

**EVALUASI PENGARUH *CHI-SQUARE FEATURE SELECTION*
DAN *OVERSAMPLING* LDW-SMOTE TERHADAP KINERJA
NAÏVE BAYES UNTUK DETEKSI EMAIL SPAM**

SKRIPSI

**OLEH
NURUS SYAFI'AH
NIM. 220601110057**



**PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026**

**EVALUASI PENGARUH *CHI-SQUARE FEATURE SELECTION*
DAN *OVERSAMPLING* LDW-SMOTE TERHADAP KINERJA
NAÏVE BAYES UNTUK DETEKSI EMAIL SPAM**

SKRIPSI

**Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)**

**Oleh
Nurus Syafi'ah
NIM. 220601110057**

**PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026**

**EVALUASI PENGARUH *CHI-SQUARE FEATURE SELECTION*
DAN *OVERSAMPLING* LDW-SMOTE TERHADAP KINERJA
NAÏVE BAYES UNTUK DETEKSI EMAIL SPAM**

SKRIPSI

**Oleh
Nurus Syafi'ah
NIM. 220601110057**

Telah Disetujui Untuk Diuji

Malang, 14 April 2026

Dosen Pembimbing I



Hisyam Fahmi, M.Kom.
NIP. 19890727 201903 1 018

Dosen Pembimbing II



Erna Herawati, M.Pd.
NIPPPK. 19760723 202321 2 006



Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrudin Rozi, M.Si.
NIP. 19800527 200801 1 012

EVALUASI PENGARUH *CHI-SQUARE FEATURE SELECTION* DAN *OVERSAMPLING LDW-SMOTE* TERHADAP KINERJA *NAÏVE BAYES* UNTUK DETEKSI EMAIL SPAM

SKRIPSI

Oleh
Nurus Syafi'ah
NIM. 220601110057

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat.)

Tanggal 12 Mei 2026

Ketua Penguji : Dr. Mohammad Jamhuri, M.Si.

Anggota Penguji 1 : Muhammad Khudzaifah, M.Si.

Anggota Penguji 2 : Hisyam Fahmi, M.Kom.

Anggota Penguji 3 : Erna Herawati, M.Pd.



Mengetahui,
Ketua Program Studi Matematika



Dr. Fachrudin Rozi, M.Si.
NIM. 19800527 200801 1 012

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : Nurus Syafi'ah
NIM : 220601110057
Program Studi : Matematika
Fakultas : Sains dan Teknologi
Judul Skripsi : Evaluasi Pengaruh *Chi-Square Feature Selection* dan
Oversampling LDW-SMOTE terhadap Kinerja *Naïve Bayes* untuk Deteksi Email Spam

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan dan pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut

Malang, 12 Mei 2026

Yang membuat pernyataan



Nurus Syafi'ah

NIM 220601110057

MOTO

“Jika Allah menanamkan keinginan di hatiku, kupercaya itu bukan untuk dibiarkan, tapi untuk ku temui nanti”

-Nurussyafi-

“Jika amanah sudah diberi, maka pantang untuk lari”

-UmmahNury-

“Perilakumu cerminan ilmumu”

-UmmahNury-

PERSEMBAHAN

Dengan segala kerendahan hati dan rasa syukur yang tak terhingga kepada Allah SWT. atas segala rahmat dan pertolongan-Nya, sehingga skripsi ini dapat terselesaikan dengan baik. Karya ini penulis persembahkan untuk:

Orang tua tercinta dan keluarga besarnya, yang doanya tak pernah putus bahkan saat penulis nyaris kehilangan arah, yang menjadi alasan untuk tetap melangkah hingga sampai di titik ini. Ummah dan Buya, yang tidak hanya menanamkan ilmu, tetapi juga menuntun penulis memahami arti arah dan makna dalam setiap perjalanan. Teman-teman seperjalanan, yang hadir di antara lelah dan ragu, yang menguatkan tanpa banyak kata, dan membuat langkah yang berat terasa lebih ringan. Serta untuk diri penulis sendiri, yang terus berjalan dalam diam, yang belajar memahami setiap proses, hingga akhirnya mampu sampai di titik ini. Terima kasih untuk setiap perjalanan yang tidak selalu mudah, karena dari sanalah penulis mengerti bahwa setiap langkah memiliki arti yang berharga.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Puji syukur penulis panjatkan ke hadirat Allah SWT. atas segala limpahan rahmat, taufik, dan hidayah-Nya sehingga skripsi yang berjudul “Evaluasi Pengaruh *Chi-Square Feature Selection* dan *Oversampling* LDW-SMOTE terhadap Kinerja *Naïve Bayes* untuk Deteksi Email Spam” ini dapat disusun sebagai langkah awal dalam menyelesaikan tugas akhir pada Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Shalawat serta salam semoga senantiasa tercurahkan kepada Nabi Muhammad SAW, yang telah membawa umat manusia dari zaman kegelapan menuju era yang penuh cahaya ilmu dan iman, yaitu *addinul Islam wal-iman*. Oleh karena itu, dengan penuh rasa hormat dan terima kasih, penulis menyampaikan apresiasi yang sebesar-besarnya kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Agus Mulyono, M.Kes, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Fachrur Rozi, M.Si., selaku Ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim Malang yang telah memberikan nasihat demi keberhasilan penelitian.
4. Hisyam Fahmi, M.Kom., selaku Dosen Pembimbing I yang telah memberikan bimbingan, arahan, serta motivasi dalam proses penyusunan proposal ini, sehingga penulis dapat menyelesaikannya dengan baik.
5. Erna Herawati, M.Pd. selaku Dosen Pembimbing II yang telah memberikan dukungan, masukan, serta arahan yang konstruktif dalam proses penyusunan proposal ini, sehingga penulis dapat menyusunnya dengan lebih terarah.
6. Dr. Mohammad Jamhuri, M.Si. selaku Ketua Penguji yang telah memberikan arahan, masukan, serta kritik yang membangun sehingga penulis dapat memperbaiki dan menyempurnakan skripsi ini.

7. Muhammad Khudzaifah, M.Si. selaku Anggota Penguji 1 yang telah meluangkan waktu, memberikan saran, serta masukan yang sangat berharga demi kesempurnaan skripsi ini.
8. Seluruh Dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
9. Orang tua tercinta dan seluruh keluarga besar yang senantiasa memberikan doa, kasih sayang, serta dukungan tiada henti dalam setiap langkah penulis.
10. Pengasuh PP Daruzzahra Arrifa'i, Buya Nadhif Anwar dan Ummah Nury Firdausia, yang telah memberikan bimbingan, nasihat, serta teladan dalam menanamkan nilai-nilai kedisiplinan, keikhlasan, dan semangat dalam menuntut ilmu sehingga menjadi bekal berharga bagi penulis dalam menyelesaikan penelitian ini.
11. Teman-teman dekat penulis, Ayu, Amira, Mutiara, Ochi, Isma, yang selalu memberikan dukungan, kebersamaan, serta menjadi bagian dari proses perjalanan penulis.
12. Teman-teman pondok di PP Daruzzahra Arrifa'i 1, yang telah menemani hari-hari dengan kebersamaan, saling menguatkan, dan menjadi tempat bertumbuh dalam proses belajar.
13. Teman-teman Komputasi Eva, Farah, Diah, Bintang, Thoriq, Fisal, Yusuf, yang telah bekerja sama, berbagi ilmu, serta saling membantu selama proses perkuliahan.
14. Seluruh mahasiswa angkatan 2022 Mathdeux.

Semoga Allah SWT. memberikan balasan yang terbaik atas segala bantuan dan kebaikan yang telah diberikan kepada penulis.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Malang, 16 Oktober 2025

Penulis

DAFTAR ISI

HALAMAN JUDUL.....	i
HALAMAN PENGAJUAN.....	ii
HALAMAN PERSETUJUAN.....	iii
HALAMAN PENGESAHAN.....	iv
PERNYATAAN KEASLIAN TULISAN	v
MOTO	vi
PERSEMBAHAN.....	vii
KATA PENGANTAR.....	viii
DAFTAR ISI.....	x
DAFTAR TABEL	xii
DAFTAR GAMBAR.....	xiii
DAFTAR SIMBOL	xiv
DAFTAR LAMPIRAN	xv
ABSTRAK	xvi
ABSTRACT	xvii
الملخص البحث.....	xviii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	5
1.3 Tujuan Penelitian	6
1.4 Manfaat Penelitian	6
1.5 Batasan Masalah	7
1.6 Definisi Istilah	8
BAB II KAJIAN TEORI	10
2.1 Teori Pendukung	10
2.1.1 <i>Machine learning</i>	10
2.1.2 Klasifikasi	11
2.1.3 <i>Text Mining</i>	12
2.1.4 <i>Term Frequency–Inverse Document Frequency (TF-IDF)</i> .	16
2.1.5 <i>Chi-Square Feature Selection</i>	18
2.1.6 <i>Synthetic Minority Oversampling Technique (SMOTE)</i>	20
2.1.7 <i>Local Density Weighting SMOTE (LDW-SMOTE)</i>	21
2.1.8 Algoritma <i>Naïve Bayes</i>	24
2.1.9 <i>Multinomial Naïve Bayes</i>	26
2.1.10 Evaluasi Kinerja Model	29
2.1.11 Spam Email.....	32
2.2 Kajian Integrasi Topik dengan Al-Quran/Hadits.....	32
2.3 Kajian Topik Dengan Teori Pendukung.....	35
BAB III METODE PENELITIAN	38
3.1 Jenis Penelitian	38
3.2 Data dan Sumber Data	38
3.3 Teknik Analisis Data	39
3.3.1 <i>Exploratory Data Analysis (EDA)</i>	41

3.3.2	<i>Preprocessing Teks</i>	41
3.3.3	<i>Split Data</i>	43
3.3.4	Ekstraksi Fitur TF-IDF	43
3.3.5	<i>Chi-Square Feature Selection</i>	44
3.3.6	LDW-SMOTE	44
3.3.7	Pelatihan Model <i>Multinomial Naïve Bayes</i>	45
3.3.8	Evaluasi Model	46
BAB IV	HASIL DAN PEMBAHASAN	47
4.1	Hasil.....	47
4.1.1	<i>Exploratory Data Analysis (EDA)</i>	47
4.1.2	<i>Preprocessing Teks</i>	50
4.1.3	<i>Split Data</i>	58
4.1.4	Ekstraksi Fitur TF-IDF	58
4.1.5	<i>Chi-Square Feature Selection</i>	68
4.1.6	LDW-SMOTE	70
4.1.7	Pelatihan Model <i>Multinomial Naïve Bayes</i>	75
4.1.8	Evaluasi Model	84
4.2	Pembahasan	97
4.3	Klasifikasi Email Spam dalam Perspektif Islam	101
BAB V	PENUTUP	105
5.1	Kesimpulan.....	105
5.2	Saran	106
DAFTAR PUSTAKA		108
LAMPIRAN		115
RIWAYAT HIDUP		132

DAFTAR TABEL

Tabel 2.1	<i>Confusion Matrix</i>	30
Tabel 3.1	Deskripsi Data.....	39
Tabel 4.1	Sampel Data Email.....	47
Tabel 4.2	Contoh Hasil <i>Lowercasing</i> dan <i>Cleansing</i>	50
Tabel 4.3	Contoh Hasil <i>Tokenization</i>	52
Tabel 4.4	Contoh Hasil <i>Stopwords Removal</i>	54
Tabel 4.5	Contoh Hasil <i>Lemmatization</i>	56
Tabel 4.6	Kalimat Sampel.....	59
Tabel 4.7	Frekuensi Kemunculan Kata pada Dokumen.....	65
Tabel 4.8	Hasil Perhitungan TF	66
Tabel 4.9	Hasil Perhitungan IDF	66
Tabel 4.10	Hasil Perhitungan TF-IDF	67
Tabel 4.11	Hasil Perhitungan <i>Chi-Square Feature Selection</i>	68
Tabel 4.12	Keputusan Uji <i>Feature Selection Chi-Square</i>	70
Tabel 4.13	Frekuensi Kemunculan Kata pada Dokumen.....	76
Tabel 4.14	Hasil <i>Cross Validation</i> MNB tanpa <i>Chi-Square</i>	81
Tabel 4.15	Hasil <i>Cross Validation</i> MNB dengan <i>Chi-Square</i>	82
Tabel 4.16	Hasil <i>Cross Validation</i> MNB dengan <i>Chi-Square</i> & LDW-SMOTE.....	83
Tabel 4.17	Hasil <i>Cross Validation</i> MNB dengan LDW-SMOTE	84
Tabel 4.18	Hasil Perbandingan Evaluasi dengan <i>Chi-Square</i>	93
Tabel 4.19	Hasil Perbandingan dengan LDW-SMOTE.....	94
Tabel 4.20	Hasil Perbandingan tanpa <i>Chi-Square</i>	95
Tabel 4.21	Hasil Perbandingan Model.....	100

DAFTAR GAMBAR

Gambar 3.1 <i>Flowchart</i> Analisis Data.....	40
Gambar 4.1 Distribusi Kelas	49
Gambar 4.2 Diagram Distribusi Kelas Setelah LDW-SMOTE	74
Gambar 4.3 <i>Confusion Matrix</i> tanpa <i>Chi-Square</i>	85
Gambar 4.4 <i>Confusion Matrix</i> dengan <i>Chi-Square</i> tanpa LDW-SMOTE.....	86
Gambar 4.5 <i>Confusion Matrix</i> dengan <i>Chi-Square</i> dan LDW-SMOTE.....	88
Gambar 4.6 <i>Confusion Matrix</i> dengan LDW-SMOTE	90

DAFTAR SIMBOL

$\chi^2(t.C)$: Nilai *Chi-Square* antara fitur (kata) t dan kelas C .

Df : Derajat kebebasan

x_i : Sampel minortas asli.

x_j : Tetangga terdekat dari x_i .

$P(d)$: *Evidence*, yang mengacu pada total data d tanpa memperhitungkan kelas yang terkait.

$P(C)$: *Prior probability* dari hipotesis C , yakni peluang awal tanpa memperhitungkan data d .

N_t : Jumlah total fitur di seluruh dokumen dalam dataset.

D_C : Jumlah dokumen yang termasuk kelas C .

D : Jumlah seluruh dokumen.

DAFTAR LAMPIRAN

Lampiran 1 Dataset Penelitian	115
Lampiran 2 Hasil <i>Preprocessing</i>	115
Lampiran 3 Hasil Evaluasi Model <i>Chi-Square</i>	115
Lampiran 4 Hasil Evaluasi Model tanpa <i>Chi-Square</i>	115
Lampiran 5 Hasil Evaluasi Model <i>Chi-Square</i> dan LDW-SMOTE	116
Lampiran 6 Hasil Evaluasi Model LDW-SMOTE.....	116
Lampiran 7 Hasil Perbandingan dengan Model lain.....	116
Lampiran 8 <i>Source Code</i> Klasifikasi Email Spam.....	117
Lampiran 9 Tabel <i>Chi-Square</i>	131

ABSTRAK

Syafi'ah, Nurus. 2026. **Evaluasi Pengaruh *Chi-Square Feature Selection* dan *Oversampling* LDW-SMOTE terhadap Kinerja *Naive Bayes* untuk Deteksi Email Spam.** Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Hisyam Fahmi, M.Kom. (II) Erna Herawati, M.Pd.

Kata Kunci: *Multinomial Naive Bayes*, *Chi-Square*, LDW-SMOTE, Klasifikasi, Email Spam

Peningkatan penggunaan email menyebabkan bertambahnya jumlah spam yang mengganggu dan berisiko bagi pengguna. Penelitian ini bertujuan menganalisis kinerja *Multinomial Naive Bayes* dengan *Chi-Square Feature Selection* dan LDW-SMOTE dalam klasifikasi email spam. Data berbahasa Inggris diproses menggunakan TF-IDF dan direduksi menggunakan *Chi-Square*. Hasil menunjukkan bahwa *Chi-Square* meningkatkan akurasi dan presisi, namun menurunkan *recall* spam. Sebaliknya, LDW-SMOTE meningkatkan *recall* spam secara signifikan, tetapi menurunkan akurasi dan presisi. Hal ini menunjukkan adanya perbedaan fokus antara ketepatan model dan kemampuan deteksi spam. Pemilihan metode perlu disesuaikan dengan tujuan sistem. Penelitian selanjutnya disarankan mengoptimalkan jumlah fitur *Chi-Square*, parameter LDW-SMOTE, serta membandingkan metode penyeimbangan data lain.

ABSTRACT

Syafi'ah, Nurus. 2026. **Evaluation of the Influence of Chi-Square Feature Selection and Oversampling of LDW-SMOTE on Naive Bayes Performance for Spam Email Detection.** Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim, Malang. Advisors: (I) Hisyam Fahmi, M.Kom. (II) Erna Herawati, M.Pd.

Keywords: Multinomial Naïve Bayes, Chi-Square, LDW-SMOTE, Classification, Email Spam

The increased use of email leads to an increase in the amount of spam that is annoying and risky for users. This study aims to analyze the performance of Multinomial Naïve Bayes with Chi-Square Feature Selection and LDW-SMOTE in the classification of spam emails. English-language data was processed using TF-IDF with and reduced using Chi-Square. The results showed that Chi-Square improved accuracy and precision, but decreased spam recalls. In contrast, LDW-SMOTE significantly improved spam recalls, but decreased accuracy and precision. This shows a difference in focus between model accuracy and spam detection capabilities. The selection of methods needs to be adjusted to the objectives of the system. Further research is recommended to optimize the number of Chi-Square features, LDW-SMOTE parameters, and compare other data balancing methods.

مستخلص البحث

الشافعة، نور. ٢٠٢٦. **تقويم تأثير *Chi-Square Feature Selection* و *Oversampling* من *LDW-SMOTE* على أداء *Naïve Bayes* في اكتشاف البريد الإلكتروني المزعج**. بحث الجامعي. قسم الرياضيات، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية، مالانج. المشرف: (الأول) هسام فهمي، الماجستير في العلوم الحاسوب (الثاني) إرنا هيراواتي، الماجستير في التربية.

الكلمات الأساسية: *LDW-SMOTE*، *Chi-Square*، *Multinomial Naïve Bayes*، التصنيف، البريد الإلكتروني المزعج

يؤدي الاستخدام المتزايد للبريد الإلكتروني إلى زيادة كمية الرسائل المزعجة والمخوفة بالمخاطر للمستخدمين. هدفت هذه الدراسة إلى تحليل أداء *Multinomial Naïve Bayes* باستخدام ميزة اختيار ميزات *Chi-Square* و *LDW-SMOTE* في تصنيف رسائل البريد الإلكتروني المزعج. تمت معالجة البيانات باللغة الإنجليزية باستخدام *TF-IDF* مع وتقليل استخدام *Chi-Square*. أظهرت النتائج أن *Chi-Square* حسن الدقة والدقة، لكنه قلل من استدعاءات الرسائل المزعجة. على النقيض من ذلك، حسن *LDW-SMOTE* بشكل ملحوظ عمليات استدعاء الرسائل المزعجة، لكنه قلل من الدقة والدقة. ظهر هذا فرق في التركيز بين دقة النموذج وقدرات اكتشاف الرسائل المزعجة. يجب تعديل اختيار الطرق وفقا لأهداف النظام. يوصى بإجراء الدراسة المستقبلية بتحسين عدد الخصائص في *Chi-Square*، وضبط معاملات *LDW-SMOTE*، إضافة إلى مقارنة طرائق أخرى لموازنة البيانات.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi telah mendorong pemanfaatan *machine learning* secara luas untuk menyelesaikan berbagai permasalahan berbasis data, termasuk pada pengolahan teks. Melalui pendekatan *machine learning*, sistem komputer dapat mempelajari pola dari data tanpa perlu diprogram secara eksplisit (Sulianta, 2025). Salah satu bidang penerapan dari *machine learning* adalah *text classification*, yang berfokus pada pengelompokkan dokumen teks dalam jumlah besar ke dalam kategori tertentu agar lebih mudah dianalisis dan dimanfaatkan (Pane & Caroline, 2024). Dalam bidang ini, sejumlah algoritma telah dikembangkan, namun penelitian terkini menunjukkan bahwa algoritma *Naïve Bayes* tetap konsisten memberikan kinerja yang kompetitif untuk klasifikasi teks.

Algoritma *Naïve Bayes* bekerja dengan memanfaatkan prinsip Teorema Bayes yang secara matematis menjelaskan hubungan antara probabilitas bersyarat suatu kelas dengan fitur-fitur yang diamati. Keunggulan algoritma ini terletak pada kesederhanaannya, efisiensi komputasi, serta kemampuannya menangani data berdimensi tinggi seperti teks (Kumalasari dkk., 2025). Penelitian yang dilakukan oleh (Drogkoula dkk., 2025) menunjukkan bahwa *Naïve Bayes* mampu memberikan hasil klasifikasi teks yang kompetitif dibandingkan algoritma lain seperti *Support Vector Machine* (SVM) dan *Random Forest* (RF). Dalam penelitian tersebut, *Naïve Bayes* digunakan untuk mengklasifikasikan 6.000 artikel akademik berdasarkan kata kunci dan abstrak menggunakan pembobotan TF-IDF dan

mencapai akurasi sebesar 90,16% lebih tinggi dari SVM (86,54%) dan RF (84,61%). Dalam hal ini, terbukti bahwa *Naïve Bayes* masih menjadi metode yang andal dalam klasifikasi teks.

Selain algoritma klasifikasi, performa model juga dipengaruhi oleh jumlah fitur yang dihasilkan dari representasi data teks. Proses representasi ini umumnya menghasilkan jumlah fitur yang sangat besar, namun tidak semuanya memiliki kontribusi yang signifikan. Beberapa fitur bahkan bersifat redundan dan dapat menurunkan performa model (Suharmin dkk., 2025). Oleh karena itu, diperlukan metode seleksi fitur yang dapat mengurangi dimensi data tanpa kehilangan informasi penting.

Dalam penelitiannya (Barbieri dkk., 2024) menegaskan bahwa metode seleksi fitur seperti *Chi-Square* tidak hanya membantu mereduksi dimensi dan redundansi, tetapi juga meningkatkan stabilitas dan keandalan performa model ketika dihadapkan pada data yang berdimensi tinggi. Hal ini sejalan dengan penelitian (Jusia dkk., 2025) yang mengombinasikan algoritma *Naïve Bayes* dengan seleksi fitur *Chi-Square* dalam analisis sentimen. Kombinasi tersebut terbukti mampu meningkatkan akurasi model dari 80,66% menjadi 86,22%. Penelitian lain oleh (Sari & Fahmi, 2021) menunjukkan bahwa penerapan *Chi-Square Feature Selection* dan *Naïve Bayes* dapat meningkatkan akurasi klasifikasi sebesar 11% pada dataset berukuran besar. Temuan ini menunjukkan bahwa penerapan metode seleksi fitur *Chi-Square* yang dikombinasikan dengan algoritma *Naïve Bayes* dapat memberikan peningkatan kinerja yang signifikan pada model klasifikasi.

Selain reduksi dimensi, distribusi data juga memiliki peranan penting dalam menentukan performa model klasifikasi. Dalam dataset teks, seringkali terjadi

ketidakseimbangan antara kelas mayoritas dan kelas minoritas, yang dapat menyebabkan model bias terhadap kelas dengan jumlah data lebih banyak. Kondisi ini dapat menurunkan kemampuan model dalam mengenali pola dari kelas minoritas. Salah satu pendekatan statistik untuk mengatasi masalah tersebut adalah *Synthetic Minority Oversampling Technique* (SMOTE), yang bekerja dengan menghasilkan sampel sintesis baru pada kelas minoritas berdasarkan jarak antara titik data (Paramita dkk., 2025).

Penelitian terbaru menunjukkan bahwa penerapan SMOTE dapat memberikan perbaikan yang signifikan dalam menangani ketidakseimbangan kelas. Misalnya penelitian yang dilakukan (Sartika dkk., 2025) pada klasifikasi status gizi balita dengan menggunakan *Naïve Bayes* membuktikan bahwa sebelum penerapan SMOTE nilai *recall* model hanya mencapai 65%, sedangkan setelah penerapan SMOTE meningkat menjadi 76,80%. Hasil ini membuktikan bahwa teknik *oversampling* seperti SMOTE efektif dalam menangani ketidakseimbangan kelas, sehingga membantu model meningkatkan kemampuan dalam memprediksi kelas minoritas secara lebih akurat.

Meskipun SMOTE terbukti efektif dalam mengatasi ketidakseimbangan kelas, SMOTE juga memiliki kelemahan karena menghasilkan sampel sintesis secara seragam tanpa mempertimbangkan variasi kepadatan lokal pada data minoritas. Kondisi tersebut dapat menyebabkan terbentuknya sampel yang kurang representatif. Untuk mengatasi hal tersebut, penelitian ini memberikan kontribusi matematis melalui pengembangan metode *Local Density Weighting SMOTE* (LDW-SMOTE), yang merupakan modifikasi dari SMOTE klasik. Dengan

pendekatan ini, LDW-SMOTE diharapkan mampu meningkatkan kualitas representasi data minoritas dan performa model dalam klasifikasi email spam.

Klasifikasi email spam dengan *machine learning* menjadi salah satu implementasi penting karena tingginya volume email dan dominasi spam secara global, sehingga metode manual sudah tidak relevan untuk diterapkan pada era komunikasi digital saat ini. Email spam juga telah menjadi ancaman serius khususnya dalam komunikasi digital sehari-hari. Banyak spam yang mengandung konten berbahaya atau menyesatkan terutama ketika mengandung konten *phishing*. Banyak pengguna internet kesulitan dalam memilah email penting karena tingginya volume spam yang diterima setiap harinya.

Berdasarkan laporan Lanskap Keamanan Siber Indonesia 2024 yang diterbitkan oleh Badan Siber dan Sandi Negara (BSSN), tercatat sebanyak 26.771.610 aktivitas *phishing* terjadi di Indonesia sepanjang tahun 2024. Angka tersebut menunjukkan bahwa *phishing* masih menjadi salah satu bentuk serangan siber paling dominan di Indonesia. Aktivitas tertinggi terjadi pada bulan Desember dengan total 6.191.413 kasus, sedangkan aktivitas terendah tercatat pada bulan Oktober sebanyak 1.135.012 kasus (Badan Siber dan Sandi Negara, 2025). Kondisi ini menegaskan bahwa ancaman *phishing* melalui email maupun media digital lainnya masih menjadi tantangan serius dalam menjaga keamanan siber nasional.

Dalam perspektif Islam, Islam mengajarkan pentingnya menjaga komunikasi yang jujur dan bermanfaat. Banyak email spam yang berisi kebohongan atau informasi yang tidak dapat dipertanggungjawabkan, yang bertentangan dengan etika komunikasi Islam. Dalam Al-Qur'an, QS. Al-Hujurat ayat 6 yang berbunyi:

يَا أَيُّهَا الَّذِينَ آمَنُوا إِنْ جَاءَكُمْ فَاسِقٌ بِنَبَأٍ فَتَبَيَّنُوا أَنْ تُصِيبُوا قَوْمًا بِجَهَالَةٍ فَتُصْحَبُوا عَلَىٰ مَا فَعَلْتُمْ
نُدِمِينَ ﴿٦٠﴾

“Wahai orang-orang yang beriman, jika seorang yang fasik datang kepadamu membawa suatu berita, maka telitilah kebenarannya, agar kamu tidak mencelakakan suatu kaum karena kebodohan (kecerobohan) yang akhirnya kamu menyesali perbuatan itu” (Kementrian Agama, 2022).

Ayat ini menegaskan bahwa informasi yang diterima harus diverifikasi terlebih dahulu sebelum dipercaya dan disebarkan. Oleh karena itu, penerapan teknologi seperti klasifikasi email spam dapat menjadi salah satu cara untuk menyeleksi informasi yang masuk agar tetap sesuai dengan prinsip kebenaran yang diajarkan oleh Islam.

Berdasarkan uraian tersebut, penelitian ini bertujuan untuk mengevaluasi pengaruh metode *Chi-Square* sebagai seleksi fitur dan LDW-SMOTE untuk mengatasi ketidakseimbangan kelas terhadap kinerja algoritma *Naïve Bayes* dalam klasifikasi email spam pada dataset Enron. Beberapa penelitian sebelumnya menunjukkan bahwa masing-masing metode memiliki keunggulan dalam meningkatkan performa klasifikasi. Namun kombinasi ketiganya belum banyak dieksplorasi secara mendalam. Berdasarkan landasan tersebut, hipotesis penelitian ini, penerapan kombinasi *Chi-Square Feature Selection* dan LDW-SMOTE menghasilkan kinerja algoritma *Naïve Bayes* yang baik dalam klasifikasi email spam pada dataset Enron Email.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana performa algoritma *Naïve Bayes* dalam melakukan klasifikasi email spam pada dataset Enron Email dengan metode *Chi-Square Feature Selection*?
2. Bagaimana pengaruh penerapan *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) terhadap penanganan ketidakseimbangan kelas menggunakan algoritma *Naïve Bayes* dalam klasifikasi email spam pada dataset Enron Email?

1.3 Tujuan Penelitian

Berdasarkan Rumusan Masalah, maka tujuan penelitian ini adalah sebagai berikut:

1. Untuk Mengevaluasi performa algoritma *Naïve Bayes* dalam melakukan klasifikasi email spam pada dataset Enron email dengan metode *Chi-Square Feature Selection* melalui pengukuran akurasi, presisi, *recall*, dan *f1-score*.
2. Untuk Menganalisis pengaruh penerapan *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) terhadap penanganan ketidakseimbangan kelas menggunakan algoritma *Naïve Bayes* dalam klasifikasi email spam pada dataset Enron Email.

1.4 Manfaat Penelitian

A. Manfaat Teoritis

1. Menambah pengetahuan dan wawasan keilmuan mengenai penerapan algoritma *Naïve Bayes* dalam klasifikasi email spam pada dataset tidak seimbang.
2. Memberikan kontribusi matematis pada pengembangan metode LDW-SMOTE, khususnya dalam perhitungan kepadatan lokal, bobot

oversampling, dan interpolasi berbobot sebagai dasar penelitian lanjutan terkait ketidakseimbangan kelas.

B. Manfaat Praktis

1. Membantu sistem filter email dalam mengklasifikasikan email spam dan non-spam secara otomatis berdasarkan pola kata pada email, terutama pada kondisi data yang tidak seimbang.
2. Menjadi referensi bagi peneliti dan praktisi dalam penerapan teknik *oversampling* dan seleksi fitur pada klasifikasi email spam.

1.5 Batasan Masalah

Batasan masalah dalam penelitian ini sebagai berikut:

1. Penelitian ini difokuskan pada pemanfaatan atribut *subject*, *body*, *from*, dan label dari dataset Enron Email yang didasarkan pada pertimbangan bahwa informasi yang terkandung di dalamnya memiliki relevansi yang signifikan terhadap proses klasifikasi email spam, sedangkan atribut lain diabaikan karena dianggap kurang berkontribusi secara langsung terhadap tujuan penelitian.
2. Penelitian ini hanya berfokus pada deteksi email spam yang bersifat merugikan atau mengganggu, khususnya spam yang mengandung indikasi penipuan (*phishing*), promosi yang tidak diinginkan, atau konten berbahaya lainnya. Email yang secara teknis terklasifikasi sebagai “spam” namun disukai atau diinginkan oleh penerima tidak termasuk dalam ruang lingkup penelitian ini.
3. Seluruh tahapan *preprocessing* pada penelitian ini dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan *library* yang tersedia, sehingga proses pembersihan teks, normalisasi, dan pembentukan data siap olah bergantung pada fungsi dan konfigurasi *library* yang digunakan.

4. Pada penelitian ini, jumlah fitur dibatasi untuk efisiensi waktu komputasi dan penyimpanan. Pada tahap TF-IDF digunakan 2000 fitur, kemudian pada tahap Chi-Square direduksi menjadi 1000 fitur yang paling relevan.
5. Pada penelitian ini, dataset yang digunakan berupa teks berbahasa Inggris, sehingga proses *preprocessing* dan ekstraksi fitur disesuaikan dengan karakteristik bahasa tersebut.

1.6 Definisi Istilah

Terdapat beberapa istilah yang digunakan dalam penelitian ini, yakni sebagai berikut:

1. *Phishing* : Salah satu bentuk serangan di mana pelaku menipu korban dengan menyamar sebagai pihak terpercaya guna mendapatkan informasi pribadi seperti *password* atau data penting lainnya (Febriansyah, 2025).
2. Algoritma : Serangkaian langkah sistematis dan logis yang dirancang untuk menyelesaikan suatu masalah atau menjalankan sebuah proses, termasuk dalam hal *machine learning* (Supriyono dkk., 2025).
3. Seleksi Fitur : Proses pemilihan sejumlah fitur paling relevan dari sekumpulan fitur yang tersedia untuk digunakan dalam pembangunan model *machine learning* (Rahayu dkk., 2024).
4. Redundansi : Kondisi di mana terdapat fitur-fitur yang memberikan informasi serupa atau berulang dalam suatu dataset (Olson, 2008).

5. *Oversampling* : Teknik penyeimbangan data yang dilakukan dengan menambah jumlah sampel pada kelas minoritas (Mola dkk., 2025)
6. *Dataset* : Kumpulan data yang digunakan untuk melatih dan menguji model *machine learning* (Maulani dkk., 2025).
7. *Fitur* : Atribut atau variabel dalam dataset yang digunakan sebagai *input* dalam model (Rianto & Santosa, 2025).
8. *Klasifikasi* : Proses memetakan *input* ke dalam kategori atau kelas yang telah ditentukan (Rustiyana dkk., 2025).
9. *Regresi* : Metode *machine learning* untuk memprediksi nilai kontinu berdasarkan data *input* (Sulianta, 2025).
10. *Malware* : Perangkat lunak jahat yang dirancang untuk merusak sistem atau mencuri data secara ilegal (Sulianta, 2025).
11. *Training* : Proses melatih model dengan data agar dapat mengenali pola atau hubungan dalam dataset (Sari dkk., 2025).
12. *Testing* : Proses mengevaluasi model menggunakan data baru untuk mengukur kinerja dan generalisasi model (Al-Husaini dkk., 2024).

