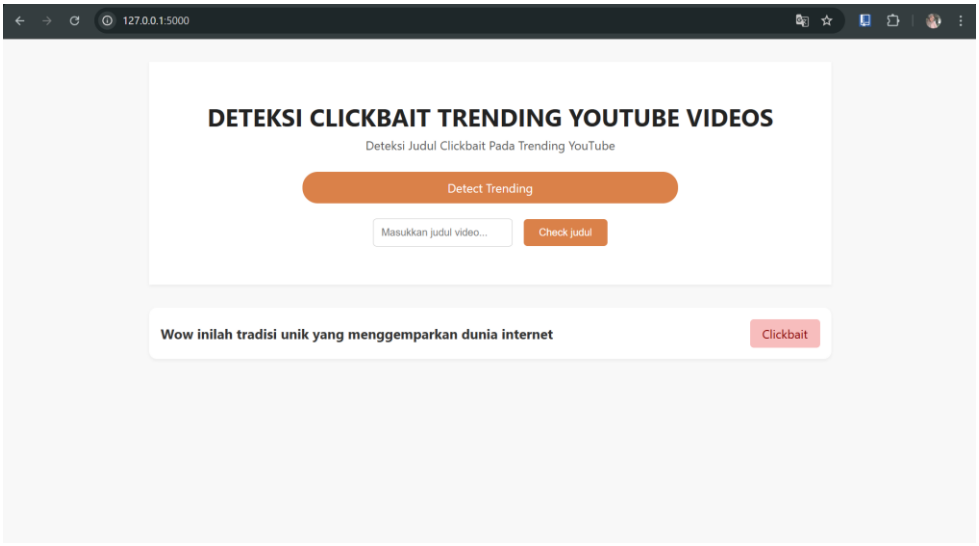
Aplikasi dijalankan secara lokal di perangkat penulis menggunakan server bawaan *Flask*. Karena tidak dilakukan proses *hosting* atau *deployment*, aplikasi hanya dapat diakses melalui *browser* lokal dengan alamat *default* http://127.0.0.1:5000/. Aplikasi web ini memiliki antarmuka yang terdiri dari dua halaman utama:

a. *Home* (/): Halaman awal saat pengguna mengakses aplikasi. Gambar 4.5 berikut merupakan tampilan dari halaman *home* pada web:



Gambar 4.5 Halaman *Home* pada Web

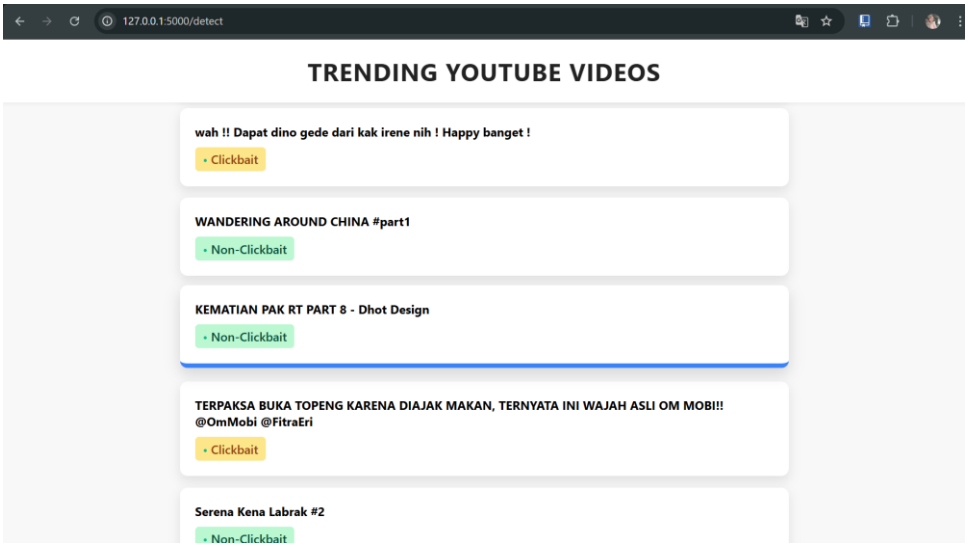
Gambar 4.5 merupakan tampilan dari halaman *home*, terdapat tombol *"Detect Trending"* yang ketika di tekan maka akan mengambil data *trending YouTube* secara *realtime* dan menampilkan hasil deteksi dari setiap judul. Kemudian, terdapat form untuk menginputkan judul video yang ketika di *submit* dengan menekan "Check judul" akan ditampilkan hasil deteksi dari judul. Tampilan ketika dilakukan *check* judul yang diinputkan dapat dilihat pada gambar 4.6 berikut ini:



Gambar 4.6 Halaman Home Setelah Cek Judul

Pada gambar 4.6 setelah judul diinputkan dan menekan tombol "Check judul" maka judul yang telah diinputkan sebelumnya akan ditampilkan dan dideteksi labelnya. Dalam contoh tersebut judul "Wow inilah tradisi unik yang menggemparkan dunia internet" terdeteksi sebagai *clickbait*.

b. *Detect Clickbait (/detect):* Halaman utama yang menampilkan hasil deteksi *clickbait* dari judul video trending.



Gambar 4.7 Tampilan Halaman Detect

Dari gambar 4.7 tersebut terlihat bahwa implementasi model ke dalam aplikasi web telah berhasil dilakukan. Dengan akurasi sebesar 86.44% model mampu membedakan judul-judul yang termasuk kategori *clickbait* dan *non-*

clickbait dengan cukup baik. Meskipun demikian, terdapat beberapa judul (sebagian kecil) yang seharusnya termasuk dalam kategori *clickbait*, namun terdeteksi sebagai *non-clickbait*. Kesalahan ini kemungkinan disebabkan oleh keterbatasan cakupan dataset pelatihan, sehingga model belum mendapatkan cukup variasi untuk mengenali semua pola judul *clickbait*. Namun secara keseluruhan, sebagian besar judul berhasil diklasifikasikan dengan tepat, yang menunjukkan bahwa model dengan akurasi 0.864 tersebut sudah cukup baik untuk diterapkan dalam sistem deteksi *clickbait* berbasis web.

4.8.4 Pengujian Sistem

Pengujian sistem dilakukan dengan menggunakan metode black box untuk memastikan bahwa setiap fungsi pada sistem berjalan sesuai dengan yang diharapkan. Metode black box berfokus pada pengujian fungsionalitas sistem berdasarkan input dan output. Tabel 4.12 berikut menampilkan hasil pengujian fungsional sistem menggunakan metode black box testing:

Tabel 4. 12 Hasil Pengujian Sistem

No	Fitur yang Diuji	Input	Output yang Diharapkan	Hasil Pengujian	Keterangan
1	Tombol “Detect Trending”	Klik tombol	Menampilkan daftar video trending YouTube dan label deteksinya	Berhasil	Sesuai
2	Input manual judul video	“Wow inilah tradisi unik yang menggemparkan dunia internet”	Menampilkan label prediksi “Clickbait”	Berhasil	Sesuai
3	Input manual judul video	“Cara Menanak Nasi”	Menampilkan label prediksi “Non-Clickbait”	Berhasil	Sesuai
4	Labeling hasil prediksi video trending	Judul trending otomatis dari YouTube	Menampilkan label "Clickbait" atau "Non-Clickbait"	Berhasil	Akurat dan sesuai model
5	Tombol “Check Judul”	Klik setelah mengisi input	Menampilkan judul dan label deteksinya	Berhasil	Tidak terjadi error
6	Tidak mengisi input, lalu klik "Check Judul"	Kosong	Muncul pesan “Anda belum memasukkan judul!”	Berhasil	Menolak input kosong

Berdasarkan tabel 4.12 tersebut menunjukkan bahwa sistem deteksi clickbait yang dibangun telah berjalan dengan baik dan sesuai dengan fungsionalitas yang diharapkan. Sistem mampu menerima input, memproses klasifikasi menggunakan model SVM yang telah dibangun, dan menampilkan hasil secara responsif melalui antarmuka web.

