

INFORMASI PROYEK

Judul Proyek :

Prediksi Komentar Spam pada YouTube Menggunakan Machine Learning dan Deep Learning

Nama Mahasiswa	:	ABYAN RAGA KUSUMA
NIM	:	234311001
Program Studi	:	TEKNOLOGI REKAYASA PERANGKAT LUNAK
Mata Kuliah	:	DATA SCIENCE
Dosen Pengampu	:	GUS NANANG SYAIFUDDIIN
Tahun Akademik	:	2025
Link GitHub Repository	:	https://github.com/abyan441/234311001_UAS_DataScience
Link Video Pembahasan	:	https://youtu.be/qc0GdaXAhw0

1. LEARNING OUTCOMES

Pada proyek ini, mahasiswa diharapkan dapat:

1. Memahami konteks masalah dan merumuskan problem statement secara jelas
2. Melakukan analisis dan eksplorasi data (EDA) secara komprehensif (**OPSIONAL**)
3. Melakukan data preparation yang sesuai dengan karakteristik dataset
4. Mengembangkan tiga model machine learning yang terdiri dari (**WAJIB**):
 - Model baseline
 - Model machine learning / advanced
 - Model deep learning (**WAJIB**)
5. Menggunakan metrik evaluasi yang relevan dengan jenis tugas ML
6. Melaporkan hasil eksperimen secara ilmiah dan sistematis
7. Mengunggah seluruh kode proyek ke GitHub (**WAJIB**)
8. Menerapkan prinsip software engineering dalam pengembangan proyek

2. PROJECT OVERVIEW

2.1 Latar Belakang

YouTube sebagai platform berbagi video terbesar menerima jutaan komentar setiap hari. Namun, banyak komentar tersebut merupakan spam, seperti promosi tidak relevan, tautan berbahaya, atau pesan otomatis yang mengganggu kualitas diskusi dan dapat merugikan pengguna. Hal ini membuat deteksi komentar spam menjadi kebutuhan penting pada sistem moderasi konten.

Penelitian oleh (Alberto dkk., 2015) menunjukkan bahwa komentar spam di YouTube merupakan masalah nyata, dan teknik klasifikasi otomatis seperti Naive Bayes serta Random Forest mampu meningkatkan efektivitas penyaringan komentar (*TubeSpam: Comment Spam Filtering on YouTube*). Sementara itu, studi terbaru oleh (Airlangga, 2024) menemukan bahwa model deep learning—khususnya LSTM—dapat mencapai akurasi hingga 95,65% dalam mendeteksi spam, melampaui metode machine learning klasik. Kedua penelitian ini menegaskan pentingnya pengembangan sistem otomatis berbasis NLP untuk deteksi komentar spam.

Dengan demikian, proyek ini penting untuk dilakukan guna membantu moderasi komentar secara otomatis, meningkatkan kenyamanan pengguna, dan memberikan kontribusi terhadap pengembangan penelitian di bidang text classification dan NLP.

3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING

3.1 Problem Statements

1. Sistem perlu mampu mengklasifikasikan komentar YouTube menjadi spam atau non-spam dengan tingkat akurasi yang cukup tinggi, meskipun komentar sering pendek, tidak baku, dan mengandung variasi bahasa informal.
2. Dataset komentar mengandung banyak noise seperti emoticon, tautan, huruf kapital berlebih, serta teks tidak terstruktur, sehingga membutuhkan proses preprocessing yang tepat sebelum dilakukan pemodelan.
3. Model machine learning tradisional memiliki keterbatasan dalam memahami konteks teks, sehingga diperlukan model deep learning seperti LSTM yang mampu mempelajari pola urutan kata secara lebih efektif.
4. Sistem deteksi spam harus dapat bekerja secara otomatis dan stabil untuk membantu moderasi komentar dalam skala besar, sehingga solusi yang dikembangkan harus tepat, efisien, dan mudah diimplementasikan.

3.2 Goals

1. Mengembangkan tiga model prediksi (baseline, advanced machine learning, dan deep learning) untuk mengklasifikasikan komentar YouTube menjadi spam atau non-spam, serta membandingkan performanya secara sistematis.
2. Mencapai performa akurasi dan F1-score yang tinggi pada model akhir, sehingga mampu mendeteksi komentar spam secara efektif meskipun teks pendek, informal, dan bervariasi.
3. Menerapkan preprocessing teks yang tepat untuk mengatasi noise pada data, seperti pembersihan karakter khusus, normalisasi teks, penghapusan tautan, dan tokenisasi, sehingga meningkatkan kualitas input data.
4. Mengidentifikasi model terbaik berdasarkan metrik evaluasi (Accuracy, Precision, Recall, dan F1-Score) yang paling sesuai dalam memfilter komentar spam secara otomatis.

3.3 Solution Approach

Model 1 – Baseline Model (Naive Bayes)

Alasan Pemilihan:

Naive Bayes merupakan model dasar yang sangat umum digunakan pada tugas klasifikasi teks. Model ini bekerja baik pada data berbasis frekuensi kata (TF-IDF), memiliki komputasi cepat, dan dapat menjadi pembanding awal terhadap model yang lebih kompleks.

Kelebihan:

- Sangat cepat dilatih dan diimplementasikan
- Cocok untuk dataset teks pendek
- Performanya stabil sebagai baseline

Peran dalam proyek:

Menjadi acuan apakah model lebih kompleks benar-benar memberikan peningkatan performa.

Model 2 – Advanced / ML Model (Random Forest Classifier)

Alasan Pemilihan:

Random Forest mampu menangkap pola non-linear dan bekerja baik pada data hasil vektorisasi TF-IDF. Selain itu, model ini tahan terhadap overfitting dan cocok untuk dataset berukuran kecil-menengah seperti komentar YouTube.

Kelebihan:

- Robust terhadap noise
- Dapat mempelajari interaksi antar fitur

Peran dalam proyek:

Menjadi model machine learning tingkat lanjut untuk dibandingkan dengan baseline dan model deep learning.

Model 3 – Deep Learning Model (LSTM)

Alasan Pemilihan:

Berdasarkan penelitian terbaru, LSTM terbukti unggul untuk tugas klasifikasi komentar YouTube karena mampu memahami urutan dan konteks kata. LSTM juga menunjukkan performa terbaik dalam penelitian *Spam Detection in YouTube Comments Using Deep Learning Models* (Airlangga, 2024) dengan akurasi mencapai 95,65%.

Kelebihan:

- Mampu memahami konteks antar kata
- Unggul pada teks pendek dan tidak terstruktur
- Cocok untuk data NLP berbasis urutan

Peran dalam proyek:

Menjadi model utama untuk mencapai performa tertinggi dalam mendeteksi komentar spam.

4. DATA UNDERSTANDING

4.1 Informasi Dataset

Sumber Dataset:

<https://archive.ics.uci.edu/dataset/380/youtube+spam+collection>

Dataset yang digunakan merupakan bagian dari *YouTube Spam Collection Dataset*, khususnya file *Youtube02-KatyPerry.csv*, yang berisi kumpulan komentar YouTube pada video artis Katy Perry.