BAB II

KAJIAN TEORI

2.1 Teori Pendukung

2.1.1 *Machine learning*

Machine learning (ML) merupakan salah satu bagian dari kecerdasan buatan (*Artificial Intelligence*) yang memungkinkan sistem komputer belajar dari data dan meningkatkan kinerjanya tanpa harus diprogram secara eksplisit (Sulianta, 2025). *Machine learning* bekerja dengan mengenali pola dan hubungan dari kumpulan data yang telah diberi label, sehingga dapat melakukan prediksi terhadap data baru yang belum pernah dilihat sebelumnya (Maulani dkk., 2025). Dalam konteks penerapan matematikanya, *machine learning* memanfaatkan konsep optimasi, statistik, dan probabilitas dalam setiap proses pelatihannya (Rajendra dkk., 2024). Oleh karena itu, *machine learning* menjadi pendekatan penting untuk menyelesaikan berbagai permasalahan klasifikasi email spam.

Secara umum, *machine learning* terbagi menjadi tiga jenis utama, yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. *Supervised learning*, menggunakan data yang dilabeli untuk mempelajari hubungan antara *input* dan *output*, seperti pada kasus klasifikasi dan regresi. Sebaliknya, *unsupervised learning* digunakan ketika data tidak memiliki label sehingga model harus menemukan pola tersembunyi melalui teknik seperti *clustering*. Sementara itu, *reinforcement learning* merupakan jenis *machine learning* yang melibatkan agen yang berinteraksi dengan lingkungan dan belajar

dari umpan balik berupa penghargaan atau hukuman terhadap tindakan yang diambil (Maulani dkk., 2025).

Dalam implementasinya, *machine learning* memiliki berbagai algoritma yang dapat dipilih sesuai karakteristik data dan tujuan analisis. Setiap algoritma memiliki keunggulan dan keterbatasan tersendiri dalam hal kompleksitas, kecepatan pelatihan, serta akurasi hasil prediksi (Menarianti dkk., 2025). Salah satu algoritmanya adalah algoritma *Naïve Bayes* yang dikenal efisien untuk klasifikasi teks karena mampu bekerja dengan baik pada data berdimensi tinggi dan mengasumsikan independensi antar fitur (Slam dkk., 2025). Selain itu algoritma lain seperti *Support Vector Machine (SVM)*, *Random Forest*, *Decision Tree*, *Regresi Logistic* juga sering digunakan sebagai pembanding dalam evaluasi performa model (Fadillah dkk., 2025; Rosyidi & Rochmawati, 2023).

Dalam konteks penyaringan spam email, pendekatan yang umum digunakan adalah *supervised learning*. Pada pendekatan ini, data pelatihan telah diberi label yang menunjukkan kategori “spam” dan “non-spam”. Model kemudian mempelajari pola dari data berlabel tersebut untuk mengenali ciri-ciri teks yang menjadi pembeda utama antar kategori. Dengan demikian, penerapan *machine learning* berperan penting dalam meningkatkan keamanan dan efisiensi sistem komunikasi digital.

2.1.2 Klasifikasi

Klasifikasi merupakan bagian penting dalam *machine learning* yang berfungsi memetakan data masukan ke dalam kategori tertentu berdasarkan pola yang dipelajari dari data berlabel sebelumnya (Sulianta, 2025). Tujuan utama klasifikasi adalah menghasilkan model yang mampu mengenali pola pada saat

pelatihan dan menggunakannya untuk memprediksi kelas data baru secara akurat. Klasifikasi termasuk *supervised learning*, karena dilakukan dengan menggunakan data yang sudah memiliki label kelas. Model belajar dari pasangan data yang berisi masukan dan label untuk mempelajari pola hubungan antara keduanya. Model kemudian dievaluasi berdasarkan seberapa baik ia mampu memprediksi label data yang belum pernah dilihat sebelumnya. Pendekatan ini dianggap efektif karena menggunakan pengalaman masa lalu (data latih) untuk membuat keputusan yang lebih akurat terhadap data baru (Swastika dkk., 2023).

Pada konteks klasifikasi teks, model digunakan untuk mengelompokkan dokumen ke dalam kategori tertentu berdasarkan isi dan konteksnya. Klasifikasi teks menjadi teknik penting dalam berbagai aplikasi seperti analisis sentimen, sistem rekomendasi, dan deteksi spam. Proses ini biasanya diawali dengan *preprocessing* data dan transformasi teks menjadi representasi numerik agar dapat diolah oleh algoritma *machine learning* (Mola dkk., 2025). Dalam penelitian ini, klasifikasi digunakan untuk membedakan antara email spam dan non-spam sebagai bentuk penerapan *supervised learning*. Model dilatih pada data yang telah dilabeli, di mana setiap email diberi tanda sebagai spam atau non-spam berdasarkan isinya. Pendekatan ini memiliki relevansi langsung dengan penelitian ini, karena menjadi dasar bagi implementasi algoritma *Naïve Bayes* dalam membedakan spam dan non-spam.

2.1.3 *Text Mining*

Text mining merupakan salah satu teknik dalam *data mining* yang berfokus pada proses mengekstraksi informasi bermakna dari kumpulan data teks yang tidak terstruktur. *Text mining* berperan dalam mengubah teks mentah menjadi

informasi yang terstruktur dan dapat diinterpretasikan secara kuantitatif (Mola dkk., 2025). Data teks seperti email, ulasan produk, atau dokumen digital sering kali mengandung pola tersembunyi yang dapat diungkap melalui proses analisis berbasis teks. Dalam konteks ini, email spam merupakan salah satu bentuk penerapan *text mining* karena melibatkan analisis isi pesan untuk mengklasifikasikan teks ke dalam kategori spam dan non-spam secara sistematis.

Secara umum, proses *text mining* terdiri atas beberapa tahapan penting, yaitu pengumpulan data, *preprocessing*, dan ekstraksi fitur (Faisal dkk., 2023). Tahap *preprocessing* memiliki peran yang sangat penting dalam *text mining*, sebab data teks mentah sering kali mengandung berbagai gangguan seperti *noise*, duplikasi, serta ketidakkonsistenan format. Proses ini diperlukan untuk memastikan bahwa data yang dianalisis telah bersih, konsisten, dan siap digunakan dalam tahap analisis lanjutan. Penelitian terbaru oleh (Nursahid dkk., 2025) menegaskan bahwa strategi *preprocessing* yang tepat berkontribusi signifikan terhadap peningkatan performa model. Dengan demikian, *preprocessing* menjadi landasan utama yang menjembatani teks mentah menuju representasi numerik yang dapat dianalisis secara matematis.

Tahapan *preprocessing* merupakan fondasi penting dalam pengolahan teks yang bertujuan mengubah data mentah menjadi format yang siap untuk dianalisis secara matematis. Langkah-langkah utama dalam *text preprocessing* mencakup *case folding*, *tokenizing*, *cleansing*, normalisasi, *stopword removal*, dan *stemming* atau *lemmatization* (Fajar & Maulina, 2024). Pertama, *case folding* berfungsi menyeragamkan seluruh huruf menjadi huruf kecil (*lowercase*) untuk menghindari perbedaan arti antara kata yang sama tetapi berbeda bentuk huruf,

misalnya “Email” dan “email” (Dariato dkk., 2024). Kedua, *tokenizing* memecah teks menjadi satuan kata atau token agar setiap elemen teks dapat dihitung dan dianalisis secara individual (Asri dkk., 2025). Langkah ini merupakan dasar pembentukan fitur dalam representasi teks berbasis *bag-of-words* maupun TF-IDF.

Langkah selanjutnya, *cleansing* dilakukan untuk menghapus karakter yang tidak relevan seperti angka, tanda baca, simbol, tautan, atau HTML tag yang sering muncul dalam data teks daring (Zaman & Purwanti, 2025). Selanjutnya, normalisasi teks dilakukan untuk menyamakan bentuk kata yang tidak baku, salah ejaan, atau bentuk tidak standar agar data teks lebih konsisten dan mudah dianalisis, seperti “u” menjadi bentuk “you” (Rizkia dkk., 2025). Tahap berikutnya, *stopword removal* menghapus kata umum yang tidak memiliki makna penting dalam konteks klasifikasi, misalnya kata sambung atau kata ganti seperti “and”, “is”, atau “the”. Proses ini membantu model fokus pada kata-kata yang memiliki nilai diskriminatif tinggi terhadap kelas tertentu (Azzahra & Majid, 2025).

Stemming adalah proses mengubah kata menjadi bentuk dasarnya seperti “closely”, “close”, “closing”, “closed” diwakili oleh satu bentuk yang sama, yaitu “close” (Habibillah dkk., 2025). Menurut algoritma *stemming* dapat dibagi menjadi beberapa jenis. Pertama, algoritma *rule-based* yang menggunakan aturan tertentu untuk menghapus awalan atau akhiran. Contohnya adalah *Lovins Stemmer* (1968) yang menggunakan daftar panjang akhiran untuk dipotong, dan *Porter Stemmer* (1980) yang paling banyak digunakan karena memiliki aturan yang sistematis dalam lima tahap pemrosesan. *Snowball Stemmer* (2001)

merupakan versi pengembangan dari *Porter* yang memiliki aturan lebih sederhana dan hasil pemotongan yang lebih konsisten. Selain itu, *Paice/Husk Stemmer* (1990) bekerja secara berulang dengan banyak aturan pemotongan, sedangkan *Dawson Stemmer* (1974) memiliki daftar akhiran yang lebih lengkap untuk meningkatkan ketepatan hasil *stemming* (Haroon, 2018; Memon dkk., 2020).

Selain itu, terdapat algoritma *statistical-based* yang tidak bergantung pada aturan bahasa, tetapi menggunakan pola statistik dari kumpulan data teks. Misalnya, *N-Gram Stemmer* membandingkan kemiripan urutan huruf antar kata, sedangkan *Hidden Markov Model (HMM) Stemmer* menggunakan pendekatan probabilitas untuk mengenali pola awalan dan akhiran kata. Ada juga YASS (*Yet Another Suffix Stripper*) yang mengelompokkan kata-kata berdasarkan kemiripan bentuk huruf tanpa aturan bahasa tertentu. Di samping itu, ada pendekatan lain seperti *corpus-based* yang melihat frekuensi kemunculan kata dalam teks, dan *context-sensitive stemming* yang memperhatikan makna kata dalam kalimat. Semua algoritma ini dikembangkan untuk menghasilkan proses pemotongan kata yang cepat, akurat, dan sesuai dengan konteks teks (Memon dkk., 2020).

Berbeda dengan *stemming*, *lemmatization* adalah proses mengubah kata ke bentuk dasarnya (*lemma*) dengan memperhatikan makna dan struktur gramatikalnya. Jika *stemming* hanya memotong akhiran atau awalan berdasarkan aturan, maka *lemmatization* menggunakan kamus (*lexicon*) dan analisis tata bahasa (*part-of-speech tagging*) untuk menentukan bentuk kata yang benar. Misalnya, kata “*running*”, “*ran*”, dan “*runs*” semuanya akan diubah menjadi “*run*” karena sistem memahami bahwa ketiganya berasal dari kata kerja yang sama. Pendekatan ini menghasilkan bentuk dasar yang lebih akurat dan bermakna

dibandingkan *stemming*, meskipun membutuhkan waktu proses yang lebih lama (Angriani dkk., 2025)

Sejalan dengan berbagai penelitian sebelumnya, *preprocessing* dalam *text mining* tidak hanya berfungsi sebagai tahap awal pengolahan teks, tetapi juga sebagai fondasi dalam memastikan kualitas data yang digunakan untuk model klasifikasi. Penelitian oleh (Fahmi dkk., 2024) membuktikan bahwa pemilihan strategi *preprocessing* yang tepat berpengaruh signifikan terhadap peningkatan performa model. Dalam penelitian tersebut, algoritma *Multinomial Naïve Bayes* diuji dengan beberapa kombinasi teknik *preprocessing*, dan hasil terbaik dicapai pada skenario *stemming* dengan akurasi 83%, presisi 78%, *recall* 94%, dan *f1-score* 85%. Fakta ini menegaskan bahwa penerapan *preprocessing* yang efektif merupakan komponen krusial dalam membangun model klasifikasi teks yang presisi dan efisien.

2.1.4 *Term Frequency–Inverse Document Frequency (TF-IDF)*

Term Frequency–Inverse Document Frequency (TF-IDF) merupakan metode pembobotan teks yang digunakan untuk mengukur seberapa penting suatu kata dalam sebuah dokumen relatif terhadap seluruh korpus. Secara konseptual, TF-IDF menggabungkan frekuensi kemunculan kata dalam dokumen dengan tingkat kelangkaannya di seluruh dokumen untuk menghasilkan nilai bobot yang representatif (Laksana dkk., 2025). TF-IDF terdiri dari dua komponen utama, yaitu:

1. *Term Frequency (TF)*, yang mengukur seberapa sering kata t muncul dalam sebuah dokumen d , dibandingkan dengan total kata di dalam dokumen tersebut. Frekuensi ini menggambarkan tingkat kepentingan kata tersebut di

dalam dokumen tersebut saja (Abdurrazik & Wirawan, 2025).

$$TF(t.d) = \frac{f_{t.d}}{\sum_k f_{k.d}} \quad (2.1)$$

di mana:

$TF(t.d)$: Frekuensi kemunculan kata t dalam dokumen d .

$f_{t.d}$: Jumlah kemunculan kata t dalam dokumen d .

$\sum_k f_{k.d}$: Jumlah semua kata dalam dokumen d .

2. *Inverse Document Frequency* (IDF), yang mengukur seberapa unik atau jarang kata tersebut muncul di seluruh kumpulan dokumen. Kata-kata yang sering muncul di banyak dokumen cenderung memiliki nilai IDF yang rendah karena dianggap tidak informatif (seperti kata “and”, “the”, dll) (Abdurrazik & Wirawan, 2025).

$$IDF(t) = \log\left(\frac{D}{df_t}\right) \quad (2.2)$$

di mana:

D : Jumlah total dokumen dalam dataset.

df_t : Jumlah dokumen yang mengandung kata t .

Kombinasi keduanya menghasilkan bobot yang tinggi untuk kata-kata yang sering muncul dalam satu dokumen tetapi jarang muncul di dokumen lain. Menurut penelitian oleh (Arasy dkk., 2025) pendekatan TF-IDF efektif dalam mengidentifikasi kata-kata kunci yang relevan dalam proses klasifikasi teks, termasuk deteksi spam. Secara matematis, rumus TF-IDF didefinisikan sebagai berikut:

$$TFIDF(t.d) = TF(t.d) \times IDF(t) \quad (2.3)$$

Dalam penelitian ini, TF-IDF digunakan untuk mengubah teks email menjadi representasi numerik yang dapat digunakan sebagai input bagi algoritma

klasifikasi. Proses ini dilakukan setelah tahap *preprocessing* seperti tokenisasi dan *stopword removal*, sehingga hanya kata-kata bermakna yang diberi bobot. TF-IDF membantu mengidentifikasi kata-kata yang paling informatif dalam membedakan email spam dan non-spam. Dengan demikian, TF-IDF menjadi komponen penting dalam kerangka kerja *text mining* yang digunakan dalam penelitian ini.

2.1.5 *Chi-Square Feature Selection*

Chi-Square Feature Selection merupakan metode statistik yang digunakan untuk mengukur tingkat ketergantungan antara fitur dan kelas dalam data kategorikal (Irawan dkk., 2025). Metode ini menguji apakah kemunculan suatu kata (fitur) dalam dokumen berkorelasi secara signifikan dengan kelas tertentu, seperti spam atau non-spam. Semakin besar nilai *Chi-Square* yang dihasilkan, semakin kuat hubungan antara kata tersebut dengan kelas tertentu. Oleh karena itu, fitur dengan nilai *Chi-Square* tinggi dianggap lebih relevan dan penting untuk digunakan dalam proses klasifikasi (Noviyanti & Putra, 2025).

Metode ini didasarkan pada distribusi *Chi-Square* (χ^2) yang berfungsi untuk mengukur kekuatan hubungan antara fitur dan kelas. Prinsip tersebut diterapkan dalam seleksi fitur untuk mengidentifikasi kata-kata yang paling berpengaruh dalam membedakan satu kelas dengan kelas lainnya. Dengan cara ini, *Chi-Square Feature Selection* mampu mengurangi jumlah fitur yang tidak relevan sekaligus mempertahankan kata-kata yang paling informatif bagi model klasifikasi (Hokijuliandy dkk., 2023). Fungsi dari uji *Chi-Square* sebuah kata terhadap suatu kategori diperoleh dari Persamaan (2.4).

$$\chi^2(t.C) = \frac{D(AG - BE)^2}{(A + E)(B + G)(A + B)(E + G)} \quad (2.4)$$

di mana:

$\chi^2(t.C)$: Nilai *Chi-Square* antara fitur (kata) t dan kelas C .

D : Jumlah total dokumen dalam dataset.

A : Jumlah dokumen yang mengandung kata t dan termasuk kelas C .

B : Jumlah dokumen yang mengandung kata t tetapi bukan kelas C .

E : Jumlah dokumen yang tidak mengandung kata t tetapi termasuk kelas C .

G : Jumlah dokumen yang tidak mengandung kata t tetapi bukan kelas C .

Nilai χ^2 yang diperoleh dari rumus tersebut mempresentasikan kekuatan hubungan antara kemunculan kata dengan kelas dokumen. Kata dengan nilai χ^2 tinggi dianggap memiliki pengaruh besar terhadap pembentukan model klasifikasi karena sering muncul dalam kelas tertentu dan jarang muncul pada kelas lainnya. Sebaliknya, kata dengan nilai χ^2 rendah cenderung tidak memberikan informasi yang signifikan bagi proses pemisahan kelas. Hal ini menjadikan metode *Chi-Square* sangat berguna dalam pemilihan fitur yang relevan untuk mode klasifikasi teks.

Untuk menentukan signifikansi keterkaitan fitur terhadap kelas, nilai χ^2_{hitung} dapat dibandingkan dengan χ^2_{tabel} berdasarkan derajat kebebasan (Df) dan taraf signifikansi α . Derajat kebebasan pada uji *Chi-Square* untuk tabel kontingensi dihitung dengan menggunakan rumus:

$$Df = (r - 1)(c - 1) \quad 2.5$$

Dengan r menyatakan jumlah baris dan c menyatakan jumlah kolom pada tabel kontingensi.

Dalam penerapannya pada klasifikasi teks seperti deteksi email spam, *Chi-Square Feature Selection* berperan penting dalam mengoptimalkan kinerja model. Dengan memilih hanya fitur-fitur yang memiliki nilai *Chi-Square* tertinggi, algoritma seperti *Multinomial Naïve Bayes* dapat bekerja lebih efisien dan menghasilkan tingkat akurasi yang lebih baik. Selain itu, metode ini membantu mengurangi kompleksitas model karena hanya menggunakan fitur yang paling relevan. Dengan demikian, metode *Chi-Square* tidak hanya berfungsi sebagai alat seleksi fitur, tetapi juga sebagai pendekatan statistik yang memperkuat proses *machine learning* pada data teks yang bersifat besar dan kompleks.

2.1.6 Synthetic Minority Oversampling Technique (SMOTE)

Synthetic Minority Oversampling Technique (SMOTE) merupakan metode yang digunakan untuk menangani masalah ketidakseimbangan kelas dalam data, yaitu kondisi ketika jumlah data pada satu kelas jauh lebih sedikit dibandingkan kelas lainnya, khususnya dalam konteks klasifikasi. Masalah ini, dapat menyebabkan model *machine learning* cenderung memprioritaskan kelas mayoritas dan mengabaikan kelas minoritas. Akibatnya, hasil klasifikasi menjadi tidak akurat, terutama untuk kelas dengan jumlah data yang lebih sedikit (Fadillah dkk., 2025).

Secara konseptual, SMOTE bekerja dengan menciptakan sampel sintetis dari kelas minoritas dengan membuat sampel baru yang berada di antara dua sampel kelas minoritas yang sudah ada. Proses ini dilakukan melalui interpolasi linier di antara sampel-sampel minoritas (Sidiq dkk., 2025). Dalam penelitian ini, SMOTE digunakan untuk menyeimbangkan distribusi antara email spam dan non-spam sebelum proses pelatihan model. Oleh karena itu, SMOTE berfungsi sebagai

pendekatan operasional untuk meningkatkan representasi kelas minoritas dalam dataset.

2.1.7 *Local Density Weighting SMOTE (LDW-SMOTE)*

SMOTE bekerja dengan melakukan proses *oversampling* secara seragam tanpa mempertimbangkan kondisi lokal tiap sampel minoritas. Akibatnya, sampel minoritas yang berada pada daerah padat memperoleh sampel sintetis terlalu banyak, sementara sampel yang berada pada daerah jarang tetap kurang terwakili. Untuk mengatasi masalah tersebut, dikembangkan pendekatan berbasis kepadatan lokal. Breunig dkk. (2000) memperkenalkan konsep *Local Outlier Factor* (LOF) yang mendefinisikan *local reability density*, yaitu ukuran kerapatan suatu titik berdasarkan rata-rata jarak ke tetangga terdekatnya. Semakin kecil rata-rata jaraknya, semakin tinggi kepadatan lokalnya. Prinsip inilah yang menjadi dasar pembentukan pada *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE).

LDW-SMOTE merupakan pengembangan dari SMOTE. LDW-SMOTE mempertimbangkan kepadatan lokal tiap sampel minoritas untuk menentukan jumlah dan karakteristik sampel sintetis yang dihasilkan. Konsep ini sejalan dengan penelitian terbaru oleh Lyu dkk. (2025) yang menggunakan pendekatan *local density estimation*. Di mana densitas dihitung berdasarkan jarak rata-rata ke tetangga terdekat untuk mengetahui apakah suatu titik berada pada daerah jarang atau padat.

Secara matematis, tahapan kerja LDW-SMOTE diawali dengan menghitung jarak antara satu data minoritas dengan data minoritas lainnya. Proses

ini dilakukan dengan memilih titik data dari kelas minoritas dan menentukan tetangga terdekatnya menggunakan rumus jarak *Euclidean* sebagai berikut:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{N_t} (x_{ik} - x_{jk})^2} \quad (2.6)$$

di mana:

$d(x_i, x_j)$: Jarak *Euclidean* antara dua titik data x_i dan x_j .

N_t : Jumlah fitur yang digunakan dalam model.

x_{ik} : Nilai fitur ke- k pada sampel minoritas x_i .

x_{jk} : Nilai fitur ke- k pada tetangga x_j

Setelah itu, dicari k tetangga terdekat dari tiap sampel untuk mendapatkan rata-rata jarak (*average distance*):

$$\text{avg_dist}_i = \frac{1}{k} \sum_{j=1}^k d(x_i, x_j) \quad (2.7)$$

di mana:

avg_dist_i : Rata-rata jarak dari sampel x_i ke k tetangga minoritas terdekat.

k : Jumlah tetangga terdekat yang digunakan.

$d(x_i, x_j)$: Jarak *Euclidean* antara dua titik data x_i dan x_j .

Nilai ini mencerminkan apakah sebuah titik berada pada daerah padat atau jarang. Semakin kecil nilai avg_dist_i maka semakin padat titik tersebut. Nilai avg_dist_i kemudian ditransformasikan menjadi *density* (kepadatan lokal sampel minoritas x_i):

$$\text{density}_i = \frac{1}{1 + \text{avg_dist}_i} \quad (2.8)$$

Dan nilai *rarity* (tingkat kelangkaan sampel x_i) dihitung sebagai kebalikan dari *density*:

$$rarity_i = \frac{1}{density_i} \quad (2.9)$$

Nilai *rarity* kemudian dinormalisasikan sehingga menjadi bobot *oversampling*:

$$weight_i = \frac{rarity_i}{\sum rarity} \quad (2.10)$$

Bobot ini menentukan berapa banyak sampel sintetis yang harus dibuat dari setiap titik minoritas. Titik dengan *density* rendah (jarang) mendapatkan bobot besar sehingga *oversampling* lebih banyak, sedangkan titik yang padat mendapatkan bobot kecil. Setelah bobot dihitung, LDW-SMOTE menghasilkan sampel sintetis melalui interpolasi linier berbobot antara titik minoritas x_i dan tetangganya x_j .

$$x_{synthetic} = x_i + (\delta)(weight_i)(x_j - x_i) \quad (2.11)$$

di mana:

$x_{synthetic}$: Data sintetis yang baru dihasilkan.

$weight_i$: Bobot *oversampling* untuk sampel x_i ,

x_i : Sampel minoritas asli.

x_j : Tetangga terdekat dari x_i .

δ : Bilangan acak antara 0 dan 1.

Proses ini dilakukan berulang-ulang sampai jumlah data pada kelas minoritas setara dengan jumlah data pada kelas mayoritas. Hasil akhirnya adalah dataset yang seimbang, di mana jumlah data pada kedua kelas menjadi sama. Dengan demikian, LDW-SMOTE merupakan pengembangan adaptif dari SMOTE yang mampu menghasilkan representasi kelas minoritas yang lebih baik dan stabil, terutama pada data teks berdimensi tinggi seperti pada kasus klasifikasi spam.

2.1.8 Algoritma *Naïve Bayes*

Naïve Bayes merupakan algoritma klasifikasi berbasis probabilitas yang menggunakan teorema Bayes dengan asumsi independensi antar fitur (Pati dkk., 2025). Secara konseptual, algoritma ini menghitung probabilitas suatu data termasuk dalam kelas tertentu berdasarkan nilai fitur yang dimilikinya (Aryanda dkk., 2024). Dalam konteks penelitian ini, *Naïve Bayes* digunakan untuk mengklasifikasikan email ke dalam kategori spam atau non-spam berdasarkan fitur teks yang telah diekstraksi. Oleh karena itu, *Naïve Bayes* berfungsi sebagai pendekatan operasional dalam tahap klasifikasi akhir dari sistem deteksi spam.

Naïve Bayes didasarkan pada teorema Bayes yang secara matematis dirumuskan sebagai berikut:

$$P(C|d) = \frac{P(d|C) \times P(C)}{P(d)} \quad (2.12)$$

di mana:

$P(C|d)$: *Posterior probability*, yaitu peluang hipotesis C benar setelah mempertimbangkan data d .

$P(d|C)$: *Likelihood*, yang merujuk pada kemunculan data d dengan asumsi bahwa hipotesis C adalah benar.

$P(d)$: *Evidence*, yang mengacu pada total data d tanpa memperhitungkan kelas yang terkait.

$P(C)$: *Prior probability* dari hipotesis C , yakni peluang awal tanpa memperhitungkan data d .

Prior menggambarkan peluang awal suatu kelas sebelum melihat data. *Likelihood* menunjukkan seberapa besar kemungkinan fitur tertentu muncul dalam kelas tertentu. *Posterior* merupakan hasil akhir, yaitu peluang bahwa suatu data

termasuk ke dalam kelas tertentu setelah mempertimbangkan bukti yang ada (Enjelia dkk., 2025). Tujuan *Naïve Bayes* adalah mencari nilai *posterior probability* tertinggi agar sistem dapat menentukan data termasuk ke kelas mana. Karena nilai $P(d)$ bersifat sama untuk semua kelas, maka nilai $P(d)$ dapat diabaikan dalam perbandingan. Dengan demikian, hubungan ketiga komponen tersebut dapat disederhanakan menjadi:

$$P(C|d) \propto P(d|C) \times P(C) \quad (2.13)$$

Asumsi utama dalam algoritma *Naïve Bayes* adalah bahwa setiap fitur atau atribut pada data bersifat independen, artinya kemunculan satu fitur tidak mempengaruhi fitur lainnya dalam menentukan kelas (Dinata & Hasdyna, 2025). Berdasarkan asumsi ini, peluang kemunculan suatu dokumen d pada kelas C dapat dihitung dengan mengalikan peluang kemunculan setiap kata t_i pada kelas tersebut, sebagaimana ditunjukkan pada Persamaan (2.9).

$$P(d|C) = \prod_{i=1}^{N_t} P(t_i|C) \quad (2.14)$$

Pendekatan ini membuat perhitungan lebih sederhana dan efisien tanpa mempertimbangkan hubungan antarfitur. Meskipun asumsi ini jarang benar sepenuhnya, *Naïve Bayes* tetap menunjukkan kinerja yang baik untuk data besar dan berdimensi tinggi seperti klasifikasi email spam (Dinata & Hasdyna, 2025).

Terdapat beberapa varian dari algoritma *Naïve Bayes*, antara lain *Gaussian Naïve Bayes*, *Multinomial Naïve Bayes*, dan *Bernoulli Naïve Bayes* (Mutoffar dkk., 2025). *Gaussian Naïve Bayes* digunakan untuk data numerik atau kontinu yang berdistribusi normal, di mana probabilitas setiap fitur dihitung berdasarkan fungsi distribusi Gaussian (normal). Varian ini sering digunakan pada data yang

mengandung nilai-nilai kontinu seperti tinggi badan, berat, atau nilai sensor (Januarto dkk., 2025). *Multinomial Naïve Bayes* merupakan varian yang paling umum digunakan dalam klasifikasi teks karena mempertimbangkan frekuensi kemunculan kata dalam dokumen (Laome dkk., 2025). Sementara itu, *Bernoulli Naïve Bayes* lebih cocok untuk data biner, seperti keberadaan atau ketiadaan kata dalam dokumen (Januarto dkk., 2025).

Dalam penelitian ini, algoritma yang digunakan adalah *Multinomial Naïve Bayes*, yaitu salah satu varian dari metode *Naïve Bayes* yang paling sesuai untuk data berbasis teks. Algoritma ini digunakan untuk mengklasifikasikan email ke dalam kategori spam atau non-spam berdasarkan fitur-fitur yang telah diproses sebelumnya. *Multinomial Naïve Bayes* dipilih karena mampu menangani data teks dalam jumlah besar serta mempertimbangkan frekuensi kemunculan kata (*term frequency*) pada setiap dokumen sebagai dasar perhitungannya. Dengan pendekatan probabilistik yang sederhana namun efektif, algoritma ini dapat menghasilkan prediksi yang cepat dan akurat dalam mendeteksi pesan spam. Oleh karena itu, penggunaan *Multinomial Naïve Bayes* dianggap paling tepat untuk mendukung sistem klasifikasi berbasis *text mining* pada penelitian ini.

2.1.9 *Multinomial Naïve Bayes*

Multinomial Naïve Bayes merupakan varian dari algoritma *Naïve Bayes* yang dirancang khusus untuk menangani data diskrit seperti teks. Secara konseptual, algoritma ini mengasumsikan bahwa fitur berupa kata-kata dalam dokumen mengikuti distribusi *Multinomial* berdasarkan frekuensi kemunculannya. Artinya semakin kata tertentu muncul, semakin besar pula kontribusinya terhadap penentuan kelas (Mardivta & Izzati, 2024). *Multinomial*

Naïve Bayes menggunakan Teorema Bayes yang dimodifikasi untuk menghitung probabilitas kelas berdasarkan distribusi kata.

Probabilitas sebuah dokumen d termasuk dalam kelas C dapat dihitung menggunakan rumus utama berikut:

$$P(C|d) \propto P(C) \prod_{i=1}^{N_t} P(t_i|C) \quad (2.15)$$

di mana:

- $P(C|d)$: Probabilitas dokumen d berada pada kelas C .
- $P(C)$: Probabilitas awal (*prior*) suatu kelas.
- $P(t_i|C)$: Probabilitas kemunculan kata ke- i (token) pada kelas C .
- N_t : Jumlah total fitur di seluruh dokumen dalam dataset.

Rumus tersebut menunjukkan bahwa peluang suatu dokumen d termasuk ke dalam kelas C ditentukan oleh probabilitas awal kelas $P(C)$ dan hasil kali dari seluruh probabilitas kemunculan kata t_i pada kelas tersebut. Setiap kelas akan menghasilkan nilai probabilitas yang berbeda-beda, tergantung pada kata-kata yang muncul di dokumen. Semakin sering kata tertentu muncul dalam dokumen dan juga sering muncul pada kelas tertentu di data latih, maka semakin besar peluang dokumen tersebut termasuk ke dalam kelas tersebut.

Setelah semua probabilitas kelas dihitung, model akan menentukan kelas terbaik, yaitu kelas yang memiliki nilai probabilitas paling tinggi. Proses ini disebut dengan *Maximum a Posteriori* (MAP), di mana sistem memilih kelas yang paling mungkin sesuai dengan data yang diberikan. Secara matematis, dapat dituliskan sebagai berikut:

$$C_{MAP} = \operatorname{argmax}_{C \in K} P(C) \prod_{i=1}^{N_t} P(t_i|C) \quad (2.16)$$

Namun, karena hasil perkalian dari banyak nilai probabilitas kecil dapat menghasilkan angka yang sangat kecil dan menyebabkan kesalahan komputasi, maka perhitungan biasanya diubah ke dalam bentuk logaritma agar lebih stabil.

Bentuk logaritmanya ditulis sebagai berikut:

$$C_{MAP} = \operatorname{argmax}_{C \in K} \left[\log P(C) + \sum_{i=1}^{N_t} \log P(t_i|C) \right] \quad (2.17)$$

Perubahan ke bentuk log ini tidak memengaruhi hasil akhir karena fungsi logaritma bersifat monoton, sehingga nilai probabilitas tertinggi tetap akan menghasilkan kelas terbaik yang sama. Selanjutnya, nilai *prior probability* $P(C)$ dihitung berdasarkan jumlah dokumen pada kelas C dibandingkan dengan jumlah seluruh dokumen dalam dataset. Rumusnya adalah sebagai berikut:

$$\hat{P}(C) = \frac{D_C}{D} \quad (2.18)$$

di mana:

$\hat{P}(C)$: *Prior* probabilitas suatu dokumen berada di kelas C .

D_C : Jumlah dokumen yang termasuk kelas C .

D : Jumlah seluruh dokumen.

Sementara itu, nilai *likelihood* atau probabilitas kemunculan kata t_i pada kelas C dalam penelitian ini dihitung dengan mempertimbangkan bobot TF-IDF (*Term Frequency–Inverse Document Frequency*) dari setiap kata pada kelas tertentu. Untuk menghindari probabilitas nol pada kata yang tidak muncul, digunakan teknik *Laplace smoothing* (*add-one smoothing*). Rumusnya dituliskan sebagai berikut:

$$P(t_i|C) = \frac{n_i + 1}{n + |kosakata|} \quad (2.19)$$

di mana:

n_i : Frekuensi kemunculan kata t_i pada kelas C .

n : Jumlah total kata pada kelas C .

