Indonesian Journal of Digital Business

Journal homepage: https://ejournal.upi.edu/index.php/IJDB

Implementasi Algoritma Naive Bayes untuk Filtrasi Spam Komentar Judi Online pada YouTube

Faiz Jauhari Makarim Riza [1], Rangga Gelar Guntara [1], Muhammad Rizki Nugraha [1]

[1]Program Studi Bisnis Digital, Universitas Pendidikan Indonesia, Jawa Barat, Indonesia
Correspondence: E-mail: ranggagelar@upi.edu

ABSTRACT

Perkembangan interaksi pengguna di platform YouTube turut menghadirkan tantangan baru, salah satunya adalah maraknya komentar spam yang mengandung unsur promosi perjudian online. Jenis komentar ini berdampak negatif terhadap komunitas yang terdapat di kanal. Penelitian ini bertujuan merancang sebuah sistem klasifikasi komentar spam dengan memanfaatkan algoritma *Naive Bayes*. Pengembangan sistem dilakukan berdasarkan tahapan CRISP-DM, dimulai dari proses pengambilan data komentar menggunakan YouTube API, dilanjutkan tahapan *text preprocessing* seperti *unicode normalization*, *case folding*, *tokenizing*, *stopword removal*, *filtering*, hingga pelabelan data. Selanjutnya, setiap kata diberi bobot menggunakan metode TF-IDF. Evaluasi model dilakukan menggunakan teknik *K-Fold Cross Validation* dan analisis *Confusion Matrix*. Hasil evaluasi menunjukkan bahwa model memiliki performa yang baik dengan akurasi sebesar 97,1%, *precision* 96,4%, *recall* 95,6%, dan *f1-score* 96%. Model ini kemudian diimplementasikan dalam bentuk aplikasi berbasis *Command Line Interface* (CLI) yang dapat digunakan oleh pemilik kanal YouTube untuk mendeteksi serta menghapus komentar spam secara otomatis. Berdasarkan hasil pengujian, sistem bekerja secara efektif sehingga membuktikan bahwa proses preprocessing dan algoritma *Naive Bayes* dapat menghasilkan sistem deteksi spam akurat.

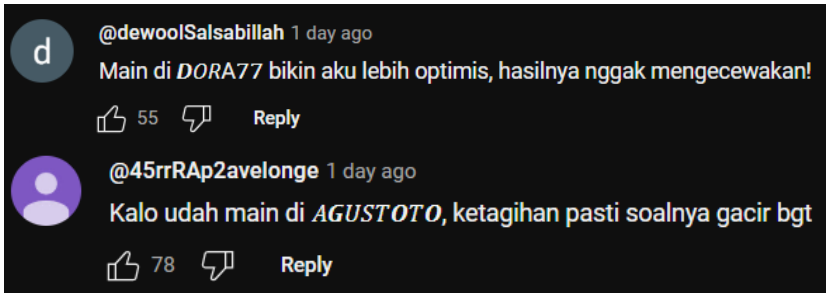
ARTICLE INFO

Article History:
Submitted/Received 12 Juni 2025
First Revised 15 Juli 2025
Accepted 20 Juli 2025
First Available online 31 Juli 2025
Publication Date 31 Juli 2025

Keyword:
YouTube; Spam, Naive Bayes, Term Frequency-Inverse Document Frequency, Classifier, Command Line Interface.

1. PENDAHULUAN

Perkembangan teknologi digital telah mendorong pertumbuhan aktivitas perjudian berbasis daring, di mana individu dapat dengan mudah mengakses dan mengikuti praktik perjudian melalui situs-situs yang terhubung dengan internet. Aspek psikologis memainkan peran signifikan dalam mendorong partisipasi individu dalam perjudian online. Pemain pemula kerap kali mengalami kemenangan pada tahap awal, yang kemudian membentuk persepsi keliru bahwa peluang untuk menang cukup tinggi (Makarin & Astuti, 2023). Persepsi ini diperkuat oleh strategi pemasaran yang agresif di berbagai platform digital, yang secara aktif mendorong keterlibatan masyarakat. Kampanye promosi yang masif, termasuk yang dilakukan oleh influencer dengan narasi keuntungan instan, turut memperkuat daya tarik perjudian online di tengah masyarakat. Tingginya paparan terhadap iklan semacam ini tidak hanya berkontribusi pada normalisasi praktik perjudian dalam kehidupan sehari-hari, tetapi juga mendorong peningkatan interaksi dan transaksi pada platform-platform judi digital (Paramartha et al., 2021). Salah satu strategi pemasaran yang semakin marak digunakan adalah praktik spamming pada kolom komentar video YouTube yang sedang populer (Hayoung, 2021). Praktik ini memanfaatkan tingginya jumlah penonton pada video yang masuk dalam kategori trending untuk menyebarkan nama situs dan informasi yang mengarah ke situs perjudian daring, dengan biaya pemasaran yang minimal. Gambar 1 memperlihatkan contoh promosi judi online yang disisipkan melalui kolom komentar pada sebuah video YouTube. Komentar tersebut mempromosikan pengguna ke situs terkait, sehingga menjadikan platform YouTube sebagai medium yang dimanfaatkan untuk menarik perhatian calon pengguna layanan ilegal tersebut. Fenomena ini menyebabkan kolom komentar yang semestinya berfungsi sebagai ruang interaksi antar pengguna justru berubah menjadi sarana promosi yang tidak relevan dan mengganggu.