Deskripsi Dataset:

- Jumlah baris (rows): 350
- Jumlah kolom (columns/features): 5
- Tipe data: Text
- Ukuran dataset: <1MB
- Format file: CSV

4.2 Deskripsi Fitur

Jelaskan setiap fitur/kolom yang ada dalam dataset. **Contoh tabel:**

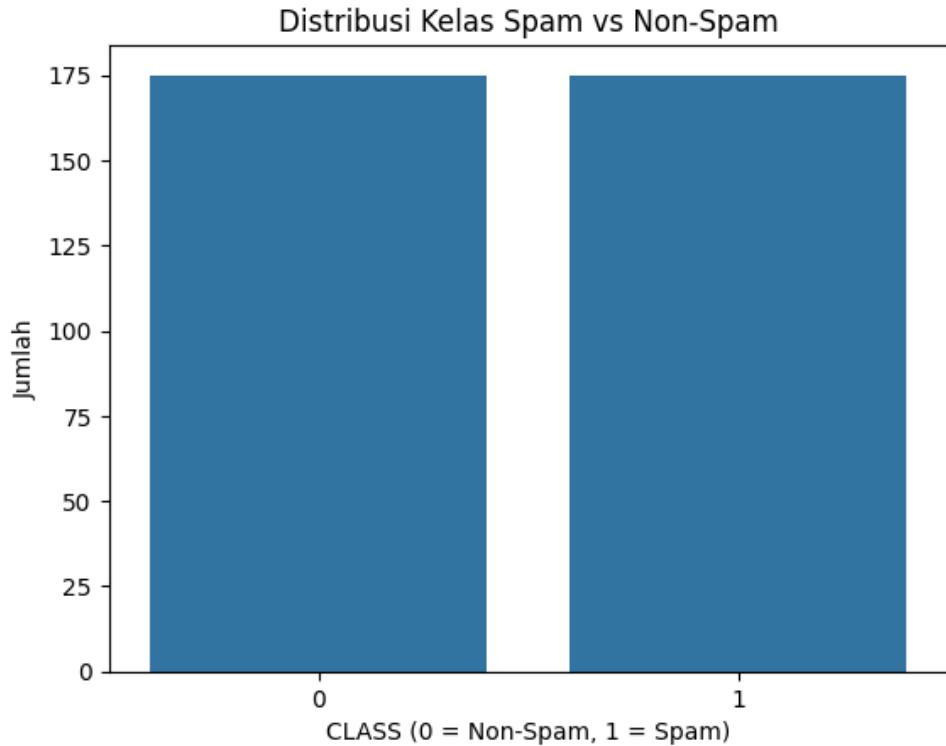
Nama Fitur	Tipe Data	Deskripsi	Contoh Nilai
AUTHOR	Categorical	Nama pengguna yang memberikan komentar	Erica Ross
DATE	Categorical	Tanggal komentar diposting	2014-07-27T02:51:43
CONTENT	Categorical	Isi komentar YouTube	“Hey guys! Please join me in my fight to help a...”

4.3 Kondisi Data

- **Missing Values:** Tidak ada
- **Duplicate Data:** Tidak ada
- **Outliers:** Ada, terutama komentar yang jauh lebih panjang ($\geq 80\text{--}100$ kata) dibanding komentar lain
- **Imbalanced Data:** Tidak ada (jumlah spam = jumlah non-spam)
- **Noise:** sangat umum dalam komentar YouTube
 - emoji   
 - tautan: <http://bit.ly/>
 - spam promosi: “subscribe my channel pls!!”
 - teks tidak baku: “chek this out!!!”
 - huruf kapital berlebih
- **Data Quality Issues:** Tidak ada

4.4 Exploratory Data Analysis (EDA) - (OPSIONAL)

Visualisasi 1: Distribusi Kelas Spam vs Non-Spam (Bar Plot)



Insight:

Visualisasi menunjukkan bahwa jumlah komentar spam (kelas 1) dan non-spam (kelas 0) berada dalam jumlah yang seimbang, masing-masing sekitar 175 komentar. Kondisi ini memberikan beberapa keuntungan penting dalam proses pemodelan:

1. Tidak Ada Masalah Class Imbalance

Dataset yang seimbang memastikan bahwa model tidak memiliki kecenderungan untuk memprediksi salah satu kelas lebih dominan dibanding kelas lainnya. Ini membuat proses training lebih stabil.

2. Metrik Evaluasi Seperti Accuracy Lebih Valid

Pada dataset yang tidak seimbang, accuracy biasanya menyesatkan. Namun karena kedua kelas memiliki proporsi yang sama, akurasi dapat digunakan sebagai metrik evaluasi yang representatif.

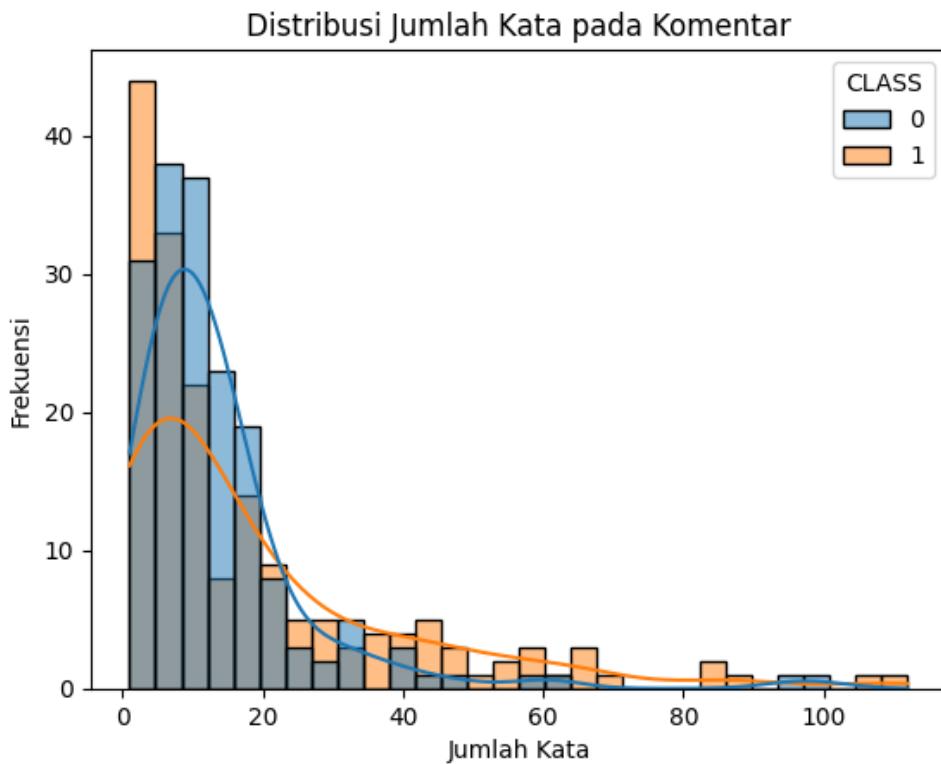
3. Tidak Memerlukan Teknik Penyeimbangan Data

Karena distribusi kelas sudah proporsional, tidak diperlukan metode seperti SMOTE, undersampling, atau oversampling.

4. Model Berpeluang Belajar Pola Secara Adil

Model machine learning akan memiliki kesempatan yang sama untuk mempelajari karakteristik komentar spam dan non-spam, sehingga dapat menghasilkan performa yang lebih baik dan tidak bias.

Visualisasi 2: Distribusi Jumlah Kata pada Komentar (Histogram + KDE)



Insight:

Visualisasi distribusi jumlah kata pada komentar menunjukkan bahwa komentar spam cenderung memiliki panjang yang lebih pendek, umumnya berada pada rentang 1–10 kata. Hal ini sesuai dengan pola komentar spam yang biasanya berbentuk ajakan singkat, tautan promosi, atau pesan berulang yang sering dihasilkan oleh bot. Sementara itu, komentar non-spam memiliki distribusi panjang yang lebih bervariasi, dengan sebagian pengguna menulis komentar yang lebih panjang (20–40 kata) untuk menjelaskan pendapat, memberikan apresiasi, atau berdiskusi mengenai isi video. Meskipun demikian, kedua kelas sama-sama didominasi oleh komentar pendek, yaitu kurang dari 15 kata, sehingga dataset ini dapat dikategorikan sebagai *short-text dataset*. Kondisi ini penting dalam pemilihan model dan teknik NLP, seperti penggunaan TF-IDF serta penerapan preprocessing yang tepat. Selain itu, ditemukan adanya outlier pada kelas spam berupa komentar yang sangat panjang (lebih dari 80 kata), yang kemungkinan besar berasal dari pesan promosi panjang atau script spam otomatis. Keberadaan outlier ini menunjukkan adanya variasi gaya penulisan spam, meskipun mayoritas spam tetap berupa komentar singkat.

Visualisasi 3: Wordcloud Spam & Wordcloud Non-Spam



Insight:

Visualisasi wordcloud menunjukkan perbedaan yang sangat jelas antara pola kata pada komentar spam dan non-spam. Pada wordcloud komentar spam, kata-kata yang paling dominan adalah “*please*”, “*subscribe*”, “*check*”, “*channel*”, “*video*”, “*facebook*”, dan “*https*”. Dominasi kata-kata ini menunjukkan bahwa komentar spam umumnya berisi ajakan untuk mengunjungi tautan tertentu, mempromosikan channel, atau mengarahkan pengguna ke platform lain. Kehadiran kata seperti “*free*”, “*amp*”, “*help*”, dan “*watch*” juga mengindikasikan bahwa spam sering bersifat promosi, repetitif, dan terkadang mengandung tautan berpotensi berbahaya. Pola ini sejalan dengan sifat spam yang sering kali dibuat secara otomatis atau massal.