$|kosakata|$: Jumlah kata unik dalam *vocabulary*.

Dalam penelitian ini, *Multinomial Naïve Bayes* digunakan untuk mengklasifikasikan email ke dalam kategori spam atau non-spam berdasarkan frekuensi kata. Algoritma ini dipilih karena kemampuannya dalam menangani data teks yang telah dibobot menggunakan TF-IDF. Dengan pendekatan probabilistik berbasis distribusi kata, algoritma ini dapat memberikan prediksi yang cepat dan akurat. Oleh karena itu, *Multinomial Naïve Bayes* menjadi komponen utama dalam sistem klasifikasi yang dikembangkan dalam penelitian ini.

2.1.10 Evaluasi Kinerja Model

Evaluasi kinerja model merupakan proses pengukuran efektivitas suatu algoritma dalam melakukan klasifikasi atau prediksi berdasarkan data uji. Jenis evaluasi yang digunakan meliputi *confusion matrix* dan metrik evaluasi yakni akurasi, presisi, *recall*, dan *f1-score* (Ridwan dkk., 2025). *Confusion matrix* berbentuk tabel dua dimensi yang terdiri dari empat komponen utama, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Nilai TP menunjukkan jumlah data positif yang berhasil diklasifikasikan dengan benar, sedangkan TN menunjukkan jumlah data negatif yang juga diprediksi benar. FP menggambarkan data negatif yang salah diklasifikasikan sebagai positif, dan FN menunjukkan data positif yang gagal terdeteksi oleh model (Brian

dkk., 2025). Struktur umum dari *confusion matrix* dapat digambarkan sebagai berikut:

Tabel 2.1 *Confusion Matrix*

	Prediksi Positif	Prediksi Negatif
Aktual Positif	TP	FN
Aktual Negatif	FP	TN

Dari tabel tersebut, dapat dihitung berbagai metrik evaluasi yang menggambarkan kualitas performa model. Metrik pertama adalah akurasi, yaitu persentase benar model dalam melakukan klasifikasi (Yana dkk., 2025). Rumus akurasi dapat dituliskan sebagai berikut:

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.20)$$

Semakin tinggi nilai akurasi, semakin baik kinerja model dalam memprediksi data dengan benar. Namun, akurasi saja tidak selalu cukup, terutama jika jumlah data pada tiap kelas tidak seimbang, karena model dapat terlihat akurat hanya karena lebih sering menebak kelas mayoritas.

Untuk itu digunakan metrik lain, yaitu presisi dan *recall*, yang lebih fokus pada performa model terhadap kelas positif. Presisi mengukur sejauh mana hasil prediksi positif model benar-benar relevan (Brian dkk., 2025). Dalam konteks deteksi spam, presisi menunjukkan proporsi email yang benar-benar spam dari seluruh email yang diprediksi spam oleh model. Rumusnya dituliskan sebagai berikut:

$$\text{Presisi} = \frac{TP}{TP + FP} \quad (2.21)$$

Nilai presisi yang tinggi menandakan bahwa model jarang salah dalam menganggap email non-spam sebagai spam.

Berbeda dengan presisi, *recall* mengukur proporsi data positif yang berhasil diprediksi dengan benar oleh model dibandingkan dengan seluruh data yang sebenarnya positif (Pinem & Putra, 2025). Dalam kasus email spam, *recall* menunjukkan seberapa banyak email spam yang berhasil terdeteksi oleh model dari seluruh spam yang ada. Rumus *recall* dituliskan sebagai:

$$Recall = \frac{TP}{TP + FN} \quad (2.22)$$

Nilai *recall* yang tinggi menunjukkan bahwa model tidak banyak melewatkan data yang seharusnya terdeteksi.

Namun, dalam banyak kasus, presisi dan *recall* perlu dipertimbangkan secara bersamaan, karena model yang memiliki presisi tinggi bisa saja memiliki *recall* rendah, atau sebaliknya. Untuk menyeimbangkan keduanya digunakan metrik *f1-score*, yang merupakan rata-rata harmonik antara presisi dan *recall*. *F1-score* memberikan ukuran tunggal yang merepresentasikan keseimbangan antara ketepatan dan kelengkapan model dalam melakukan prediksi (Arasy dkk., 2025). Rumusnya adalah sebagai berikut:

$$F1 - Score = 2 \times \frac{Presisi \times Recall}{Presisi + Recall} \quad (2.23)$$

Nilai *f1-score* berkisar antara 0 hingga 1, di mana nilai 1 menunjukkan kinerja model yang sempurna dan nilai mendekati 0 menunjukkan performa yang buruk. Dengan demikian, evaluasi kinerja model melalui *confusion matrix* serta metrik akurasi, presisi, *recall*, dan *f1-score* memberikan gambaran menyeluruh mengenai kemampuan model dalam melakukan klasifikasi.

2.1.11 Spam Email

Spam email merupakan pesan elektronik yang dikirim secara massal dan tidak diinginkan oleh penerima, sering kali berisi iklan, penipuan, atau konten berbahaya (Rahmawati dkk., 2024). Spam dikategorikan sebagai bentuk komunikasi digital yang tidak sah dan mengganggu, serta dapat membahayakan privasi dan keamanan pengguna. Spam email dapat diklasifikasikan ke dalam beberapa kategori, seperti iklan komersial, *phishing*, *malware*, dan penipuan finansial (Nurcahyo dkk., 2025). Klasifikasi ini penting untuk memahami motif dan strategi yang digunakan oleh pengirim spam dalam menyusun konten pesan. Dalam konteks klasifikasi otomatis, spam email dibedakan dari email biasa berdasarkan pola teks, metadata pengirim, dan struktur pesan.

2.2 Kajian Integrasi Topik dengan Al-Quran/Hadits

Islam menempatkan kebenaran, keadilan, dan kehati-hatian dalam memilah serta menyampaikan informasi sebagai nilai yang sangat utama. Dalam konteks perkembangan teknologi informasi modern, penerapan algoritma dan metode dalam bidang *machine learning* dapat dipandang bukan hanya sebagai upaya ilmiah, tetapi juga sebagai refleksi nilai-nilai Islam tentang ketepatan, kejujuran, dan keseimbangan. Penelitian ini yang berfokus pada *Chi-Square Feature Selection* dan LSW-SMOTE (*Local Density Weighting Synthetic Minority Oversampling Technique*) menunjukkan keterkaitan antara prinsip ilmiah dan ajaran Islam dalam menjaga kemurnian data serta keadilan analisis. Dalam konteks penelitian ini, algoritma *Naïve Bayes* yang digunakan untuk mengklasifikasikan pesan spam dan non-spam dapat diibaratkan sebagai bentuk “*tabayyun*” sebuah penerapan prinsip

kehati-hatian dalam era modern, di mana sistem komputer membantu manusia menyeleksi informasi yang benar dan bermanfaat dari yang menyesatkan.

Penerapan metode *Chi-Square Feature Selection* yang berfungsi untuk memilih fitur-fitur paling relevan dalam proses klasifikasi sejalan dengan nilai yang terkandung dalam QS. Ar-Ra'd ayat 17, yaitu:

أَنْزَلَ مِنَ السَّمَاءِ مَاءً فَسَالَتْ أَوْدِيَةٌ بِقَدَرِهَا فَاحْتَمَلَ السَّيْلُ زَبَدًا رَابِيًا وَمِمَّا يُوقِدُونَ عَلَيْهِ فِي النَّارِ ابْتِغَاءَ حِلْيَةٍ أَوْ مَتَاعٍ زَبَدٌ مِثْلُ بَثَلٍ كَذَلِكَ يَضْرِبُ اللَّهُ الْحَقَّ وَالْبَاطِلَ ه فَاَمَّا الزَّبَدُ فَيَذْهَبُ جُفَاءً وَاَمَّا مَا يَنْفَعُ النَّاسَ فَيَمْكُثُ فِي الْأَرْضِ كَذَلِكَ يَضْرِبُ اللَّهُ الْأَمْثَالَ ﴿١٧﴾

“Dia telah menurunkan air dari langit, lalu mengalirlah air itu di lembah-lembah sesuai dengan ukurannya. Arus itu membawa buih yang mengambang. Dari apa (logam) yang mereka lebur dalam api untuk membuat perhiasan atau alat-alat, ada (pula) buih seperti (buih arus) itu. Demikianlah Allah membuat perumpamaan tentang hak dan batil. Buih akan hilang tidak berguna, sedangkan yang bermanfaat bagi manusia akan menetap di dalam bumi. Demikianlah Allah membuat perumpamaan.” (Kementrian Agama, 2022)

Ayat ini turun untuk menggambarkan perbandingan antara kebenaran dan kebatilan. Menurut Tafsir Tahlili (NU Online), kebenaran dan kebatilan itu apabila bercampur, seperti arus air yang bercampur buih. Maka sebagaimana buih yang berada di atas air akan lenyap setelah tiupan angin, demikian pula perkara yang batil akan hilang dan musnah apabila datang hak dan kebenaran yang jelas.

Dalam asbābun nuzul-nya (NU Online), para ulama menjelaskan bahwa Allah SWT menurunkan perumpamaan ini agar manusia memahami bahwa kebenaran dan manfaat akan bertahan, sedangkan kebatilan dan hal yang tidak berguna akan lenyap seperti buih di atas air. Perumpamaan ini juga menunjukkan bahwa Allah SWT menyingkap kebenaran melalui proses penyaringan, di mana sesuatu yang bermanfaat dipertahankan dan yang tidak bernilai dibuang. Konsep tersebut sejalan dengan prinsip *Chi-Square Feature Selection*, yaitu proses penyaringan fitur untuk

memisahkan kata-kata yang tidak relevan atau “buih” data dari fitur yang benar-benar bermakna. Dengan demikian, hanya fitur yang informatif dan bernilai tinggi yang dipertahankan agar model klasifikasi menjadi lebih efisien dan akurat.

Selain itu, Islam juga menekankan pentingnya ketelitian dan selektivitas dalam menerima informasi. Hal ini sejalan dengan hadis Nabi Muhammad SAW:

عَنْ أَبِي هُرَيْرَةَ رَضِيَ اللَّهُ عَنْهُ عَنِ النَّبِيِّ صَلَّى اللَّهُ عَلَيْهِ وَسَلَّمَ قَالَ : كَفَى بِالْمَرْءِ كَذِبًا، أَنْ يُحَدِّثَ بِكُلِّ مَا سَمِعَ (رواه مسلم)

Dari Abu Hurairah RA dari Nabi saw beliau bersabda, "Cukuplah seseorang (dianggap) berdusta jika ia menceritakan semua yang ia dengar." (HR. Muslim) (Muslim bin al-Hajjaj)

Hadist tersebut, menurut ulama seperti imam Nawawi dalam kitab *Shahih Muslim*, menunjukkan larangan menyebarkan informasi tanpa proses seleksi. Dalam hal ini teknologi klasifikasi spam email dengan menggunakan *Chi-Square Feature Selection* dapat mencerminkan nilai spiritual Islam tentang pentingnya penyucian, ketelitian, dan selektivitas dalam memilah informasi yang benar dan bermanfaat.

Adapun metode LDW-SMOTE yang digunakan dalam penelitian ini juga memiliki nilai filosofis yang selaras dengan ajaran Islam. LDW-SMOTE bertujuan menyeimbangkan distribusi data agar hasil klasifikasi menjadi adil dan tidak bias. Prinsip ini sejalan dengan nilai-nilai Islam yang menekankan pentingnya keadilan (*'Al-adl*) dalam setiap aspek kehidupan. Allah SWT berfirman dalam QS. An-Nahl ayat 90:

إِنَّ اللَّهَ يَأْمُرُ بِالْعَدْلِ وَالْإِحْسَانِ وَإِيتَائِ ذِي الْقُرْبَىٰ وَيَنْهَىٰ عَنِ الْفَحْشَاءِ وَالْمُنْكَرِ وَالْبَغْيِ يَعِظُكُم لَعَلَّكُمْ تَذَكَّرُونَ ﴿٩٠﴾

“Sesungguhnya Allah menyuruh berlaku adil, berbuat kebajikan, dan memberikan bantuan kepada kerabat. Dia (juga) melarang perbuatan keji, kemungkaran, dan permusuhan. Dia memberi pelajaran kepadamu agar kamu selalu ingat.” (Kementrian Agama, 2022)

Dalam Tafsir Tahlili (NU Online), ayat ini dimaknai bahwa Allah SWT memerintahkan kaum muslimin untuk berbuat adil dalam semua aspek kehidupan. Adil berarti mewujudkan kesamaan dan keseimbangan hak dan kewajiban. Dari ayat tersebut, dapat dipahami bahwa upaya menyeimbangkan data melalui teknik LDW-SMOTE mencerminkan nilai keadilan dalam konteks ilmiah, yaitu memastikan bahwa model tidak memihak pada kelas mayoritas dan tetap memberikan kesempatan yang sama bagi seluruh kelas untuk diolah secara proporsional. Hal ini menunjukkan bahwa pendekatan komputasi modern pun dapat mengandung nilai-nilai etis Islam apabila diterapkan dengan tujuan yang benar.

Dengan demikian, dapat disimpulkan bahwa penelitian ini tidak hanya berorientasi pada pencapaian teknis dalam meningkatkan kinerja algoritma *Naïve Bayes* melalui *Chi-Square Feature Selection* dan LDW-SMOTE, tetapi juga mengandung dimensi nilai Islami yang menekankan pentingnya kejujuran, keadilan, penyaringan kebenaran, dan tanggung jawab informasi. Integrasi nilai-nilai Islam ini menunjukkan bahwa ilmu pengetahuan dan teknologi tidak terlepas dari aspek moralitas, melainkan dapat menjadi sarana untuk menerapkan ajaran Islam dalam menghadapi tantangan digital masa kini.

2.3 Kajian Topik Dengan Teori Pendukung

Penelitian ini berfokus pada penerapan algoritma *Multinomial Naïve Bayes* dalam klasifikasi email spam, yang didukung oleh dua pendekatan penting, yaitu *Chi-Square Feature Selection* untuk seleksi fitur dan *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) untuk menangani

ketidakseimbangan kelas. Ketiga metode ini saling berkaitan dan memiliki landasan teori yang kuat dalam bidang *machine learning*. Kombinasi ketiganya diharapkan mampu meningkatkan efektivitas sistem dalam mengenali dan memisahkan email spam dari email non-spam secara otomatis.

Algoritma *Naïve Bayes* digunakan karena memiliki kemampuan tinggi dalam memproses data berdimensi besar seperti teks, serta efisien dalam waktu komputasi. Algoritma ini bekerja berdasarkan prinsip probabilitas dengan mengasumsikan bahwa setiap fitur bersifat independen. Dalam konteks penelitian ini, *Multinomial Naïve Bayes* dipilih karena paling sesuai untuk data teks yang dihitung berdasarkan frekuensi kemunculan kata. Pendekatan probabilistik ini membantu sistem dalam menilai seberapa besar kemungkinan suatu email tergolong spam berdasarkan kata-kata yang terkandung di dalamnya. Hasil penelitian terdahulu juga menunjukkan bahwa *Multinomial Naïve Bayes* seringkali menghasilkan akurasi tinggi dalam klasifikasi teks, sehingga relevan diterapkan pada kasus deteksi spam.

Selain algoritma klasifikasi, penelitian ini juga memanfaatkan metode *Chi-Square Feature Selection* untuk meningkatkan kualitas fitur yang digunakan oleh model. Metode ini bekerja dengan mengukur hubungan antara fitur (kata) dan kelas (spam atau non-spam) untuk memilih kata yang paling relevan dalam proses klasifikasi. Dengan menghapus fitur yang tidak signifikan, model menjadi lebih sederhana, cepat, dan akurat karena hanya menggunakan fitur yang benar-benar berpengaruh terhadap hasil klasifikasi. Pendekatan ini sangat penting dalam kasus data teks seperti email yang biasanya memiliki jumlah fitur sangat banyak, sehingga proses seleksi fitur menjadi langkah kunci dalam meningkatkan kinerja model.

Untuk mengatasi masalah ketidakseimbangan kelas yang sering terjadi pada dataset spam, di mana jumlah email non-spam jauh lebih banyak dibandingkan spam, penelitian ini menerapkan *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) sebagai pengembangan dari SMOTE klasik. Berbeda dengan SMOTE yang menghasilkan data sintetis secara seragam, LDW-SMOTE mempertimbangkan kepadatan lokal data minoritas dengan menghitung rata-rata jarak antar sampel untuk menentukan bobot *oversampling*. Pendekatan ini memungkinkan pembuatan sampel sintetis yang lebih adaptif dan representatif terhadap distribusi lokal kelas spam, sehingga model dapat mempelajari karakteristik kelas minoritas secara lebih optimal.

Secara keseluruhan, kombinasi antara *Multinomial Naïve Bayes*, *Chi-Square* Feature Selection, dan LDW-SMOTE membentuk kerangka kerja yang kuat untuk klasifikasi email spam. *Naïve Bayes* berperan sebagai model probabilistik utama, *Chi-Square* meningkatkan relevansi fitur, dan LDW-SMOTE memperbaiki distribusi data melalui pendekatan matematis berbasis kepadatan lokal. Ketiganya saling melengkapi untuk mencapai tujuan penelitian, yaitu menghasilkan sistem klasifikasi spam yang akurat, efisien, dan mampu menangani data teks berskala besar secara optimal.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Penelitian ini merupakan penelitian kuantitatif eksperimental yang berfokus pada evaluasi pengaruh metode *Chi-Square Feature Selection* dan teknik *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) terhadap kinerja algoritma *Naïve Bayes* dalam mendeteksi email spam dan non-spam. Pendekatan kuantitatif dipilih karena meskipun data awal berupa teks, setelah melalui tahapan *preprocessing* dan transformasi TF-IDF, data tersebut diubah menjadi bentuk numerik sehingga dapat dianalisis secara komputasional. Eksperimen dalam penelitian ini dilakukan melalui serangkaian tahapan yang mencakup eksplorasi data, *preprocessing*, pembagian data latih dan data uji, TF-IDF, seleksi fitur dengan metode *Chi-Square*, penyeimbangan data menggunakan LDW-SMOTE, pemodelan dengan algoritma *Multinomial Naïve Bayes*, serta evaluasi kinerja model menggunakan metrik evaluasi.

3.2 Data dan Sumber Data

Data yang digunakan dalam penelitian ini adalah (Enron Fraud Email Dataset), sebuah kumpulan data publik yang tersedia di platform Kaggle (<https://www.kaggle.com/datasets/advaithsrao/Enron-fraud-email-dataset>) yang terdiri dari 447.417 baris dan 32 atribut atau kolom. Namun, tidak semua atribut digunakan dalam penelitian ini. Pertimbangan tersebut didasarkan pada kondisi bahwa sebagian atribut memiliki banyak nilai kosong, beberapa hanya memuat satu kata yang sama pada seluruh baris sehingga tidak memberikan variasi informasi,

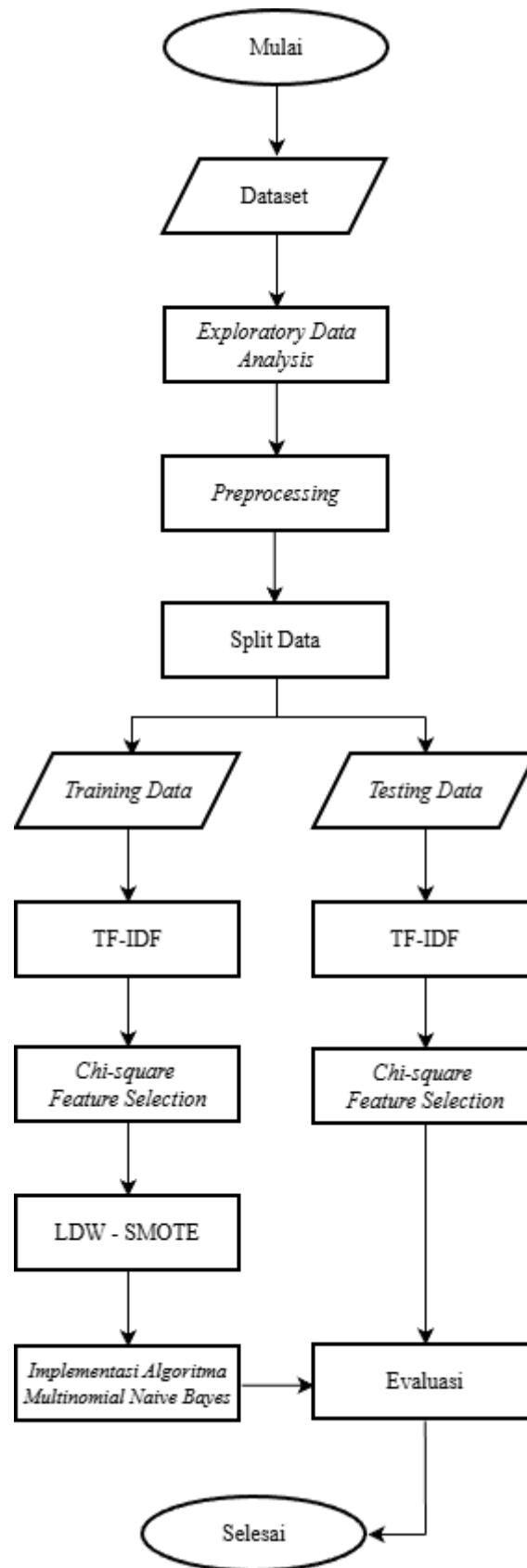
dan beberapa atribut bersifat unik pada setiap baris seperti ID sehingga tidak relevan untuk proses klasifikasi. Dengan dasar tersebut, penelitian ini difokuskan pada pemanfaatan empat atribut utama, yaitu *subject*, *body*, *from*, dan label, sedangkan atribut lainnya diabaikan karena tidak memiliki kontribusi signifikan terhadap tujuan klasifikasi email spam. Beberapa kolom utama dalam dataset ini meliputi:

Tabel 3.1 Deskripsi Data

Nama Kolom	Deskripsi
<i>Body</i>	Isi utama dari pesan email (konten teks lengkap yang dikirim).
<i>Subject</i>	Judul atau topik email yang dituliskan oleh pengirim.
<i>From</i>	Alamat email pengirim.
Label	Target klasifikasi: 0 untuk non-spam, 1 untuk spam.

3.3 Teknik Analisis Data

Analisis data pada penelitian ini dilakukan melalui beberapa tahapan sebagaimana ditunjukkan pada Gambar 3.1. Setiap tahapan memiliki peran penting dalam memastikan bahwa data yang digunakan siap diproses dan menghasilkan model klasifikasi yang optimal. Proses analisis dimulai dengan tahap *Exploratory Data Analysis* (EDA), *preprocessing*, pembagian data, ekstraksi fitur menggunakan (TF-IDF), *Chi-Square Feature Selection*, *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE), algoritma *Multinomial Naïve Bayes* diimplementasikan sebagai model utama dalam melakukan klasifikasi email spam dan tahap akhir dilakukan evaluasi model dengan menggunakan metrik seperti akurasi, presisi, *recall*, dan *f1-score* untuk menilai kinerja serta keakuratan hasil klasifikasi yang dihasilkan oleh model.

Gambar 3.1 *Flowchart* Analisis Data

3.3.1 *Exploratory Data Analysis (EDA)*

Exploratory Data Analysis (EDA) merupakan tahap awal yang penting dalam proses analisis data, dengan tujuan memahami karakteristik dasar dataset sebelum dilakukan pemodelan. Pada tahap ini dilakukan identifikasi terhadap keberadaan *missing value*, data duplikat, serta distribusi kelas. Dalam penelitian ini, EDA dilakukan untuk memahami struktur dan distribusi email spam maupun non-spam. Analisis distribusi kelas dilakukan dengan menghitung proporsi masing-masing kategori untuk mengetahui apakah terdapat ketidakseimbangan kelas (*class imbalance*). Ketidakseimbangan ini penting diperhatikan karena dapat memengaruhi performa model klasifikasi, di mana model cenderung bias terhadap kelas mayoritas. Hasil distribusi kelas kemudian divisualisasikan menggunakan diagram batang, sehingga perbandingan proporsi email spam dan non-spam dapat terlihat secara lebih jelas.

3.3.2 *Preprocessing Teks*

Tahap *preprocessing* teks merupakan langkah krusial dalam pengolahan data teks, khususnya dalam klasifikasi email spam. Tahap ini bertujuan untuk membersihkan dan menormalisasi data agar lebih terstruktur serta dapat diproses secara efektif oleh algoritma *machine learning*. Proses ini dilakukan melalui beberapa tahapan sistematis dengan menggunakan fungsi-fungsi pembersih khusus yang dirancang untuk menghapus informasi tidak relevan dan mengubah teks menjadi lebih bersih, konsisten, dan siap untuk dilakukan proses representasi fitur dan pemodelan.

Langkah pertama yang dilakukan adalah perbaikan *missing value* dan data duplikat dengan menggunakan *library pandas*. Baris data yang tidak memiliki

label dihapus karena tidak dapat digunakan dalam proses pelatihan model. Nilai yang hilang pada kolom *Subject*, *Body*, dan *From* diisi dengan string kosong, serta baris dengan *Subject* dan *Body* yang sama sama kosong juga dihapus dari dataset. Sementara itu, data yang memiliki kombinasi *Subject* dan *Body* yang identik dianggap sebagai duplikat, dan hanya satu entri pertama dari setiap pasangan duplikat yang dipertahankan.

Tahapan berikutnya adalah *lowercasing* dan *cleaning* dengan menggunakan *regular library pandas* dan *re*. Pada tahap ini, teks dari kolom *Subject*, *Body*, dan *From* menjadi satu kolom baru untuk menyatukan seluruh informasi yang relevan dari setiap email. Seluruh teks kemudian diubah menjadi huruf kecil agar tidak terjadi perbedaan interpretasi antara huruf besar dan kecil. Setelah itu dilakukan pembersihan teks dengan menghapus elemen-elemen yang tidak relevan seperti awalan “re:” atau “fw:”, URL, tag HTML, angka karakter non-alfabetis. Spasi ganda juga dihapus untuk menjaga konsistensi format teks.

Selanjutnya, teks yang telah dibersihkan dipecah menjadi potongan-potongan kata melalui proses *tokenization* dengan menggunakan *library re*. Setelah diperoleh token, dilakukan penghilangan *stopwords* dengan menggunakan *library NLTK* khususnya modul *nlk.corpus.stopwords*. pada bagian ini, kata-kata umum yang memiliki frekuensi tinggi namun nilai informasi rendah, seperti “the”, “a”, “is” dan sejenisnya akan dihapus. *Stopwords* tidak memberikan kontribusi signifikan terhadap proses klasifikasi sehingga perlu dihilangkan. Tahapan terakhir adalah proses *Lemmatization* dengan menggunakan *library NLTK* khususnya modul *nlk.stem.WordNetLemmatizer*, yakni mengubah kata ke bentuk dasarnya, misalnya “*running*” menjadi “*run*”. Proses ini penting agar variasi kata

dengan akar yang sama dianggap sebagai satu entitas. Hasil *preprocessing* ini kemudian digunakan pada tahap transformasi fitur menggunakan metode TF-IDF.

3.3.3 *Split Data*

Pada tahap ini, dataset yang telah melalui proses *preprocessing* dibagi menjadi dua bagian, yaitu data latih (*training data*) dan data uji (*testing data*). Pembagian ini bertujuan agar model dapat dilatih menggunakan data latih dan kemudian diuji performanya pada data uji yang belum pernah dilihat sebelumnya. Dalam penelitian ini, 20% dari setiap kelas (spam dan non-spam) digunakan sebagai data uji, sementara 80% sisanya digunakan sebagai data latih. Proporsi data uji yang kecil dipilih agar data latih memiliki jumlah sampel yang memadai untuk proses penyeimbangan kelas menggunakan metode LDW-SMOTE pada tahap selanjutnya.

3.3.4 Ekstraksi Fitur TF-IDF

Setelah data teks melalui tahap *preprocessing* dan *split data*, langkah selanjutnya adalah mengubah data tersebut ke dalam bentuk numerik agar dapat diproses oleh algoritma *machine learning*. Metode yang digunakan dalam penelitian ini adalah TF-IDF (*Term Frequency–Inverse Document Frequency*). Dalam penelitian ini, transformasi TF-IDF dilakukan dengan mempertahankan 2000 kata paling relevan sebagai fitur utama. Prosesnya melibatkan perhitungan seberapa sering suatu kata muncul dalam sebuah email dan matriks numerik yang mempresentasikan tingkat kepentingan setiap kata pada tiap email. Matriks TF-IDF dari data latih digunakan untuk membangun model, sedangkan data uji diubah dengan kosatakata yang sama agar konsisten. Representasi ini kemudian menjadi dasar untuk tahap selanjutnya.

3.3.5 *Chi-Square Feature Selection*

Setelah representasi teks dalam bentuk matriks TF-IDF diperoleh, tahap berikutnya adalah seleksi fitur. Pada tahap ini, digunakan metode *Chi-Square Feature Selection* untuk menentukan fitur yang memiliki hubungan signifikan dengan kelas target. Uji signifikansi dilakukan menggunakan taraf signifikansi (α) sebesar 0,05 (5%), sesuai dengan ketentuan umum dalam penelitian kuantitatif bahwa tingkat kesalahan 5% masih dapat diterima (Ghozali, 2018; Sugiyono, 2016). Setelah dilakukan perhitungan nilai *Chi-Square* untuk seluruh fitur, hasilnya diurutkan berdasarkan skor tertinggi, dan dipilih 1.000 fitur teratas untuk proses klasifikasi. Proses seleksi dilakukan pada data latih yang telah ditransformasikan ke dalam bentuk matriks TF-IDF, kemudian fitur-fitur terpilih tersebut diterapkan pula pada data uji agar representasinya tetap konsisten. Hasil dari tahap ini berupa matriks fitur berdimensi lebih rendah yang siap digunakan pada tahap berikutnya, yaitu penanganan ketidakseimbangan kelas menggunakan LDW-SMOTE dan pelatihan model klasifikasi.

3.3.6 LDW-SMOTE

Salah satu tantangan umum dalam klasifikasi email spam adalah adanya ketidakseimbangan kelas (*class imbalance*), yaitu jumlah data non-spam jauh lebih besar dibandingkan jumlah data spam. Kondisi ini dapat menyebabkan model cenderung bias terhadap kelas mayoritas (non-spam), sehingga mengabaikan kelas minoritas (spam) yang justru merupakan kelas penting dalam konteks deteksi spam. Untuk mengatasi permasalahan tersebut, penelitian ini menerapkan teknik LDW-SMOTE (*Local Density Weighting -Synthetic Minority Oversampling Technique*). Teknik ini bekerja dengan menghasilkan sampel baru

dari kelas minoritas melalui proses interpolasi berbobot antara titik data minoritas yang ada dengan tetangganya, sehingga menghasilkan data baru yang lebih bervariasi dan menyerupai distribusi alami secara lebih akurat.

Penerapan LDW-SMOTE dilakukan hanya pada data latih. Karena supaya model dapat belajar dari data yang seimbang tanpa mempengaruhi validitas evaluasi model pada data uji yang tetap mempertahankan distribusi asli. Dengan demikian, model yang dihasilkan diharapkan mampu mengenali pola dari kedua kelas dengan lebih baik dan tidak bias terhadap salah satu kelas saja. Langkah ini penting dalam penelitian klasifikasi dengan data tidak seimbang karena dapat secara signifikan meningkatkan *recall* dan *f1-score* pada kelas minoritas, tanpa menurunkan kinerja keseluruhan model.

3.3.7 Pelatihan Model *Multinomial Naïve Bayes*

Tahap pelatihan model merupakan inti dari proses klasifikasi, di mana algoritma belajar dari data latih untuk mengenali pola yang membedakan antara email spam dan non-spam. Dalam penelitian ini, algoritma yang digunakan adalah *Multinomial Naïve Bayes*, yaitu varian dari algoritma *Naïve Bayes* yang dirancang khusus untuk data berbasis frekuensi atau jumlah kemunculan kata, seperti hasil transformasi dari metode TF-IDF. Proses pelatihan dilakukan dengan menerapkan *Stratified K-Fold Cross-Validation* sebanyak 5 *fold*, yang bertujuan untuk memperoleh estimasi performa model yang lebih stabil dan menghindari bias pada pembagian data. Setelah proses pelatihan, dilakukan evaluasi awal terhadap data latih untuk memperoleh gambaran mengenai kemampuan model dalam mengenali pola data sebelum dilakukan pengujian pada data uji yang akan dibahas pada tahap selanjutnya.

3.3.8 Evaluasi Model

Tahap evaluasi model dilakukan setelah algoritma *Multinomial Naïve Bayes* selesai dilatih menggunakan data latih. Evaluasi ini bertujuan untuk mengukur kemampuan model dalam menggeneralisasi pola data baru dengan menggunakan data uji yang tidak pernah digunakan selama pelatihan. Pada penelitian ini, proses evaluasi dilakukan melalui empat skenario yang digunakan meliputi MNB tanpa *Chi-Square*, MNB dengan *Chi-Square*, MNB dengan *Chi-Square* disertai LDW-SMOTE, serta MNB tanpa *Chi-Square* dengan penerapan LDW-SMOTE. Setiap model dievaluasi menggunakan beberapa metrik evaluasi standar, yaitu akurasi, presisi, *recall*, *f1-score* dan *confusion matrix*. Untuk memperoleh hasil evaluasi yang lebih representatif dan mengurangi pengaruh variasi pada setiap percobaan, proses evaluasi pada masing-masing skenario dilakukan sebanyak lima kali *running*. Hasil dari setiap *running* kemudian digunakan untuk menghitung nilai rata-rata dan standar deviasi sehingga dapat memberikan gambaran yang lebih akurat mengenai performa model.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil

4.1.1 *Exploratory Data Analysis* (EDA)

Exploratory Data Analysis (EDA) dilakukan untuk memahami karakteristik awal dataset Enron Email sebelum memasuki tahap *preprocessing* dan pemodelan. Analisis diawali dengan peninjauan sampel data serta pemeriksaan *missing value* pada atribut yang digunakan dalam penelitian, yaitu *subject*, *body*, dan *from*. Contoh sampel data email yang digunakan dalam penelitian disajikan pada Tabel 4.1. Hasil analisis menunjukkan bahwa kolom *subject* memiliki 18.020 *missing value*, sedangkan kolom *body* dan *from* tidak memiliki *missing value*. Selain itu, pemeriksaan data duplikat berdasarkan kombinasi *subject*, *body*, dan *from* menunjukkan adanya 203.030 data duplikat, sehingga data tersebut perlu dihapus untuk menghindari redundansi dan potensi bias dalam proses pelatihan model.