Gambar 1. Promosi Judi Online pada Kolom Komentar di Video YouTube

Meskipun YouTube telah menerapkan sistem filter untuk menyaring komentar spam, mekanisme tersebut belum sepenuhnya efektif. (Fernando et al., 2019) menyatakan bahwa sistem deteksi spam pada YouTube masih memiliki kelemahan, khususnya karena tidak selalu mampu mengenali komentar bersifat rasis maupun spam yang ditulis dalam bahasa Indonesia. Kendala lain yang didapatkan untuk memfiltrasi komentar spam judi online adalah fitur pencarian biasa tidak dapat untuk mendeteksi spam tersebut karena para pelaku spam yang kerap menggunakan kombinasi karakter *Unicode*, *alfabet Cyrillic*, serta simbol atau karakter khusus. Sejalan dengan temuan tersebut, (Mahmoud et al., 2014) menyatakan bahwa pola karakter tertentu seperti karakter yang dikodekan atau penggunaan *alfabet non-standard* dapat membingungkan sistem penyaringan spam secara otomatis.

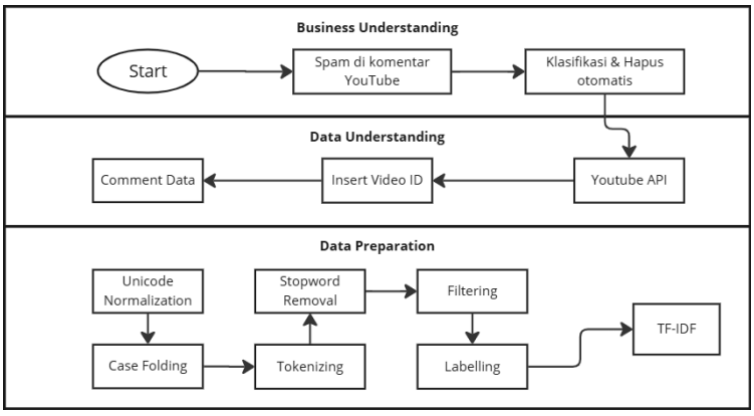
Berbagai penelitian telah dilakukan dalam upaya mengklasifikasikan komentar spam, terutama menggunakan pendekatan machine learning. Salah satu algoritma yang cukup populer dan sering digunakan adalah *Naive Bayes*, yang terbukti memberikan hasil yang

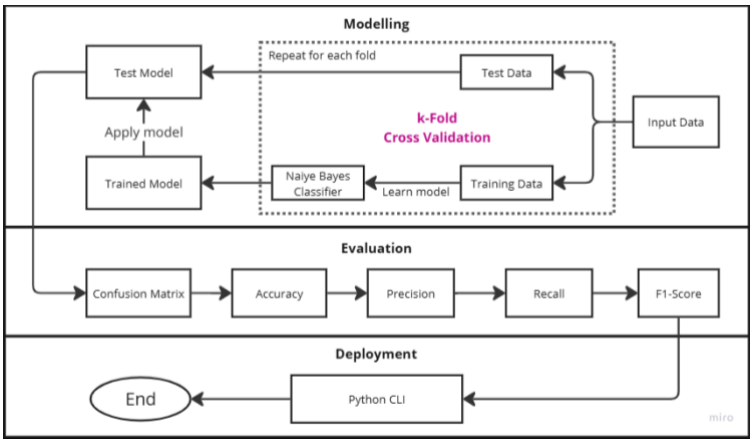
cukup andal. Studi oleh (Samsudin et al., 2019) menunjukkan bahwa *Naive Bayes* mampu mencapai akurasi di atas 80% dalam mendeteksi komentar spam. Penelitian lain oleh (Fernando et al., 2019) bahkan mencatatkan performa yang lebih tinggi, dengan akurasi sebesar 87%, *precision* 80%, *recall* 97%, dan *F-1 Score* 88%. Temuan-temuan ini memperkuat posisi *Naive Bayes* sebagai *baseline* yang kuat dalam klasifikasi spam, terutama untuk data berbasis teks. Selain *Naive Bayes*, studi lainnya juga mengeksplorasi penggunaan algoritma yang lebih kompleks. (Abdullah et al., 2018) melaporkan bahwa algoritma *Adaptive Genetic Algorithm* (AGA) berhasil mencapai akurasi hingga 99,11%, tertinggi di antara sembilan algoritma yang dibandingkan. Dalam studi terbaru oleh (Ghatasheh et al., 2022), digunakan pendekatan *XGBoost* yang dioptimasi menggunakan AGA untuk seleksi fitur. Walaupun fokus utama studi tersebut adalah pada *XGBoost*, mereka juga menyertakan hasil *baseline* dari *Naive Bayes* tanpa optimasi, dengan representasi fitur TF-IDF, yang mencatat akurasi sebesar 92% pada dataset Twitter. Dataset ini memiliki karakteristik serupa dengan komentar spam di platform lain, yaitu teks singkat dan penggunaan bahasa informal.

Efektivitas algoritma *Naive Bayes* dalam klasifikasi komentar spam telah dibuktikan melalui berbagai studi, masih terdapat gap penelitian yang relevan untuk ditinjau lebih lanjut. Mayoritas penelitian hanya berfokus pada evaluasi model secara eksperimental tanpa dilanjutkan ke tahap pengembangan dan implementasi sistem yang dapat digunakan dalam konteks aplikasi nyata. Selain itu, belum seluruh studi terdahulu menerapkan metode *k-fold cross validation*, yang penting untuk meningkatkan reliabilitas model dan optimalisasi penggunaan data. Fenomena meningkatnya spam komentar terkait promosi judi online menjadi dasar dilakukannya penelitian ini, dengan tujuan mengembangkan sistem deteksi spam berbasis *Naive Bayes*. Sistem ini dirancang menggunakan YouTube API untuk mengumpulkan data komentar, menghapus komentar spam, serta melakukan klasifikasi melalui pendekatan *machine learning*. Implementasi dilakukan dalam bentuk antarmuka *Command Line Interface* (CLI) berbasis *Python*, di mana pengguna dapat memasukkan ID video YouTube, melakukan autentikasi menggunakan OAuth 2.0, dan menjalankan proses klasifikasi secara langsung.