Sebaliknya, wordcloud komentar non-spam didominasi oleh kata-kata seperti “*Katy*”, “*Perry*”, “*song*”, “*video*”, “*love*”, dan “*Roar*”. Kata-kata ini menunjukkan bahwa komentar non-spam lebih fokus pada isi konten video, opini penonton, dan apresiasi terhadap karya artis. Kata-kata seperti “*awesone*”, “*good*”, “*amazing*”, dan “*music*” menggambarkan percakapan organik dari pengguna yang benar-benar menonton video tersebut.

Secara keseluruhan, kedua wordcloud ini memperlihatkan bahwa komentar spam memiliki pola kata yang berorientasi pada promosi dan tindakan tertentu, sedangkan komentar non-spam berorientasi pada interaksi alami dan tanggapan terhadap konten video. Perbedaan pola ini menjadi dasar yang kuat bagi model NLP untuk membedakan spam dan non-spam secara efektif.

5. DATA PREPARATION

Bagian ini menjelaskan **semua** proses transformasi dan preprocessing data yang dilakukan.

5.1 Data Cleaning

Pada tahap data cleaning, dilakukan pemeriksaan kualitas data untuk memastikan dataset siap digunakan pada proses selanjutnya. Hasil analisis menunjukkan bahwa dataset tidak memiliki missing values pada seluruh fitur, sehingga tidak diperlukan proses imputasi. Pemeriksaan duplikasi melalui `df.duplicated()` juga menunjukkan bahwa tidak terdapat komentar yang duplikat, sehingga tidak ada baris yang perlu dihapus. Selain itu, ditemukan beberapa outlier berupa komentar yang memiliki panjang kata jauh lebih besar dibandingkan komentar lainnya, namun outlier tersebut tetap dipertahankan karena mencerminkan pola spam tertentu seperti promosi panjang atau script otomatis yang relevan bagi proses pembelajaran model. Dari sisi tipe data, seluruh kolom telah memiliki tipe yang sesuai—teks disimpan sebagai object dan label CLASS sebagai `int64`—sehingga tidak diperlukan perubahan tipe data tambahan. Secara keseluruhan, dataset dinyatakan bersih dan dapat langsung digunakan pada tahap preprocessing dan feature engineering berikutnya.

5.2 Feature Engineering

Feature selection dilakukan secara manual dengan memilih hanya fitur CONTENT sebagai masukan model. Dari tiga fitur yang tersedia (AUTHOR, DATE, dan CONTENT), hanya CONTENT yang relevan karena berisi isi komentar yang menjadi dasar penentuan spam. Fitur AUTHOR dan DATE tidak digunakan karena tidak memiliki hubungan logis dengan klasifikasi spam dan berpotensi menambah noise tanpa memberikan informasi yang bermakna.

Fitur yang Dipilih

- CONTENT (isi komentar Youtube)

Alasan Pemilihan Fitur

- Isi komentar adalah sumber informasi utama yang membedakan spam dan non-spam.
- Menghindari noise dari fitur yang tidak relevan (AUTHOR dan DATE).
- Memperkuat fokus analisis sesuai tujuan proyek, yaitu klasifikasi komentar berdasarkan isi teks.
- Mengurangi risiko overfitting, terutama pada model deep learning dengan dataset berukuran kecil.

5.3 Data Transformation

Transformasi untuk Model Machine Learning (Naive Bayes & Random Forest)

1. Lowercasing

Seluruh teks pada kolom CONTENT diubah menjadi huruf kecil agar model tidak membedakan kata yang sama dengan kapitalisasi berbeda. Contoh: “Check” dan “check” dianggap sama.

2. Removing Punctuation & Symbols

Tanda baca, emoji, dan simbol yang tidak relevan dihapus untuk mengurangi noise.

Contoh penghapusan: !, ?, :), 😊, http://, www.

3. Removing Stopwords

Kata umum seperti “the”, “is”, “and” dihapus agar model fokus pada kata bermakna.

4. Stemming/Lemmatization

Kata dikembalikan ke bentuk dasar untuk mengurangi variasi kata. Contoh: “running” → “run”.

5. Tokenization

Komentar dipecah menjadi token kata untuk memudahkan transformasi berikutnya.

6. TF-IDF Vectorization

Teks diubah menjadi representasi numerik menggunakan Term Frequency–Inverse Document Frequency, yang efektif untuk teks pendek seperti komentar YouTube.

TF-IDF digunakan sebagai input untuk model Naive Bayes dan Random Forest.

Transformasi untuk Model Deep Learning (LSTM)

1. Tokenization (Word Indexing)

Setiap kata pada komentar diubah menjadi indeks angka menggunakan tokenizer.

2. Padding Sequences

Seluruh komentar diseragamkan panjangnya menggunakan padding (misalnya padding ke panjang 100 token).

Tujuan: agar semua input memiliki dimensi sama saat diproses oleh LSTM.

5.4 Data Splitting

Pada tahap ini, dataset dibagi menjadi dua bagian utama, yaitu training set dan test set. Pembagian dilakukan menggunakan teknik Stratified Split, yaitu metode pemisahan data yang memastikan distribusi kelas (spam vs non-spam) tetap seimbang pada kedua subset data. Hal ini penting karena model klasifikasi akan belajar secara lebih stabil ketika proporsi kelas pada training set dan test set tidak berubah.

Proses splitting dilakukan dengan rasio:

- Training set: 80% → 280 sampel
- Test set: 20% → 70 sampel
- Random State: 42 untuk memastikan hasil pembagian dapat direproduksi (reproducibility)

Penggunaan stratified split dipilih karena dataset memiliki dua kelas yang seimbang (175 spam, 175 non-spam). Dengan cara ini, masing-masing subset tetap mempertahankan proporsi yang sama, sehingga evaluasi model lebih representatif dan tidak bias.

5.5 Data Balancing (jika diperlukan)

Pada tahap ini dilakukan pemeriksaan distribusi kelas untuk menentukan apakah diperlukan teknik penyeimbangan data. Hasil analisis menunjukkan bahwa dataset memiliki distribusi kelas yang sepenuhnya seimbang, yaitu:

- Kelas Spam: 175 sampel
- Kelas Non-Spam: 175 sampel

Karena kedua kelas memiliki jumlah sampel yang sama, maka proses data balancing tidak diperlukan. Teknik seperti SMOTE, Random Undersampling, Oversampling, maupun penyesuaian class weights tidak digunakan karena dapat berpotensi menambah noise atau menurunkan kualitas data yang sebenarnya sudah ideal.

Kondisi dataset yang seimbang ini juga membantu model belajar secara lebih stabil dan menghindari bias terhadap salah satu kelas, sehingga proses pelatihan dapat dilakukan langsung tanpa manipulasi distribusi data.

5.6 Ringkasan Data Preparation

1. Data Cleaning

Apa yang dilakukan:

Membersihkan dataset dari potensi masalah seperti missing values, duplikasi, dan noise pada teks komentar.

Mengapa penting:

Data yang bersih memastikan proses pelatihan model berjalan dengan baik, mengurangi bias, dan mencegah error selama pemodelan.

Bagaimana dilakukan:

Dilakukan pengecekan menggunakan fungsi isnull() untuk missing values dan duplicated() untuk duplikasi, yang menunjukkan bahwa keduanya tidak ditemukan. Outlier panjang komentar tetap dipertahankan karena merepresentasikan pola spam tertentu. Semua tipe data sudah sesuai sehingga tidak diperlukan konversi tambahan.

2. Feature Engineering

Apa yang dilakukan:

Memilih fitur yang relevan untuk pemodelan, yaitu hanya kolom CONTENT.

Mengapa penting:

Hanya isi komentar yang memberikan informasi signifikan untuk menentukan apakah komentar merupakan spam atau bukan. Fitur lain seperti AUTHOR dan DATE berpotensi menambah noise.