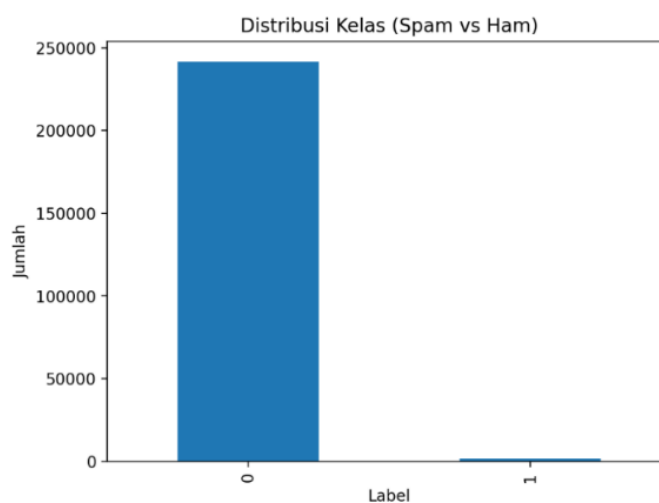
Tabel 4.1 Sampel Data Email

<i>Subject</i>	<i>Body</i>	<i>From</i>	<i>Label</i>
ALL daily charts and matrice s as hot	ALL daily charts and matrices as hot links 10/17-resend The information contained herein is based on sources that we believe to be reliable, but we do not represent that it is accurate or complete. Nothing contained herein should be considered as an offer to sell	sobland er@car rfut.co m	Non-spam

links 10/17- resend	or a solicitation of an offer to buy any financial instruments discussed herein. Any opinions expressed herein are solely those of the author. As such, they may differ in material respects from those of, or expressed or published by on behalf of Carr Futures or its officers, directors, employees or affiliates. 2001 Carr Futures The charts are now available on the web by clicking on the hot link(s) contained in this email. If for any reason you are unable to receive the charts via the web, please contact me via email and I will email the charts to you as attachments. Crude <URL> Natural Gas <URL> Distillate <URL> Unleaded <URL> Dec WTI/Brent Spread <URL> Dec Heat Crack <URL> Dec Gas Crack <URL> Dec/May Heat Spread <URL> Jan/Feb Heat Spread <URL> Nov Gas/Heat Spread <URL> Dec Gas/Heat Spread <URL> Nov/Mar Unlead Spread <URL> Nat Gas Strip Matrix <URL> Nat Gas Spread Matrix <URL> Crude and Products Spread Matrix <URL> Scott Oblander 312-762-1015 312-762-1014 fax Carr Futures 150 S. Wacker Suite 1500 Chicago, IL 60606		
-----------------------------------	--	--	--

Selanjutnya, dilakukan analisis distribusi kelas untuk mengetahui proporsi jumlah email spam dan non-spam dalam dataset yang digunakan. Analisis ini bertujuan untuk memahami karakteristik awal serta mengidentifikasi potensi ketidakseimbangan kelas yang dapat memengaruhi kinerja model klasifikasi. Berdasarkan hasil perhitungan, diketahui bahwa jumlah non-spam (label 0) sebanyak 445.090 data, sedangkan email spam (label 1) sebanyak 2.327 data. Proporsi tersebut menunjukkan bahwa kelas non-spam mencapai sekitar 99,48% dan kelas spam hanya sekitar 0,52% dari keseluruhan data

Distribusi kelas tersebut kemudian divisualisasikan menggunakan diagram batang yang memperlihatkan perbedaan jumlah data yang sangat signifikan antara kedua kelas, sebagaimana ditunjukkan pada Gambar 4.1. Kondisi ini menunjukkan adanya ketidakseimbangan kelas (*class imbalance*) dengan dominasi yang sangat tinggi pada kelas non-spam. Oleh karena itu, diperlukan teknik penanganan ketidakseimbangan data pada tahap selanjutnya agar model dapat bekerja secara lebih seimbang dan efektif.



Gambar 4.1 Distribusi Kelas

4.1.2 *Preprocessing* Teks

Tahap *preprocessing* teks dilakukan untuk membersihkan dan menyiapkan data teks email agar dapat diolah secara optimal pada tahap selanjutnya. Proses *preprocessing* diawali dengan *lowercasing* dan *cleansing*, yaitu mengubah seluruh huruf pada teks email menjadi huruf kecil serta menghapus karakter yang tidak relevan seperti angka, tanda baca, simbol khusus, dan karakter non-alfabet. Langkah ini bertujuan untuk menyeragamkan representasi kata dan mengurangi *noise* pada data teks. Contoh hasil penerapan *lowercasing* dan *cleansing* pada data email ditampilkan pada Tabel 4.2.

Tabel 4.2 Contoh Hasil *Lowercasing* dan *Cleansing*

Sebelum	Sesudah
ALL daily charts and matrices as hot links 10/17-resend ALL daily charts and matrices as hot links 10/17-resend The information contained herein is based on sources that we believe to be reliable, but we do not represent that it is accurate or complete. Nothing contained herein should be considered as an offer to sell or a solicitation of an offer to buy any financial instruments discussed herein. Any opinions expressed herein are solely those of the author. As such, they may differ in	all daily chart and matrix a hot link resend all daily chart and matrix a hot link resend the information contain herein be base on source that we believe to be reliable but we do not represent that it be accurate or complete nothing contain herein should be consider a an offer to sell or a solicitation of an offer to buy any financial instrument discuss herein any opinion express herein be solely those of the author a such they may differ in material respect from those of

material respects from those of, or expressed or published by on behalf of Carr Futures or its officers, directors, employees or affiliates. 2001 Carr Futures The charts are now available on the web by clicking on the hot link(s) contained in this email. If for any reason you are unable to receive the charts via the web, please contact me via email and I will email the charts to you as attachments. Crude <URL> Natural Gas <URL> Distillate <URL> Unleaded <URL> Dec WTI/Brent Spread <URL> Dec Heat Crack <URL> Dec Gas Crack <URL> Dec/May Heat Spread <URL> Jan/Feb Heat Spread <URL> Nov Gas/Heat Spread <URL> Dec Gas/Heat Spread <URL> Nov/Mar Unlead Spread <URL> Nat Gas Strip Matrix <URL> Nat Gas Spread Matrix <URL> Crude and Products Spread Matrix <URL> Scott Oblander 312-762-1015 312-762-1014 fax Carr Futures 150 S.	or express or publish by on behalf of carr future or it officer director employee or affiliate carr future the chart be now available on the web by click on the hot link s contain in this email if for any reason you be unable to receive the chart via the web please contact me via email and i will email the chart to you a attachment crude natural gas distillate unleaded dec wti brent spread dec heat crack dec gas crack dec may heat spread jan feb heat spread nov gas heat spread dec gas heat spread nov mar unlead spread nat gas strip matrix nat gas spread matrix crude and product spread matrix scott oblander fax carr future s wacker suite chicago il
---	--

Wacker Suite 1500 Chicago, IL 60606 soblander@carrfut.com	
--	--

Setelah proses *lowercasing* dan *cleansing*, tahap selanjutnya adalah *tokenization*, yaitu memecah teks email menjadi unit-unit kata atau token berdasarkan spasi atau pemisah tertentu. Proses ini memungkinkan setiap kata diperlakukan sebagai fitur yang berdiri sendiri dalam analisis teks. Hasil tokenisasi memberikan representasi awal struktur kata dalam setiap dokumen email. Contoh hasil tokenisasi pada data email disajikan pada Tabel 4.3.

Tabel 4.3 Contoh Hasil *Tokenization*

Sebelum	Sesudah
all daily chart and matrix a hot link resend all daily chart and matrix a hot link resend the information contain herein be base on source that we believe to be reliable but we do not represent that it be accurate or complete nothing contain herein should be consider a an offer to sell or a solicitation of an offer to buy any financial instrument discuss herein any opinion express herein be solely those of the author a such they may differ in material respect from those of	['all', 'daily', 'charts', 'and', 'matrices', 'as', 'hot', 'links', 'resend', 'all', 'daily', 'charts', 'and', 'matrices', 'as', 'hot', 'links', 'resend', 'the', 'information', 'contained', 'herein', 'is', 'based', 'on', 'sources', 'that', 'we', 'believe', 'to', 'be', 'reliable', 'but', 'we', 'do', 'not', 'represent', 'that', 'it', 'is', 'accurate', 'or', 'complete', 'nothing', 'contained', 'herein', 'should', 'be', 'considered', 'as', 'an', 'offer', 'to', 'sell', 'or', 'a', 'solicitation', 'of', 'an', 'offer', 'to', 'buy', 'any', 'financial', 'instruments',

or express or publish by on behalf of carr future or it officer director employee or affiliate carr future the chart be now available on the web by click on the hot link s contain in this email if for any reason you be unable to receive the chart via the web please contact me via email and i will email the chart to you a attachment crude natural gas distillate unleaded dec wti brent spread dec heat crack dec gas crack dec may heat spread jan feb heat spread nov gas heat spread dec gas heat spread nov mar unlead spread nat gas strip matrix nat gas spread matrix crude and product spread matrix scott oblender fax carr future s wacker suite chicago il	'discussed', 'herein', 'any', 'opinions', 'expressed', 'herein', 'are', 'solely', 'those', 'of', 'the', 'author', 'as', 'such', 'they', 'may', 'differ', 'in', 'material', 'respects', 'from', 'those', 'of', 'or', 'expressed', 'or', 'published', 'by', 'on', 'behalf', 'of', 'carr', 'futures', 'or', 'its', 'officers', 'directors', 'employees', 'or', 'affiliates', 'carr', 'futures', 'the', 'charts', 'are', 'now', 'available', 'on', 'the', 'web', 'by', 'clicking', 'on', 'the', 'hot', 'link', 's', 'contained', 'in', 'this', 'email', 'if', 'for', 'any', 'reason', 'you', 'are', 'unable', 'to', 'receive', 'the', 'charts', 'via', 'the', 'web', 'please', 'contact', 'me', 'via', 'email', 'and', 'i', 'will', 'email', 'the', 'charts', 'to', 'you', 'as', 'attachments', 'crude', 'natural', 'gas', 'distillate', 'unleaded', 'dec', 'wti', 'brent', 'spread', 'dec', 'heat', 'crack', 'dec', 'gas', 'crack', 'dec', 'may', 'heat', 'spread', 'jan', 'feb', 'heat', 'spread', 'nov', 'gas', 'heat', 'spread', 'dec', 'gas', 'heat', 'spread', 'nov', 'mar', 'unlead', 'spread', 'nat', 'gas', 'strip',
---	---

	'matrix', 'nat', 'gas', 'spread', 'matrix', 'crude', 'and', 'products', 'spread', 'matrix', 'scott', 'oblender', 'fax', 'carr', 'futures', 's', 'wacker', 'suite', 'chicago', 'il']
--	---

Tahap berikutnya adalah *stopwords removal*, yaitu proses menghapus kata-kata umum yang sering muncul namun tidak memiliki makna diskriminatif dalam proses klasifikasi, seperti kata penghubung dan kata depan. *Stopwords removal* bertujuan untuk mengurangi dimensi fitur serta meningkatkan fokus model pada kata-kata yang lebih informatif. Hasil penghilangan *stopwords* pada data email ditampilkan pada Tabel 4.4.

Tabel 4.4 Contoh Hasil *Stopwords Removal*

Sebelum	Sesudah
['all', 'daily', 'charts', 'and', 'matrices', 'as', 'hot', 'links', 'resend', 'all', 'daily', 'charts', 'and', 'matrices', 'as', 'hot', 'links', 'resend', 'the', 'information', 'contained', 'herein', 'is', 'based', 'on', 'sources', 'that', 'we', 'believe', 'to', 'be', 'reliable', 'but', 'we', 'do', 'not', 'represent', 'that', 'it', 'is', 'accurate', 'or', 'complete', 'nothing', 'contained', 'herein', 'should', 'be', 'considered', 'as',	['daily', 'charts', 'matrices', 'hot', 'links', 'resend', 'daily', 'charts', 'matrices', 'hot', 'links', 'resend', 'information', 'contained', 'herein', 'based', 'sources', 'believe', 'reliable', 'represent', 'accurate', 'complete', 'nothing', 'contained', 'herein', 'considered', 'offer', 'sell', 'solicitation', 'offer', 'buy', 'financial', 'instruments', 'discussed', 'herein', 'opinions', 'expressed',

'an', 'offer', 'to', 'sell', 'or', 'a', 'solicitation', 'of', 'an', 'offer', 'to', 'buy', 'any', 'financial', 'instruments', 'discussed', 'herein', 'any', 'opinions', 'expressed', 'herein', 'are', 'solely', 'those', 'of', 'the', 'author', 'as', 'such', 'they', 'may', 'differ', 'in', 'material', 'respects', 'from', 'those', 'of', 'or', 'expressed', 'or', 'published', 'by', 'on', 'behalf', 'of', 'carr', 'futures', 'or', 'its', 'officers', 'directors', 'employees', 'or', 'affiliates', 'carr', 'futures', 'the', 'charts', 'are', 'now', 'available', 'on', 'the', 'web', 'by', 'clicking', 'on', 'the', 'hot', 'link', 's', 'contained', 'in', 'this', 'email', 'if', 'for', 'any', 'reason', 'you', 'are', 'unable', 'to', 'receive', 'the', 'charts', 'via', 'the', 'web', 'please', 'contact', 'me', 'via', 'email', 'and', 'i', 'will', 'email', 'the', 'charts', 'to', 'you', 'as', 'attachments', 'crude', 'natural', 'gas', 'distillate', 'unleaded', 'dec', 'wti', 'brent', 'spread', 'dec', 'heat', 'crack', 'dec', 'gas', 'crack', 'dec', 'may', 'heat', 'spread', 'jan', 'feb', 'heat',	'herein', 'solely', 'author', 'may', 'differ', 'material', 'respects', 'expressed', 'published', 'behalf', 'carr', 'futures', 'officers', 'directors', 'employees', 'affiliates', 'carr', 'futures', 'charts', 'available', 'web', 'clicking', 'hot', 'link', 'contained', 'email', 'reason', 'unable', 'receive', 'charts', 'via', 'web', 'please', 'contact', 'via', 'email', 'email', 'charts', 'attachments', 'crude', 'natural', 'gas', 'distillate', 'unleaded', 'dec', 'wti', 'brent', 'spread', 'dec', 'heat', 'crack', 'dec', 'gas', 'crack', 'dec', 'may', 'heat', 'spread', 'jan', 'feb', 'heat', 'spread', 'nov', 'gas', 'heat', 'spread', 'dec', 'gas', 'heat', 'spread', 'nov', 'mar', 'unlead', 'spread', 'nat', 'gas', 'strip', 'matrix', 'nat', 'gas', 'spread', 'matrix', 'crude', 'products', 'spread', 'matrix', 'scott', 'oblander', 'fax', 'carr', 'futures', 'wacker', 'suite', 'chicago', 'il']
---	--

'spread', 'nov', 'gas', 'heat', 'spread', 'dec', 'gas', 'heat', 'spread', 'nov', 'mar', 'unlead', 'spread', 'nat', 'gas', 'strip', 'matrix', 'nat', 'gas', 'spread', 'matrix', 'crude', 'and', 'products', 'spread', 'matrix', 'scott', 'oblender', 'fax', 'carr', 'futures', 's', 'wacker', 'suite', 'chicago', 'il']	
---	--

Tahap terakhir dalam *preprocessing* teks adalah *lemmatization*, yaitu proses mengubah kata ke bentuk dasar atau lema tanpa mengubah makna kata tersebut. Proses ini bertujuan untuk menyatukan variasi bentuk kata sehingga kata-kata yang memiliki akar makna yang sama dapat direpresentasikan secara konsisten. Dengan demikian, jumlah fitur yang dihasilkan menjadi lebih terkontrol dan representatif. Contoh hasil *lemmatization* pada data email disajikan pada Tabel 4.5.

Tabel 4.5 Contoh Hasil *Lemmatization*

Sebelum	Sesudah
['daily', 'charts', 'matrices', 'hot', 'links', 'resend', 'daily', 'charts', 'matrices', 'hot', 'links', 'resend', 'information', 'contained', 'herein', 'based', 'sources', 'believe', 'reliable', 'represent', 'accurate', 'complete', 'nothing',	daily chart matrix hot link resend daily chart matrix hot link resend information contain herein base source believe reliable represent accurate complete nothing contain herein consider offer sell solicitation offer

'contained', 'herein', 'considered', 'offer', 'sell', 'solicitation', 'offer', 'buy', 'financial', 'instruments', 'discussed', 'herein', 'opinions', 'expressed', 'herein', 'solely', 'author', 'may', 'differ', 'material', 'respects', 'expressed', 'published', 'behalf', 'carr', 'futures', 'officers', 'directors', 'employees', 'affiliates', 'carr', 'futures', 'charts', 'available', 'web', 'clicking', 'hot', 'link', 'contained', 'email', 'reason', 'unable', 'receive', 'charts', 'via', 'web', 'please', 'contact', 'via', 'email', 'email', 'charts', 'attachments', 'crude', 'natural', 'gas', 'distillate', 'unleaded', 'dec', 'wti', 'brent', 'spread', 'dec', 'heat', 'crack', 'dec', 'gas', 'crack', 'dec', 'may', 'heat', 'spread', 'jan', 'feb', 'heat', 'spread', 'nov', 'gas', 'heat', 'spread', 'dec', 'gas', 'heat', 'spread', 'nov', 'mar', 'unlead', 'spread', 'nat', 'gas', 'strip', 'matrix', 'nat', 'gas', 'spread', 'matrix', 'crude', 'products', 'spread', 'matrix', 'scott',	buy financial instrument discuss herein opinion express herein solely author may differ material respect express publish behalf carr future officer director employee affiliate carr future chart available web click hot link contain email reason unable receive chart via web please contact via email email chart attachment crude natural gas distillate unleaded dec wti brent spread dec heat crack dec gas crack dec may heat spread jan feb heat spread nov gas heat spread dec gas heat spread nov mar unlead spread nat gas strip matrix nat gas spread matrix crude product spread matrix scott oblander fax carr future wacker suite chicago il
---	---

'oblender', 'fax', 'carr', 'futures', 'wacker', 'suite', 'chicago', 'il']	
--	--

4.1.3 Split Data

Pada tahap ini, dataset dibagi menjadi data latih (*training data*) dan data uji (*testing data*) dengan rasio 80:20. Dari total 244.387 data email, sebanyak 195.510 data digunakan sebagai data latih, sedangkan 48.877 data digunakan sebagai data uji. Pembagian ini bertujuan untuk memastikan bahwa model memiliki jumlah data yang cukup untuk proses pembelajaran sekaligus dapat diuji secara objektif pada data yang belum pernah dilihat sebelumnya.

Distribusi label pada data latih menunjukkan bahwa kelas non-spam (label 0) berjumlah 194.110 data, sedangkan kelas spam (label 1) berjumlah 1.400 data. Sementara itu, pada data uji, jumlah data non-spam tercatat sebanyak 48.527 data dan data spam sebanyak 350 data. Distribusi ini menunjukkan bahwa kondisi ketidakseimbangan kelas (*class imbalance*) tetap dipertahankan pada data latih dan data uji, sehingga mencerminkan kondisi data asli dan memungkinkan evaluasi model dilakukan secara lebih realistis terhadap performa klasifikasi email spam.

4.1.4 Ekstraksi Fitur TF-IDF

Pada tahap ekstraksi fitur, data teks yang telah melalui proses *preprocessing* selanjutnya direpresentasikan ke dalam bentuk numerik menggunakan metode *Term Frequency–Inverse Document Frequency* (TF-IDF). Proses ini bertujuan untuk memberikan bobot pada setiap kata berdasarkan tingkat kemunculannya dalam dokumen serta tingkat keunikannya di seluruh dokumen. Untuk mempermudah pemahaman terhadap alur perhitungan yang dilakukan,

penelitian ini menggunakan delapan data sebagai sampel agar proses perhitungan sampel disajikan secara manual dan lebih rinci. Tabel 4.6 menampilkan contoh sampel yang digunakan sebagai dasar dalam proses perhitungan. Sampel tersebut merupakan data awal yang telah melalui tahap *preprocessing* sehingga siap digunakan untuk proses pembobotan. Tabel ini memberikan gambaran awal mengenai kata-kata yang akan dianalisis serta menjadi dasar dalam proses perhitungan TF-IDF dan pada tahap-tahap selanjutnya.

Tabel 4.6 Kalimat Sampel

Dokumen	Kalimat
D1	all daily chart and matrix a hot link resend all daily chart and matrix a hot link resend the information contain herein be base on source that we believe to be reliable but we do not represent that it be accurate or complete nothing contain herein should be consider a an offer to sell or a solicitation of an offer to buy any financial instrument discuss herein any opinion express herein be solely those of the author a such they may differ in material respect from those of or express or publish by on behalf of carr future or it officer director employee or affiliate carr future the chart be now available on the web by click on the hot link s contain in this email if for any reason you be unable to receive the chart via the web please contact me via email and i will email the chart to you a attachment crude natural gas distillate unleaded dec wti brent spread dec heat crack dec gas crack dec may heat spread jan feb heat spread nov gas heat spread dec gas heat spread nov mar unlead spread nat gas strip

	matrix nat gas spread matrix crude and product spread matrix scott oblander fax carr future s wacker suite chicago il
D2	reserve release dpr reserve release dpr the mm profit in power canada on friday turn out to be release of reserve a well so total reserve release be mm v the we knew of in frevert meeting gas ee canada power day p l so day day loss w o reserve release qtd p l so qtd profit w o reserve release the gas power reserve release leaf mm in north america s schedule c d although most of that be specific deal reserve Cassandra
D3	new power product new power product ice friday october intercontinentalexchange launch hourly power well nd rd th week strip hourly power offered al west power hub pjw east new market manually added portfolio nd rd th week strip available power hub currently available exchange automatically added portfolio containing next day strip pa rticular hub credit must set financial power credit filter pjw hour ly power credit filter west hourly market hourly ower market arranged chronological order portfolio editor window please call ice representative assistance needed question concern directed hour help desk at
D4	got debt tsomk got debt tsomk got debt help using debt consolidation owe usd consolidate debt payment let u handle rest nice worry fee instead half dozen think home need another mortgage credit check required approval within business day available u citizen free obligation consultation please fill form return u paying bill chore life

	easy simple possible take advantage great offer please note deal unsecured debt mortgage car loan etc cannot apply please enter appropriate work phone secondary cell number full name address city state zip code home phone work phone best time call mail address estimated debt size thank receive offer company regarding matter matter please reply mail word remove subject line tz qu
D5	financial tool today economy financial tool today economy image image image image dear jennifer arnold like many u economic security important ever quicken deluxe delivers new feature designed help continue working oward financial goal current economy know busy designed easy learn time saving use upgrade toda shipping handling free plus save reg ular price special previous user mail rebate coupon included software shipment order today go ken com saving enter preferred customer <i>priority</i> code billing shipping page new account management feature help ou build financial cushion careful monitoring income expense help increase saving new automatic categorization improves tracking efficiency categorizing transaction automatically enter run even accurate report analyze spending pattern new step step budgeting tool quickly create budget based past transaction build one scra tch help control household spending feature like improved one step update automatic reconcile online accoun t save time give fast accurate read financial situation new tax tool feature new tax law updated th year tax rate important upgrade tax rate earlier version quicken course obsolete due new tax

	la w saving money tax depends careful planning da te tool quicken includes latest rate support new law new tax implication view give insight different inve stment holding period could impact tax improved capital ga in estimator cover loss carryovers employee stock option purch ase plan improved tax forecasting tax withholding estimator trans fer quicken data turbotax included assess asset mix take action new po rtfolio analyzer help evaluate portfolio stand teps could take optimize get helpful advice act holding performance asset allocation one easy step also quicken deluxe make easier set inves tment account support decimal based pricing includes improved supp ort bond short sale alternative stock plus find performance indicator portfolio view one click acc es investment research well valuable investment alert via il wireless web quicken deluxe brings together l need respond changing economic landscape wait use preferred customer <i>priority</i> code order online get quicken deluxe window previous user mail bate order today shipping handling free place order oday enter preferred customer pri ority code billing shipping page sincerely uicken team rebate previous quicken user complete rebate etails found clicking link offer valid nove mber note term condition pricing special offer featur e service option subject change without notice internet access required online feature service fee may apply online banking nd online bill pay subject application approval service fee may vary participating financial institution intuit inc quicken intuit registered
--	--

	trademark intuit inc quicken com trademark intuit inc trademark sole proper ty respective owner need help placing order call monday friday pm pacific time menti <i>priority</i> code image image important intuit respect personal nature mai communication every effort made offer information may value business wish receive marketing mail intuit future please click would like change mail address database please click mail sent following address dlxf ima ge
D6	password changed password changed yahoo id nymexdesk password account recently changed need anything message simply notification protect security account please note new password may take activate work first try please try later reply message help contact support see always change password signing yahoo service clicking account info link choosing change password cannot find account info link find change password link following way yahoo yahoo chat click account information upper right corner yahoo mail click option left column next screen select user information main screen yahoo finance click yahoo view yahoo page click account information upper right corner
D7	welcome to the texas military force museum ezboard online community welcome to the texas military force museum ezboard online community congratulation jreitme you be now a register member of the texas military force museum ezboard online community here be local account information login name jreitme password aggieswin a a local user your account will only work on

	this particular ezboard you may use the follow link to get there a a register ezboard user we invite you to join our community support community csc program csc be a low cost way in which you can support your community and ezboard in return for your support we d like to give you a personal no ad account no ezboard sponsor advertisement wherever you go on the site we ll also give you a unique community supporter icon and enhance customer service at only for six month this be your opportunity to show the world how much your community mean to you by become a csc member to learn more please visit our csc info page here please keep this email for your file a it contains your account information
D8	daily loss for ena and ee daily loss for ena and ee due to the release of schedule c for ena and ee the attach spreadsheet provide the trading p l without prudency and the associate daily loss limit notification additionally the spreadsheet reflect the net effect on the total trading line if the corp var remains at mm then a daily loss limit notification would be sent out however the current opinion be that the corp var be mm include discretionary var which eliminates the notification please let me know if you have any quetions thanks chris abel

Sebelum dilakukan pembobotan fitur menggunakan TF-IDF, terlebih dahulu dihitung frekuensi kemunculan kata pada setiap dokumen sebagai dasar pembentukan representasi numerik. Frekuensi kemunculan kata diperoleh dengan menghitung jumlah kemunculan setiap kata fitur pada masing-masing dokumen

sampel setelah data melalui tahap *preprocessing*. Hasil frekuensi kemunculan kata pada dokumen sampel disajikan pada Tabel 4.7.

Tabel 4.7 Frekuensi Kemunculan Kata pada Dokumen

Dokumen	Power	Spread	Yahoo	Debt	Password	Account
D1	0	8	0	0	0	0
D2	3	0	0	0	0	0
D3	8	0	0	0	0	0
D4	0	0	0	7	0	0
D5	0	0	0	0	0	2
D6	0	0	8	0	7	6
D7	0	0	0	0	1	4
D8	0	2	0	0	0	0

Tabel 4.7 menunjukkan representasi frekuensi kemunculan kata pada dokumen yang dianalisis. Setiap baris pada tabel merepresentasikan dokumen, sedangkan setiap kolom merepresentasikan kata fitur yang diamati. Nilai pada setiap sel menunjukkan jumlah kemunculan kata fitur tersebut pada dokumen terkait. Dengan demikian, tabel ini memperlihatkan bahwa kemunculan kata pada dokumen bersifat bervariasi dan tidak semua kata muncul pada setiap dokumen, sehingga pola kemunculan kata dapat menjadi pembeda karakteristik antar dokumen. Setelah frekuensi kemunculan kata pada masing-masing dokumen diketahui pada Tabel 4.7, nilai tersebut kemudian dinyatakan dalam bentuk bobot yang lebih representatif melalui perhitungan Term Frequency (TF). TF digunakan untuk menunjukkan tingkat kepentingan suatu kata di dalam dokumen dengan mempertimbangkan proporsi kemunculannya. Perhitungan TF mengacu pada Persamaan (2.1). Hasil perhitungan TF untuk setiap kata pada dokumen sampel disajikan pada Tabel 4.8.

Tabel 4.8 Hasil Perhitungan TF

Dokumen	Power	Spread	Yahoo	Debt	Password	Account
D1	0	0,06061	0	0	0	0
D2	0,03529	0	0	0	0	0
D3	0,08696	0	0	0	0	0
D4	0	0	0	0,06087	0	0
D5	0	0	0	0	0	0,00424
D6	0	0	0,08247	0	0,07216	0,06186
D7	0	0	0	0	0,00532	0,02128
D8	0	0,0202	0	0	0	0

Berdasarkan Tabel 4.8, terlihat bahwa nilai TF berada pada rentang 0 hingga mendekati 0,1. Nilai 0 menandakan kata tidak muncul pada dokumen tersebut, sedangkan nilai TF yang lebih besar menunjukkan kata tersebut relatif lebih dominan dibanding kata lain pada dokumen yang sama. Sebagai contoh, dokumen D3 memiliki nilai TF tertinggi pada kata *Power* sebesar 0,08696, sedangkan dokumen D1 menonjol pada kata *Spread* sebesar 0,06061. Dengan demikian, TF memberikan gambaran tingkat kepentingan kata pada level dokumen, namun belum mempertimbangkan apakah kata tersebut umum atau langka pada keseluruhan dokumen. Oleh karena itu, diperlukan perhitungan *Inverse Document Frequency* (IDF) untuk menilai tingkat keunikan kata pada kumpulan dokumen dengan menggunakan Persamaan (2.2), yang hasilnya disajikan pada Tabel 4.9.

Tabel 4.9 Hasil Perhitungan IDF

Kata	DF	IDF
Power	2	0,60206
Spread	2	0,60206
Yahoo	1	0,90309
Debt	1	0,90309
Password	2	0,60206
Account	3	0,42597

Berdasarkan Tabel 4.9, terlihat bahwa nilai IDF dipengaruhi langsung oleh nilai DF, yaitu jumlah dokumen yang memuat suatu kata. Kata *Yahoo* dan *Debt* memiliki $DF = 1$ sehingga menghasilkan nilai IDF paling tinggi sebesar 0,90309, yang menunjukkan bahwa kedua kata tersebut relatif lebih unik karena hanya muncul pada satu dokumen. Sementara itu, Kata *Account* memiliki $DF = 3$ dan nilai IDF paling kecil sebesar 0,42597, yang berarti kata ini paling umum dibanding kata fitur lain karena muncul pada lebih banyak dokumen. Nilai TF pada Tabel 4.8 kemudian dikalikan dengan nilai IDF pada Tabel 4.9 menggunakan Persamaan (2.3) untuk memperoleh bobot TF-IDF yang merepresentasikan kepentingan kata secara lebih menyeluruh. Hasil perhitungan TF-IDF disajikan pada Tabel 4.10.

Tabel 4.10 Hasil Perhitungan TF-IDF

Dokumen	Power	Spread	Yahoo	Debt	Password	Account
D1	0	0,03649	0	0	0	0
D2	0,02125	0	0	0	0	0
D3	0,05235	0	0	0	0	0
D4	0	0	0	0,05497	0	0
D5	0	0	0	0	0	0,0018
D6	0	0	0,07448	0	0,04345	0,02635
D7	0	0	0	0	0,0032	0,00906
D8	0	0,01216	0	0	0	0

Berdasarkan Tabel 4.10, terlihat bahwa apabila nilai TF-IDF menjadi 0 artinya suatu kata tidak muncul pada dokumen tertentu, sedangkan nilai yang lebih besar menunjukkan bahwa kata tersebut tidak hanya muncul pada dokumen tersebut, tetapi juga tidak terlalu umum pada dokumen lain. Sebagai contoh, kata *Yahoo* pada dokumen D6 memiliki bobot TF-IDF 0,07448, yang mencerminkan bahwa kata tersebut cukup dominan di dokumen D6 dan memiliki tingkat keunikan yang tinggi berdasarkan nilai IDF sebelumnya. Sementara itu, kata yang

lebih umum seperti *Account* memiliki bobot TF-IDF yang relatif lebih kecil pada dokumen yang memuatnya, karena nilai IDF-nya lebih rendah akibat kemunculannya pada lebih banyak dokumen. Dengan demikian, hasil pada Tabel 4.10 menunjukkan bahwa TF-IDF mampu merepresentasikan dokumen dalam bentuk numerik dengan menekankan kata-kata yang paling informatif, yaitu kata yang sering muncul pada dokumen tertentu namun tidak terlalu sering muncul pada dokumen lain. Hasil perhitungan TF-IDF ini kemudian digunakan pada tahap selanjutnya.

4.1.5 *Chi-Square Feature Selection*

Pada tahap ini dilakukan proses seleksi fitur menggunakan metode *Chi-Square* (χ^2) untuk mengetahui tingkat keterkaitan antara setiap kata dengan kelas dokumen. Metode ini digunakan untuk memilih fitur yang paling relevan sehingga dapat meningkatkan kinerja proses klasifikasi. Perhitungan nilai metode *Chi-Square* (χ^2) dilakukan dengan menggunakan Persamaan (2.4) untuk memperoleh nilai χ^2_{hitung} pada masing-masing kata atau fitur. Hasil perhitungan χ^2_{hitung} disajikan pada table Tabel 4.11.