2. METODE

Penelitian ini menggunakan pendekatan *Research and Development* (R&D), yaitu jenis penelitian yang bertujuan untuk menghasilkan produk baru sekaligus menguji efektivitasnya (Sugiyono, 2013). Dalam konteks ini, pendekatan R&D digunakan untuk mengembangkan sistem klasifikasi komentar spam dengan algoritma *Naive Bayes*. Proses pengembangan mengikuti model CRISP-DM, yang terdiri dari tahapan *Business Understanding*, *Data Understanding*, *Data Preparation*, *Modelling*, *Evaluation*, dan *Deployment*, sebagaimana diilustrasikan pada Gambar 2. Penerapan metode R&D memungkinkan dilakukannya pengujian dan penyempurnaan model secara iteratif hingga mencapai hasil yang optimal.





Gambar 2. Tahapan Penelitian

2.1 *Business Understanding*

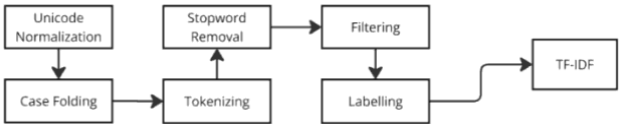
Business understanding merupakan tahap awal yang berfokus pada penetapan tujuan bisnis, analisis terhadap kondisi serta konteks permasalahan saat penelitian dilakukan, dan perumusan arah penelitian. Tahapan ini menjadi dasar dalam mengaitkan permasalahan yang dihadapi dengan solusi yang akan dikembangkan melalui pendekatan *data mining* (Schröer et al., 2021). Tahapan ini berfokus pada pemahaman terhadap fenomena maraknya komentar spam di platform YouTube, khususnya yang berisi promosi judi online. Permasalahan utama terletak pada ketidakefektifan sistem moderasi YouTube dalam menyaring komentar. Penelitian ini menerapkan tahapan CRISP-DM serta algoritma *Naive Bayes* sebagai metode klasifikasi untuk mengenali pola-pola yang umum digunakan dalam komentar spam.

2.2 *Data Understanding*

Pada fase *Data Understanding*, dilakukan serangkaian langkah untuk memahami karakteristik data secara menyeluruh. Tahapan ini mencakup pengumpulan data, mengambil atribut penting, serta verifikasi kualitas data guna memastikan konsistensi dan kelayakan untuk dianalisis (Schröer et al., 2021). Data dalam penelitian ini diperoleh dari 26 video YouTube yang sedang trending, dengan rentang unggahan tahun 2024 hingga 2025. Proses pengambilan data dilakukan menggunakan YouTube API melalui Python, sehingga terkumpul 56.546 komentar dalam format JSON. Sebagian dari komentar tersebut terindikasi mengandung konten spam yang berkaitan dengan promosi judi online.

2.3 *Data Preparation*

Data preparation merupakan tahap yang mencakup serangkaian proses untuk memastikan data memiliki kualitas tinggi, terstruktur, dan siap digunakan dalam analisis lanjutan karena tahap ini mempengaruhi akurasi model serta efisiensi pemrosesan data (Schröer et al., 2021). Pada penelitian ini, tahap data preparation meliputi proses pembersihan dan transformasi data sebelum pemodelan dilakukan. Langkah awal dimulai dengan *unicode normalization* untuk menyeragamkan format karakter, diikuti oleh *case folding*, *tokenizing*, *stopword removal*, serta *filtering*. Setelah proses *cleaning* selesai, data diberi label secara manual menjadi dua kategori, yaitu spam dan *non-spam*. Tahap akhir melibatkan pembobotan kata menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF) guna mengubah teks menjadi representasi numerik yang dapat diolah oleh algoritma klasifikasi. Gambar 3 menunjukan keseluruhan tahapan dari data preparation.



Gambar 3. Tahapan Text Preprocessing

2.4 *Modelling*

2.4.1 *K-Fold Cross Validation*

Validasi terhadap performa model menjadi aspek krusial agar hasil yang diperoleh baik pada data pelatihan. Menurut Yadav & Shukla (2016) metode ini bekerja dengan membagi dataset ke dalam *k* bagian (*fold*), umumnya sebanyak 5 hingga 10 bagian. Pada setiap iterasi, satu *fold* digunakan sebagai data uji, sementara *k–1 fold* lainnya digunakan untuk *training*. Proses ini dilakukan berulang hingga setiap *fold* pernah menjadi *data testing*. Teknik ini memberikan evaluasi model yang lebih menyeluruh dan tidak bergantung pada satu subset data serta membantu mengurangi risiko *overfitting* dan meningkatkan kemampuan generalisasi model. Adapun proporsi data yang digunakan dalam penelitian ini terdiri atas 63% data *non-spam* dan 37% data spam.

2.4.2 *Naive Bayes*

Secara umum, algoritma *Naive Bayes* bekerja dengan menghitung kemungkinan suatu kelas target berdasarkan fitur-fitur yang terdapat dalam data input. Dalam prosesnya, algoritma ini menilai seberapa besar peluang setiap fitur muncul pada masing-masing kelas, lalu menggabungkan seluruh probabilitas tersebut untuk menentukan kelas yang paling mungkin sesuai dengan data yang diberikan (Rangga Gelar Guntara, 2023). Berdasarkan asumsi tersebut, dihasilkan model klasifikasi yang dikenal sebagai *Naive Bayes*. Algoritma ini terdiri atas dua tahapan utama, yaitu tahap pelatihan (*training*) dan tahap klasifikasi (*classification*). Pada tahap pelatihan, dilakukan analisis terhadap dokumen sampel untuk mengidentifikasi kata-kata yang memiliki kemungkinan tinggi muncul dalam koleksi data. Probabilitas untuk setiap kategori dihitung berdasarkan distribusi atribut yang terdapat dalam data (Murphy, 2012). Model yang dihasilkan kemudian digunakan untuk memprediksi kelas dari dokumen baru berdasarkan probabilitas tertinggi.