Bagaimana dilakukan:

Dilakukan manual feature selection berdasarkan analisis domain, mempertahankan hanya fitur CONTENT sebagai input model.

3. Data Transformation

Apa yang dilakukan:

Melakukan preprocessing teks untuk mengubah komentar mentah menjadi representasi yang dapat dibaca model.

Mengapa penting:

Model machine learning dan deep learning tidak bisa memproses teks mentah, sehingga harus diubah menjadi format numerik.

Bagaimana dilakukan:

Transformasi meliputi lowercasing, penghapusan URL dan simbol, stopword removal, lemmatization, tokenization, dan pembuatan representasi numerik menggunakan TF-IDF untuk model ML serta padding sequence untuk calon model LSTM.

4. Data Splitting

Apa yang dilakukan:

Membagi dataset menjadi training dan test set.

Mengapa penting:

Agar model dapat dievaluasi secara objektif menggunakan data yang tidak pernah dilihat sebelumnya, serta menghindari overfitting.

Bagaimana dilakukan:

Menggunakan stratified train-test split dengan perbandingan 80:20, sehingga distribusi kelas tetap seimbang pada kedua subset. Random state = 42 digunakan untuk memastikan hasil dapat direproduksi.

5. Data Balancing

Apa yang dilakukan:

Mengecek apakah diperlukan teknik penyeimbangan kelas seperti SMOTE atau class weights.

Mengapa penting:

Dataset yang tidak seimbang dapat membuat model bias pada kelas mayoritas.

Bagaimana dilakukan:

Dilakukan analisis distribusi kelas dan ditemukan bahwa jumlah data spam dan non-spam sama (175:175), sehingga proses balancing tidak diperlukan.

6. MODELING

6.1 Model 1 — Baseline Model

6.1.1 Deskripsi Model

Nama Model: Multinomial Naive Bayes

Teori Singkat:

Multinomial Naive Bayes adalah model probabilistik yang bekerja berdasarkan Teorema Bayes, yaitu menghitung kemungkinan sebuah teks termasuk ke dalam kelas tertentu (spam atau non-spam) berdasarkan frekuensi kata-katanya. Model ini mengasumsikan bahwa setiap fitur (kata) bersifat independen, sehingga perhitungannya sangat efisien dan cepat. Dalam konteks pemrosesan teks, model ini menghitung probabilitas suatu komentar mengandung kata-kata tertentu, lalu menentukan kelas dengan probabilitas tertinggi.

Model ini sangat efektif digunakan pada representasi teks berbasis TF-IDF atau bag-of-words, terutama untuk dataset dengan komentar pendek seperti YouTube comments.

Alasan Pemilihan:

Model Multinomial Naive Bayes dipilih sebagai baseline karena:

1. Sederhana dan sangat cepat, sehingga cocok untuk membangun tolok ukur awal sebelum menggunakan model yang lebih kompleks.
2. Efektif untuk data teks, terutama pada kasus spam detection, di mana pola kata sangat berpengaruh.
3. Kinerja stabil pada dataset kecil, sehingga tepat untuk dataset 350 komentar seperti pada penelitian ini.
4. Banyak literatur sebelumnya menyebutkan bahwa Naive Bayes merupakan standar baseline dalam klasifikasi komentar spam.

6.1.2 Hyperparameter

Parameter yang digunakan:

- `alpha = 1.0`

Parameter smoothing (Laplace smoothing). Digunakan untuk mencegah probabilitas nol pada kata yang jarang muncul.

- `fit_prior = True`

Model menghitung prior probability dari masing-masing kelas berdasarkan distribusi data. Karena dataset seimbang, nilai prior dapat digunakan apa adanya.

- `class_prior = None`

Tidak ditentukan secara manual, sehingga model menghitungnya otomatis.

6.1.3 Implementasi (Ringkas)

```
from sklearn.naive_bayes import MultinomialNB  
  
model_baseline = MultinomialNB()  
  
model_baseline.fit(X_train, y_train)  
y_pred_baseline = model_baseline.predict(X_test)
```

6.1.4 Hasil Awal

Classification Report:				
	precision	recall	f1-score	support
Non-Spam	0.77	0.86	0.81	35
Spam	0.84	0.74	0.79	35
accuracy			0.80	70
macro avg	0.80	0.80	0.80	70
weighted avg	0.80	0.80	0.80	70

Accuracy = 0.80 (80%) Artinya:

Dari seluruh data uji (70 komentar), model berhasil mengklasifikasikan 80% komentar dengan benar.

Untuk baseline model, akurasi ini cukup baik.

Precision dan Recall per Kelas

1. Non-Spam (class = 0)

- Precision = 0.77
Dari semua komentar yang diprediksi *Non-Spam*, 77% benar-benar non-spam.
- Recall = 0.86
Dari semua komentar non-spam yang sebenarnya ada, model berhasil menemukan 86%.

Model lebih mudah mengenali Non-Spam (recall tinggi).

2. Spam (class = 1)

- Precision = 0.84
Dari semua komentar yang diprediksi *Spam*, 84% benar-benar spam.
- Recall = 0.74
Dari semua komentar spam yang sebenarnya ada, model hanya menemukan 74%.

Model lebih hati-hati dalam memutuskan komentar itu spam (cenderung sedikit meleset pada recall).

3. F1-Score

- Non-Spam → 0.81

- Spam → 0.79

F1-score sekitar 0.8 menunjukkan:

Model seimbang antara kemampuan mendeteksi spam dan non-spam.

6.2 Model 2 — ML / Advanced Model

6.2.1 Deskripsi Model

Nama Model: Random Forest Classifier

Teori Singkat:

Teori Random Forest adalah algoritma ensemble learning yang bekerja dengan membangun banyak decision tree secara independen dan menggabungkan hasil prediksinya melalui mekanisme voting. Setiap pohon dilatih menggunakan subset data dan subset fitur yang dipilih secara acak (bootstrap aggregating). Dengan pendekatan ini, Random Forest mampu mengurangi risiko overfitting yang sering terjadi pada decision tree tunggal serta menghasilkan prediksi yang lebih stabil dan akurat.

Dalam konteks klasifikasi teks, Random Forest memanfaatkan fitur numerik hasil ekstraksi teks (seperti TF-IDF) untuk mempelajari pola kombinasi kata yang membedakan komentar spam dan non-spam. Singkat:

Alasan Pemilihan:

Random Forest dipilih sebagai model lanjutan karena:

1. Mampu menangkap hubungan non-linear antar fitur yang tidak dapat ditangkap oleh model probabilistik sederhana seperti Naive Bayes.
2. Memiliki performa yang baik pada berbagai jenis dataset tanpa membutuhkan tuning yang terlalu kompleks.
3. Cocok digunakan sebagai pembanding untuk mengukur peningkatan performa dibanding baseline model.

Keunggulan:

- Mampu menangani data berdimensi tinggi seperti hasil TF-IDF.
- Lebih robust terhadap noise dan outlier dibanding decision tree tunggal.
- Mengurangi overfitting melalui mekanisme ensemble.
- Tidak terlalu sensitif terhadap skala fitur.

Kelemahan:

- Waktu pelatihan lebih lama dibanding Naive Bayes.
- Model lebih kompleks dan kurang interpretatif.
- Membutuhkan memori lebih besar karena menyimpan banyak pohon keputusan.

6.2.2 Hyperparameter

Parameter yang digunakan:

```
rf_model = RandomForestClassifier(  
    n_estimators=100,  
    max_depth=None,  
    min_samples_split=2,  
    random_state=42,  
    n_jobs=-1  
)
```

Model Random Forest Classifier dikonfigurasi menggunakan beberapa hyperparameter utama yang berpengaruh langsung terhadap proses pembelajaran dan kompleksitas model. Hyperparameter yang digunakan dalam penelitian ini adalah sebagai berikut:

- n_estimators = 100
Menentukan jumlah decision tree yang dibangun dalam ensemble Random Forest. Jumlah pohon yang cukup besar membantu meningkatkan stabilitas dan akurasi prediksi.
- max_depth = None
Menunjukkan bahwa kedalaman maksimum pohon tidak dibatasi, sehingga model dapat mempelajari pola data secara lebih fleksibel.
- min_samples_split = 2
Menentukan jumlah minimum sampel yang diperlukan untuk membagi sebuah node. Nilai default digunakan untuk memungkinkan proses pemisahan node berlangsung secara optimal.