Tabel 4.11 Hasil Perhitungan *Chi-Square Feature Selection*

Kata	A	B	E	G	D	χ^2_{hitung}
<i>Power</i>	0	2	4	2	8	2,666666667
<i>Spread</i>	0	2	4	2	8	2,666666667
<i>Yahoo</i>	1	0	3	4	8	1,142857143
<i>Debt</i>	1	0	3	4	8	1,142857143
<i>Password</i>	2	0	2	4	8	2,666666667
<i>Account</i>	3	0	1	4	8	4,8

Pada Tabel 4.11, setiap baris merepresentasikan kata atau fitur yang dianalisis, seperti *Power*, *Spread*, *Yahoo*, *Debt*, *Password*, dan *Account*. Kolom A, B, E, dan G menunjukkan nilai-nilai frekuensi pada tabel kontingensi yang

digunakan dalam perhitungan *Chi-Square*, yaitu jumlah dokumen pada masing-masing kombinasi kemunculan kata dan kelas dokumen. Setiap kata atau fitur dihitung nilai keterkaitannya terhadap kelas dokumen menggunakan Persamaan (2.4), sehingga perhitungannya seperti di bawah ini:

$$\chi^2(power) = \frac{8(0 \times 2 - 2 \times 4)^2}{(0+4)(2+2)(0+2)(4+2)} = 2,66666667$$

$$\chi^2(spread) = \frac{8(0 \times 2 - 2 \times 4)^2}{(0+4)(2+2)(0+2)(4+2)} = 2,66666667$$

$$\chi^2(yahoo) = \frac{8(1 \times 4 - 0 \times 3)^2}{(1+3)(0+4)(1+0)(3+4)} = 1,14285714$$

$$\chi^2(debt) = \frac{8(1 \times 4 - 0 \times 3)^2}{(1+3)(0+4)(1+0)(3+4)} = 1,14285714$$

$$\chi^2(password) = \frac{8(2 \times 4 - 0 \times 2)^2}{(2+2)(0+4)(2+0)(2+4)} = 2,66666667$$

$$\chi^2(account) = \frac{8(3 \times 4 - 0 \times 1)^2}{(3+1)(0+4)(3+0)(1+4)} = 4,8$$

Selanjutnya, untuk menentukan keputusan seleksi fitur, nilai χ^2_{hitung} perlu dibandingkan dengan nilai acuan χ^2_{tabel} , sehingga terlebih dahulu ditentukan derajat kebebasan (Df) yang digunakan pada uji *Chi-Square*. Derajat kebebasan dihitung menggunakan Persamaan (2.5).

$$Df = (r - 1)(c - 1)$$

$$Df = (2 - 1)(2 - 1)$$

$$Df = 1$$

Penentuan derajat kebebasan ini diperlukan karena nilai χ^2_{tabel} bergantung pada Df dan taraf signifikansi yang digunakan. Berdasarkan $\alpha = 0,05$ dan $Df = 1$, diperoleh nilai $\chi^2_{tabel} = 3,841459$ sebagai nilai pembanding.

Berdasarkan nilai χ^2_{hitung} dan χ^2_{tabel} tersebut, dilakukan perbandingan untuk menentukan Keputusan uji pada masing-masing fitur. Kriteria yang

digunakan adalah jika $\chi^2_{hitung} > \chi^2_{tabel}$ maka H_0 ditolak, sedangkan jika $\chi^2_{hitung} \leq \chi^2_{tabel}$ maka H_0 gagal ditolak. Hasil keputusan uji seleksi fitur dengan menggunakan *Chi-Square* disajikan dalam Tabel 4.12.

Tabel 4.12 Keputusan Uji *Feature Selection Chi-Square*

Kata	χ^2_{hitung}	χ^2_{tabel}	Keputusan
<i>Power</i>	2,666666667	3,841459	Gagal Tolak H_0 , karena $2,666666667 < 3,841459$
<i>Spread</i>	2,666666667	3,841459	Gagal Tolak H_0 , karena $2,666666667 < 3,841459$
<i>Yahoo</i>	1,142857143	3,841459	Gagal Tolak H_0 , karena $1,142857143 < 3,841459$
<i>Debt</i>	1,142857143	3,841459	Gagal Tolak H_0 , karena $1,142857143 < 3,841459$
<i>Password</i>	2,666666667	3,841459	Gagal Tolak H_0 , karena $2,666666667 < 3,841459$
<i>Account</i>	4,8	3,841459	Tolak H_0 , karena $4,8 > 3,841459$

Hasil perbandingan menunjukkan bahwa kata *Power* (2,6667), *Spread* (2,6667), *Yahoo* (1,1429), *Debt* (1,1429), dan *Password* (2,6667) berada di bawah χ^2_{tabel} , sehingga semua kata tersebut dinyatakan gagal tolak H_0 . Sementara itu, kata *Account* memiliki χ^2_{hitung} sebesar 4,8 yang lebih besar dari χ^2_{tabel} sebesar 3,841459, sehingga kata tersebut dinyatakan tolak H_0 dan memiliki keterkaitan yang signifikan terhadap kelas dokumen.

4.1.6 LDW-SMOTE

LDW-SMOTE bekerja dengan memilih beberapa sampel dari kelas minoritas yang sudah direpresentasikan menjadi angka hasil TF-IDF. Misalnya, dipilih tiga sampel minoritas yaitu D5, D6, dan D7 berdasarkan dua fitur kata terpilih yaitu kata *password* dan *account*. Nilai TF-IDF untuk D5 adalah (0 dan 0,0018), untuk D6 adalah (0,04345 dan 0,02635), dan untuk D7 adalah (0,0032 dan 0,00906). Ketiga sampel ini kemudian digunakan sebagai titik awal untuk membuat sampel baru dengan menggunakan LDW-SMOTE.

Langkah awal dilakukan dengan menghitung jarak antar sampel minoritas menggunakan jarak *Euclidean* pada Persamaan (2.6). Hasil jarak menunjukkan adanya perbedaan kedekatan antar titik minoritas, sehingga dapat terlihat pasangan titik yang posisinya paling berdekatan dan titik yang cenderung lebih jauh dari sampel lainnya. Informasi jarak ini penting karena menjadi dasar pemilihan tetangga terdekat pada proses LDW-SMOTE. Perhitungannya seperti dibawah ini:

Jarak dari D5:

$$d(x_1, x_2) = \sqrt{(0 - 0,04345)^2 + (0,0018 - 0,02635)^2} = 0,0499$$

$$d(x_1, x_3) = \sqrt{(0 - 0,0032)^2 + (0,0018 - 0,00906)^2} = 0,0079$$

Jarak dari D6:

$$d(x_2, x_1) = \sqrt{(0,04345 - 0)^2 + (0,02635 - 0,0018)^2} = 0,0499$$

$$d(x_2, x_3) = \sqrt{(0,04345 - 0,0032)^2 + (0,02635 - 0,00906)^2} = 0,0438$$

Jarak dari D7:

$$d(x_3, x_1) = \sqrt{(0,0032 - 0)^2 + (0,00906 - 0,0018)^2} = 0,0079$$

$$d(x_3, x_2) = \sqrt{(0,0032 - 0,04345)^2 + (0,00906 - 0,02635)^2} = 0,0438$$

Setelah jarak antar titik diperoleh, penelitian ini menggunakan nilai $k = 2$ sehingga setiap sampel minoritas memiliki dua tetangga terdekat. Nilai rata-rata jarak (*average distance*) dihitung menggunakan Persamaan (2.7) untuk masing-masing sampel. Dari nilai ini terlihat bahwa sampel dengan *average distance* lebih kecil berada pada area yang lebih padat (lebih dekat dengan dua sampel lainnya), sedangkan sampel yang *average distance*-nya lebih besar menunjukkan posisi yang relatif lebih jarang. Perhitungannya seperti dibawah ini:

$$avg_dist(x_1) = \frac{0,0499 + 0,0079}{2} = 0,0289$$

$$avg_dist(x_2) = \frac{0,0499 + 0,0438}{2} = 0,0469$$

$$avg_dist(x_3) = \frac{0,0079 + 0,0438}{2} = 0,0259$$

Nilai *average distance* selanjutnya ditransformasikan menjadi *density* atau kepadatan lokal melalui Persamaan (2.8). Pada hasil ini, sampel yang sebelumnya memiliki *average distance* kecil akan menghasilkan *density* yang lebih tinggi (area lebih padat), sedangkan sampel dengan *average distance* besar akan menghasilkan *density* yang lebih rendah (area lebih jarang). Dengan demikian, tahap *density* memperjelas kondisi penyebaran data minoritas di sekitar masing-masing titik. Perhitungannya seperti dibawah ini:

$$density(x_1) = \frac{1}{1 + 0,0289} = 0,9719$$

$$density(x_2) = \frac{1}{1 + 0,0469} = 0,9552$$

$$density(x_3) = \frac{1}{1 + 0,0259} = 0,0259$$

Untuk melihat tingkat kelangkaan sampel minoritas, dilakukan perhitungan *rarity* sebagai kebalikan dari *density* sesuai Persamaan (2.9). Hasil *rarity* menunjukkan sampel mana yang paling jarang (*rarity* terbesar) dan sampel mana yang paling padat (*rarity* terkecil). Sampel dengan *rarity* terbesar inilah yang menjadi perhatian utama karena representasinya paling rendah dibandingkan sampel minoritas lainnya. Perhitungannya seperti dibawah ini:

$$rarity(x_1) = \frac{1}{0,9719} = 1,0289$$

$$rarity(x_2) = \frac{1}{0,9552} = 1,0469$$

$$rarity(x_2) = \frac{1}{0,0259} = 1,0259$$

$$\text{Jumlah } rarity = 1,0289 + 1,0469 + 1,0259 = 3,1017$$

Nilai *rarity* kemudian dinormalisasi menjadi bobot *oversampling* menggunakan Persamaan (2.10). Bobot ini menggambarkan kontribusi setiap titik minoritas dalam pembuatan data sintetis. Semakin besar bobot yang diperoleh suatu sampel, semakin besar pula porsi pembuatan sampel sintetis yang berasal dari area titik tersebut, sehingga distribusi minoritas menjadi lebih merata.

$$w_1 = \frac{1,0289}{3,1017} = 0,3317$$

$$w_2 = \frac{1,0469}{3,1017} = 0,3375$$

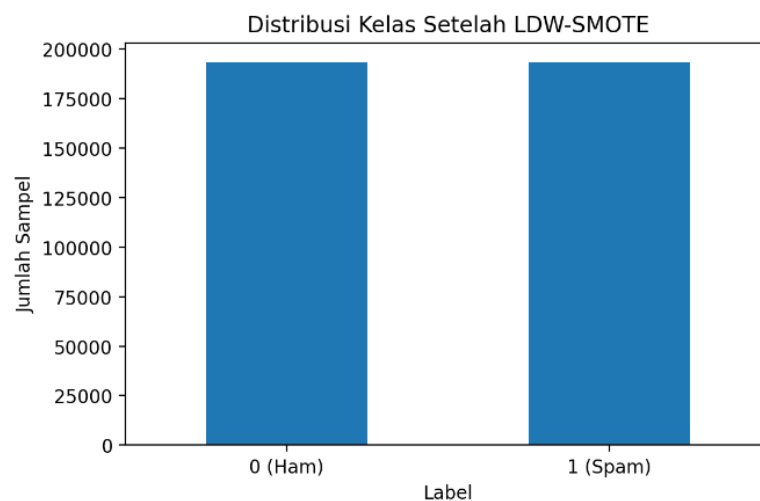
$$w_3 = \frac{1,0259}{3,1017} = 0,3308$$

Tahap akhir adalah pembentukan sampel sintetis menggunakan interpolasi linier berbobot sesuai Persamaan (2.11). Pada proses ini dipilih titik minoritas yang paling jarang (memiliki *rarity* terbesar), yaitu (0,04345 dan 0,02635), kemudian dipilih salah satu tetangga terdekatnya, yaitu (0,0032 dan 0,00906). Selanjutnya digunakan nilai delta sebesar 0,5 sebagai faktor interpolasi, sehingga titik sintetis dibentuk di antara kedua titik tersebut.

$$\begin{aligned} x_{new} &= x_2 + (\delta)(w_2)(x_3 - x_2) \\ &= (0,04345 ; 0,02635) + (0,5)(0,3375)(0,0032 - 0,04345 ; 0,00906 - 0,02635) \\ &= (0,04345 ; 0,02635) + (0,1688)(-0,0403 ; -0,0173) \end{aligned}$$

$$\begin{aligned}
&= (0,04345 ; 0,02635) + (-0,0068; -0,0029) \\
&= (0,04345 + (-0,0068); 0,02635 + (-0,0029)) \\
&= (0,0367 ; 0,0234)
\end{aligned}$$

Berdasarkan hasil perhitungan interpolasi berbobot pada tahap LDW-SMOTE, diperoleh satu sampel sintetis baru dengan nilai fitur sebesar (0.0367 dan 0.0234). Nilai ini berada di antara titik minoritas yang paling jarang dan tetangga terdekatnya, sehingga menunjukkan bahwa proses pembuatan data sintetis tidak menghasilkan data yang menyimpang dari karakteristik kelas minoritas. Sampel sintetis yang terbentuk tetap merepresentasikan pola distribusi data minoritas, namun memiliki variasi baru yang dapat memperkaya sebaran data latih.



Gambar 4.2 Diagram Distribusi Kelas Setelah LDW-SMOTE

Berdasarkan Gambar 4.2, terlihat bahwa proses penambahan sampel sintetis membuat jumlah data pada masing-masing kelas menjadi setara, ditunjukkan oleh tinggi batang label 0 (non-spam) dan label 1 (spam) yang sama. Pada kondisi awal, kelas minoritas spam hanya berjumlah 1.400 sampel, sedangkan kelas mayoritas non-spam mencapai 194.110 sampel, sehingga

ketimpangan distribusi sangat besar dan berpotensi menyebabkan model lebih condong memprediksi kelas mayoritas. Setelah penerapan LDW-SMOTE, jumlah sampel pada kelas spam meningkat hingga 194.110, sehingga distribusi kedua kelas menjadi seimbang. Hal ini menunjukkan bahwa LDW-SMOTE mampu menambah data minoritas secara adaptif dengan mempertimbangkan kondisi lokal tiap sampel, sehingga diharapkan dapat mengurangi bias kelas dan meningkatkan kinerja model klasifikasi pada tahap selanjutnya.

4.1.7 Pelatihan Model *Multinomial Naïve Bayes*

Tahap pelatihan model dilakukan menggunakan *Multinomial Naïve Bayes* untuk mengklasifikasikan email ke dalam kelas spam dan non-spam. Pada perhitungan manual ini digunakan delapan dokumen dengan enam fitur kata, yaitu *Power*, *Spread*, *Yahoo*, *Debt*, *Password*, dan *Account*. Proses pelatihan disusun bertahap agar terlihat jelas alur probabilitas mulai dari peluang awal kelas hingga keputusan kelas akhir. Perhitungan mengacu pada persamaan *Multinomial Naïve Bayes* pada bab sebelumnya, terutama persamaan *prior probability*, *likelihood*, dan *posterior probability*.

Nilai *prior probability* pada data latih dihitung menggunakan Persamaan (2.18) dan menunjukkan peluang awal masing-masing kelas sebelum mempertimbangkan kata yang muncul. Berdasarkan data sampel, dokumen non-spam terdiri dari D1, D2, D3, dan D8 sedangkan dokumen spam terdiri dari D4, D5, D6 dan D7.

$$P(spam) = \frac{4}{8} = \frac{1}{2} = 0,5$$

$$P(non\ spam) = \frac{4}{8} = \frac{1}{2} = 0,5$$

Hasilnya menunjukkan bahwa *prior* untuk spam dan non-spam sama besar karena masing-masing memiliki 4 dokumen dari total 8 dokumen. Dengan demikian, nilai *prior* spam dan non-spam pada hasil ini seimbang, sehingga perbedaan keputusan klasifikasi selanjutnya lebih dipengaruhi oleh nilai *likelihood* dan *posterior*.

Setelah *prior* diperoleh, tahap berikutnya adalah menghitung *likelihood* untuk melihat seberapa besar peluang kemunculan kata pada masing-masing kelas. *Likelihood* pada *Multinomial Naïve Bayes* dihitung berdasarkan frekuensi kemunculan kata pada dokumen-dokumen dalam kelas tertentu, sesuai Persamaan (2.19). Untuk itu, data frekuensi kemunculan kata pada setiap dokumen digunakan sebagai dasar perhitungan. Ringkasan frekuensi kata dan label kelas ditunjukkan pada tabel berikut agar proses penjumlahan setiap kelas dapat dilakukan secara sistematis.

Tabel 4.13 Frekuensi Kemunculan Kata pada Dokumen

Dokumen	Power	Spread	Yahoo	Debt	Password	Account
D1	0	8	0	0	0	0
D2	3	0	0	0	0	0
D3	8	0	0	0	0	0
D4	0	0	0	7	0	0
D5	0	0	0	0	0	2
D6	0	0	8	0	7	6
D7	0	0	0	0	1	4
D8	0	2	0	0	0	0

Berdasarkan Tabel 4.13, distribusi kemunculan kata pada tiap dokumen terlihat tidak merata, sehingga ketika data dikelompokkan ke kelas non-spam, total frekuensi kata setiap fitur menjadi pembeda utama antar kelas. Pada beberapa dokumen terdapat kata yang dominan, misalnya pada D6 kata *Yahoo* muncul 8 kali, kata *Password* muncul 7 kali dan kata *Account* muncul 6 kali, sedangkan pada D5 hanya terlihat *Account* muncul 2 kali. Perbedaan pola kemunculan ini

membuat total frekuensi kata pada kelas non-spam perlu dirangkum terlebih dahulu sebelum dihitung peluang kemunculannya. Hasil total frekuensi setiap kata pada kelas non-spam kemudian menjadi dasar untuk menghitung nilai *likelihood* pada tahap berikutnya.

Perhitungan *likelihood* diawali dengan menjumlahkan total frekuensi masing-masing kata pada kelas spam. Karena kelas spam pada sampel terdiri dari D4, D5, D6, dan D7, maka setiap kata dijumlahkan hanya dari empat dokumen tersebut. Hasil penjumlahan ini menghasilkan total kemunculan kata setiap fitur untuk kelas spam, sekaligus total seluruh kata pada kelas spam sebagai penyebut probabilitas. Nilai total inilah yang digunakan pada tahap berikutnya untuk menghitung peluang $P(\text{kata}|\text{spam})$ dengan smoothing.

$$\text{power} = 0 + 0 + 0 + 0 = 0$$

$$\text{spread} = 0 + 0 + 0 + 0 = 0$$

$$\text{yahoo} = 0 + 0 + 8 + 0 = 8$$

$$\text{debt} = 7 + 0 + 0 + 0 = 7$$

$$\text{password} = 0 + 0 + 7 + 1 = 8$$

$$\text{account} = 0 + 2 + 6 + 4 = 12$$

$$\text{total kata spam} = 0 + 0 + 8 + 7 + 8 + 12 = 35$$

Langkah yang sama dilakukan untuk kelas non-spam dengan menjumlahkan total frekuensi kata pada D1, D2, D3, dan D8. Penjumlahan dilakukan setiap fitur agar terlihat kata mana yang paling sering muncul pada kelas non-spam dan mana yang jarang muncul. Selain total setiap fitur, dihitung juga total seluruh kata pada kelas non-spam sebagai nilai pembagi dalam perhitungan

likelihood. Dengan demikian, peluang $P(\text{kata}|\text{non-spam})$ dapat dihitung secara konsisten menggunakan Persamaan (2.19).

$$\text{power} = 0 + 3 + 8 + 0 = 11$$

$$\text{spread} = 8 + 0 + 0 + 2 = 10$$

$$\text{yahoo} = 0 + 0 + 0 + 0 = 0$$

$$\text{debt} = 0 + 0 + 0 + 0 = 0$$

$$\text{password} = 0 + 0 + 0 + 0 = 0$$

$$\text{account} = 0 + 0 + 0 + 0 = 0$$

$$\text{total kata non-spam} = 11 + 10 + 0 + 0 + 0 + 0 = 21$$

Setelah total frekuensi kata pada masing-masing kelas diperoleh, *likelihood* dihitung menggunakan Persamaan (2.19) yang memasukkan konsep *Laplace smoothing (add-one)*. *Smoothing* diperlukan agar kata yang tidak muncul pada suatu kelas tidak menghasilkan probabilitas nol, karena hal tersebut dapat membuat hasil *posterior* langsung menjadi nol saat dikalikan. Dalam perhitungan ini, setiap frekuensi kata ditambah 1, kemudian dibagi dengan total seluruh kata pada kelas tersebut ditambah jumlah kata unik pada *vocabulary*. Hasil *likelihood* yang diperoleh kemudian menjadi komponen utama dalam menghitung *posterior* pada langkah berikutnya.

Kelas spam:

$$P(\text{power}|\text{spam}) = \frac{0 + 1}{35 + 6} = 0,0243$$

$$P(\text{spread}|\text{spam}) = \frac{0 + 1}{35 + 6} = 0,0243$$

$$P(\text{yahoo}|\text{spam}) = \frac{8 + 1}{35 + 6} = 0,2195$$

$$P(debt|spam) = \frac{7 + 1}{35 + 6} = 0,1951$$

$$P(password|spam) = \frac{8 + 1}{35 + 6} = 0,2195$$

$$P(account|spam) = \frac{12 + 1}{35 + 6} = 0,3170$$

Kelas non-spam:

$$P(power|non spam) = \frac{11 + 1}{21 + 6} = 0,4444$$

$$P(spread|non spam) = \frac{10 + 1}{21 + 6} = 0,4074$$

$$P(yahoo|non spam) = \frac{0 + 1}{21 + 6} = 0,0370$$

$$P(debt|non spam) = \frac{0 + 1}{21 + 6} = 0,0370$$

$$P(password|non spam) = \frac{0 + 1}{21 + 6} = 0,0370$$

$$P(account|non spam) = \frac{0 + 1}{21 + 6} = 0,0370$$

Hasil likelihood memperlihatkan bahwa perbedaan frekuensi kemunculan kata pada data latih benar-benar memengaruhi besarnya peluang kata pada masing-masing kelas. Kata yang lebih sering muncul pada dokumen tertentu akan mendorong peluang kelas yang sejenis menjadi lebih besar ketika kata tersebut muncul pada dokumen uji. Sebaliknya, kata yang jarang muncul akan menghasilkan peluang yang kecil, sehingga kontribusinya terhadap skor akhir juga lebih lemah. Ringkasan ini kemudian digunakan untuk menyusun perkalian peluang kata-kata pada dokumen uji ketika masuk ke tahap *posterior probability*.

Tahap *posterior probability* menghitung skor peluang kelas dengan menggabungkan *prior* dan *likelihood* sesuai Persamaan (2.15) pada *Multinomial*

Naïve Bayes. Pada tahap *posterior probability*, dokumen uji diasumsikan mengandung kata *Yahoo* dan *Account*, sehingga perhitungan hanya memanfaatkan *likelihood* kedua kata tersebut pada masing-masing kelas. Skor *posterior* untuk setiap kelas diperoleh dari penggabungan *prior* kelas dan *likelihood* kata-kata yang muncul, sehingga menghasilkan dua skor yang bisa langsung dibandingkan.

$$\begin{aligned} P(\text{spam}|x) &= P(\text{spam}) \times P(\text{yahoo}|\text{spam}) \times P(\text{account}|\text{spam}) \\ &= 0,5 \times 0,2195 \times 0,3170 \\ &= 0,0348 \end{aligned}$$

$$\begin{aligned} P(\text{non spam}|x) &= P(\text{non spam}) \times P(\text{yahoo}|\text{non spam}) \\ &\quad \times P(\text{account}|\text{non spam}) \\ &= 0,5 \times 0,058824 \times 0,176471 \\ &= 0,005197 \end{aligned}$$

Hasil perhitungan menunjukkan skor untuk kelas spam lebih besar dibanding kelas non-spam pada kasus dokumen yang memuat kata *Yahoo* dan *Account*. Oleh karena itu, dokumen uji tersebut diprediksi termasuk ke dalam kelas spam, sesuai kesimpulan pada bagian akhir perhitungan.

Hasil perhitungan manual pada pembahasan sebelumnya dilakukan menggunakan sejumlah kecil data sebagai sampel untuk memperlihatkan secara rinci proses kerja model *Multinomial Naïve Bayes*. Perhitungan tersebut bertujuan sebagai ilustrasi agar mekanisme pelatihan dan penentuan kelas dapat dipahami dengan jelas. Namun, hasil dari perhitungan manual tersebut belum dapat menggambarkan kinerja model secara menyeluruh karena hanya menggunakan data dalam jumlah terbatas. Oleh karena itu, diperlukan pengujian lanjutan menggunakan data yang lebih besar agar diperoleh hasil evaluasi yang lebih akurat dan representatif. Untuk memperoleh hasil evaluasi yang representatif,

selanjutnya dilakukan pengujian menggunakan metode *K-Fold Cross Validation* pada data *training* yang sebenarnya, yang berasal dari keseluruhan dataset penelitian. Hasil pengujian tersebut disajikan pada Tabel 4.14 sampai dengan Tabel 4.17.

Tabel 4.14 Hasil *Cross Validation* MNB tanpa *Chi-Square*

K	Akurasi	Presisi		Recall		F1-score		Training Time (s)
		0	1	0	1	0	1	
1	0,9914	0,9938	0,2977	0,9976	0,1392	0,9957	0,1897	0,0878
2	0,9917	0,9939	0,3333	0,9977	0,1535	0,9958	0,2102	0,0825
3	0,9910	0,9936	0,2481	0,9973	0,1214	0,9955	0,1630	0,0890
4	0,9913	0,9936	0,2540	0,9976	0,1107	0,9956	0,1542	0,0848
5	0,9917	0,9939	0,3384	0,9977	0,1571	0,9958	0,2146	0,0841
Rata Rata	0,9914	0,9938	0,2943	0,9976	0,1364	0,9957	0,1863	0,0856

Berdasarkan Tabel 4.14, rata-rata akurasi yang diperoleh sebesar 0,9914. Nilai ini menunjukkan model memiliki ketepatan prediksi yang sangat tinggi secara keseluruhan, terutama karena kontribusi kelas non-spam (kelas 0) yang dominan. Hal tersebut terlihat dari metrik kelas 0 yang sangat tinggi, yaitu presisi sebesar 0,9938, *recall* sebesar 0,9976, dan *f1-score* sebesar 0,9957. Namun, kinerja pada kelas spam (kelas 1) masih relatif rendah, ditunjukkan oleh presisi sebesar 0,2943, *recall* sebesar 0,1364, dan *f1-score* sebesar 0,1863. Nilai *recall* yang rendah menunjukkan bahwa sebagian besar email spam belum berhasil terdeteksi dengan baik oleh model, sehingga model cenderung lebih banyak memprediksi email sebagai non-spam.. Dari sisi waktu komputasi, waktu tercepat terdapat pada *fold* ke-2 yaitu sebesar 0,0825 detik, sedangkan waktu terlama terdapat pada *fold* ke-3 yaitu sebesar 0,0890 detik. Perbedaan waktu ini relatif kecil, sehingga menunjukkan bahwa proses pelatihan model memiliki waktu komputasi yang stabil pada setiap *fold*.

Tabel 4.15 Hasil *Cross Validation* MNB dengan *Chi-Square*

K	Akurasi	Presisi		Recall		F1-score		Training Time (s)
		0	1	0	1	0	1	
1	0,9918	0,9936	0,3177	0,9981	0,1214	0,9959	0,1757	0,0543
2	0,9918	0,9937	0,3180	0,9980	0,1241	0,9558	0,1785	0,0443
3	0,9918	0,9937	0,3279	0,9981	0,1258	0,9959	0,1819	0,0376
4	0,9918	0,9936	0,3171	0,9981	0,1223	0,9958	0,1765	0,0406
5	0,9920	0,9937	0,3438	0,9982	0,1267	0,9959	0,1852	0,0420
Rata Rata	0,9918	0,9937	0,3249	0,9981	0,1241	0,9879	0,1796	0,0438

Selanjutnya, berdasarkan Tabel 4.15 rata-rata akurasi meningkat menjadi 0,9918 dibandingkan skenario tanpa *Chi-Square*. Penerapan *Chi-Square* memberikan peningkatan pada presisi kelas spam (kelas 1) yang mencapai 0,3249, dibandingkan skenario tanpa *Chi-Square*. Hal ini menunjukkan bahwa ketika model memprediksi suatu email sebagai spam, tingkat ketepatan prediksi menjadi lebih baik sehingga kesalahan *false positive* relatif lebih terkendali. Namun, *recall* spam masih rendah yaitu 0,1241. Nilai tersebut menunjukkan bahwa sebagian besar email spam masih belum berhasil terdeteksi oleh model. Kondisi tersebut berdampak pada *f1-score* spam yang hanya mencapai 0,1796, sehingga secara umum *Chi-Square* lebih menonjol dalam meningkatkan ketepatan prediksi spam (presisi), tetapi belum mampu meningkatkan kemampuan model dalam mendeteksi lebih banyak email spam (*recall*) pada data yang tidak seimbang. Dari sisi waktu komputasi, proses pelatihan model menjadi lebih efisien dibandingkan skenario tanpa *Chi-Square*. Waktu komputasi tercepat terdapat pada *fold* ke-3 sebesar 0,0376 detik, sedangkan waktu terlama terdapat pada *fold* ke-1 sebesar 0,0543 detik. Perbedaan waktu tersebut relatif kecil sehingga menunjukkan kestabilan waktu komputasi pada setiap *fold*. Selain itu, reduksi fitur menggunakan *Chi-Square* juga membantu mempercepat proses pelatihan model karena jumlah fitur yang diproses menjadi lebih sedikit.

Tabel 4.16 Hasil *Cross Validation* MNB dengan *Chi-Square* & LDW-SMOTE

K	Akurasi	Presisi		Recall		F1-score		Training Time (s)
		0	1	0	1	0	1	
1	0,9341	0,9693	0,9038	0,8966	0,9716	0,9315	0,9365	0,1200
2	0,9329	0,9699	0,9013	0,8936	0,9722	0,9302	0,9354	0,1000
3	0,9342	0,9683	0,9048	0,8979	0,9706	0,9318	0,9365	0,1100
4	0,9331	0,9694	0,9691	0,8944	0,9718	0,9304	0,9356	0,1000
5	0,9349	0,9691	0,9054	0,8985	0,9713	0,9324	0,9372	0,1100
Rata Rata	0,9338	0,9692	0,9169	0,8962	0,9715	0,9313	0,9362	0,1080

Pada Tabel 4.16 pola performa berubah signifikan dibandingkan skenario tanpa penyeimbangan data. Rata-rata akurasi turun menjadi 0,9338, namun kemampuan menangkap spam meningkat sangat signifikan, ditunjukkan oleh *recall* spam (kelas 1) sebesar 0,9715 dan *f1-score* spam sebesar 0,9362. Peningkatan ini menunjukkan bahwa penerapan LDW-SMOTE berhasil membuat model lebih sensitif terhadap kelas minoritas, sehingga sebagian besar email spam yang sebelumnya tidak terdeteksi dapat dikenali dengan baik pada hampir seluruh *fold* pengujian.. Akan tetapi, terdapat *trade-off* pada kelas non-spam (kelas 0), di mana *recall* non-spam turun menjadi 0,8962, yang menandakan meningkatnya kemungkinan non-spam terdeteksi sebagai spam (*false positive*). Hal ini menunjukkan bahwa peningkatan kemampuan deteksi spam diperoleh dengan konsekuensi menurunnya performa klasifikasi pada kelas non-spam.

Dari sisi waktu komputasi, penerapan LDW-SMOTE menyebabkan waktu pelatihan menjadi lebih lama dibandingkan model tanpa *oversampling*. Waktu komputasi tercepat berada pada *fold* ke-2 dan ke-4 sebesar 0,1000 detik, sedangkan waktu terlama terdapat pada *fold* ke-1 sebesar 0,1200 detik. Peningkatan waktu komputasi ini disebabkan oleh penambahan data sintetis pada proses *oversampling*, sehingga jumlah data pelatihan yang diproses model

menjadi lebih besar. Namun demikian, peningkatan waktu tersebut sebanding dengan peningkatan kemampuan model dalam mendeteksi email spam.

Tabel 4.17 Hasil *Cross Validation* MNB dengan LDW-SMOTE

K	Akurasi	Presisi		Recall		F1-score		Training Time (s)
		0	1	0	1	0	1	
1	0,9347	0,9551	0,9160	0,9122	0,9571	0,9332	0,9361	0,1500
2	0,9342	0,9529	0,9171	0,9137	0,9548	0,9329	0,9548	0,1400
3	0,9373	0,9547	0,9212	0,9182	0,9564	0,9361	0,9385	0,1400
4	0,9334	0,9523	0,9159	0,9124	0,9543	0,9319	0,9347	0,1500
5	0,9364	0,9530	0,9209	0,9180	0,9548	0,9352	0,9375	0,1400
Rata Rata	0,9352	0,9536	0,9182	0,9149	0,9555	0,9339	0,9403	0,1440

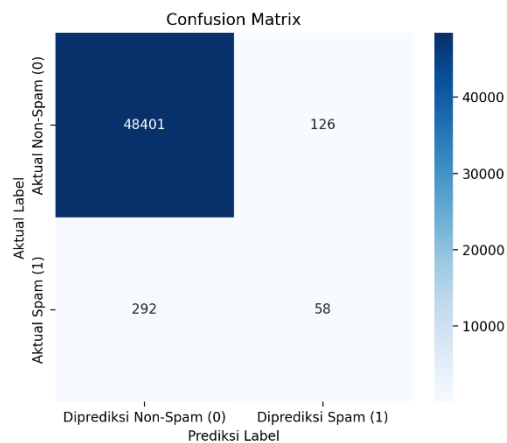
Adapun Tabel 4.17 menunjukkan hasil performa model *Multinomial Naïve Bayes* dengan penerapan LDW-SMOTE tanpa *Chi-Square*. Jika dibandingkan dengan model awal tanpa LDW-SMOTE dan tanpa *Chi-Square*, terlihat bahwa *recall* spam meningkat sangat drastis dari 0,1364 menjadi 0,9555, meskipun akurasi menurun dari 0,9914 menjadi 0,9352. Penurunan ini menunjukkan adanya *trade-off* akibat penyeimbangan data, di mana model menjadi lebih sensitif dalam mendeteksi spam tetapi meningkatkan kemungkinan email non-spam terdeteksi sebagai spam (*false positive*). Selain itu, waktu komputasi pada skenario ini juga lebih lama, yaitu berada pada rentang 0,1400 hingga 0,1500 detik, dibandingkan model awal yang hanya sekitar 0,0825 hingga 0,0890 detik. Hal ini menunjukkan bahwa penerapan LDW-SMOTE secara signifikan meningkatkan kemampuan deteksi spam, namun dengan konsekuensi penurunan akurasi dan peningkatan waktu komputasi.