2.5 *Evaluation*

Evaluasi dilakukan untuk menilai kinerja model dalam mengklasifikasi komentar spam serta memastikan bahwa seluruh tahapan pemodelan telah berjalan sesuai prosedur. Pada penelitian ini, model *Naive Bayes* dievaluasi menggunakan *confusion matrix* dengan beberapa metrik utama *accuracy*, *precision*, *recall*, dan *F1-score*. *Confusion matrix* digunakan untuk memetakan hasil prediksi model terhadap label sebenarnya, sehingga memudahkan analisis terhadap kesalahan klasifikasi. Metrik tersebut dapat diukur berdasarkan value hasil prediksi yang disajikan dalam *confusion matrix*.

Menurut Zeng (2020), evaluasi model klasifikasi melibatkan empat kemungkinan hasil prediksi yang dihasilkan *confusion matrix* yaitu *True Negative* (TN), *False Negative* (FN), *True Positive* (TP), *False Positive* (FP).

Accuracy mengukur persentase prediksi yang benar dari seluruh data uji, sementara *precision* dan *recall* mengevaluasi keseimbangan antara prediksi positif yang tepat dan total sampel positif yang teridentifikasi. *F1-score* digunakan untuk menilai performa model dalam kondisi data dengan distribusi kelas yang tidak seimbang (Han et al., 2022). Rumus yang digunakan untuk menghitung metrik evaluasi berdasarkan confusion matrix adalah sebagai berikut:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

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

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \quad (4)$$

2.6 *Deployment*

Pada tahap *deployment*, seluruh pemahaman yang diperoleh dari setiap tahapan dalam kerangka CRISP-DM digunakan sebagai dasar dalam pengembangan model, guna

mengoptimalkan kinerja algoritma yang diterapkan (Schröer et al., 2021). Penelitian ini menghasilkan dua luaran utama, yaitu evaluasi kinerja algoritma *Naive Bayes* dalam mengklasifikasi komentar spam serta sistem deteksi spam yang dapat dijalankan secara praktis. Sistem dibangun menggunakan bahasa pemrograman *Python* dengan penerapan algoritma *Naive Bayes*, dan dikemas dalam format *executable* (.exe) menggunakan *PyInstaller* agar dapat dijalankan secara mandiri. Antarmuka sistem menggunakan *Command Line Interface* (CLI), yang dapat digunakan pengguna untuk memasukkan ID video YouTube secara spesifik guna mendeteksi komentar spam secara otomatis.

3. HASIL DAN PEMBAHASAN

3.1 Persiapan Data

Tahap awal dalam membangun sistem deteksi spam otomatis berbasis algoritma *Naive Bayes* dimulai dengan pengambilan data yang nantinya akan digunakan sebagai *dataset*. Data diperoleh melalui YouTube API dengan memanfaatkan *library Google API Client* yang tersedia dalam Bahasa pemrograman *Python* untuk mengambil data komentar dari video YouTube. Langkah awal yang dilakukan dalam proses ini adalah mengimpor *library* yang dibutuhkan sebelum masuk ke tahap *scraping* dan analisis data. Gambar 4 menunjukkan cuplikan kode program untuk proses pengambilan data komentar dan menampilkan data melalui YouTube API. Data komentar yang berhasil dikumpulkan berjumlah 56.546 komentar dari 27 ID video berbeda yang sedang trending dan disimpan dalam format JSON.

```
video_id = input("input id video: ").strip()

comments = get_all_comments(youtube, video_id)

output_file = f"dat_{video_id}.json"
with open(output_file, "w", encoding="utf-8") as f:
    json.dump(comments, f, indent=2, ensure_ascii=False)

print(f"{len(comments)} successfully saved  {output_file}")
```

```
{
  "author": "@EllLubab",
  "text": "Pembuatan vidionya keren sih, semoga kedepanya semakin bagus aja kalo makin sukses",
  "published_at": "2025-05-15T04:04:33Z",
  "like_count": 0
},
```

Gambar 4. Source Code Comment Scraper

Setelah data komentar berhasil diperoleh dalam format JSON, tahap selanjutnya adalah mengelompokkan komentar berdasarkan kemiripan isi atau konteks. Tujuan dari proses ini adalah untuk mempermudah proses pelabelan manual ke dalam dua kategori, yaitu spam dan *non-spam*. Proses pengelompokan dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan *library scikit-learn*, khususnya modul TfidfVectorizer untuk menghitung bobot kata berdasarkan metode *Term Frequency-Inverse Document Frequency* (TF-IDF), serta fungsi *cosine_similarity* untuk mengukur tingkat kemiripan antar komentar. Hasil dari proses ini berupa pengelompokan komentar yang semula acak dalam struktur *raw* JSON menjadi lebih terorganisasi berdasarkan kesamaan makna atau topik. Gambar 5 menampilkan cuplikan hasil pengelompokan komentar yang telah dilakukan.

```
Kelompok 8 (3 kalimat):
  'itu stir bukan karna kepanasan tapi kena cairan,
  'Stir mobil ngelupas karna kena cincin'
  'stir yaris memang gituu, mudah ngelupas'
```

Gambar 5. Pengelompokan Teks

Tahap selanjutnya adalah menggabungkan dataset dari 27 ID video YouTube menggunakan *Microsoft Excel*. Penggabungan ini tidak hanya bertujuan untuk menyatukan seluruh data, tetapi juga untuk memastikan konsistensi melalui proses *filtering*, seperti menghapus duplikasi dan mengeliminasi kata-kata yang tidak memiliki bobot makna signifikan. Hasil dari

pembersihan ini menyebabkan jumlah komentar berkurang dari 56.546 menjadi 4.568 entri data yang dianggap relevan untuk analisis lebih lanjut. Pada tahap ini, dilakukan pelabelan data secara manual, dengan pemberian label 1 untuk komentar yang teridentifikasi sebagai spam, dan label 0 untuk komentar *non-spam*.