Selain hyperparameter utama tersebut, digunakan pula parameter pendukung untuk keperluan teknis:

- random_state = 42
Digunakan untuk memastikan hasil pelatihan model bersifat reproduksibel.
- n_jobs = -1
Digunakan untuk memanfaatkan seluruh core CPU yang tersedia guna meningkatkan efisiensi komputasi, tanpa memengaruhi hasil pembelajaran model.

Hyperparameter Tuning (jika dilakukan):

hyperparameter tuning tidak dilakukan, karena model Random Forest digunakan sebagai model lanjutan untuk dibandingkan dengan baseline model menggunakan konfigurasi standar yang umum digunakan.

6.2.3 Implementasi (Ringkas)

```
rf_model = RandomForestClassifier(  
    n_estimators=100,  
    max_depth=None,  
    min_samples_split=2,  
    random_state=42,  
    n_jobs=-1  
)  
  
# Melatih model  
rf_model.fit(X_train, y_train)  
  
# Melakukan prediksi pada data test  
y_pred_rf = rf_model.predict(X_test)
```

6.2.4 Hasil Model

Classification Report:				
	precision	recall	f1-score	support
Non-Spam	0.85	0.63	0.72	35
Spam	0.70	0.89	0.78	35
accuracy			0.76	70
macro avg	0.78	0.76	0.75	70
weighted avg	0.78	0.76	0.75	70

Berdasarkan hasil evaluasi, model Random Forest Classifier mencapai akurasi sebesar 76%. Model menunjukkan kinerja yang baik dalam mendeteksi komentar spam dengan nilai recall tinggi sebesar 0.89, namun memiliki recall yang lebih rendah pada kelas non-spam (0.63), yang menandakan adanya beberapa komentar non-spam yang salah diklasifikasikan sebagai spam. Nilai F1-score yang relatif seimbang pada kedua kelas menunjukkan bahwa model mampu menjaga keseimbangan antara precision dan recall, meskipun cenderung memprioritaskan pendekripsiannya.

6.3 Model 3 — Deep Learning Model (WAJIB)

6.3.1 Deskripsi Model

Nama Model: Long Short-Term Memory (LSTM)

Alasan Pemilihan:

Model LSTM dipilih karena dataset yang digunakan berupa teks komentar YouTube yang memiliki struktur urutan kata. LSTM mampu menangkap hubungan antar kata dalam sebuah komentar, sehingga lebih sesuai untuk tugas klasifikasi teks dibandingkan model machine learning konvensional yang tidak mempertimbangkan urutan kata. Selain itu, LSTM relatif mudah diimplementasikan dan cukup efektif untuk dataset berukuran kecil hingga menengah.

6.3.2 Arsitektur Model

Deskripsi Layer:

1. Input Layer
Menerima input berupa urutan token hasil tokenisasi teks dengan panjang maksimum (*max sequence length*) tertentu.
2. Embedding Layer
Mengubah indeks kata menjadi vektor numerik berdimensi tetap sehingga dapat merepresentasikan makna kata secara lebih informatif.
 - Output dimension: 128
3. LSTM Layer
Layer LSTM dengan 64 unit digunakan untuk mempelajari pola urutan kata dan dependensi konteks dalam komentar teks.
 - Units: 64
 - Return sequences: False
4. Dropout Layer
Digunakan untuk mengurangi overfitting dengan menonaktifkan sebagian neuron secara acak selama proses pelatihan.
 - Dropout rate: 0.3
5. Output Layer
Layer output dengan satu neuron untuk klasifikasi biner spam dan non-spam.
 - Units: 1
 - Activation: Sigmoid

No	Layer	Output Shape	Parameter	Keterangan
1	Input Layer	(100,)	-	Sequence token hasil tokenisasi
2	Embedding	(100, 128)	vocab_size × 128	Representasi kata
3	LSTM (64 units)	(64,)	49,480	Menangkap dependensi urutan kata
4	Dropout (0.3)	(64,)	0	Mencegah overfitting
5	Dense (1, Sigmoid)	(1,)	65	Output klasifikasi spam / non-spam

6.3.3 Input & Preprocessing Khusus

Input shape:

Input yang digunakan pada model deep learning berupa urutan token hasil tokenisasi teks komentar. Setiap komentar direpresentasikan sebagai vektor dengan panjang tetap (*fixed-length sequence*).

- Input shape: $(max_len,)$
- max_len ditentukan berdasarkan panjang maksimum komentar setelah proses tokenisasi dan padding.

Preprocessing khusus untuk DL:

- Tokenization
Teks komentar diubah menjadi urutan indeks kata menggunakan tokenizer, sehingga dapat diproses oleh model neural network.
- Padding Sequences
Seluruh urutan token diseragamkan panjangnya menggunakan *padding* agar memiliki dimensi input yang konsisten.
- Lowercasing
Seluruh teks diubah menjadi huruf kecil untuk mengurangi variasi kata yang tidak perlu.
- Removal of Noise
Karakter khusus, tanda baca, URL, dan simbol yang tidak relevan dihilangkan untuk meningkatkan kualitas representasi teks.
- Word Embedding
Embedding layer digunakan untuk mengubah indeks kata menjadi vektor numerik berdimensi tetap, sehingga model dapat menangkap hubungan semantik antar kata.

6.3.4 Hyperparameter

Training Configuration:

- Optimizer: Adam
- Learning rate: 0.001 (default dari Adam)
- Loss function: Binary Crossentropy
- Metrics: Accuracy
- Batch size: 32
- Epochs: 10
- Validation split: 0.2
- Callbacks: EarlyStopping (Monitor: val_loss, Patience: 2, Restore: best weights:True)

6.3.5 Implementasi (Ringkas)

Framework: TensorFlow/Keras

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping

# Arsitektur model LSTM
model_dl = Sequential([
    Embedding(input_dim=MAX_WORDS, output_dim=128, input_length=MAX_LEN),
    LSTM(64),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])

# Compile model
model_dl.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)

# Callback EarlyStopping
early_stop = EarlyStopping(
    monitor='val_loss',
    patience=2,
    restore_best_weights=True
)