4.1.8 Evaluasi Model

Pada tahap evaluasi, penelitian ini membandingkan beberapa skenario model untuk melihat pengaruh *Chi-Square Feature Selection* dan LDW-SMOTE

terhadap kinerja klasifikasi. Empat skenario yang digunakan meliputi MNB tanpa *Chi-Square*, MNB dengan *Chi-Square*, MNB dengan *Chi-Square* disertai LDW-SMOTE, serta MNB tanpa *Chi-Square* dengan penerapan LDW-SMOTE. Perbandingan dilakukan secara bertahap agar pengaruh masing-masing metode dapat terlihat dengan jelas, yaitu antara model dengan dan tanpa *Chi-Square*, kemudian antara model *Chi-Square* dengan dan tanpa LDW-SMOTE, serta antara model tanpa *Chi-Square* dengan dan tanpa LDW-SMOTE.

Dalam penyajian hasil evaluasi, digunakan pendekatan berbasis satu kali *running* pertama, di mana *confusion matrix* serta perhitungan manual metrik evaluasi yang disajikan secara rinci untuk memberikan gambaran yang lebih jelas dan terstruktur mengenai proses evaluasi model. Untuk skenario MNB tanpa *Chi-Square*, hasil prediksi model disajikan dalam bentuk *confusion matrix* pada Gambar 4.3.



Gambar 4.3 *Confusion Matrix* tanpa *Chi-Square*

Berdasarkan Gambar 4.3, nilai pada *confusion matrix* selanjutnya digunakan sebagai dasar perhitungan manual untuk memperoleh nilai akurasi, presisi, *recall*, dan *f1-score* sebagai berikut:

$$\text{Akurasi} = \frac{48401+58}{48401+58+126+292} = 0,9914$$

$$\text{Presisi kelas spam} = \frac{58}{58+126} = 0,3152$$

$$\text{Presisi kelas non-spam} = \frac{48401}{48401+292} = 0,9940$$

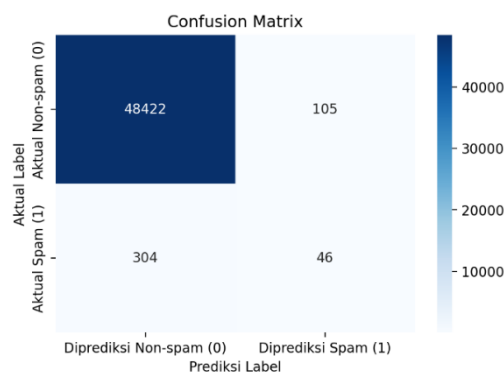
$$\text{Recall kelas spam} = \frac{58}{58+292} = 0,1657$$

$$\text{Recall kelas non-spam} = \frac{48401}{48401+126} = 0,9974$$

$$\text{F1-score kelas spam} = 2 \times \frac{0,3152 \times 0,1657}{0,3152 + 0,1657} = 0,2172$$

$$\text{F1-score kelas non-spam} = 2 \times \frac{0,9940 \times 0,9974}{0,9940 + 0,9974} = 0,9957$$

Selanjutnya, untuk skenario MNB dengan *Chi-Square* dan tanpa LDW-SMOTE, hasil prediksi model disajikan dalam bentuk *confusion matrix* pada Gambar 4.4.



Gambar 4.4 *Confusion Matrix* dengan *Chi-Square* tanpa LDW-SMOTE

Berdasarkan Gambar 4.4, nilai pada *confusion matrix* selanjutnya digunakan sebagai dasar perhitungan manual untuk memperoleh nilai akurasi, presisi, *recall*, dan *f1-score* sebagai berikut:

$$\text{Akurasi} = \frac{48422+46}{48422+46+304+105} = 0,9916$$

$$\text{Presisi kelas spam} = \frac{46}{46+105} = 0,3046$$

$$\text{Presisi kelas non-spam} = \frac{48422}{48422+304} = 0,9938$$

$$\text{Recall kelas spam} = \frac{46}{46+304} = 0,1314$$

$$\text{Recall kelas non-spam} = \frac{48422}{48422+105} = 0,9978$$

$$\text{F1-score kelas spam} = 2 \times \frac{0,3046 \times 0,1314}{0,3046 + 0,1314} = 0,1836$$

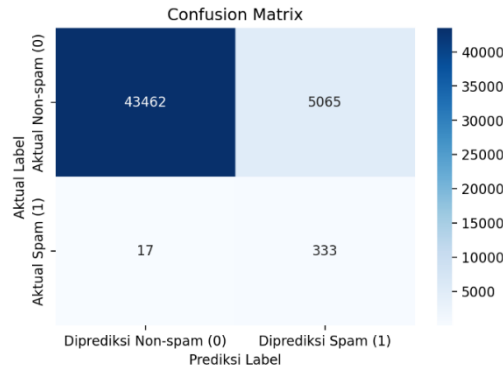
$$\text{F1-score kelas non-spam} = 2 \times \frac{0,9938 \times 0,9978}{0,9938 + 0,9978} = 0,9958$$

Berdasarkan hasil perhitungan di atas antara MNB dengan *Chi-Square* dan MNB tanpa *Chi-Square*, MNB dengan *Chi-Square* menunjukkan perbedaan performa dibandingkan model tanpa *Chi-Square*. Nilai akurasi meningkat dari 0,9914 menjadi 0,9916, yang menunjukkan adanya perbaikan ketepatan prediksi secara keseluruhan. Pada kelas spam, nilai presisi menurun sedikit dari 0,3152 menjadi 0,3046, sedangkan *recall* spam juga menurun dari 0,1657 menjadi 0,1314. Penurunan *recall* ini terjadi karena jumlah *true positive* menurun dari 58 menjadi 46 data dan *false negative* meningkat dari 292 menjadi 304 data. Kondisi tersebut menunjukkan bahwa model dengan *Chi-Square* menjadi lebih sulit mendeteksi email spam.

Namun demikian, jumlah *false positive* berhasil ditekan dari 126 menjadi 105 data dan *true negative* meningkat dari 48.401 menjadi 48.422 data. Hal ini menunjukkan bahwa penerapan *Chi-Square* membantu model mengurangi kesalahan dalam mengklasifikasikan email non-spam sebagai spam. Sementara itu, performa pada kelas non-spam tetap sangat baik dengan nilai *f1-score* sebesar 0,9957 pada model tanpa *Chi-Square* dan 0,9958 pada model dengan *Chi-Square*. Dengan demikian, penerapan *Chi-Square* memberikan peningkatan pada stabilitas

klasifikasi kelas non-spam dan efisiensi fitur, meskipun kemampuan deteksi spam masih relatif rendah pada kondisi data yang tidak seimbang.

Adapun untuk skenario MNB dengan *Chi-Square* dan LDW-SMOTE, hasil prediksi model disajikan dalam bentuk *confusion matrix* pada Gambar 4.5.



Gambar 4.5 *Confusion Matrix* dengan *Chi-Square* dan LDW-SMOTE

Berdasarkan Gambar 4.5, nilai pada *confusion matrix* selanjutnya digunakan sebagai dasar perhitungan manual untuk memperoleh nilai akurasi, presisi, *recall*, dan *f1-score* sebagai berikut:

$$\text{Akurasi} = \frac{43462+333}{43462+5065+17+333} = 0,8960$$

$$\text{Presisi kelas spam} = \frac{333}{333+5065} = 0,0617$$

$$\text{Presisi kelas non-spam} = \frac{43462}{43462+17} = 0,9996$$

$$\text{Recall kelas spam} = \frac{333}{333+17} = 0,9514$$

$$\text{Recall kelas non-spam} = \frac{43462}{43462+5065} = 0,8956$$

$$\text{F1-score kelas spam} = 2 \times \frac{0,0617 \times 0,9514}{0,0617 + 0,9514} = 0,1159$$

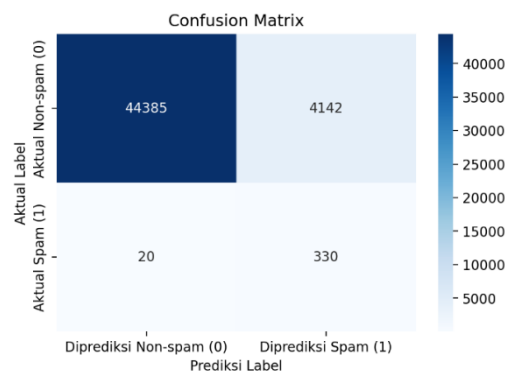
$$\text{F1-score kelas non-spam} = 2 \times \frac{0,9996 \times 0,8956}{0,9996 + 0,8956} = 0,9448$$

Selanjutnya, berdasarkan perhitungan di atas antara MNB *Chi-Square* dengan LDW-SMOTE dan MNB *Chi-Square* tanpa LDW-SMOTE, penerapan LDW-SMOTE pada model MNB *Chi-Square* memberikan perubahan performa yang sangat signifikan, terutama pada kemampuan mendeteksi kelas spam. Hal ini terlihat dari peningkatan *recall* spam dari 0,1314 menjadi 0,9514. Peningkatan tersebut menunjukkan bahwa sebagian besar email spam berhasil terdeteksi sehingga jumlah spam yang lolos (*false negative*) menurun secara drastis. Jika dikaitkan dengan *confusion matrix*, pada model tanpa LDW-SMOTE jumlah *true positive* sebesar 46 data dan *false negative* sebesar 304 data, sedangkan pada model dengan LDW-SMOTE jumlah *true positive* meningkat menjadi 333 data dan *false negative* menurun menjadi 17 data. Kondisi ini menunjukkan bahwa LDW-SMOTE sangat efektif dalam meningkatkan sensitivitas model terhadap kelas minoritas (spam).

Namun demikian, peningkatan kemampuan deteksi spam tersebut diikuti dengan penurunan presisi spam dari 0,3046 menjadi 0,0617. Penurunan ini disebabkan oleh meningkatnya jumlah false positive dari 105 data menjadi 5.065 data, yaitu email non-spam yang salah diklasifikasikan sebagai spam. Selain itu, jumlah true negative juga menurun dari 48.422 data menjadi 43.462 data, yang menunjukkan bahwa lebih banyak email non-spam ikut terdeteksi sebagai spam. Dampak lain yang terlihat adalah penurunan akurasi dari 0,9916 menjadi 0,8960, yang menunjukkan bahwa meskipun kemampuan deteksi spam meningkat, kesalahan klasifikasi secara keseluruhan juga meningkat akibat bertambahnya *false positive*.

Dengan demikian, penerapan LDW-SMOTE terbukti sangat efektif dalam meningkatkan kemampuan deteksi spam pada data tidak seimbang. Namun, peningkatan tersebut menghasilkan *trade-off* berupa menurunnya ketepatan klasifikasi pada kelas non-spam. Oleh karena itu, pemilihan skenario terbaik dapat disesuaikan dengan tujuan sistem, yaitu apabila fokus utama adalah meminimalkan spam yang lolos maka penggunaan LDW-SMOTE lebih diunggulkan, sedangkan apabila fokus utama adalah menjaga akurasi dan mengurangi false positive maka model tanpa LDW-SMOTE lebih sesuai.

Adapun untuk skenario MNB dengan LDW-SMOTE dan tanpa *Chi-Square*, hasil prediksi model disajikan dalam bentuk *confusion matrix* pada Gambar 4.6.



Gambar 4.6 *Confusion Matrix* dengan LDW-SMOTE

Berdasarkan Gambar 4.6, nilai pada *confusion matrix* selanjutnya digunakan sebagai dasar perhitungan manual untuk memperoleh nilai akurasi, presisi, *recall*, dan *f1-score* sebagai berikut:

$$\text{Akurasi} = \frac{44385 + 330}{44385 + 4142 + 20 + 330} = 0,9148$$

$$\text{Presisi kelas spam} = \frac{330}{330 + 4142} = 0,0738$$

$$\text{Presisi kelas non-spam} = \frac{44385}{44385+20} = 0,9995$$

$$\text{Recall kelas spam} = \frac{330}{330+20} = 0,9429$$

$$\text{Recall kelas non-spam} = \frac{44385}{44385+4142} = 0,9146$$

$$\text{F1-score kelas spam} = 2 \times \frac{0,0738 \times 0,9429}{0,0738 + 0,9429} = 0,1369$$

$$\text{F1-score kelas non-spam} = 2 \times \frac{0,9995 \times 0,9146}{0,9995 + 0,9146} = 0,9552$$

Berdasarkan perhitungan di atas antara MNB tanpa *Chi-Square* dengan dan tanpa LDW SMOTE, penerapan LDW-SMOTE memberikan perubahan performa yang signifikan, khususnya pada kemampuan mendeteksi kelas spam. Hal ini terlihat dari nilai *recall* spam yang meningkat drastis dari 0,1657 menjadi 0,9429, yang menunjukkan bahwa sebagian besar spam berhasil terdeteksi sehingga jumlah spam yang lolos (*false negative*) berkurang secara signifikan. Jika dikaitkan dengan *confusion matrix*, pada model tanpa LDW-SMOTE jumlah *true positive* sebesar 58 data dan *false negative* sebesar 292 data, sedangkan pada model dengan LDW-SMOTE jumlah *true positive* meningkat menjadi 330 data dan *false negative* menurun menjadi 20 data. Kondisi ini menunjukkan bahwa LDW-SMOTE sangat efektif dalam meningkatkan sensitivitas model terhadap kelas minoritas (spam).

Namun demikian, peningkatan kemampuan deteksi spam tersebut diikuti oleh penurunan presisi spam dari 0,3152 menjadi 0,0738. Penurunan ini disebabkan oleh meningkatnya jumlah *false positive* dari 126 data menjadi 4.142 data, yaitu email non-spam yang salah diklasifikasikan sebagai spam. Selain itu, jumlah *true negative* juga menurun dari 48.401 data menjadi 44.385 data, yang menunjukkan bahwa lebih banyak email non-spam ikut terdeteksi sebagai spam.

Dampak lainnya adalah penurunan akurasi dari 0,9914 menjadi 0,9148, yang menunjukkan bahwa meskipun kemampuan deteksi spam meningkat secara signifikan, jumlah kesalahan klasifikasi keseluruhan juga meningkat akibat bertambahnya *false positive*.

Selain menggunakan satu kali *running* sebagai dasar perhitungan manual, hasil evaluasi juga disajikan menggunakan pendekatan rata-rata dan standar deviasi dari lima kali percobaan (*running*). Pendekatan ini bertujuan untuk memberikan gambaran performa model yang lebih representatif serta mempertimbangkan variasi yang mungkin terjadi pada setiap percobaan, sehingga hasil evaluasi tidak bergantung pada satu kondisi tertentu saja.

Setelah diperoleh nilai akurasi, presisi, *recall*, dan *f1-score* melalui perhitungan manual pada masing-masing skenario, hasil tersebut kemudian dirangkum untuk memudahkan proses perbandingan antar model. Perbandingan ini diperlukan agar dampak penerapan *Chi-Square Feature Selection* dan LDW-SMOTE terhadap kinerja klasifikasi dapat terlihat secara lebih jelas pada metrik evaluasi yang sama. Oleh karena itu, ringkasan hasil evaluasi antara model dengan *Chi-Square* dan tanpa *Chi-Square* disajikan pada Tabel 4.18, sedangkan ringkasan perbandingan MNB *Chi-Square* antara model dengan LDW-SMOTE dan tanpa LDW-SMOTE disajikan pada Tabel 4.19. Selain itu, perbandingan model tanpa *Chi-Square* dengan dan tanpa LDW-SMOTE juga disajikan pada Tabel 4.20. Adapun hasil lengkap dari setiap *running* disajikan pada bagian Lampiran 3 hingga Lampiran 6.

Tabel 4.18 Hasil Perbandingan Evaluasi dengan *Chi-Square*

Hasil Evaluasi		MNB <i>Chi-Square</i>	MNB Tanpa <i>Chi-Square</i>
Akurasi		$0,9917 \pm 0,0002$	$0,9911 \pm 0,0003$
Presisi	spam	$0,3157 \pm 0,0356$	$0,2797 \pm 0,0292$
	non-spam	$0,9937 \pm 0,0001$	$0,9938 \pm 0,0002$
Recall	spam	$0,1274 \pm 0,0147$	$0,1480 \pm 0,0167$
	non-spam	$0,9979 \pm 0,0002$	$0,9972 \pm 0,0002$
F1-score	spam	$0,1814 \pm 0,0200$	$0,1935 \pm 0,0210$
	non-spam	$0,9958 \pm 0,0001$	$0,9955 \pm 0,0001$
Testing Time (s)		$0,0104 \pm 0,0020$	$0,0157 \pm 0,0050$

Berdasarkan Tabel 4.18, model *Multinomial Naïve Bayes* dengan *Chi-Square* menunjukkan perbedaan performa dibandingkan model tanpa *Chi-Square*. Nilai akurasi meningkat dari 0,9911 menjadi 0,9917, yang menunjukkan adanya peningkatan ketepatan prediksi secara keseluruhan. Pada kelas spam, nilai presisi meningkat dari 0,2797 menjadi 0,3157. Peningkatan tersebut menunjukkan bahwa model dengan *Chi-Square* lebih mampu mengurangi kesalahan dalam mengklasifikasikan email non-spam sebagai spam (*false positive*). Namun, *recall* spam menurun dari 0,1480 menjadi 0,1274, yang menunjukkan bahwa jumlah spam yang berhasil terdeteksi menjadi lebih sedikit sehingga masih terdapat cukup banyak spam yang tidak terklasifikasi dengan baik (*false negative*). Selain itu, nilai *f1-score* spam juga menurun dari 0,1935 menjadi 0,1814. Hal ini menunjukkan bahwa meskipun presisi meningkat, keseimbangan performa antara presisi dan *recall* pada kelas spam masih belum optimal.

Sementara itu, pada kelas non-spam performa model relatif stabil. Hal ini terlihat dari nilai presisi non-spam yang tetap sangat tinggi yaitu 0,9937 pada model dengan *Chi-Square* dan 0,9938 pada model tanpa *Chi-Square*, serta nilai *f1-score* non-spam yang tidak mengalami perubahan signifikan, yaitu 0,9958 dan 0,9955. Kondisi tersebut menunjukkan bahwa kemampuan model dalam

mengenali email non-spam tetap terjaga dengan baik. Dari sisi waktu komputasi, model dengan *Chi-Square* memiliki *testing time* sebesar 0,0104 detik, sedangkan model tanpa *Chi-Square* sebesar 0,0157 detik. Hal ini menunjukkan bahwa penerapan *Chi-Square* mampu meningkatkan efisiensi proses pengujian model. Pengurangan jumlah fitur melalui seleksi fitur menyebabkan data yang diproses menjadi lebih sederhana sehingga proses klasifikasi dapat dilakukan lebih cepat.

Tabel 4.19 Hasil Perbandingan dengan LDW-SMOTE

Hasil Evaluasi		MNB <i>Chi-Square</i> dengan LDW-SMOTE	MNB <i>Chi-Square</i> tanpa LDW-SMOTE
Akurasi		$0,8986 \pm 0,0024$	$0,9917 \pm 0,0002$
Presisi	spam	$0,0633 \pm 0,0017$	$0,3157 \pm 0,0356$
	non-spam	$0,9996 \pm 0,00005$	$0,9937 \pm 0,0001$
Recall	spam	$0,9421 \pm 0,0226$	$0,1274 \pm 0,0147$
	non-spam	$0,9098 \pm 0,0281$	$0,9979 \pm 0,0002$
<i>F1-score</i>	spam	$0,1188 \pm 0,0030$	$0,1814 \pm 0,0200$
	non-spam	$0,9462 \pm 0,0013$	$0,9958 \pm 0,0001$
<i>Testing Time (s)</i>		$0,0100 \pm 0,0000$	$0,0104 \pm 0,0020$

Selanjutnya, berdasarkan Tabel 4.19, penerapan LDW-SMOTE pada model MNB *Chi-Square* memberikan perubahan performa yang signifikan, terutama pada kemampuan mendeteksi kelas spam. Hal ini terlihat dari nilai *recall* spam yang meningkat drastis dari 0,1274 menjadi 0,9421, yang menunjukkan bahwa sebagian besar data spam berhasil terdeteksi. Namun, peningkatan tersebut diikuti oleh penurunan presisi spam dari 0,3157 menjadi 0,0633. Hal ini menunjukkan bahwa meskipun model menjadi lebih sensitif dalam mendeteksi spam, jumlah kesalahan dalam mengklasifikasikan email non-spam sebagai spam (*false positive*) juga meningkat secara signifikan.

Dampak lain yang terlihat adalah penurunan akurasi dari 0,9917 menjadi 0,8996, yang mengindikasikan bahwa secara keseluruhan tingkat ketepatan

prediksi model mengalami penurunan. Selain itu, pada kelas non-spam, nilai *recall* juga menurun dari 0,9979 menjadi 0,9098 dan nilai *f1-score* menurun dari 0,9958 menjadi 0,9462, yang menunjukkan bahwa performa model dalam mengenali email non-spam ikut terdampak. Dari sisi waktu komputasi, model dengan LDW-SMOTE memiliki *testing time* sebesar 0,0100 detik, sedangkan model tanpa LDW-SMOTE sebesar 0,0104 detik. Meskipun penerapan LDW-SMOTE umumnya menambah beban komputasi pada tahap pelatihan karena adanya penambahan data sintesis, pada tahap pengujian model waktu komputasi yang diperoleh relatif sedikit lebih cepat. Hal ini menunjukkan bahwa model yang telah dilatih menggunakan data yang lebih seimbang mampu melakukan proses klasifikasi secara lebih stabil pada data pengujian.

Tabel 4.20 Hasil Perbandingan tanpa *Chi-Square*

Hasil Evaluasi		MNB dengan LDW SMOTE	MNB tanpa LDW SMOTE
akurasi		0,9158 ± 0,0010	0,9911 ± 0,0003
presisi	spam	0,0744 ± 0,0007	0,2797 ± 0,0292
	non-spam	0,9995 ± 0,0001	0,9938 ± 0,0002
recall	spam	0,9423 ± 0,0118	0,1480 ± 0,0167
	non-spam	0,9155 ± 0,0008	0,9972 ± 0,0002
F1-score	spam	0,1380 ± 0,0013	0,1935 ± 0,0210
	non-spam	0,9557 ± 0,0004	0,9955 ± 0,0001
Testing Time (s)		0,0140 ± 0,0055	0,0157 ± 0,0050

Berdasarkan Tabel 4.20, penerapan LDW-SMOTE pada model *Multinomial Naive Bayes* (MNB) tanpa *Chi-Square* memberikan perubahan performa yang cukup signifikan, khususnya dalam mendeteksi kelas spam. Hal ini terlihat dari nilai *recall* spam yang meningkat drastis dari 0,1480 menjadi 0,9423, yang menunjukkan bahwa sebagian besar data spam berhasil terdeteksi sehingga

jumlah *false negative* berkurang secara signifikan. Namun, peningkatan kemampuan deteksi tersebut diikuti oleh penurunan presisi spam dari 0,2797 menjadi 0,0744. Hal ini mengindikasikan bahwa banyak data non-spam yang salah diklasifikasikan sebagai spam (*false positive*), sehingga ketepatan prediksi untuk kelas spam menjadi lebih rendah. Dampak lain dari kondisi ini juga terlihat pada penurunan nilai akurasi secara keseluruhan dari 0,9911 menjadi 0,9158, yang menunjukkan bahwa meskipun model menjadi lebih sensitif dalam mendeteksi spam, kesalahan klasifikasi total meningkat.

Selain itu, nilai *recall* untuk kelas non-spam mengalami penurunan dari 0,9972 menjadi 0,9155, yang menandakan bahwa sebagian data non-spam mulai salah terklasifikasi sebagai spam. Sebaliknya, presisi non-spam meningkat dari 0,9938 menjadi 0,9995, yang menunjukkan bahwa ketika model memprediksi non-spam, hasilnya cenderung sangat akurat. Dari sisi *f1-score*, nilai pada kelas spam menurun dari 0,1935 menjadi 0,1380, sedangkan untuk kelas non-spam juga mengalami penurunan dari 0,9955 menjadi 0,9557. Hal ini mencerminkan adanya *trade-off* antara peningkatan kemampuan deteksi spam dan penurunan performa pada kelas non-spam.

Dari sisi waktu komputasi, model dengan LDW-SMOTE memiliki *testing time* sebesar 0,0140 detik, sedangkan model tanpa LDW-SMOTE sebesar 0,0157 detik. Meskipun penerapan *oversampling* umumnya menambah beban komputasi pada tahap pelatihan karena adanya penambahan data sintesis, pada tahap pengujian model waktu komputasi yang diperoleh relatif sedikit lebih cepat. Hal ini menunjukkan bahwa model yang telah dilatih menggunakan data yang lebih seimbang mampu melakukan proses klasifikasi secara lebih stabil pada data

pengujian. Secara keseluruhan, LDW-SMOTE terbukti efektif dalam meningkatkan kemampuan model dalam mendeteksi spam, khususnya pada kelas minoritas. Selain itu, waktu pengujian model tetap relatif cepat dan stabil meskipun proses pelatihan melibatkan penambahan data sintetis. Hal ini menunjukkan bahwa LDW-SMOTE mampu membantu model mempelajari pola data spam dengan lebih baik tanpa memberikan peningkatan beban komputasi yang signifikan pada tahap pengujian. Namun, peningkatan kemampuan deteksi tersebut diikuti oleh meningkatnya kesalahan klasifikasi pada data non-spam serta penurunan akurasi total.

4.2 Pembahasan

Berdasarkan hasil penelitian, penerapan *Multinomial Naïve Bayes* dengan *Chi-Square Feature Selection* menunjukkan perbedaan performa dibandingkan tanpa *Chi-Square*. Hal ini ditunjukkan oleh nilai akurasi model MNB *Chi-Square* sebesar 0,9917, sedikit lebih tinggi dibandingkan MNB tanpa *Chi-Square* sebesar 0,9911. Selain itu, *Chi-Square* juga meningkatkan ketepatan prediksi kelas spam, terlihat dari presisi spam yang naik dari 0,2797 menjadi 0,3157. Peningkatan ini terjadi karena jumlah *false positive* berhasil ditekan, sehingga lebih sedikit email non-spam yang salah diklasifikasikan sebagai spam.

Namun demikian, peningkatan presisi tersebut belum diikuti oleh peningkatan kemampuan model dalam menangkap spam secara lebih luas. Pada kelas spam, nilai *recall* pada model *Chi-Square* adalah 0,1274, lebih rendah dibandingkan tanpa *Chi-Square* sebesar 0,1480. Hal ini menunjukkan bahwa jumlah spam yang berhasil terdeteksi masih relatif sedikit karena jumlah *true positive* menurun dan *false negative* meningkat. Dampaknya, nilai *f1-score* spam juga menurun dari 0,1935

menjadi 0,1814 karena *f1-score* dipengaruhi oleh keseimbangan antara presisi dan *recall*. Sementara itu, performa kelas non-spam tetap sangat tinggi pada kedua skenario, ditunjukkan oleh *f1-score* non-spam sebesar 0,9958 pada model *Chi-Square* dan 0,9955 pada model tanpa *Chi-Square*, yang didukung oleh tingginya kemampuan model dalam mengenali email non-spam.

Selanjutnya, penerapan LDW-SMOTE pada model MNB dengan *Chi-Square* memberikan perubahan performa yang sangat signifikan, terutama pada kemampuan mendeteksi spam. Nilai akurasi pada skenario ini sebesar 0,8986 lebih rendah dibandingkan model *Chi-Square* tanpa LDW-SMOTE sebesar 0,9917, namun kemampuan mendeteksi spam meningkat drastis. Hal ini terlihat dari nilai *recall* spam yang meningkat dari 0,1274 menjadi 0,9421, yang disebabkan oleh meningkatnya jumlah *true positive*, serta menurunnya jumlah *false negative*, sehingga hampir seluruh spam berhasil terdeteksi.

Meskipun demikian, peningkatan *recall* spam tersebut diikuti oleh penurunan presisi *spam* yang sangat tajam, yaitu dari 0,3157 menjadi 0,0633. Hal ini disebabkan oleh meningkatnya jumlah *false positive* secara signifikan, sehingga banyak email non-spam yang salah diklasifikasikan sebagai spam. Kondisi ini juga berdampak pada penurunan *recall* non-spam menjadi 0,9098, serta *f1-score* non-spam menjadi 0,9462.

Dari sisi waktu komputasi, model dengan LDW-SMOTE memiliki *testing time* sebesar 0,0100 detik, sedikit lebih cepat dibandingkan model tanpa LDW-SMOTE sebesar 0,0104 detik. Meskipun penerapan *oversampling* umumnya meningkatkan beban komputasi pada tahap pelatihan karena adanya penambahan data sintesis, pada tahap pengujian model waktu komputasi yang diperoleh tetap relatif cepat dan

stabil. Hal ini menunjukkan bahwa model yang telah dilatih menggunakan data yang lebih seimbang mampu melakukan proses klasifikasi secara efisien pada data pengujian.

Secara keseluruhan, hasil penelitian menunjukkan bahwa model terbaik dalam hal akurasi dan ketepatan klasifikasi adalah model *Multinomial Naïve Bayes* dengan *Chi-Square* tanpa LDW-SMOTE. Hal ini ditunjukkan oleh nilai akurasi yang tinggi serta presisi spam yang lebih baik dengan jumlah *false positive* yang lebih rendah. Sementara itu, model terbaik dalam hal kemampuan mendeteksi spam adalah model *Multinomial Naïve Bayes* dengan *Chi-Square* dan LDW-SMOTE. Hal ini terlihat dari nilai *recall* spam yang sangat tinggi, sehingga jumlah spam yang tidak terdeteksi (*false negative*) berhasil ditekan secara signifikan. Dari sisi waktu komputasi, model tercepat pada tahap pengujian adalah *Multinomial Naïve Bayes* dengan *Chi-Square* dan LDW-SMOTE dengan *testing time* sebesar 0,0100 detik. Hal ini menunjukkan bahwa meskipun LDW-SMOTE menambahkan data sintetis pada tahap pelatihan, proses klasifikasi pada tahap pengujian tetap dapat berjalan dengan cepat dan stabil.

Setelah memahami performa model *Multinomial Naïve Bayes*, selanjutnya dilakukan perbandingan dengan beberapa algoritma lain untuk melihat posisi kinerja model yang diusulkan secara lebih komprehensif. Perbandingan ini bertujuan untuk mengetahui keunggulan dan kelemahan masing-masing model dalam mendeteksi email spam. Pada penelitian ini, seluruh model yang dibandingkan diberikan perlakuan yang sama, yaitu menggunakan *Chi-Square Feature Selection* dan LDW-SMOTE. Dengan demikian, hasil evaluasi yang diperoleh dapat dibandingkan secara adil dan objektif.

Tabel 4.21 Hasil Perbandingan Model

Model	Akurasi	Presisi		Recall		F1-score		Testing Time (s)
		0	1	0	1	0	1	
<i>Multinomial Naïve Bayes</i>	0,8986 ± 0,0024	0,9996 ± 0,00005	0,0633 ± 0,0017	0,9098 ± 0,0281	0,9421 ± 0,0226	0,9462 ± 0,0013	0,1188 ± 0,0030	0,0100 ± 0,0000
<i>Bernoulli Naïve Bayes</i>	0,9284 ± 0,0027	0,9974 ± 0,0001	0,0647 ± 0,0013	0,9302 ± 0,0024	0,6697 ± 0,0154	0,9606 ± 0,0048	0,1180 ± 0,0021	0,0471 ± 0,0403
Regresi Logistik	0,9624 ± 0,0007	0,9993 ± 0,0001	0,1483 ± 0,0032	0,9626 ± 0,0007	0,9023 ± 0,0209	0,9806 ± 0,0003	0,2548 ± 0,0054	0,0111 ± 0,0019

Berdasarkan Tabel 4.21, skenario Multinomial Naïve Bayes dengan *Chi-Square* dan LDW-SMOTE tetap menunjukkan keunggulan utama pada kemampuan mendeteksi kelas spam. Hal ini terlihat dari nilai *recall* spam sebesar 0,9421, yang merupakan nilai tertinggi dibandingkan *Bernoulli Naïve Bayes* sebesar 0,6697 maupun Regresi Logistik sebesar 0,9023. Nilai *recall* yang lebih tinggi ini mengindikasikan bahwa model mampu mendeteksi sebagian besar email spam sehingga jumlah spam yang lolos sebagai non-spam dapat ditekan secara signifikan.

Namun demikian, pada metrik presisi spam, *Multinomial Naïve Bayes* memperoleh nilai sebesar 0,0633, yang sedikit lebih rendah dibandingkan *Bernoulli Naïve Bayes* sebesar 0,0647, serta jauh di bawah Regresi Logistik sebesar 0,1483. Hal ini menunjukkan bahwa meskipun model sangat baik dalam menangkap spam, namun masih banyak email non-spam yang ikut terklasifikasi sebagai spam.

Jika dibandingkan dengan Regresi Logistik, model tersebut menunjukkan performa yang lebih seimbang, dengan akurasi sebesar 0,9624 serta *f1-score* spam sebesar 0,2548 yang merupakan nilai tertinggi dibandingkan model lainnya. Sementara itu, *Bernoulli Naïve Bayes* memiliki akurasi sebesar 0,9284 dan *f1-score*

spam sebesar 0,1180, yang relatif medekati *Multinomial Naïve Bayes* namun dengan kemampuan deteksi spam *Bernoulli Naïve Bayes* yang masih lebih rendah.

Dari sisi waktu komputasi, model tercepat pada tahap pengujian adalah *Multinomial Naïve Bayes* dengan *testing time* sebesar 0,0100 detik, diikuti Regresi Logistik sebesar 0,0111 detik, sedangkan *Bernoulli Naïve Bayes* memiliki waktu pengujian paling lama yaitu 0,0471 detik. Hal ini menunjukkan bahwa *Multinomial Naïve Bayes* tidak hanya unggul dalam kemampuan mendeteksi spam, tetapi juga memiliki efisiensi waktu pengujian yang lebih baik dibandingkan model lainnya.