3.2 *Text Preprocessing*

Text preprocessing dilakukan untuk mempersiapkan data sebelum proses klasifikasi. Tahapan ini bertujuan untuk membersihkan dan menyeragamkan teks agar model dapat bekerja secara optimal. Proses mencakup *unicode normalization*, *case folding*, *tokenizing*, *stopword removal*, *filtering*, dan *labelling* manual untuk membedakan komentar spam dan non-spam. Seluruh tahapan dijalankan secara *real time* menggunakan blok kode *Python* dalam satu alur otomatis.

3.2.1 *Unicode Normalization*

Unicode normalization dilakukan untuk menyeragamkan karakter khusus, seperti kombinasi Unicode dan Cyrillic, ke dalam format standar (Shahriar, 2024). Langkah ini penting agar algoritma dapat mengenali komentar spam secara konsisten serta membuat konsistensi data lebih baik. Normalisasi NFC dan NFKC banyak diterapkan dalam berbagai aplikasi *Natural Language Processing (NLP)* karena kemampuannya dalam meningkatkan konsistensi data serta dukungan yang baik terhadap standar Unicode, sehingga membantu menjaga reprodusibilitas dalam pemrosesan teks (Kudo & Richardson, 2018).

Proses normalisasi dilakukan menggunakan pustaka unicodedata pada Python, mengingat banyaknya variasi format karakter yang sering digunakan oleh spammer. Tabel 1 menunjukan proses unicode normalization yang menormalisasi karakter menjadi standar.

Tabel 1. Hasil *Unicode Normalization*

Sebelum *Unicode Normalization*	Sesudah *Unicode Normalization*
UPVC ini bs gantiin asbes buat atap rmh nggak sih? Ato cm cocok buat kanopi ya?	UPVC ini bs gantiin asbes buat atap rmh nggak sih? Ato cm cocok buat kanopi ya?
Baru coba udah maxwin, di DORA 77 emg bikin nagih bgt	Baru coba udah maxwin, di DORA 77 emg bikin nagih bgt

3.2.2 *Case Folding*

Case folding dilakukan dengan mengubah seluruh huruf dalam komentar menjadi huruf kecil (Hani, 2023). Tujuan utamanya adalah menjaga konsistensi data dan mengurangi variasi akibat perbedaan penulisan huruf kapital. Proses ini membantu algoritma dalam mengenali pola teks secara lebih akurat. Tabel 2 menunjukkan hasil dari case folding.

Tabel 2. Hasil *Case Folding*

Sebelum *Case Folding*	Sesudah *Case Folding*
UPVC ini bs gantiin asbes buat atap rmh nggak sih? Ato cm cocok buat kanopi ya?	upvc ini bs gantiin asbes buat atap rmh nggak sih ato cm cocok buat kanopi ya
Baru coba udah maxwin, di DORA 77 emg bikin nagih bgt	baru coba udah maxwin, di dora 77 emg bikin nagih bgt

3.2.3 *Tokenizing*

Tokenizing dilakukan untuk memecah kalimat menjadi elemen-elemen individu atau token, agar memudahkan dalam proses analisis seperti perhitungan pembobotan kata (Hani, 2023). Token yang dihasilkan dapat berupa kata, atau karakter, tergantung pada kebutuhan analisis yang digunakan sebagaimana ditunjukan pada Tabel 3. Proses ini memudahkan analisis lanjutan seperti pembobotan kata oleh TF-IDF.

Tabel 3. Hasil *Tokenizing*

Sebelum ***Tokenizing***	Sesudah ***Tokenizing***
upvc ini bs gantiin asbes buat atap rmh nggak sih ato cm cocok buat kanopi ya	[‘upvc’, ‘ini’, ‘bs’, ‘gantiin’, ‘asbes’, ‘buat’, ‘atap’, ‘rmh’, ‘nggak’, ‘sih’, ‘ato’, ‘cm’, ‘cocok’, ‘buat’, ‘kanopi’, ‘ya’]
baru coba udah maxwin, di dora 77 emg bikin nagih bgt	[‘baru’, ‘coba’, ‘udah’, ‘maxwin’, ‘di’, ‘dora’, ‘77’, ‘emg’, ‘bikin’, ‘nagih’, ‘bgt’]

3.2.4 *Stopword Removal*

Stopwords removal dilakukan untuk menghapus kata-kata umum yang tidak memiliki makna penting dalam proses klasifikasi, seperti "udah", "atau", dan "yang". Langkah ini bertujuan meningkatkan akurasi analisis dengan hanya mempertahankan kata-kata yang relevan. Proses ini dilakukan menggunakan *library nltk* dan *Sastrawi* yang menyediakan daftar stopwords dalam bahasa Indonesia dan Inggris.

Tabel 4. Hasil *Stopwords Removal*

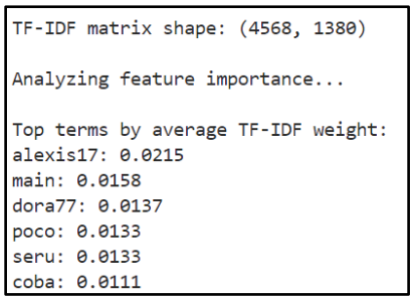
Sebelum ***Stopwords Removal***	Sesudah ***Stopwords Removal***
[‘upvc’, ‘ini’, ‘bs’, ‘gantiin’, ‘asbes’, ‘buat’, ‘atap’, ‘rmh’, ‘nggak’, ‘sih’, ‘ato’, ‘cm’, ‘cocok’, ‘buat’, ‘kanopi’, ‘ya’]	['upvc', 'gantiin', 'asbes', 'atap', 'rmh', 'cocok', 'kanopi']
[‘baru’, ‘coba’, ‘udah’, ‘maxwin’, ‘di’, ‘dora’, ‘77’, ‘emg’, ‘bikin’, ‘nagih’, ‘bgt’]	['baru', 'coba', 'maxwin', 'dora', '77', 'bikin', 'nagih', 'bgt']

3.2.5 Labelling

Data yang sudah siap kemudian diberi label secara manual untuk membedakan antara komentar yang termasuk spam dan non-spam. Setelah pelabelan selesai, label tersebut diencoding ke dalam format numerik agar dapat diproses oleh algoritma, dengan angka 1 untuk komentar spam dan 0 untuk komentar non-spam.