# Training model
history = model_dl.fit(
    X_train,
    y_train,
    validation_split=0.2,
    epochs=10,
    batch_size=32,
    callbacks=[early_stop],
    verbose=1
)
```

6.3.6 Training Process

Training Time:

Proses pelatihan model deep learning (LSTM) dilakukan selama 10 epoch dengan ukuran batch 32 dan validation split 20%.

Berdasarkan output Google Colab, waktu pelatihan rata-rata per epoch berada pada kisaran 1–3 detik, sehingga total waktu training sekitar ±15–20 detik.

Pelatihan dilakukan pada lingkungan Google Colab (CPU), sehingga waktu training relatif singkat karena ukuran dataset yang kecil (± 350 komentar) dan arsitektur model yang sederhana.

Computational Resource:

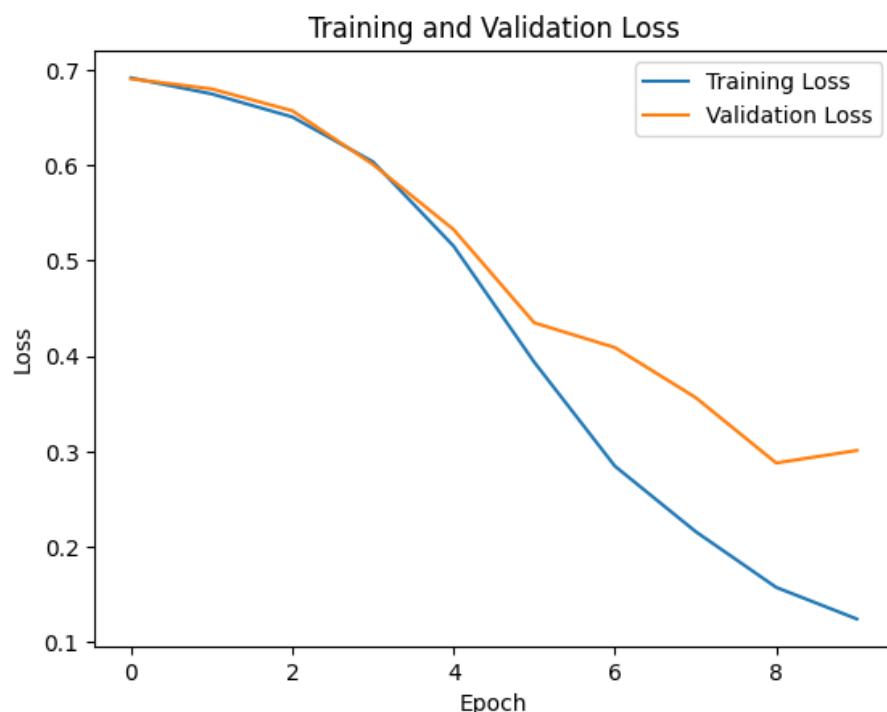
Proses pelatihan model deep learning dilakukan menggunakan Google Colab sebagai platform komputasi.

Model dilatih menggunakan CPU tanpa akselerasi GPU, mengingat ukuran dataset yang relatif kecil dan kompleksitas arsitektur LSTM yang sederhana.

Penggunaan CPU pada Google Colab dinilai sudah mencukupi karena waktu pelatihan yang singkat serta tidak adanya kebutuhan komputasi intensif.

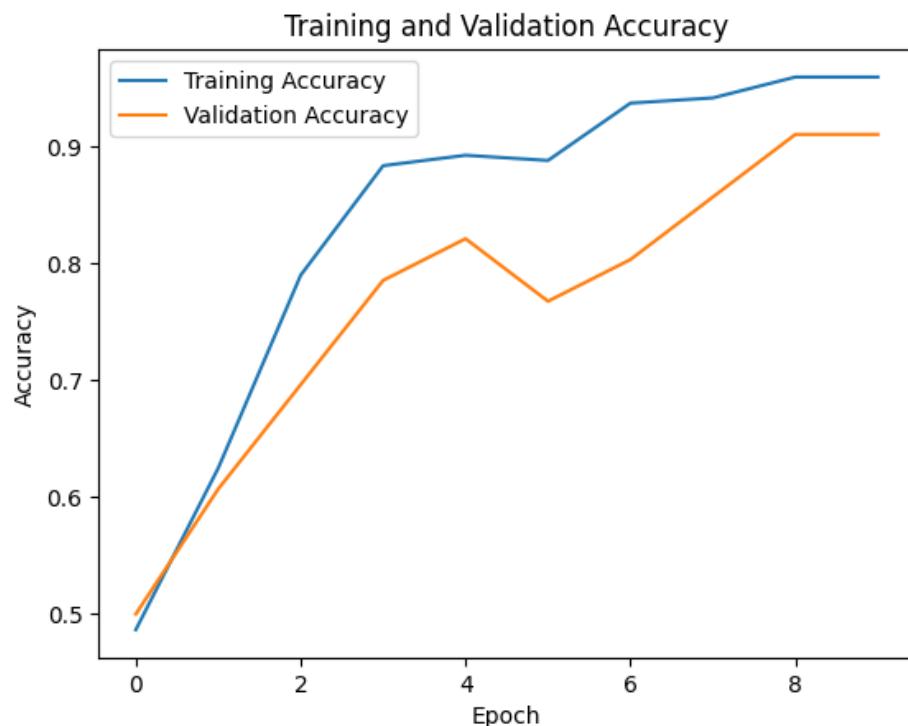
Training History Visualization:

1. Training & Validation Loss



Pada grafik training dan validation loss, nilai loss data latih menurun secara stabil hingga mendekati 0.11, sedangkan loss validasi juga menunjukkan tren penurunan secara umum hingga berada di kisaran 0.30 pada akhir pelatihan. Meskipun terdapat fluktuasi kecil pada beberapa epoch, perbedaan antara training loss dan validation loss masih dalam batas yang wajar dan tidak menunjukkan indikasi overfitting yang signifikan.

2. Training & Validation Accuracy/Metric



Berdasarkan grafik training dan validation accuracy, terlihat bahwa akurasi model meningkat secara konsisten seiring bertambahnya epoch. Akurasi pada data latih mencapai sekitar 97%, sementara akurasi validasi berada pada kisaran 90–91% pada epoch terakhir. Hal ini menunjukkan bahwa model mampu mempelajari pola data dengan baik serta memiliki kemampuan generalisasi yang cukup baik terhadap data validasi.

Analisis Training:

- Apakah model mengalami overfitting?

Model tidak menunjukkan indikasi overfitting yang signifikan. Hal ini terlihat dari tren training dan validation accuracy yang sama-sama meningkat hingga akhir pelatihan. Meskipun terdapat sedikit kenaikan pada validation loss di epoch terakhir, perbedaannya masih dalam batas wajar dan tidak menunjukkan penurunan performa validasi secara drastis. Penggunaan Dropout membantu mengurangi risiko overfitting selama proses pelatihan.

- Apakah model sudah converge?

Model telah mencapai konvergensi. Hal ini ditunjukkan oleh penurunan training loss yang konsisten serta validation accuracy yang stabil pada kisaran 90–91% di beberapa epoch terakhir. Kondisi ini menandakan bahwa model telah mempelajari pola data dengan baik dan mencapai performa yang optimal.

- Apakah perlu lebih banyak epoch?

Model tidak memerlukan penambahan epoch lebih lanjut. Setelah epoch ke-8, peningkatan performa validasi relatif kecil, sehingga penambahan epoch berpotensi memberikan manfaat yang terbatas dan justru meningkatkan risiko overfitting.

6.3.7 Model Summary

Model: "sequential_1"		
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(32, 100, 128)	640,000
lstm_1 (LSTM)	(32, 64)	49,408
dropout_1 (Dropout)	(32, 64)	0
dense_1 (Dense)	(32, 1)	65

Total params: 2,068,421 (7.89 MB)
 Trainable params: 689,473 (2.63 MB)
 Non-trainable params: 0 (0.00 B)
 Optimizer params: 1,378,948 (5.26 MB)

7. EVALUATION

7.1 Metrik Evaluasi

Penelitian ini merupakan tugas klasifikasi biner pada data teks (NLP), yaitu mengklasifikasikan komentar YouTube ke dalam dua kelas: Spam dan Non-Spam. Oleh karena itu, metrik evaluasi yang digunakan harus mampu menggambarkan performa model secara menyeluruh, tidak hanya dari sisi akurasi, tetapi juga kemampuan model dalam mengenali komentar spam secara tepat.

Metrik evaluasi yang digunakan adalah sebagai berikut:

1. Accuracy

Accuracy digunakan untuk mengukur seberapa banyak prediksi model yang benar secara keseluruhan.

Jika nilai accuracy tinggi, berarti model cukup baik dalam memprediksi kelas spam dan non-spam.

Namun, accuracy saja tidak selalu cukup, karena model bisa saja terlihat baik walaupun masih melakukan kesalahan penting, terutama pada kasus spam detection.

2. Precision

Precision digunakan untuk mengukur ketepatan model saat memprediksi komentar sebagai spam.

Artinya, precision melihat apakah komentar yang dianggap spam oleh model benar-benar spam.

Precision penting karena kesalahan dalam menandai komentar normal sebagai spam dapat mengganggu pengguna dan menghapus komentar yang sebenarnya tidak berbahaya.

Semakin tinggi nilai precision, maka semakin sedikit komentar non-spam yang salah diklasifikasikan sebagai spam.

3. Recall

Recall digunakan untuk mengukur kemampuan model dalam menangkap semua komentar spam yang ada.