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan dari hasil penelitian ini disajikan sebagai berikut:

1. Penelitian ini berhasil menerapkan algoritma *Support Vector Machine* (SVM) untuk membangun model klasifikasi *clickbait* pada judul video *trending YouTube* berbahasa Indonesia. Dataset yang digunakan terdiri dari 2.062 judul video yang proses pelabelannya dilakukan secara manual oleh lima orang melalui sistem voting yang diklasifikasikan dalam dua kelas, yaitu *clickbait* (1) dan *non-clickbait* (0). Sebelum proses pemodelan, data melalui tahap *preprocessing* yang meliputi *case folding, cleaning, normalizing, tokenizing, filtering*, dan *stemming*. Kemudian pembobotan kata dilakukan dengan menggunakan pendekatan *Term Frequency–Inverse Document Frequency* (TF-IDF). Dataset kemudian dibagi dengan tiga rasio perbandingan yaitu 70:30, 80:20, dan 90:10, dengan hasil terbaik yang diperoleh yaitu pada rasio 80:20. Berdasarkan hasil evaluasi, model SVM yang dibangun mampu mengklasifikasikan judul dengan akurasi mencapai 86,44%. Hasil ini menunjukkan bahwa model yang telah dibangun memiliki performa yang baik dalam membedakan antara judul *clickbait* dan *non-clickbait*.
2. Selain membangun model klasifikasi, penelitian ini juga berhasil merancang dan mengimplementasikan sistem deteksi *clickbait* dalam bentuk aplikasi web yang dapat digunakan untuk mendeteksi judul video *trending* secara *realtime*. Meskipun demikian, masih terdapat beberapa kesalahan dalam hasil deteksi, yang kemungkinan disebabkan oleh keterbatasan variasi dalam dataset pelatihan. Hal tersebut membuat model belum sepenuhnya mampu mengenali berbagai pola judul *clickbait* secara menyeluruh. Berdasarkan hasil pengujian menggunakan metode *black box*, sistem yang dibangun telah berjalan dengan baik dan sesuai dengan fungsionalitas yang diharapkan.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, berikut beberapa saran yang dapat diberikan untuk pengembangan penelitian selanjutnya:

1. Dataset dalam penelitian ini tergolong terbatas. Maka, penambahan jumlah serta keragaman data latih seperti menambah dataset dengan Bahasa Inggris atau lainnya sangat disarankan agar model dapat mengenali pola *clickbait* dan *non-clickbait* secara lebih akurat. Semakin banyak dan beragam data yang digunakan, maka semakin baik pula kemampuan model dalam mendeteksi berbagai bentuk *clickbait*.
2. Untuk meningkatkan performa model SVM, pencarian parameter terbaik (hyperparameter tuning) seperti pengaturan nilai C, gamma, dan pemilihan jenis kernel dapat dilakukan. Meskipun model saat ini telah menunjukkan performa yang cukup baik, penerapan strategi tuning ini dapat menjadi fokus pengembangan lebih lanjut pada penelitian selanjutnya.

DAFTAR PUSTAKA

Agustian, S., & Nazir, A. (2024). Klasifikasi Sentimen Terhadap Pengangkatan Kaesang Sebagai Ketua Umum Partai PSI Menggunakan Metode Support Vector Machine. *Technology and Science (BITS)*, *6*(1). https://doi.org/10.47065/bits.v6i1.5340

Dewi Trustyanda, N., Ratu Afni Rizki, C., & Okto Sri, Q. (2021). *BUDAYA CLICKBAIT PADA JUDUL BERITA DI ERA DIGITAL 4.0*. *6*(9). https://doi.org/10.36418/syntax

Goel, U. (2023). *Clickbait Identification from YouTube Video Titles*. www.ijnrd.org

Hadiyat, Y. D. (2019). Clickbait on Indonesia Online Media. *Journal Pekommas*, *4*(1), 1. https://doi.org/10.30818/jpkm.2019.2040101

Ibrohim, M. O., & Budi, I. (2019). Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter. *Proceedings of the Third Workshop on Abusive Language Online*, 46–57. https://doi.org/10.18653/v1/W19-3506

Jurafsky, D., & Martin, J. H. (n.d.). *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models Third Edition draft Summary of Contents*.