3.3 Pembobotan Pembobotan *Term Frequency-Inverse Document Frequency*

Dalam bidang *machine learning* maupun *deep learning*, TF-IDF berfungsi untuk mengonversi data teks menjadi representasi numerik, sehingga memungkinkan algoritma bekerja secara lebih optimal dalam berbagai tugas seperti klasifikasi teks, pencarian informasi hingga *text mining* (Christian et al., 2016). Tahap TF-IDF dilakukan untuk memberikan bobot pada kata-kata yang sudah melewati proses tokenisasi. Tujuannya adalah untuk mengubah data teks menjadi bentuk numerik agar bisa dikenali oleh model. Proses ini menghasilkan matriks yang menunjukkan seberapa penting sebuah kata dalam suatu komentar dibandingkan dengan seluruh kumpulan komentar. Kata-kata yang paling relevan dalam konteks komentar dapat teridentifikasi. Hasil akhir dari proses ini berupa data vektorisasi dengan total 4.568 baris dan 1.380 kolom. Gambar 6 menunjukkan cuplikan hasil pembobotan kata menggunakan TF-IDF.



Gambar 6. Implementasi TF-IDF

3.4 *K-Fold Cross Validation*

Sebelum memasuki tahap klasifikasi dengan algoritma *Naive Bayes*, data terlebih dahulu dibagi menggunakan teknik *k-fold cross-validation*. Proses ini dilakukan dengan modul *KFold* dari *library sklearn.model_selection*, yang membagi data menjadi beberapa subset untuk memastikan setiap bagian mendapat bagian sebagai data uji dan latih. Untuk mengoptimalkan model, digunakan metode *Grid Search* dengan 56 kombinasi *parameter* dalam skema 5-*fold*, menghasilkan total 280 proses pelatihan. Meskipun *cross-validation* dikenal sebagai metode evaluasi yang memiliki tingkat bias yang rendah, dalam beberapa kasus dengan ukuran sampel yang kecil, hasil yang diperoleh dapat menunjukkan ketidakstabilan. Untuk mengatasi hal tersebut, proses *cross-validation* dapat dilakukan berulang kali guna meminimalkan varians pada hasil evaluasi model (Tantithamthavorn et al., 2017). Dari hasil evaluasi, model terbaik mencapai skor F1-macro sebesar 0.9689, menandakan performa klasifikasi yang seimbang dan akurat antar kelas. Gambar 7 memperlihatkan hasil evaluasi dari proses tersebut.

```
Optimized MultinomialNB with hyperparameter tuning:
Fitting 5 folds for each of 56 candidates, totalling 280 fits
Best parameters found: {'alpha': np.float64(0.1), 'class_prior': None, 'fit_prior': True}
Best F1-macro score: 0.9689
```

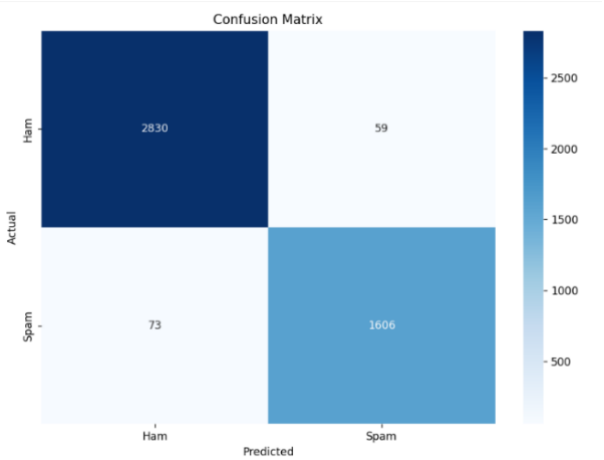
Gambar 7. Proses K-Fold

3.5 Implementasi

Setelah melalui tahapan text preprocessing dan pembobotan fitur dengan TF-IDF, proses pelatihan model dilakukan menggunakan teknik k-fold cross-validation untuk memastikan evaluasi mencakup seluruh bagian dataset. Algoritma yang digunakan adalah *Multinomial Naive Bayes*, algoritma ini merupakan penerapan dari metode *Naive Bayes* yang dirancang khusus untuk menangani data dengan distribusi multinomial. Mekanismenya melibatkan perhitungan probabilitas kemunculan setiap kata dalam masing-masing kelas, lalu memanfaatkan informasi tersebut untuk memprediksi kelas dari teks yang baru (Daniel & Martin, 2024). diimplementasikan melalui modul *MultinomialNB* dari *sklearn.naive_bayes*. Proses pelatihan dilakukan berdasarkan skema *K-Fold Cross Validation*, di mana model dilatih dan diuji secara bergantian pada setiap *fold* untuk memastikan evaluasi yang lebih menyeluruh.

3.6 Evaluasi

Tahapan selanjutnya, kinerja model yang sudah di *modelling* kemudian dievaluasi menggunakan metrik *F1-score*, *precision*, *recall*, dan *accuracy* yang dihitung dari *confusion matrix*. Gambar 8 menampilkan *confusion matrix* beserta nilai evaluasinya sebagai indikator performa keseluruhan dari model klasifikasi.