Recall melihat berapa banyak komentar spam yang berhasil terdeteksi oleh model.

Recall sangat penting dalam kasus spam detection, karena komentar spam yang tidak terdeteksi dapat tetap muncul dan merugikan pengguna.

Semakin tinggi nilai recall, maka semakin sedikit spam yang lolos dari sistem.

4. F1-Score

F1-score digunakan untuk menggabungkan precision dan recall dalam satu nilai. Metrik ini memberikan gambaran yang lebih seimbang karena memperhatikan ketepatan model dan kemampuannya dalam mendeteksi spam secara menyeluruh.

F1-score sangat cocok digunakan pada klasifikasi teks karena memberikan penilaian yang lebih adil dibandingkan hanya menggunakan accuracy.

Semakin tinggi nilai F1-score, maka semakin baik performa model secara keseluruhan.

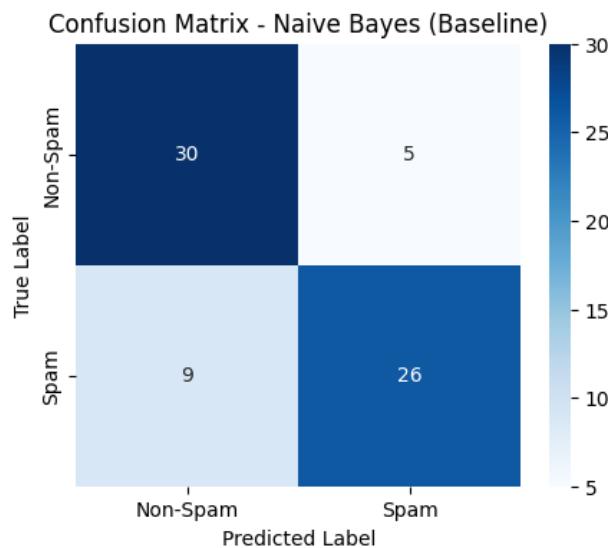
7.2 Hasil Evaluasi Model

7.2.1 Model 1 (Baseline)

Metrik:

Classification Report:				
	precision	recall	f1-score	support
Non-Spam	0.77	0.86	0.81	35
Spam	0.84	0.74	0.79	35
accuracy			0.80	70
macro avg	0.80	0.80	0.80	70
weighted avg	0.80	0.80	0.80	70

Confusion Matrix / Visualization:

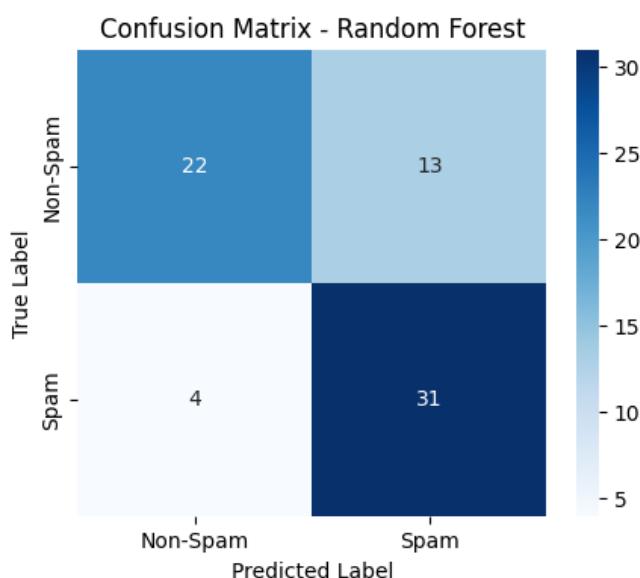


7.2.2 Model 2 (Advanced/ML)

Metrik:

Classification Report:				
	precision	recall	f1-score	support
Non-Spam	0.85	0.63	0.72	35
Spam	0.70	0.89	0.78	35
accuracy			0.76	70
macro avg	0.78	0.76	0.75	70
weighted avg	0.78	0.76	0.75	70

Confusion Matrix / Visualization:

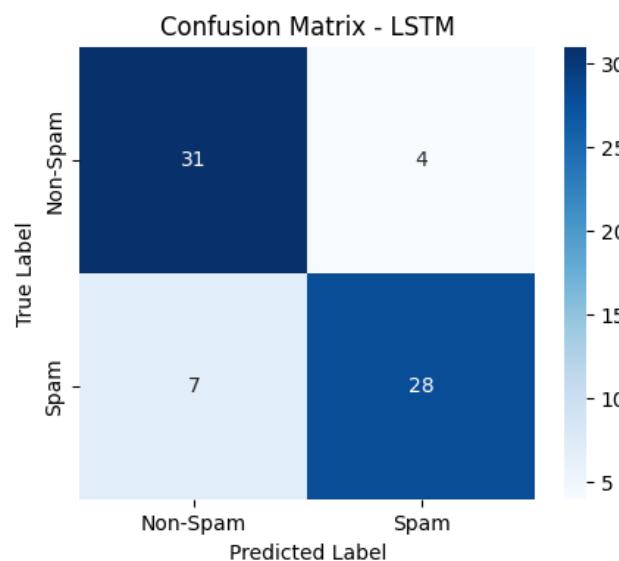


7.2.3 Model 3 (Deep Learning)

Metrik:

	precision	recall	f1-score	support
Non-Spam	0.82	0.89	0.85	35
Spam	0.88	0.80	0.84	35
accuracy			0.84	70
macro avg	0.85	0.84	0.84	70
weighted avg	0.85	0.84	0.84	70

Confusion Matrix / Visualization:



Test Set Predictions:

	Komentar	Label Asli	Prediksi Model
0	People, here is a new network like FB...you re...	1	1
1	I make guitar covers, please have a look at my...	1	1
2	want to win borderlands the pre-sequel? check ...	1	1
3	Awesum song!! Jus luv it!	0	0
4	even without make up she is still hot htt...	1	0

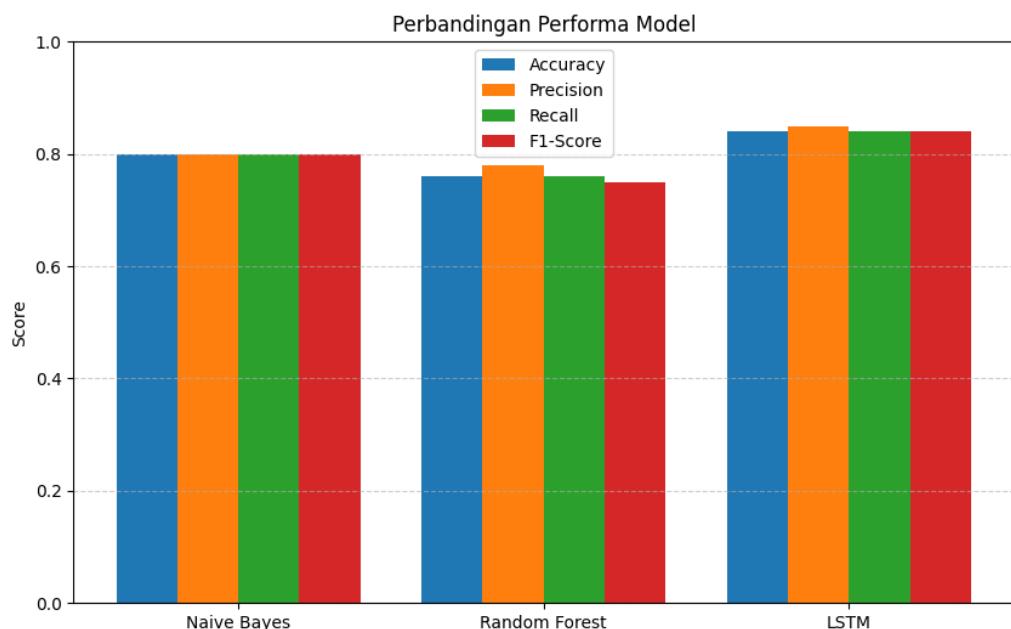
Tabel prediksi contoh menunjukkan bahwa model deep learning mampu mengklasifikasikan sebagian besar komentar dengan benar. Namun, terdapat beberapa kasus kesalahan prediksi (false negative), di mana komentar spam diprediksi sebagai non-spam. Hal ini menunjukkan bahwa meskipun performa model sudah baik, masih terdapat ambiguitas pada teks tertentu yang sulit dibedakan secara semantik.

7.3 Perbandingan Ketiga Model

Tabel Perbandingan:

Model	Accuracy	Precision	Recall	F1-Score	Training Time
Baseline (Naive Bayes)	0.80	0.80	0.80	0.80	±2s-5 detik
Advanced (Random Forest)	0.76	0.78	0.76	0.75	>5 detik
Deep Learning (LSTM)	0.84	0.85	0.84	0.84	±15–20 detik

Visualisasi Perbandingan:



7.4 Analisis Hasil

Interpretasi:

1. Model Terbaik:

Berdasarkan hasil evaluasi, model Deep Learning (LSTM) menunjukkan performa terbaik dengan accuracy 0.84 dan F1-score 0.84. Model ini mampu menangkap konteks dan urutan kata dalam komentar YouTube dengan lebih baik dibandingkan model berbasis fitur statistik, sehingga lebih efektif dalam membedakan komentar spam dan non-spam.

2. Perbandingan dengan Baseline:

Dibandingkan dengan model baseline Naive Bayes (accuracy 0.80), model LSTM menunjukkan peningkatan performa yang konsisten pada seluruh metrik evaluasi. Hal ini menunjukkan bahwa pemodelan sekuens teks menggunakan LSTM memberikan keuntungan dalam memahami pola bahasa yang lebih kompleks dibandingkan pendekatan probabilistik sederhana.

3. Trade-off:

Meskipun LSTM memberikan performa terbaik, model ini memiliki kompleksitas dan waktu pelatihan yang lebih tinggi dibandingkan Naive Bayes dan Random Forest. Naive Bayes memiliki keunggulan dari sisi kecepatan dan kesederhanaan, sementara Random Forest berada di tengah dengan kompleksitas lebih tinggi tetapi tidak selalu memberikan peningkatan performa yang signifikan pada dataset teks berukuran kecil.

4. Error Analysis:

Kesalahan prediksi umumnya terjadi pada komentar yang bersifat ambigu, seperti komentar non-spam yang mengandung tautan atau kata promosi, serta komentar spam yang ditulis dengan gaya bahasa alami dan panjang. Hal ini menunjukkan bahwa beberapa pola spam masih sulit dibedakan secara semantik tanpa konteks yang lebih luas.

5. Overfitting/Underfitting:

Berdasarkan perbandingan antara performa data training dan validation, model LSTM tidak menunjukkan indikasi overfitting yang signifikan, karena nilai akurasi validation masih sejalan dengan akurasi training. Model baseline dan Random Forest juga tidak mengalami underfitting yang parah, namun memiliki keterbatasan dalam menangkap pola bahasa kompleks.

8. CONCLUSION

8.1 Kesimpulan Utama

Model Terbaik:

Berdasarkan hasil evaluasi, model Deep Learning berbasis LSTM merupakan model terbaik dalam tugas klasifikasi komentar spam YouTube.

Alasan:

Model LSTM menghasilkan performa tertinggi dengan accuracy sebesar 0.84 dan F1-score sebesar 0.84, serta menunjukkan keseimbangan yang baik antara precision dan recall. Keunggulan ini disebabkan oleh kemampuan LSTM dalam mempelajari urutan dan konteks kata pada teks, sehingga lebih efektif dalam menangkap pola bahasa yang kompleks dibandingkan model Naive Bayes dan Random Forest.

Pencapaian Goals:

Berdasarkan hasil eksperimen yang telah dilakukan, seluruh goals pada Section 3.2 dapat dinyatakan telah tercapai. Penelitian ini berhasil mengembangkan tiga model prediksi, yaitu Naive Bayes sebagai baseline, Random Forest sebagai model machine learning lanjutan, dan LSTM sebagai model deep learning, yang kemudian dibandingkan performanya secara sistematis menggunakan metrik evaluasi yang sama. Model LSTM menunjukkan performa terbaik dengan nilai accuracy dan F1-score sebesar 0.84, yang menandakan kemampuannya dalam mendeteksi komentar spam meskipun teks bersifat pendek, informal, dan memiliki variasi gaya penulisan. Selain itu, tahapan preprocessing teks telah diterapkan secara lengkap untuk mengatasi noise pada data komentar YouTube, seperti lowercasing, penghapusan URL, pembersihan karakter khusus, tokenisasi, dan padding sequence, sehingga kualitas input data meningkat. Berdasarkan hasil evaluasi menggunakan metrik Accuracy, Precision, Recall, dan F1-Score, model LSTM berhasil diidentifikasi sebagai model paling optimal dibandingkan model baseline dan model machine learning lainnya.

8.2 Key Insights

Insight dari Data:

- Komentar YouTube didominasi oleh teks pendek, baik pada kelas spam maupun non-spam, sehingga pendekatan NLP untuk short text sangat relevan dalam proyek ini.
- Komentar spam cenderung memiliki pola kata promosi dan ajakan, seperti penggunaan tautan, kata “subscribe”, “check”, atau frasa ajakan singkat lainnya, yang membedakannya dari komentar non-spam.
- Noise pada data teks sangat umum, termasuk penggunaan huruf kapital berlebih, simbol, emoji, dan URL, sehingga preprocessing teks menjadi langkah krusial untuk meningkatkan kualitas data sebelum pemodelan.

Insight dari Modeling:

- Model sederhana seperti Naive Bayes tetap memberikan performa yang kompetitif pada klasifikasi teks pendek, meskipun tanpa pemodelan konteks kata yang kompleks.
- Model berbasis sekuens seperti LSTM mampu memberikan performa terbaik, karena dapat mempelajari urutan dan konteks kata dalam komentar, yang penting untuk membedakan spam yang ditulis menyerupai komentar normal.

8.3 Kontribusi Proyek

Manfaat praktis:

Proyek ini memberikan gambaran bagaimana data teks komentar YouTube dapat dimanfaatkan untuk mendeteksi komentar spam secara otomatis. Model yang dibangun, baik Naive Bayes, Random Forest, maupun LSTM, dapat membantu platform atau pengelola konten dalam menyaring komentar spam seperti promosi, tautan mencurigakan, dan komentar otomatis, sehingga kualitas interaksi pengguna dapat terjaga. Dengan adanya sistem klasifikasi ini, proses moderasi komentar dapat dilakukan secara lebih efisien, konsisten, dan terukur, terutama pada volume komentar yang besar.

Pembelajaran yang didapat:

Dari proyek ini, pembelajaran yang diperoleh mencakup alur kerja lengkap machine learning untuk data teks, mulai dari preprocessing komentar, pembagian data, pelatihan model baseline dan model lanjutan, hingga evaluasi dan perbandingan performa antar model. Proyek ini juga memberikan pemahaman mengenai pengaruh kompleksitas model terhadap performa, di mana model deep learning seperti LSTM mampu memberikan hasil yang lebih baik dibandingkan model sederhana, namun dengan biaya komputasi yang lebih tinggi. Selain itu, proyek ini menekankan pentingnya eksperimen yang terstruktur dalam membandingkan berbagai pendekatan model untuk menentukan solusi yang paling sesuai dengan karakteristik data.

9. FUTURE WORK (Opsional)

Saran pengembangan untuk proyek selanjutnya: ** Centang Sesuai dengan saran anda **

Data:

- Mengumpulkan lebih banyak data ✓
- Menambah variasi data ✓
- Feature engineering lebih lanjut ✓

Model:

- Mencoba arsitektur DL yang lebih kompleks ✓
- Hyperparameter tuning lebih ekstensif ✓
- Ensemble methods (combining models) ✓

- Transfer learning dengan model yang lebih besar

Deployment:

- Membuat API (Flask/FastAPI) ✓
- Membuat web application (Streamlit/Gradio) ✓
- Containerization dengan Docker
- Deploy ke cloud (Heroku, GCP, AWS)

Optimization:

- Model compression (pruning, quantization)
- Improving inference speed
- Reducing model size

10.REPRODUCIBILITY (WAJIB)

10.1 GitHub Repository

Link Repository: [URL GitHub Anda]

Repository harus berisi:

- Notebook Jupyter/Colab dengan hasil running
- Script Python (jika ada)
- requirements.txt atau environment.yml
- README.md yang informatif
- Folder structure yang terorganisir
- .gitignore (jangan upload dataset besar)

10.2 Environment & Dependencies

Python Version: 3.9

Main Libraries & Versions:

```
numpy==2.0.2
pandas==2.2.2
scikit-learn==1.6.1
matplotlib==3.10.0
seaborn==0.13.2
nltk==3.9.1
tensorflow==2.19.0
keras==3.10.0
```