Kemm, R. (2022). The Linguistic and Typological Features of Clickbait in Youtube Video Titles. *Social Communication*, *8*(1), 66–80. https://doi.org/10.2478/sc-2022-0007

Mufti Prasetiyo, S., Gustiawan, R., & Rizzel Albani, F. (2024). *BIIKMA : Buletin Ilmiah Ilmu Komputer dan Multimedia Analisis Pertumbuhan Pengguna Internet Di Indonesia*. *2*(1). https://jurnalmahasiswa.com/index.php/biikma

Pujahari, A., & Singh Sisodia, D. (2019). *Clickbait Detection using Multiple Categorization Techniques*. www.clickbait-challenge.org

Rama Bena Putra, P., & Setya Perdana, R. (2023). *Klasifikasi Judul Berita Online menggunakan Metode Support Vector Machine (SVM) dengan Seleksi Fitur Chi-square* (Vol. 7, Issue 5). http://j-ptiik.ub.ac.id

Ramadhan, F. A., Sitorus, S. H., & Rismawan, T. (2023). Penerapan Metode Multinomial Naïve Bayes untuk Klasifikasi Judul Berita Clickbait dengan Term Frequency - Inverse Document Frequency. *Jurnal Sistem Dan Teknologi Informasi (JustIN)*, *11*(1), 70. https://doi.org/10.26418/justin.v11i1.57452

Risnasari, M. (2022). *Konsep Dasar Data Mining (Teori dan Praktik dengan Python)*. CV. Literasi Nusantara Abadi.

Sagita, R., Enri, U., & Primajaya, A. (2020). Klasifikasi Berita Clickbait Menggunakan K-Nearest Neighbor (KNN). *JOINS (Journal of Information System)*, *5*(2), 230–239. https://doi.org/10.33633/joins.v5i2.3705

Saputra, I., & Kusnadi, A. (2021). *Belajar Text Mining dalam 24 Jam*. CV. Literasi Nusantara Abadi.

Srivastava, A., & Sahami, M. (2009). *Text Mining Classification, Clustering, and Applications*.

Swastika, R., Susanto, F., Muslihudin, M., & Ipnuwati, S. (2023). *Implementasi Data Mining (Clastering, Association, Prediction, Estimation, Classification)*. CV. Adanu Abimata.

Trishnanti, D., & Al Azies, H. (n.d.). *COMPARISON OF SUPPORT VECTOR MACHINE METHOD (SVM) AND K-NEAREST NEIGHBOR (K-NN) IN CLASSIFICATION OF HUMAN DEVELOPMENT INDEX (HDI)*. https://doi.org/10.17605/OSF.IO/NCX74

Winarto, T. S. Y., Wijaya, K., Faqih, M. A., Prasetyo, S. Y., & Muliono, Y. (2023). Tackling Clickbait with Machine Learning: A Comparative Study of Binary Classification Models for YouTube Title. *Procedia Computer Science*, *227*, 282–290. https://doi.org/10.1016/j.procs.2023.10.526

Zhang, C., & Clough, P. D. (2020). Investigating clickbait in Chinese social media: A study of WeChat. *Online Social Networks and Media*, *19*. https://doi.org/10.1016/j.osnem.2020.100095

LAMPIRAN

4.1 Dataset Judul Video Clickbait YouTube

4.2 Kode Program Clickbait Detection