4.3 Klasifikasi Email Spam dalam Perspektif Islam

Kasus spam pada media komunikasi digital, khususnya email, merupakan permasalahan yang tidak hanya bersifat teknis, tetapi juga berdampak pada kenyamanan dan keamanan pengguna. Spam dapat mengganggu aktivitas, menyita waktu, serta berpotensi memuat konten yang menyesatkan atau mengarah pada tindakan merugikan seperti penipuan dan manipulasi. Dalam konteks ini, upaya klasifikasi spam email dapat dipahami sebagai langkah preventif untuk mengurangi mudarat yang ditimbulkan oleh pesan yang tidak diinginkan. Prinsip kehati-hatian dalam menerima informasi juga sejalan dengan perintah untuk melakukan verifikasi ketika menerima kabar, sebagaimana ditegaskan dalam QS. Al-Hujurat ayat 6 yang memerintahkan agar berita diperiksa kebenarannya agar tidak menimbulkan kerugian karena ketidaktahuan.

Dalam praktiknya, identifikasi email spam juga dapat dilakukan oleh pengguna secara umum tanpa bantuan sistem. Beberapa ciri yang dapat dikenali antara lain penggunaan bahasa yang bersifat mendesak atau provokatif, seperti kalimat “akun Anda akan segera diblokir”, “segera klik tautan ini untuk menghindari kerugian”,

atau “kesempatan terbatas hanya hari ini”. Kalimat semacam ini bertujuan mendorong penerima untuk bertindak cepat tanpa melakukan verifikasi terlebih dahulu. Selain itu, spam juga sering berisi penawaran yang tidak relevan atau berlebihan, misalnya promosi produk yang tidak sesuai dengan kebutuhan pengguna, tawaran hadiah atau undian yang tidak pernah diikuti, serta janji keuntungan yang tidak masuk akal.

Ciri lainnya adalah penggunaan alamat pengirim yang tidak jelas atau mencurigakan, seperti kombinasi karakter acak yang tidak dikenal, serta adanya tautan yang mengarah ke alamat yang tidak sesuai dengan konteks pesan. Email spam juga kerap mengandung kesalahan penulisan, penggunaan tata bahasa yang kurang baik, atau format pesan yang tidak rapi, yang menunjukkan kurangnya kredibilitas pengirim. Oleh karena itu, sikap teliti dan kehati-hatian pengguna dalam menilai isi pesan menjadi langkah awal yang penting dalam mencegah dampak negatif dari spam.

Dalam perspektif Islam, komunikasi idealnya membawa manfaat serta menghindarkan individu dari informasi yang tidak jelas dan merugikan. Spam sering kali mendorong penerima untuk mengikuti tautan, instruksi, atau ajakan tertentu tanpa dasar pengetahuan yang kuat, sehingga bertentangan dengan prinsip untuk tidak mengikuti sesuatu tanpa ilmu. Hal ini selaras dengan QS. Al-Isra’ ayat 36 yang menegaskan larangan mengikuti sesuatu yang tidak memiliki pengetahuan yang jelas, serta mengingatkan bahwa pendengaran, penglihatan, dan hati akan dimintai pertanggungjawaban. Selain itu, spam juga dapat memicu sikap curiga berlebihan atau tindakan yang tidak proporsional, sehingga penting menjaga etika dan kehati-hatian dalam menilai informasi, sebagaimana QS. Al-Hujurat ayat 12

mengingatkan untuk menjauhi prasangka tertentu serta tidak melakukan tindakan yang melanggar etika sosial.

Penerapan klasifikasi spam email berbasis *machine learning* dalam penelitian ini dapat dipandang sebagai pemanfaatan ilmu pengetahuan untuk menghadirkan solusi yang bermanfaat bagi pengguna. Model klasifikasi membantu memisahkan pesan spam dan non-spam secara otomatis sehingga pengguna dapat terhindar dari paparan pesan yang tidak diinginkan dan berpotensi merugikan. Namun, dalam penerapannya tetap diperlukan perhatian terhadap prinsip keadilan agar tidak merugikan pihak yang seharusnya tidak terdampak, misalnya ketika pesan non-spam terklasifikasi sebagai spam. Prinsip keadilan ini sejalan dengan QS. Al-Ma'idah ayat 8 yang menegaskan perintah untuk berlaku adil karena keadilan lebih dekat kepada ketakwaan. Dengan demikian, implementasi sistem klasifikasi spam perlu disertai mekanisme peninjauan atau pemulihan pesan agar manfaat teknologi tetap berjalan tanpa mengabaikan aspek keadilan.

Secara keseluruhan, klasifikasi spam email selaras dengan upaya menjaga kemaslahatan dan mencegah kemudharatan dalam aktivitas komunikasi digital. Melalui penyaringan spam, pengguna dapat lebih terlindungi dari pesan yang menyesatkan, mengganggu, atau berpotensi menimbulkan kerugian, sekaligus mendorong komunikasi yang lebih aman dan bermanfaat. Pada saat yang sama, penerapannya perlu mempertimbangkan sikap kehati-hatian, verifikasi informasi, serta keadilan dalam pengambilan keputusan, sebagaimana nilai-nilai tersebut ditekankan dalam Al-Qur'an. Dengan demikian, penelitian ini tidak hanya berkontribusi pada aspek teknis pengolahan teks dan klasifikasi, tetapi juga

memiliki relevansi etis dalam menjaga kualitas komunikasi digital sesuai nilai-nilai Islam.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan rumusan masalah dan pembahasan pada bab sebelumnya maka diperoleh kesimpulan sebagai berikut:

1. Berdasarkan hasil pengujian pada dataset *Enron* Email, algoritma *Multinomial Naïve Bayes* (MNB) dengan penerapan *Chi-Square Feature Selection* menghasilkan performa klasifikasi dengan nilai akurasi sebesar 0,9917. Pada kelas spam atau kelas 1, model memperoleh presisi sebesar 0,3157, *recall* sebesar 0,1274, dan *f1-score* sebesar 0,1814. Sementara itu, pada kelas non-spam atau kelas 0, model memperoleh presisi sebesar 0,9937, *recall* sebesar 0,9979, dan *f1-score* sebesar 0,9958. Hasil tersebut menunjukkan bahwa model mampu mengklasifikasikan email dengan sangat baik pada kelas non-spam, namun kemampuan menangkap kelas spam masih terbatas jika dilihat dari nilai *recall*.
2. Penerapan *Local Density Weighting Synthetic Minority Oversampling Technique* (LDW-SMOTE) memberikan pengaruh yang signifikan terhadap penanganan ketidakseimbangan kelas pada klasifikasi spam menggunakan *Multinomial Naïve Bayes*. Pada skenario MNB *Chi-Square* dengan LDW-SMOTE, nilai *recall* kelas spam meningkat menjadi 0,9421, yang menunjukkan bahwa sebagian besar spam berhasil terdeteksi dan jumlah spam yang lolos sebagai non-spam dapat dikurangi secara signifikan. Namun, peningkatan *recall* tersebut disertai dengan penurunan presisi kelas spam menjadi 0,0633 dan penurunan akurasi menjadi 0,8986, yang

mengindikasikan meningkatnya jumlah email non-spam yang terklasifikasi sebagai spam. Hal ini disebabkan oleh peningkatan jumlah *false positive* yang cukup tinggi, sehingga model menjadi lebih sensitif dalam mendeteksi spam tetapi kurang selektif dalam membedakan antara spam dan non-spam. Dengan demikian, model MNB *Chi-Square* dengan LDW-SMOTE menjadi model paling efektif dalam mendeteksi spam sekaligus memiliki *testing time* tercepat yaitu sebesar 0,0100 detik. Sementara itu, model MNB *Chi-Square* tanpa LDW-SMOTE menjadi model terbaik dalam menjaga akurasi dan ketepatan klasifikasi secara keseluruhan.

5.2 Saran

Adapun beberapa saran yang dapat dijadikan acuan untuk penyempurnaan penelitian selanjutnya maka penulis memberikan saran berikut:

1. Penelitian selanjutnya disarankan melakukan optimasi seleksi fitur *Chi-Square*, misalnya dengan menguji beberapa nilai ambang/proporsi fitur terpilih (top-k) untuk mencari jumlah fitur paling optimal terhadap kinerja klasifikasi.
2. Penerapan LDW-SMOTE disarankan diuji dengan variasi parameter (misalnya rasio *oversampling* dan jumlah tetangga) agar peningkatan *recall* spam tetap tinggi tanpa meningkatkan kesalahan klasifikasi pada kelas non-spam secara berlebihan.
3. Untuk penanganan ketidakseimbangan data, selain LDW-SMOTE disarankan menguji teknik lain seperti ADASYN, Borderline-SMOTE, atau *undersampling*, misalnya *Random Under Sampling* atau *Tomek Links*

sehingga dapat diketahui metode penyeimbangan yang paling tepat sesuai karakteristik data email.

DAFTAR PUSTAKA

- Abdurrazik, & Wirawan, M. W. (2025). Analisis Klasifikasi Tweet Berdasarkan Topik Sosial Menggunakan SVM. *JNATIA*, 3(4). <https://doi.org/https://doi.org/10.24843/JNATIA.2025.v03.i04.p15>
- Al-Husaini, M., Saputra, P. A., Renaldi, M., & Maulana, R. A. (2024). *Prediksi Tsunami Dengan Metode Ensemble Machine Learning*. PT. Sonpedia Publishing Indonesia.
- Angriani, L., Hasan, H., Nugroho, A. Y., Panjaitan, G. H. A., Santi, D., Wulandari, A. S., Yuwono, C., Soelistianto, F. A., Kuswianto, D., Hidayat, A. S., & Yurika. (2025). Kecerdasan Buatan. In *Penerbit Tri Edukasi Ilmiah*. Penerbit Tri Edukasi Ilmiah.
- Arasy, A., Agustian, S., Handayani, L., & Iskandar, I. (2025). Klasifikasi Sentimen Menggunakan Metode Multilayer Perceptron dengan Fitur TF-IDF. *MALCOM: Indonesian Journal of Machine learning and Computer Science*, 5(3), 908–919. <https://doi.org/10.57152/malcom.v5i3.2052>
- Aryanda, H., Fachrezi, M. R., Walidein, A. D., Aptanta, D. A., & Syahputra, A. (2024). Prototype Penggabungan Fuzzy Dan *Naïve Bayes* dalam Analisis Citra Dokumen Untuk Penilaian Margin Laporan Praktikum UINSU. *IKOMTI (Jurnal Ilmu Komputer Dan Teknologi)*, 6(2), 103–110. <https://doi.org/https://doi.org/10.35960/ikomti.v6i2.1854>
- Asri, Y., Kuswardani, D., Ramadhana, S. A., TS, J. F. P., Marbun, D. U. N., Fatimah, F. N., & Qoriza, Z. (2025). *OPTIMALISASI ANALISIS SENTIMEN DENGAN SPELLING CORRECTOR*. Uwais Inspirasi Indonesia.
- Azzahra, S. A., & Majid, N. W. A. (2025). Klasifikasi dan Analisis Semantik Cyberbullying Sosial Media X: Integrasi Web Scraping dan Natural Language Processing (NLP). *Jurnal Educatio FKIP UNMA*, 11(2). <https://doi.org/10.31949/educatio.v11i2.12725>
- Badan Siber dan Sandi Negara. (2025). *Lanskap Keamanan Siber Indonesia 2024*.
- Barbieri, M. C., Grisci, B. I., & Dorn, M. (2024). Analysis and comparison of feature selection methods towards performance and stability. *Expert Systems with Applications*, 249, 123667. <https://doi.org/https://doi.org/10.1016/j.eswa.2024.123667>
- Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *Association for Computing Machinery New York, NY, United States*.

- Brian, T., Maulidhia, A. N. A., Sholikhah, E. N., & Wibowo, S. (2025). Perbandingan Model Decision Tree, Support Vector Machine dan K-Nearest Neighbors untuk Memprediksi Kualitas Air Minum. *INTEGER: Journal of Information Technology*,. <https://doi.org/https://doi.org/10.31284/j.integer.2024.v10i1.7523>
- Dariato, E., Laple, R., & Putra, S. (2024). Analisis Sentimen Komentar Pengguna Aplikasi Alfagift Di Google Playstore Menggunakan Metode Support Vector Machine (SVM). *Jatilima : Jurnal Multimedia Dan Teknologi Informasi* /, 07. <https://doi.org/https://doi.org/10.54209/jatilima.v7i03.1655>
- Dinata, R. K., & Hasdyna, N. (2025). *SUPERVISED LEARNING: Strategi Prediksi dan Klasifikasi Data*. PT. Serasi Media Teknologi.
- Drogkoula, M., Samaras, N., Iatrellis, O., Nathanail, E., & Kokkinos, K. (2025). Systematic review and topic classification of soft computing and machine learning in water resources management. *Discover Sustainability*, 6(1), 860.
- Enjelia, L., Cahyana, Y., & Wahiddin, D. (2025). Comparison of K-Nearest Neighbors and Naïve Bayes Classifier Algorithms in Sentiment Analysis of 2024 Election in Twitter (X). *Journal of Applied Informatics and Computing (JAIC)*, 9(3), 946. <https://doi.org/https://doi.org/10.30871/jaic.v9i3.9593>
- Enron Fraud Email Dataset*. (n.d.). Retrieved October 30, 2025, from <https://www.kaggle.com/datasets/advaithsrao/enron-fraud-email-dataset>
- Fadillah, D., Haerani, E., Wulandari, F., & Syafria, F. (2025). Klasifikasi Kondisi Janin Menggunakan Algoritma K-Nearest Neighbors dan Teknik SMOTE Berdasarkan Data Kardiotogram. *BULLETIN OF COMPUTER SCIENCE RESEARCH*, 5(4), 482–489. <https://doi.org/10.47065/bulletincsr.v5i4.585>
- Fahmi, K., Holle, H., Alfianita, R., & Putri, H. M. (2024). Evaluasi Teknik *Preprocessing* terhadap Kinerja *Multinomial Naïve Bayes* dalam Klasifikasi Pertanyaan Insincere. *JUSTIN (Jurnal Sistem Dan Teknologi Informasi)*, 12(4). <https://doi.org/10.26418/justin.v12i4.82758>
- Faisal, M. R., Kartini, D., Arrahimi, A. R., & Saragih, T. H. (2023). *Belajar Data Science: Text Mining Untuk Pemula I*. Scripta Cendekia. <https://books.google.co.id/books?id=5CKIEAAAQBAJ>
- Fajar, F. M., & Maulina, D. (2024). Analisis Sentimen Kurikulum Merdeka Dengan Penerapan Convolutional Neural Network Independent Curriculum Sentiment Analysis with The Application of Convolutional Neural Network. *JACIS: Journal Automation Computer Information System*, 4(1), 1–11. <https://doi.org/https://doi.org/10.47134/jacis.v4i1.68>
- Febriansyah, F. I., & SH, M. (2025). *Cybercrime: Kejahatan di Balik Layar Digital*. Najaha.

- Ghozali, I. (2018). *Aplikasi Analisis Multivariate Dengan Program IBM SPSS 26 Edisi 10*. Badan Penerbit Universitas Diponegoro.
- Habibillah, N. H., Nurlaela, R., Lestari, R. D., Mosyarrina, F., & Tsuraya, N. N. (2025). Analisis Sentimen Terhadap Aplikasi Spotify Berdasarkan Ulasan di Google Play Store. *Journal of Informatics and Interactive Technology*, 2(1), 300–306. <https://doi.org/10.63547/jiite.v2i1.47>
- Haroon, M. (2018). Comparative Analysis of Stemming Algorithms for Web Text Mining. *International Journal of Modern Education and Computer Science*, 10(9), 20–25. <https://doi.org/10.5815/ijmecs.2018.09.03>
- Hokijuliandy, E., Napitupulu, H., & Firdaniza, F. (2023). Analisis Sentimen Menggunakan Metode Klasifikasi Support Vector Machine (SVM) dan Seleksi Fitur *Chi-Square*. *SisInfo : Jurnal Sistem Informasi Dan Informatika*. <https://doi.org/https://doi.org/10.37278/sisinfo.v5i2.670>
- Irawan, Y., Pramitasari, R., Ashari, W. M., Nur, A., & Yansyah, H. (2025). Support Vector Machine Classification Algorithm for Detecting DDoS Attacks on Network Traffic. *Journal of Applied Informatics and Computing (JAIC)*, 9(4), 1945. <https://doi.org/https://doi.org/10.30871/jaic.v9i4.10003>
- Januarto, S., Murtopo, A. A., & Arif, Z. (2025). Klasifikasi Status Stunting Balita Tegal Menggunakan Teknik Smote Pada Metode Naives Bayes Gaussain. *RIGGS: Journal of Artificial Intelligence and Digital Business*, 4(3), 2484–2490. <https://doi.org/10.31004/riggs.v4i3.2349>
- Jusia, P. A., Pahlevi, R., & Simanjuntak, D. S. P. (2025). Peningkatan Performa *Naïve Bayes* dengan Fitur *Chi-Square* pada Analisis Sentimen Komentar Pengguna Aplikasi Netflix. *Bulletin of Computer Science Research*, 5(4), 614–621. <https://doi.org/https://doi.org/10.47065/bulletincsr.v5i4.532>
- Kementrian Agama. (2022). *Al-Qur'an dan Terjemahannya. Lajnah Pentashihan Mushaf Al-Qur'an Badan Litbang Kementerian Agama RI*. <https://quran.kemenag.go.id/>
- Kumalasari, K., Solihin, S., Syawaluddin, M., & Susandri, S. (2025). Klasifikasi Sentimen Umpan Balik Pelanggan dalam Bahasa Indonesia: Studi Kasus Global Bangunan Menggunakan SVM dan Naive Bayes. *Prosiding Seminar Nasional*, 1(2), 416–431.
- Laksana, J. P., Irsyad, H., & Rahman, A. (2025). Analisis Topik Dominan Dalam Paper Ilmu Komputer Menggunakan TF-IDF Dan K-Means. *Buletin Ilmiah Informatika Teknologi*, 3(3), 78–84. <https://doi.org/10.58369/biit.v2i3.122>
- Laome, L., Dian Christien Arisona, Zelianti, Gusti Ngurah Adhi Wibawa, Irma Yahya, & Mukhsar. (2025). Klasifikasi Judul Berita Online Menggunakan *Multinomial Naïve Bayes*. *Al-Khwarizmi : Jurnal Pendidikan Matematika Dan*

- Ilmu Pengetahuan Alam*, 13(1), 76–86.
<https://doi.org/10.24256/jpmipa.v13i1.6486>
- Lyu, J., Yang, J., Su, Z., & Zhu, Z. (2025). LD-SMOTE: A Novel Local Density Estimation-Based *Oversampling* Method for Imbalanced Datasets. *Symmetry*, 17(2). <https://doi.org/10.3390/sym17020160>
- Mardivta, H., & Tussyifa Izzati, Z. (2024). Deteksi Berita Hoaks Menggunakan TF-IDF dan *Multinomial Naïve Bayes* pada Data Berita Online. *Journal Of Information Technology*, 3(4).
- Maulani, G., Hasan, F. N., Setiawan, D., Bowo, I. T., Ardhana, V. Y. P., Ramdhani, Y., Inayah, I., Ardiantoro, L., Sugianto, C. A., & Chandra, R. (2025). *Machine Learning*. Mega Press Nusantara.
- Memon, S., Sindh, S. B., Mallah, G. A., KNMemon, Shaikh, A., Kaasoori, S., & Dehraj, F. U. H. (2020). Comparative Study of Truncating and Statistical *Stemming* Algorithms. *IJACSA) International Journal of Advanced Computer Science and Applications*, 11(2). www.ijacsa.thesai.org
- Menarianti, I., Kom, S., Kom, M., Wihardjo, E., Zakarijah, I. M., Arfianto, A. Z., Nampira, A. A., Khumaidi, A., ST, S., & Kolyaan, Y. (2025). *KONSEP DASAR MACHINE LEARNING DI ERA DIGITAL*. Cendikia Mulia Mandiri.
- Mola, S. A. S., Djawa, S. N. R., & Mauko, A. Y. (2025). *Text Mining: Analisis Sentimen dengan Naïve Bayes*. Kaizen Media Publishing.
- Mola, S. A. S., Roma, R. V. K. I. O., & Widiastuti, T. (2025). *Text Mining: Analisis Sentimen dengan Lexicon*. Kaizen Media Publishing.
<https://books.google.co.id/books?id=sJ90EQAAQBAJ>
- Muslim bin al-Hajjaj. *Kitab Shahih Muslim*.
- Mutoffar, M. M., ST, M. M., Retnoningsih, E., Kom, S., Kom, M., Yasik, I. Y. L., & Eliza, S. T. (2025). *Decoding Intelligence Algoritma Machine learning dalam Aksi dan Bisnis*. PT KIMHSAFI ALUNG CIPTA.
- Noviyanti, N. P. A., & Putra, Ig. N. A. C. (2025). Penerapan *Multinomial Naïve Bayes* dan *Chi-Square* pada Analisis Sentimen Makan Bergizi Gratis. *JNATIA*, 3(4). <https://doi.org/https://doi.org/10.24843/JNATIA.2025.v03.i04.p08>
- NU Online. *Tafsir Qur'an NU Online: Arab, Latin dan Terjemah Lengkap* | Retrieved October 30, 2025, from <https://quran.nu.or.id/ar-rad>
- Nurchahyo, R., Tanjung, F., & Rahman, S. (2025). Meningkatkan Deteksi Email Phising Melalui Pendekatan SVM yang Dioptimalkan NLP Enhancing Phishing Email Detection through NLP-Optimized SVM Approach.

INCODING: Journal Of Informatics and Computer Science Engineering, 2.
<http://journal.mahesacenter.org/index.php/incoding>

- Nursahid, W., Nugroho, B. I., & Syefudin, S. (2025). Optimalisasi *Preprocessing* Data Menggunakan Pendekatan CRISP-DM untuk Meningkatkan Kualitas Klasifikasi Penyakit Jantung. *RIGGS: Journal of Artificial Intelligence and Digital Business*, 4(3), 3621–3626. <https://doi.org/10.31004/riggs.v4i3.2514>
- Olson, D. (2008). *Pengantar Ilmu Penggalan Data Bisnis*. Penerbit Salemba.
- Pane, E. S., & Caroline, C. (2024). Optimalisasi Evaluasi Pelaksanaan Pelatihan Melalui Analisis Sentimen Otomatis Dengan Model *Text Classification*. *Prosiding PITNAS Widyaiswara*, 1, 141–154.
- Paramita, C., Simbolon, C. S., Pamungkas, A. S., Triono, J. M., Utomo, E. P. W., & Subhiyakto, E. R. (2025). Analisis Pengaruh SMOTE terhadap Kinerja Model KNN untuk Prediksi Risiko Stroke. *Jurnal Informatika: Jurnal Pengembangan IT*, 10(4), 978–988.
- Pati, R. M. D., Pratama, E. K., & Tuslaela. (2025). Analisis Sentimen pada Ulasan Aplikasi JakLingko Menggunakan Metode *Naïve Bayes*. *Repeater : Publikasi Teknik Informatika Dan Jaringan*, 3(4), 12–21. <https://doi.org/10.62951/repeater.v3i4.638>
- Pinem, T. H., & Putra, Z. P. (2025). Evaluasi Kinerja Algoritma Klasifikasi Deep Learning dalam Prediksi Diabetes. *Jurnal Ilmiah FIFO*, 17(1), 17. <https://doi.org/10.22441/fifo.2025.v17i1.003>
- Rahayu, P. W., Sudipa, I. G. I., Suryani, S., Surachman, A., Ridwan, A., Darmawiguna, I. G. M., Sutoyo, M. N., Slamet, I., Harlina, S., & Maysanjaya, I. M. D. (2024). *Buku ajar data mining*. PT. Sonpedia Publishing Indonesia.
- Rahmawati, D., Mila, V., Giri, L., Rienzy, K., Wiratri, A., Rizki, A., Syavia, B., & Aisyah, A. (2024). *Waspada Kejahatan Phishing Attack!* Literasi Nusantara.
- Rajendra, P., Ravi, P. V. N. H., & Meenakshi, K. (2024). *Machine learning from a mathematical perspective*. *AIP Conference Proceedings*, 3149(1), 140021.
- Rianto, I., & Santosa, I. P. I. (2025). *Data Preparation untuk Machine learning & Deep Learning*. Penerbit Andi.
- Ridwan, R., Handayani, H. H., Lestari, S. A. P., & Cahyana, Y. (2025). Evaluasi Kinerja Algoritma Random Forest Dan Gradient Boosting Untuk Klasifikasi Penyakit Jantung. *Jurnal Komtika (Komputasi Dan Informatika)*, 9(1), 112–124. <https://doi.org/10.31603/komtika.v9i1.13450>
- Rizkia, A. S., Wufron, W., & Roji, F. F. (2025). Analisis Sentimen Coretax: Perbandingan Pelabelan Data Manual, Transformers-Based, dan Lexicon-

- Based pada Performa IndoBERT. *MALCOM: Indonesian Journal of Machine learning and Computer Science*, 5(3). <https://doi.org/10.57152/malcom.v5i3.2151>
- Rosyidi, M. I. U., & Rochmawati, N. (2023). Teknik Bagging Pada Algoritma Klasifikasi Decision Tree dan SVM Untuk Klasifikasi SMS Berbahasa Indonesia. *Journal of Informatics and Computer Science (JINACS)*, 5(02), 265–271. <https://doi.org/https://doi.org/10.26740/jinacs.v5n02.p265-271>
- Rustiyana, R., Judijanto, L., Mahendra, G. S., Kamil, Z. A., Purba, D. N., Sutoyo, M. N., Hendrayana, I. G., Pasrun, Y. P., & Prayudani, S. (2025). *Data Mining: Algoritma dan Penerapannya*. PT. Sonpedia Publishing Indonesia.
- Sari, H. L., Tribuana, D., Sumah, J., & Indahsari, A. N. (2025). *Implementasi Python Untuk Machine Learning*. Serasi Media Teknologi.
- Sari, W. P., & Fahmi, H. (2021). Opinion *mining* analysis on online product reviews using *Naïve Bayes* and feature selection. *Proceedings of 2021 International Conference on Information Management and Technology, ICIMTech 2021*, 256–260. <https://doi.org/10.1109/ICIMTech53080.2021.9535081>
- Sartika, D., Florensia, Y., & Utari, M. (2025). The Effect of the SMOTE Method on the Classification of Toddler Nutritional Status Using the *Naïve Bayes* Method. *Jurnal Sisfokom (Sistem Informasi Dan Komputer)*, 14(3), 324–329.
- Slam, B. E., Irawan, F., Efranda, N., & Herikson, R. (2025). Implementasi *Machine learning* untuk Klasifikasi Buku Otomatis pada Perpustakaan Digital. *Jurnal Informatika Polinema*, 11(3), 305–310. <https://doi.org/https://doi.org/10.33795/jip.v11i3.7298>
- Sugiyono. (2016). Metode penelitian kuantitatif kualitatif dan R&D. In *Alfabeta, Bandung*. ALFABETA CV.
- Suharmin, W. N., Hasan, I. K., & Yahya, N. I. (2025). Perbandingan Seleksi Fitur Forward Selection dan Backward Elimination pada Algoritma Support Vector Machine. *The Indonesian Journal of Computer Science*, 14(2).
- Sulianta, F. (2025). *Dasar dan Konsep Machine Learning*.
- Sulianta, F. (2025). *MALWARE*.
- Supriyono, L. A., Carudin, C., Nugroho, H. A. S. A., Budiasto, J., Zulfa, I., & Kohsasih, K. L. (2025). *Buku Ajar Pengantar Ilmu Komputer*. PT. Green Pustaka Indonesia.
- Swastika, R., Mukodimah, S., Susanto, F., Muslihudin, M., & Adab, S. I. P. (2023). *IMPLEMENTASI DATA MINING (Clustering, Association, Prediction, Estimation, Classification)*. Penerbit Adab.

Yana, B. Y., Via, Y. V., & Nurlaili, A. L. (2025). Penerapan Algoritma K-Nearest Neighbor Dalam Klasifikasi Penyakit Daun Padi Menggunakan Ekstraksi HOG. *Jurnal Algoritme*, 5(3), 372–382.
<https://doi.org/10.35957/algoritme.v6i1.12154>

Zaman, B., & Purwanti, E. (2025). *Mengurai Opini Digital: Panduan Praktis Analisis Sentimen berbasis Aspek*. PT. Pustaka Saga Jawadwipa.

LAMPIRAN

Lampiran 1 Dataset Penelitian

https://drive.google.com/file/d/1WRQpaCCzMmqD7NasKj7rXswdDFels5F0/view?usp=sharing	
---	---

Lampiran 2 Hasil Preprocessing

https://drive.google.com/file/d/1OQiKa3nW7pRiqYJHoQXPc3wQuqJeH-8e/view?usp=sharing	
---	---

Lampiran 3 Hasil Evaluasi Model *Chi-Square*

hasil evaluasi		1	2	3	4	5	Rata Rata	Standar Deviasi
akurasi		0,9916	0,9917	0,9921	0,9916	0,9916	0,9917	0,0002
presisi	spam	0,3046	0,3288	0,3712	0,2916	0,2824	0,3157	0,0356
	non-spam	0,9938	0,9938	0,9938	0,9936	0,9935	0,9937	0,0001
recall	spam	0,1314	0,1400	0,1400	0,1200	0,1057	0,1274	0,0147
	non-spam	0,9978	0,9979	0,9982	0,9978	0,9980	0,9979	0,0002
F1-score	spam	0,1836	0,1963	0,2033	0,1700	0,1538	0,1814	0,0200
	non-spam	0,9958	0,9958	0,9960	0,9957	0,9958	0,9958	0,0001
Testing Time (s)		0,0122	0,0108	0,0091	0,0123	0,0078	0,0104	0,0020

Lampiran 4 Hasil Evaluasi Model tanpa *Chi-Square*

hasil evaluasi		1	2	3	4	5	Rata Rata	Standar Deviasi
akurasi		0,9914	0,9913	0,9910	0,9908	0,9911	0,9911	0,0003
presisi	spam	0,3152	0,3068	0,2569	0,2525	0,2670	0,2797	0,0292
	non-spam	0,9940	0,9940	0,9937	0,9938	0,9937	0,9938	0,0002
recall	spam	0,1657	0,1657	0,1314	0,1428	0,1342	0,1480	0,0167
	non-spam	0,9974	0,9973	0,9972	0,9969	0,9973	0,9972	0,0002
F1-score	spam	0,2172	0,2152	0,1739	0,1824	0,1787	0,1935	0,0210
	non-spam	0,9957	0,9956	0,9955	0,9953	0,9955	0,9955	0,0001
Testing Time (s)		0,0233	0,0134	0,0113	0,0123	0,0181	0,0157	0,0050

Lampiran 5 Hasil Evaluasi Model *Chi-Square* dan LDW-SMOTE

hasil evaluasi		1	2	3	4	5	Rata Rata	Standar Deviasi
akurasi		0,8960	0,8984	0,9026	0,8975	0,8984	0,8986	0,0024
presisi	spam	0,0617	0,0630	0,0661	0,0624	0,0633	0,0633	0,0017
	non-spam	0,9996	0,9996	0,9996	0,9995	0,9996	0,9996	0,00005
recall	spam	0,9514	0,9514	0,9021	0,9485	0,9571	0,9421	0,0226
	non-spam	0,8956	0,8980	0,9600	0,8972	0,8980	0,9098	0,0281
F1-score	spam	0,1159	0,1183	0,1237	0,1171	0,1189	0,1188	0,0030
	non-spam	0,9448	0,9461	0,9484	0,9456	0,9461	0,9462	0,0013
Testing Time (s)		0,0100	0,0100	0,0100	0,0100	0,0100	0,0100	0,0000

Lampiran 6 Hasil Evaluasi Model LDW-SMOTE

hasil evaluasi		1	2	3	4	5	Rata Rata	Standar Deviasi
akurasi		0,9148	0,9169	0,9152	0,9154	0,9169	0,9158	0,0010
presisi	spam	0,0738	0,0736	0,0751	0,0746	0,0751	0,0744	0,0007
	non-spam	0,9995	0,9994	0,9996	0,9995	0,9995	0,9995	0,0001
recall	spam	0,9429	0,9257	0,9571	0,9485	0,9371	0,9423	0,0118
	non-spam	0,9146	0,9160	0,9149	0,9152	0,9167	0,9155	0,0008
F1-score	spam	0,1369	0,1364	0,1392	0,1384	0,1390	0,1380	0,0013
	non-spam	0,9552	0,9559	0,9554	0,9555	0,9563	0,9557	0,0004
Testing Time (s)		0,0100	0,0100	0,0100	0,0200	0,0200	0,0140	0,0055