Gambar 8. Confusion Matrix

Setelah diperoleh empat nilai utama dari *confusion matrix*, langkah berikutnya adalah menghitung metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1-score*. Metrik-metrik ini dapat dihitung bedasarkan persamaan (5), (6), (7), (8) yang dapat digunakan sebagai indikator untuk menilai performa model secara keseluruhan, terutama dalam konteks klasifikasi spam komentar.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{1606+2830}{1606+2830+59+73} = \frac{4436}{4568} = 97.1\% \quad (5)$$

$$Precision = \frac{TP}{TP+FP} = \frac{1606}{1606+59} = 96.4\% \quad (6)$$

$$Recall = \frac{TP}{TP+FN} = \frac{1606}{1606+73} = 95.6\% \quad (7)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} = 96\% \quad (8)$$

Berdasarkan hasil perhitungan evaluasi, diperoleh nilai *accuracy* sebesar 97,1%, *precision* 96,4%, *recall* 95,6%, dan *F1-score* 96%. Nilai-nilai ini menunjukkan bahwa model memiliki kemampuan yang baik dalam mengklasifikasikan komentar spam secara akurat dan konsisten. Hal ini sejalan dengan output evaluasi yang diberikan oleh python sebagaimana ditunjukan pada Gambar 9.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.98      0.98      2889
           1       0.96      0.96      0.96      1679

    accuracy                           0.97      4568
```

Gambar 9. Hasil Evaluasi Python

3.7 *Deployment*

Setelah semua tahapan perancangan sistem selesai, seluruh kode yang digunakan untuk klasifikasi komentar spam di YouTube kemudian di konversi menjadi aplikasi untuk sistem operasi *Windows* dengan *format .exe*. Aplikasi ini dirancang agar bisa dijalankan langsung lewat *terminal* atau *PowerShell*, menggunakan interface berbasis *Command Line Interface* (CLI). Dengan begitu, pengguna tidak perlu lagi menjalankan skrip *Python* secara manual. Proses konversi dari kode *Python* ke file *.exe* dilakukan menggunakan *library PyInstaller*. Gambar 10 menunjukan tahapan proses pembangunan aplikasi menggunakan *PyInstaller*. Perintah tersebut menginstruksikan *PyInstaller* untuk melakukan proses *build* terhadap *script* dari sistem klasifikasi komentar spam.

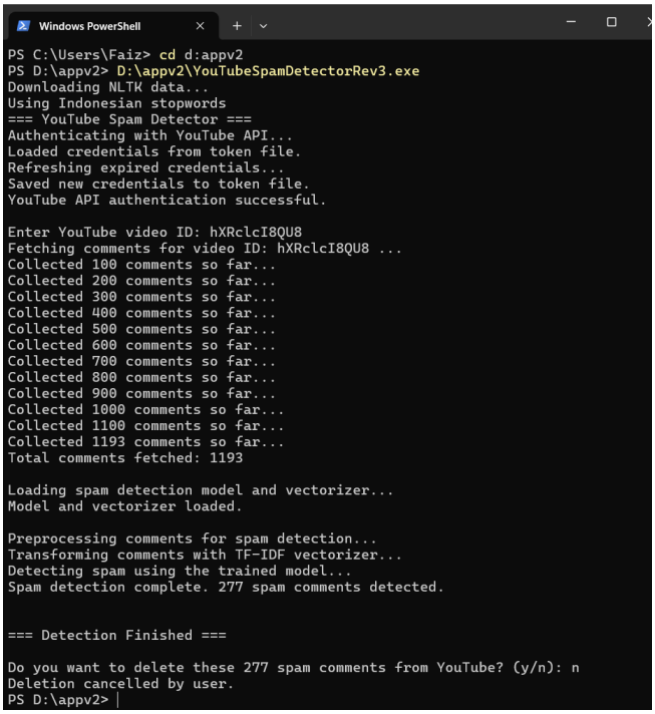
```
PS D:\BETAML> pyinstaller --onefile --name YouTubeSpamDetectorRev3
"Main App RTC EXP.py"
383 INFO: PyInstaller: 6.13.0, contrib hooks: 2025.4
383 INFO: Python: 3.11.1
397 INFO: Platform: Windows-10-10.0.26100-SP0
```

Gambar 8. Proses Konversi Script

Perintah *--onefile* digunakan untuk mengemas seluruh dependensi ke dalam satu file *executable* (.exe), sedangkan opsi *--name* digunakan untuk menentukan nama file aplikasi output yang dihasilkan. Setelah proses kompilasi selesai, *PyInstaller* akan menghasilkan file aplikasi bernama "*YouTubeSpamDetectorRev3.exe*" yang dapat langsung dijalankan oleh pengguna pada sistem operasi *Windows* tanpa *end user* untuk melakukan instalasi *environment* pengembangan *Python*.

Gambar 11 menunjukan *interface* akhir dari sistem yang telah dikembangkan dalam bentuk *Command Line Interface* (CLI). Setelah pengguna memasukkan ID video YouTube, sistem secara otomatis menjalankan seluruh rangkaian proses deteksi komentar spam. Proses ini meliputi pengambilan data komentar dari video terkait, pemuatan model klasifikasi yang telah dilatih sebelumnya, tahap *text preprocessing* dan vektorisasi menggunakan metode TF-

IDF, hingga proses klasifikasi menggunakan algoritma *Naive Bayes*. Setelah hasil klasifikasi ditampilkan, sistem menyediakan opsi bagi pengguna untuk langsung menghapus komentar-komentar yang terdeteksi sebagai spam secara otomatis



Gambar 9. Tampilan CLI Sistem Deteksi Spam

4. KESIMPULAN

Berdasarkan hasil penerapan dan evaluasi, algoritma *Multinomial Naive Bayes* terbukti mampu mengklasifikasikan komentar spam dengan tingkat akurasi yang tinggi dan performa yang stabil. Seluruh tahapan yang dirancang mulai dari pengumpulan data, *text preprocessing* dengan TF-IDF, hingga pelatihan dan validasi model berjalan efektif dan mendukung keberhasilan implementasi sistem klasifikasi. Model menunjukkan nilai metrik evaluasi yang sangat baik, dengan akurasi akhir mencapai 97%, serta skor *precision* 96,4%, *recall* 95,6%, dan *F1-score* 96%. Hal ini menunjukkan bahwa model tidak hanya akurat dalam mengenali komentar spam, tetapi juga presisi dalam meminimalkan kesalahan klasifikasi.

Pencapaian akurasi tersebut merupakan hasil dari serangkaian proses penting yang saling mendukung. Salah satu tahapan yang paling krusial dalam memperoleh performa evaluasi yang optimal adalah proses *text preprocessing*. Pada tahap ini, berbagai teknik normalisasi teks dan pengaturan *hyperparameter* diterapkan untuk menghasilkan data yang bersih dan terstruktur. Selain menghasilkan model yang akurat, sistem yang dibangun juga berhasil diimplementasikan dalam bentuk aplikasi berbasis *Command Line Interface* (CLI) menggunakan modul *PyInstaller*. Aplikasi ini dirancang dengan antarmuka yang sederhana dan dapat digunakan oleh pemilik kanal YouTube untuk mendeteksi dan menangani komentar spam secara mandiri dan efisien.

5. REFERENSI

Abdullah, A. O., Ali, M. A., Karabatak, M., & Sengur, A. (2018). A comparative analysis of common YouTube comment spam filtering techniques. *2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, 1–5. https://doi.org/10.1109/ISDFS.2018.8355315

Christian, H., Agus, M. P., & Suhartono, D. (2016). Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TF-IDF). *ComTech:*

Computer, Mathematics and Engineering Applications, *7*(4), 285. https://doi.org/10.21512/comtech.v7i4.3746

Daniel, J., & Martin, J. H. (2024). *Naive Bayes, Text Classifica tion, and Sentiment*.

Fernando, J. R., Budiraharjo, R., & Haganusa, E. (2019). Spam Classification on 2019 Indonesian President Election Youtube Comments Using Multinomial Naïve-Bayes. *Indonesian Journal of Artificial Intelligence and Data Mining*, *2*(1). https://doi.org/10.24014/ijaidm.v2i1.6445

Ghatasheh, N., Altaharwa, I., & Aldebei, K. (2022). Modified Genetic Algorithm for Feature Selection and Hyper Parameter Optimization: Case of XGBoost in Spam Prediction. *IEEE Access*, *10*, 84365–84383. https://doi.org/10.1109/ACCESS.2022.3196905

Han, J., Pei, J., & Tong, H. (2022). *Data mining: concepts and techniques*. Morgan kaufmann.

Hani, D. (2023). *Klasifikasi Masalah Pada Komunitas Marah-Marah Di Twitter Menggunakan Bidirectional Long Short-Term Memory*.

Hayoung. (2021). A YouTube Spam Comments Detection Scheme Using Cascaded Ensemble Machine Learning Model. *IEEE Access*, *9*, 144121–144128. https://doi.org/10.1109/ACCESS.2021.3121508

Kudo, T., & Richardson, J. (2018). *SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing*. http://arxiv.org/abs/1808.06226

Mahmoud, T. M., El Nashar, A. I., Abd-El-Hafeez, T., & Khairy, M. (2014). An Efficient Three-phase Email Spam Filtering Technique. In *British Journal of Mathematics & Computer Science* (Vol. 4, Issue 9). www.sciencedomain.org

Makarin, A. A., & Astuti, L. (2023). Faktor yang Mempengaruhi Mahasiswa Melakukan Perjudian Online. *Indonesian Journal of Criminal Law and Criminology (IJCLC)*, *3*(3), 180–189. https://doi.org/10.18196/ijclc.v3i3.17674

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Paramartha, P., Dewi, A., & Seputra, P. (2021). Sanksi Pidana terhadap Para Pemasang dan Promosi Iklan Bermuatan Konten Judi Online. *Jurnal Preferensi Hukum*, *2*(1), 156–160. https://doi.org/10.22225/jph.2.1.3062.156-160

Rangga Gelar Guntara. (2023). Aplikasi Deteksi Phising Berbasis Android Menggunakan Metode Pengembangan Perangkat Lunak DSRM. *Jurnal Minfo Polgan*, *12*(1), 303–310. https://doi.org/10.33395/jmp.v12i1.12379

Samsudin, N. M., Mohd Foozy, C. F. B., Alias, N., Shamala, P., Othman, N. F., & Wan Din, W. I. S. (2019). Youtube spam detection framework using naïve bayes and logistic regression. *Indonesian Journal of Electrical Engineering and Computer Science*, *14*(3), 1508–1517. https://doi.org/10.11591/ijeecs.v14.i3.pp1508-1517

Schröer, C., Kruse, F., & Gómez, J. M. (2021). A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, *181*, 526–534. https://doi.org/10.1016/j.procs.2021.01.199

Shahriar, A. (2024). *Improving Bengali and Hindi Large Language Models*.

Sugiyono, D. (2013). *Metode penelitian pendidikan pendekatan kuantitatif, kualitatif dan R&D*.

Tantithamthavorn, C., McIntosh, S., Hassan, A. E., & Matsumoto, K. (2017). An Empirical Comparison of Model Validation Techniques for Defect Prediction Models. *IEEE Transactions on Software Engineering*, *43*(1), 1–18. https://doi.org/10.1109/TSE.2016.2584050

Yadav, S., & Shukla, S. (2016). Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. *Proceedings - 6th International Advanced Computing Conference, IACC 2016*, 78–83. https://doi.org/10.1109/IACC.2016.25

Zeng, G. (2020). On the confusion matrix in credit scoring and its analytical properties. *Communications in Statistics - Theory and Methods*, *49*(9), 2080–2093. https://doi.org/10.1080/03610926.2019.1568485