Lampiran 7 Hasil Perbandingan dengan Model lain

Model	Metrik Evaluasi		Running ke-					Rata Rata	Standar Deviasi
			1	2	3	4	5		
Multinomial Naïve Bayes	Akurasi		0,8960	0,8984	0,9026	0,8975	0,8984	0,8986	0,0024
	Presisi	0	0,9996	0,9996	0,9996	0,9995	0,9996	0,9996	0,00005
		1	0,0617	0,0630	0,0661	0,0624	0,0633	0,0633	0,0017
	Recall	0	0,8956	0,8980	0,9600	0,8972	0,8980	0,9098	0,0281
		1	0,9514	0,9514	0,9021	0,9485	0,9571	0,9421	0,0226
	f1-score	0	0,9448	0,9461	0,9484	0,9456	0,9461	0,9462	0,0013
		1	0,1159	0,1183	0,1237	0,1171	0,1189	0,1188	0,0030
	Testing Time (s)		0,0100	0,0100	0,0100	0,0100	0,0100	0,0100	0,0000
Bernoulli Naïve Bayes	Akurasi		0,9293	0,9278	0,9255	0,9272	0,9317	0,9284	0,0027
	Presisi	0	0,9975	0,9975	0,9975	0,9975	0,9972	0,9974	0,0001
		1	0,0664	0,0647	0,0632	0,0639	0,0654	0,0647	0,0013
	Recall	0	0,9310	0,9296	0,9273	0,9291	0,9338	0,9302	0,0024
		1	0,6800	0,6743	0,6800	0,6714	0,6429	0,6697	0,0154
	f1-score	0	0,9631	0,9524	0,9611	0,9621	0,9645	0,9606	0,0048
		1	0,1210	0,1180	0,1156	0,1167	0,1188	0,1180	0,0021
	Testing Time (s)		0,0274	0,118	0,0339	0,0184	0,0378	0,0471	0,0403
Regresi logistik	Akurasi		0,9628	0,9629	0,9624	0,9611	0,9627	0,9624	0,0007
	Presisi	0	0,9993	0,9995	0,9991	0,9992	0,9993	0,9993	0,0001
		1	0,1501	0,1512	0,1459	0,1439	0,1504	0,1483	0,0032
	Recall	0	0,9633	0,9622	0,9630	0,9616	0,9631	0,9626	0,0007
		1	0,9000	0,9343	0,8771	0,8943	0,9057	0,9023	0,0209
	f1-score	0	0,9809	0,9805	0,9807	0,9801	0,9809	0,9806	0,0003
		1	0,2574	0,2604	0,2502	0,2479	0,2579	0,2548	0,0054
	Testing Time (s)		0,0089	0,0099	0,0107	0,0127	0,0135	0,0111	0,0019

Lampiran 8 Source Code Klasifikasi Email Spam

```

from google.colab import drive
import pandas as pd
Hubungkan Google Drive .' #
drive.mount('/content/drive')
Memuat dataset dari file CSV yang ada di Google Drive .' #
file_path = '/content/drive/MyDrive/SKRIPSI/DATASET SKRIPSI.csv' #
Ganti 'namafolder' sesuai foldermu
:try
df = pd.read_csv(file_path)
Menampilkan beberapa kolom penting .' #
print("Contoh data:")
display(df[['Body', 'Label', 'Subject', 'From']].head())
:except FileNotFoundError
print(f'Error: File not found at {file_path}. Make sure the path is correct and
your drive is mounted.')
:except Exception as e
print(f'An error occurred after mounting the drive: {e}')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
Melihat Jumlah Data ( \ --- #
)df = df.copy
print("Jumlah data:", len(df))
--- Cek Missing Value ( \ --- #
print("\n== Missing Value ==")
print(df.isnull().sum())
'Menampilkan cuplikan data dengan missing value di kolom 'Subject #
missing_subject_df = df[df['Subject'].isnull()]
print("\nCuplikan data dengan missing value pada kolom 'Subject:')
display(missing_subject_df.head())
--- Cek Duplikat ( \ --- #
)dup_count = df.duplicated(subset=['Subject', 'Body', 'From']).sum
print(f'\nJumlah duplikat (Subject+Body+From): {dup_count}')
--- Distribusi Kelas ( \ --- #
print("\n== Distribusi Kelas ==")
print(df['Label'].value_counts())
print("\nProporsi kelas:")
print(df['Label'].value_counts(normalize=True))
'config InlineBackend.figure_format = 'retina%'
Plot distribusi kelas #
df['Label'].value_counts().plot(kind='bar', title="Distribusi Kelas (Spam vs
Ham)")
plt.xlabel("Label")
plt.ylabel("Jumlah")
plt.show
----- PERBAIKI MISSING VALUE ( \ ----- #

```

```

print("\n== Tahap 1/6: PERBAIKI MISSING VALUE ==")
Label wajib (tanpa label tidak bisa training) #
before_mv = len(df)
(df = df[~df['Label'].isna()]).copy
print(f'- Baris tanpa Label dibuang: {before_mv - len(df)}')
Isi NaN di Subject/Body/From #
:for c in ['Subject','Body','From']
(n_missing = df[c].isna().sum
(df[c] = df[c].fillna
print(f'- Isi NaN pada {c}: {n_missing} -> 0")
Opsional: buang baris yang Subject & Body kosong total #
before_empty = len(df)
df = df[~((df['Subject'].str.strip()=="") & (df['Body'].str.strip()=="") &
(df['From'].str.strip()==""))].copy
print(f'- Baris kosong (Subject,Body, From kosong) dibuang: {before_empty -
len(df)}')

----- PERBAIKI DUPLIKAT (✓ ----- #
print("\n== Tahap 2/6: PERBAIKI DUPLIKAT ==")
before_dup = len(df)
df = df.drop_duplicates(subset=['Subject','Body', 'From'], keep='first')
print(f'- Duplikat (Subject, Body, From) terhapus: {before_dup - len(df)}')
import gc
import re
import pandas as pd # Re-import pandas for display if not already available in
snippet
- LOWERCASING & CLEANING DAN SIMPAN KE CSV (✓ ----- #
print("\n== Tahap 3/6: LOWERCASING & CLEANING DAN SIMPAN KE
CSV ==")
Gabungkan Subject + Body + From untuk diproses #
FIX: Include 'From' column in the text concatenation #
+ ' ' + (df['text_raw'] = (df['Subject'].astype(str).str.strip
+ ' ' + (df['Body'].astype(str).str.strip
(df['From'].astype(str).str.strip()).str.strip
Siapkan regex cleaning #
url_re = re.compile(r'(http|https):\/\/S+|www\.S+|<url>', re.IGNORECASE)
email_re = re.compile(r'\b[\w\.-]+@[\w\.-]+\.\w+\b')
html_re = re.compile(r'<.*?>')
digits_re = re.compile(r'\d+')
nonalpha = re.compile(r'^[a-z\s]')
reply_re = re.compile(r'^\s*(re|fw|fwd)\s*[:\-\s]*', re.IGNORECASE)
:def to_lower(s: str) -> str
()return s.lower().strip
:def basic_clean(s: str) -> str
s = reply_re.sub(' ', s) # 're:/'fw:' di awal subject (jika ada)
s = url_re.sub(' ', s) # hapus URL
s = email_re.sub(' ', s) # hapus email
s = html_re.sub(' ', s) # hapus tag HTML
s = digits_re.sub(' ', s) # hapus angka

```

```

s = nonalpha.sub(' ', s)      # sisakan huruf & spasi
(s = re.sub(r'\s+', ' ', s).strip
return s
df['text_lower'] = df['text_raw'].apply(to_lower)
df['text_clean'] = df['text_lower'].apply(basic_clean)
print("- Contoh text_raw vs text_lower vs text_clean:")
display(df[['text_raw', 'text_lower', 'text_clean']].head(3))
import gc
import nltk
import re # Import re for regex tokenization
from typing import List
----- TOKENIZATION (¿ ----- #
print("\n== Tahap 4/6: TOKENIZATION ==")
Gunakan tokenisasi berbasis regex sederhana sebagai alternatif #
:def tokenize(s: str) -> List[str]
:if not s
[] return
Gunakan regex untuk memecah teks menjadi kata-kata (urutan huruf) #
Ini akan memisahkan berdasarkan spasi dan menghapus karakter non- #
alfabet
tokens = re.findall(r'\b[a-z]+\b', str(s))
return tokens
df['tokens'] = df['text_clean'].apply(tokenize)
Hapus kolom perantara text_clean untuk menghemat memori #
df.drop(columns=['text_clean'], inplace=True)
(gc.collect
print("- Kolom text_clean dihapus dari memori.")
print("- Contoh tokens:")
display(df[['tokens']].head(3))
import gc
from nltk.corpus import stopwords
import nltk # Import nltk
----- STOPWORD REMOVAL (° ----- #
print("\n== Tahap 5/6: STOPWORD REMOVAL ==")
Unduh stopwords jika belum ada (pastikan ini terjadi sebelum #
STOPWORDS digunakan)
:try
nltk.data.find('corpora/stopwords')
:except LookupError
print("Downloading 'stopwords'...")
nltk.download('stopwords')
STOPWORDS = set(stopwords.words('english'))
:def remove_stop(tokens: list) -> list
:if not STOPWORDS
return tokens
return [t for t in tokens if t not in STOPWORDS and len(t) > 1]
df['tokens_no_stop'] = df['tokens'].apply(remove_stop)
Hapus kolom perantara tokens untuk menghemat memori #

```

```

df.drop(columns=['tokens'], inplace=True)
)gc.collect
print("- Kolom tokens dihapus dari memori.")
print("- Contoh tokens_no_stop:")
display(df[['tokens_no_stop']].head(3))
import gc
import nltk
from nltk.stem import WordNetLemmatizer
from typing import List
----- LEMMATIZATION (1 ----- #
print("\n== Tahap 6/6: LEMMATIZATION ==")
Define LANGUAGE variable #
'LANGUAGE = 'english
USE_LEMMATIZATION = True # Flag untuk mengaktifkan lemmatization
Unduh WordNet jika belum ada #
:try
nltk.data.find('corpora/wordnet')
:except LookupError
print("Downloading 'wordnet'...")
nltk.download('wordnet')
Unduh Averaged Perceptron Tagger untuk POS Tagging jika belum ada #
:try
Corrected resource name based on error message #
nltk.data.find('taggers/averaged_perceptron_tagger_eng')
:except LookupError
print("Downloading 'averaged_perceptron_tagger_eng'...")
Corrected download command based on error message #
nltk.download('averaged_perceptron_tagger_eng')
Inisialisasi WordNetLemmatizer #
()wordnet_lemmatizer = WordNetLemmatizer
:def get_wordnet_pos(word)
"""Maps NLTK POS tag to WordNet POS tag for lemmatization"""
()tag = nltk.pos_tag([word])[0][1][0].upper
,tag_dict = {"J": nltk.corpus.wordnet.ADJ
,N": nltk.corpus.wordnet.NOUN"
,V": nltk.corpus.wordnet.VERB"
,R": nltk.corpus.wordnet.ADV"
return tag_dict.get(tag, nltk.corpus.wordnet.NOUN)
:def normalize_tokens(tokens: List[str]) -> List[str]
:if not USE_LEMMATIZATION
return [str(t) for t in tokens if isinstance(t, str)]
[] = lemmas
:for token in tokens
:if isinstance(token, str)
Perform POS tagging and then lemmatize with the correct POS #
pos = get_wordnet_pos(token)
lemmas.append(wordnet_lemmatizer.lemmatize(token, pos=pos))
:else

```

```

lemmas.append(str(token))
return lemmas
print(f'- Kolom yang tersedia di df sebelum lemmatization:
}') # Diagnostic print() {df.columns.tolist
df['tokens_norm'] = df['tokens_no_stop'].apply(normalize_tokens)
Hapus kolom perantara tokens_no_stop untuk menghemat memori #
df.drop(columns=['tokens_no_stop'], inplace=True)
(gc.collect
print("- Kolom tokens_no_stop dihapus dari memori.")
df['text_processed'] = df['tokens_norm'].apply(lambda toks: ''.join(toks))
Hapus kolom perantara tokens_norm untuk menghemat memori #
df.drop(columns=['tokens_norm'], inplace=True)
(gc.collect
print("- Kolom tokens_norm dihapus dari memori.")
print("- Contoh text_processed:")
display(df[['text_processed']].head(3))
----- SIMPAN DATA BERSIH ----- #
import pandas as pd
import os
Path penyimpanan di Google Drive #
cleaned_path = "/content/drive/MyDrive/SKRIPSI/text_cleaned.csv" #
Mengubah path ke Google Drive
Cek dulu kolom apa saja yang ada di df #
print("\nDaftar kolom yang tersedia di df:", df.columns.tolist())
Simpan hanya kolom yang ada, pastikan nama kolom sesuai #
cols_to_save = [col for col in ['text_processed', 'Label'] if col in df.columns] #
Memperbaiki nama kolom
df[cols_to_save].to_csv(cleaned_path, index=False)
print(f'\n Data teks bersih berhasil disimpan ke: {cleaned_path}')
print("\nCuplikan hasil teks bersih:")
display(df[cols_to_save].head()) # Menggunakan display() untuk tampilan yang
lebih baik
----- SPLIT DATA MANUAL ( ----- #
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split # Import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer # Import
TfidfVectorizer
print("\n== Tahap : SPLIT DATA MANUAL ==")
Split data df into training and testing sets first #
Use stratify to maintain the proportion of labels in both sets #
X = df['text_processed']
y = df['Label']
Manual split with stratification and fixed test size #
Calculate the number of samples for the test set for each class #
total_samples = len(df)
--- MODIFIKASI: Ubah proporsi testing menjadi 10% --- #
test_size_ratio = 0.20 # 10% for testing
--- MODIFIKASI: Hitung jumlah sampel test total berdasarkan ratio --- #

```

```

total_test_size = int(total_samples * test_size_ratio)
Ensure we don't request more samples than available in the minority class #
MODIFIKASI: Hitung jumlah sampel test untuk setiap kelas berdasarkan --- #
--- ratio
test_size_class_1 = int(y.value_counts().get(1, 0) * test_size_ratio)
test_size_class_0 = int(y.value_counts().get(0, 0) * test_size_ratio)
Get indices for test set for each class #
--- MODIFIKASI: Ambil sampel dengan jumlah yang sudah dihitung --- #
test_idx_0 = y[y == 0].sample(n=test_size_class_0).index
test_idx_1 = y[y == 1].sample(n=test_size_class_1).index
test_idx = test_idx_0.union(test_idx_1)
Split data based on indices #
X_test_df = df.loc[test_idx]
y_test = y.loc[test_idx]
Get training indices by excluding test indices #
train_idx = df.index.difference(test_idx)
X_train_df = df.loc[train_idx]
y_train = y.loc[train_idx]
print("Total data:", len(df))
print("Train size (before vectorization):", len(X_train_df))
print("Test size (before vectorization):", len(X_test_df))
print("\nDistribusi label di Train (before vectorization):")
print(y_train.value_counts())
print("\nDistribusi label di Test (before vectorization):")
print(y_test.value_counts())
import pandas as pd # Import pandas for DataFrame creation
from sklearn.feature_extraction.text import TfidfVectorizer # Import
TfidfVectorizer
import joblib
import os # Import os for saving
from scipy.sparse import save_npz # Import save_npz - Keep import just in
case, though not used for saving NPZ
print("\n== Applying TF-IDF Vectorization ==")
Inisialisasi TF-IDF Vectorizer (using the same parameters as in the original #
TF-IDF cell)
tfidf_vectorizer = TfidfVectorizer(max_features=2000)
Fit on training data and transform both training and testing data #
X_train = tfidf_vectorizer.fit_transform(X_train_df['text_processed'])
X_test = tfidf_vectorizer.transform(X_test_df['text_processed'])
print(f'- Bentuk matriks TF-IDF X_train: {X_train.shape}')
print(f'- Bentuk matriks TF-IDF X_test: {X_test.shape}')
print("- Contoh fitur:", tfidf_vectorizer.get_feature_names_out()[:10])
Contoh cuplikan matriks TF-IDF (misal 5 baris pertama dan 10 fitur pertama) #
print("\nCuplikan matriks TF-IDF (Training Data):")
)tfidf_train_display = pd.DataFrame
X_train[:5, :min(10, X_train.shape[1])].toarray(), # Tambahkan min untuk
menghindari error index

```

```

columns=tfidf_vectorizer.get_feature_names_out()[:min(10,
X_train.shape[1])] # Tambahkan min
(
display(tfidf_train_display)
Tambahkan cuplikan matriks TF-IDF (Testing Data) #
print("\nCuplikan matriks TF-IDF (Testing Data):")
)tfidf_test_display = pd.DataFrame
X_test[:5, :min(10, X_test.shape[1])].toarray(), # Tambahkan min
columns=tfidf_vectorizer.get_feature_names_out()[:min(10,
X_test.shape[1])] # Tambahkan min
(
display(tfidf_test_display)
Simpan matriks TF-IDF dalam format CSV (Dense) #
'/output_dir = '/content/drive/MyDrive/SKRIPSI
os.makedirs(output_dir, exist_ok=True) # Buat folder jika belum ada
Simpan sebagai NPZ (Sparse) - Lebih efisien untuk matriks besar #
train_matrix_path = os.path.join(output_dir, 'X_train_tfidf.npz')
test_matrix_path = os.path.join(output_dir, 'X_test_tfidf.npz')
:try
save_npz(train_matrix_path, X_train)
save_npz(test_matrix_path, X_test)
print(f'\n\u2705 Matriks TF-IDF training (sparse) berhasil disimpan ke:
{train_matrix_path}')
print(f'\u2705 Matriks TF-IDF testing (sparse) berhasil disimpan ke:
{test_matrix_path}')
:except Exception as e
print(f'Error saat menyimpan file NPZ: {e}')
The variables X_train, y_train, X_test, y_test are now ready for the next steps #
(SMOTE, Model Training)
----- CHI SQUARE FEATURE SELECTION ----- #
print("\n== CHI SQUARE FEATURE SELECTION ==")
from sklearn.feature_selection import SelectKBest, chi2
import pandas as pd
import joblib
import os
Lakukan Chi-squared feature selection pada data training (sebelum SMOTE) #
Pilih jumlah fitur terbaik yang diinginkan #
k_features = 1000
selector = SelectKBest(chi2, k=k_features)
Fit the selector on the original training data and transform both training and #
testing data
Gunakan X_train (setelah TF-IDF) dan y_train (sebelum SMOTE) untuk #
fitting
Pastikan X_train dan y_train tersedia dari langkah Split data & TF-IDF #
:()if 'X_train' in locals() and 'y_train' in locals() and 'X_test' in locals
Tambahkan print statement untuk jumlah fitur awal #
print('Banyaknya fitur awal:', X_train.shape[1])
print("- Melakukan fitting selector pada data training (sebelum SMOTE)...")

```

```

selector.fit(X_train, y_train)
print("- Menerapkan transformasi pada data training dan testing...")
Transform both training and testing data using the fitted selector #
X_train_selected = selector.transform(X_train)
X_test_selected = selector.transform(X_test)
Tambahkan print statement untuk jumlah fitur setelah seleksi #
print('Banyaknya fitur setelah di seleksi:', X_train_selected.shape[1])
print(f'- Bentuk X_train setelah seleksi fitur: {X_train_selected.shape}')
print(f'- Bentuk X_test setelah seleksi fitur: {X_test_selected.shape}')
Opsional: Melihat fitur-fitur yang terpilih #
Dapatkan indeks fitur yang terpilih #
selected_feature_indices = selector.get_support(indices=True)
Dapatkan nama fitur dari vectorizer asli (Pastikan tfidf_vectorizer tersedia) #
:if 'tfidf_vectorizer' in locals() and tfidf_vectorizer is not None
()feature_names = tfidf_vectorizer.get_feature_names_out
selected_feature_names = feature_names[selected_feature_indices]
print(f'\n- Jumlah fitur yang terpilih: {len(selected_feature_names)}')
print("\n- Contoh 20 nama fitur yang terpilih:")
print(selected_feature_names[:20])
Tampilkan cuplikan matriks fitur yang terpilih (Training Data) #
print("\nCuplikan matriks fitur terpilih (Training Data):")
)selected_train_display = pd.DataFrame
,(X_train_selected[:5, :min(10, k_features)].toarray
columns=selected_feature_names[:min(10, k_features)]
(
display(selected_train_display)
Tampilkan cuplikan matriks fitur yang terpilih (Testing Data) #
print("\nCuplikan matriks fitur terpilih (Testing Data):")
)selected_test_display = pd.DataFrame
,(X_test_selected[:5, :min(10, k_features)].toarray
columns=selected_feature_names[:min(10, k_features)]
(
display(selected_test_display)
Simpan selektor (opsional) #
'selector_path = '/content/drive/MyDrive/SKRIPSI/chi2_selector.pkl
:try
joblib.dump(selector, selector_path)
print(f'\nChi-Square selector berhasil disimpan ke: {selector_path}')
:except Exception as e
print(f'Error saat menyimpan selector: {e}')
:else
print("\n- Objek tfidf_vectorizer tidak ditemukan. Tidak dapat
menampilkan nama fitur terpilih atau cuplikan data.")
Variabel X_train_selected dan X_test_selected sekarang berisi fitur-fitur #
yang sudah diseleksi (sebelum SMOTE)
:else
print("Error: Variabel X_train, y_train, atau X_test tidak ditemukan.
Pastikan langkah split data dan TF-IDF vectorization sudah dijalankan.")

```

```

from sklearn.neighbors import NearestNeighbors
from scipy.sparse import vstack, csr_matrix
import numpy as np
:def ldw_smote(X_train_selected, y_train, k=5)
    """
    Local Density Weighting SMOTE (LDW-SMOTE)
    X_train_selected : matriks fitur (sparse) setelah TF-IDF + Chi-Square
    y_train          : label (Series / array)
    k                : jumlah tetangga minoritas untuk hitung density
    """

    Hapus inisialisasi rng karena akan menggunakan np.random secara #
    langsung
    (rng = np.random.default_rng #
    Pisahkan kelas mayoritas dan minoritas .\ #
    y_train_array = y_train.values if hasattr(y_train, "values") else
    np.array(y_train)
    minority_mask = (y_train_array == 1)
    majority_mask = ~minority_mask
    X_minority = X_train_selected[minority_mask]
    X_majority = X_train_selected[majority_mask]
    [·]n_min = X_minority.shape
    [·]n_maj = X_majority.shape
    print(f'Jumlah sampel minoritas sebelum SMOTE : {n_min}')
    print(f'Jumlah sampel mayoritas sebelum SMOTE : {n_maj}')
    butuh minimal k+1 sampel minoritas #
    :if n_min < k + 1
    raise ValueError(f'Jumlah sampel minoritas ({n_min}) terlalu sedikit
    untuk k = {k}')
    Hitung jarak ke k tetangga minoritas terdekat .\ #
    nn = NearestNeighbors(n_neighbors=k+1, metric='euclidean')
    nn.fit(X_minority)
    distances, indices = nn.kneighbors(X_minority)
    jarak rata-rata ke k tetangga (tidak termasuk diri sendiri) #
    avg_dist = distances[:, 1:].mean(axis=1)
    Hitung density, rarity, dan weight .\ #
    density = 1.0 / (1.0 + avg_dist) # semakin kecil jarak → density besar
    rarity = 1.0 / density # kebalikan density (semakin jarang →
    nilai besar)
    weights = rarity / rarity.sum() # normalisasi
    print("\nStatistik density / rarity (kelas minoritas):")
    print(f' avg_dist mean : {avg_dist.mean():.4f}')
    print(f' density mean : {density.mean():.4f}')
    print(f' rarity mean : {rarity.mean():.4f}')
    Tentukan total sampel sintetis agar kelas seimbang .\ #
    n_synth_total = n_maj - n_min
    print(f'\nTotal sampel sintetis yang akan dibuat: {n_synth_total}')
    jumlah sampel sintetis per titik minoritas (dibulatkan ke bawah) #
    n_synth_per_point = np.floor(weights * n_synth_total).astype(int)

```

```

distribusikan sisa jika pembulatan menyebabkan kurang #
)diff = n_synth_total - n_synth_per_point.sum
:if diff > 0
extra_idx = np.random.choice( # Menggunakan np.random.choice
langsung
,np.arange(n_min)
,size=diff
,replace=True
p=weights
(
:for i in extra_idx
n_synth_per_point[i] += 1
Generate sampel sintetis .° #
[] = synth_samples
:for i in range(n_min)
n_i = n_synth_per_point[i]
:if n_i <= 0
continue
xi_sparse = X_minority[i:i+1, :]
neigh_idx = indices[i, 1:] # k tetangga (tanpa diri sendiri)
:for _ in range(n_i)
j = np.random.choice(neigh_idx) # Menggunakan np.random.choice
langsung
xj_sparse = X_minority[j:j+1, :]
delta = np.random.rand() # Menggunakan np.random.rand() langsung
Operasi langsung pada matriks sparse #
x_new_sparse = (1 - delta * weights[i]) * xi_sparse + (delta *
weights[i]) * xj_sparse
synth_samples.append(x_new_sparse)
:if len(synth_samples) == 0
raise ValueError("Tidak ada sampel sintetis yang berhasil dibuat.")
Gabungkan semua sampel sintetis sparse menjadi satu matriks sparse #
X_synth = vstack(synth_samples)
y_synth = np.ones(X_synth.shape[0], dtype=y_train_array.dtype)
Gabungkan dengan data training asli .↗ #
X_train_ldw = vstack([X_train_selected, X_synth])
y_train_ldw = np.concatenate([y_train_array, y_synth])
print(f"\nBentuk X_train setelah LDW-SMOTE : {X_train_ldw.shape}")
unique, counts = np.unique(y_train_ldw, return_counts=True)
print("Distribusi label setelah LDW-SMOTE :", dict(zip(unique, counts)))
return X_train_ldw, y_train_ldw
----- LDW-SMOTE ( ----- #
print("\n== Tahap: LDW-SMOTE ==")
:()if 'X_train_selected' in locals() and 'y_train' in locals
)X_train_smote, y_train_smote = ldw_smote
,X_train_selected
,y_train
k=5,          # sama seperti contoh matematik (5 tetangga)

```

```
(
:else
print("Error: X_train_selected atau y_train tidak ditemukan.")
import joblib
import time # Import time module
PEMODELAN MULTINOMIAL NAIVE BAYES dengan (^ ----- #
----- (CROSS-VALIDATION (Detail Per Fold
print("\n== Tahap 8/8: PEMODELAN MULTINOMIAL NAIVE BAYES
dengan CROSS-VALIDATION (Detail Per Fold) ==")
Pastikan X_train_smote dan y_train_smote tersedia dari langkah SMOTE #
:()if 'X_train_smote' in locals() and 'y_train_smote' in locals
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import StratifiedKFold # Import
StratifiedKFold
from sklearn.metrics import classification_report # Import
classification_report
import numpy as np # Import numpy
print("- Melatih model Multinomial Naive Bayes dengan Cross-Validation
dan menampilkan report per fold...")
()model_nb = MultinomialNB
--- Tentukan skema Cross-Validation --- #
Menggunakan StratifiedKFold karena dataset tidak seimbang #
cv_folds = 5 # Anda bisa mengubah jumlah fold jika perlu
skf = StratifiedKFold(n_splits=cv_folds, shuffle=True)
Jalankan Cross-Validation secara manual untuk menampilkan report --- #
--- per fold
[] = fold_results
fold_count = 1
Pastikan X_train_smote adalah numpy array atau scipy sparse matrix #
X_train_smote_array = X_train_smote if hasattr(X_train_smote, 'shape') else
np.array(X_train_smote)
y_train_smote_array = y_train_smote if hasattr(y_train_smote, 'shape') else
np.array(y_train_smote)
print(f'- Menjalankan {cv_folds}-Fold Stratified Cross-Validation...")
for train_index, val_index in skf.split(X_train_smote_array,
:y_train_smote_array)
print(f'\n--- Fold {fold_count}/{cv_folds} ---")
Split data untuk fold saat ini #
X_train_fold, X_val_fold = X_train_smote_array[train_index],
X_train_smote_array[val_index]
y_train_fold, y_val_fold = y_train_smote_array[train_index],
y_train_smote_array[val_index]
Latih model pada data training fold dan ukur waktunya #
()model_nb_fold = MultinomialNB
start_time = time.time() # Start timer
model_nb_fold.fit(X_train_fold, y_train_fold)
end_time = time.time() # End timer
training_time = end_time - start_time
```

```

print(f"Waktu komputasi training (Fold {fold_count}):
{training_time:.2f} detik")
Prediksi pada data validasi fold #
y_val_pred = model_nb_fold.predict(X_val_fold)
Tampilkan Classification Report untuk fold saat ini #
print(classification_report(y_val_fold, y_val_pred))
Simpan metrik utama jika perlu (opsional) #
report_dict = classification_report(y_val_fold, y_val_pred, #
output_dict=True)
fold_results.append(report_dict) #
fold_count += 1
print("\n Detail Cross-Validation per fold selesai.")
Latih model akhir pada seluruh data training setelah SMOTE (Tetap --- #
--- perlu)
Setelah mendapatkan estimasi performa dari CV, latih model pada seluruh #
data training
print("\n- Melatih model akhir pada seluruh data training (setelah
SMOTE)...")
model_nb.fit(X_train_smote, y_train_smote)
print(" Model final berhasil dilatih pada seluruh data training.")
Simpan model akhir (opsional, tetapi praktik yang baik) #
model_path =
"/content/drive/MyDrive/SKRIPSI/final_multinomial_nb_model.pkl
:try
joblib.dump(model_nb, model_path)
print(f"\n Model Multinomial Naive Bayes berhasil disimpan ke:
{model_path}")
:except Exception as e
print(f"Error saat menyimpan model: {e}")
Variabel model_nb sekarang berisi model yang dilatih pada seluruh data #
.training SMOTE-ed
.Variabel X_test_selected dan y_test masih siap untuk evaluasi final #
:else
print("Error: Variabel X_train_smote atau y_train_smote tidak ditemukan.
Pastikan langkah SMOTE sudah dijalankan.")
----- EVALUASI MODEL PADA DATA TESTING ----- #
print("\n== EVALUASI MODEL PADA DATA TESTING ==")
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
import joblib # Import joblib to load the saved model
import time # Import time module for measurement
Path ke model yang disimpan #
model_path =
"/content/drive/MyDrive/SKRIPSI/final_multinomial_nb_model.pkl
Pastikan X_test_selected dan y_test tersedia #
:()if 'X_test_selected' in locals() and 'y_test' in locals
:try
Muat model yang sudah dilatih #

```

```

model_nb = joblib.load(model_path)
print(f'- Model Multinomial Naive Bayes berhasil dimuat dari:
{model_path}')
Lakukan prediksi pada data testing dan ukur waktunya #
start_time_test = time.time() # Start timer for testing
y_test_pred = model_nb.predict(X_test_selected)
end_time_test = time.time() # End timer for testing
testing_time = end_time_test - start_time_test
print(f'- Waktu komputasi prediksi pada data testing: {testing_time:.2f}
detik")
Akurasi #
accuracy_test = accuracy_score(y_test, y_test_pred)
print(f'- Akurasi (Testing): {accuracy_test:.4f}')
Classification Report (Precision, Recall, F1-score) #
print("\n- Classification Report (Testing):")
print(classification_report(y_test, y_test_pred))
Confusion Matrix #
print("\n- Confusion Matrix (Testing):")
cm_test = confusion_matrix(y_test, y_test_pred)
print(cm_test)
:Interpretasi Confusion Matrix #
cm_test[0,0]: True Negatives (Jumlah email Ham di test set yang #
diprediksi benar sebagai Ham)
cm_test[0,1]: False Positives (Jumlah email Ham di test set yang #
diprediksi salah sebagai Spam) -> Type I error
cm_test[1,0]: False Negatives (Jumlah email Spam di test set yang #
diprediksi salah sebagai Ham) -> Type II error
cm_test[1,1]: True Positives (Jumlah email Spam di test set yang #
diprediksi benar sebagai Spam)
:except FileNotFoundError
print(f'Error: File model tidak ditemukan di {model_path}. Pastikan path
benar dan model sudah disimpan.")
:except Exception as e
print(f'Terjadi error saat memuat atau mengevaluasi model: {e}')
:else
print("Error: Variabel X_test_selected atau y_test belum tersedia. Silakan
jalankan langkah seleksi fitur terlebih dahulu.")
'config InlineBackend.figure_format = 'retina%'
----- Tampilkan Confusion Matrix ----- #
print("\n== Confusion Matrix (Testing Data) ==")
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns # Untuk visualisasi yang lebih baik
Pastikan y_test dan y_test_pred tersedia dari langkah evaluasi sebelumnya #
:()if 'y_test' in locals() and 'y_test_pred' in locals
Hitung Confusion Matrix #
cm_test = confusion_matrix(y_test, y_test_pred)
Tampilkan Confusion Matrix #

```

```
print(cm_test)
Opsional: Visualisasikan Confusion Matrix #
plt.figure(figsize=(6, 4))
, 'sns.heatmap(cm_test, annot=True, fmt='d', cmap='Blues
, xticklabels=['Predicted Ham (0)', 'Predicted Spam (1)']
, yticklabels=['Actual Ham (0)', 'Actual Spam (1)']
plt.xlabel('Predicted Label')
plt.ylabel('Actual Label')
plt.title('Confusion Matrix')
plt.show
:else
print("Error: Variabel y_test atau y_test_pred tidak ditemukan. Pastikan
langkah evaluasi model sudah dijalankan.")
```

Lampiran 9 Tabel *Chi-Square***TABEL NILAI KRITIS DISTRIBUSI CHI-SQUARE**

df	0,1	0,05	0,025	0,001	0,005
1	2,705543	3,841459	5,023886	6,634897	7,879439
2	4,605170	5,991465	7,377759	9,210340	10,596635
3	6,251389	7,814728	9,348404	11,344867	12,838156
4	7,779440	9,487729	11,143287	13,276704	14,860259
5	9,236357	11,070498	12,832502	15,086272	16,749602
6	10,644641	12,591587	14,449375	16,811894	18,547584
7	12,017037	14,067140	16,012764	18,475307	20,277740
8	13,361566	15,507313	17,534546	20,090235	21,954955
9	14,683657	16,918978	19,022768	21,665994	23,589351
10	15,987179	18,307038	20,483177	23,209251	25,188180
11	17,275009	19,675138	21,920049	24,724970	26,756849
12	18,549348	21,026070	23,336664	26,216967	28,299519
13	19,811929	22,362032	24,735605	27,688250	29,819471
14	21,064144	23,684791	26,118948	29,141238	31,319350
15	22,307130	24,995790	27,488393	30,577914	32,801321
16	23,541829	26,296228	28,845351	31,999927	34,267187
17	24,769035	27,587112	30,191009	33,408664	35,718466
18	25,989423	28,869299	31,526378	34,805306	37,156451
19	27,203571	30,143527	32,852327	36,190869	38,582257
20	28,411981	31,410433	34,169607	37,566235	39,996846
21	29,615089	32,670573	35,478876	38,932173	41,401065
22	30,813282	33,924438	36,780712	40,289360	42,795655
23	32,006900	35,172462	38,075627	41,638398	44,181275
24	33,196244	36,415029	39,364077	42,979820	45,558512
25	34,381587	37,652484	40,646469	44,314105	46,927890
26	35,563171	38,885139	41,923170	45,641683	48,289882
27	36,741217	40,113272	43,194511	46,962942	49,644915
28	37,915923	41,337138	44,460792	48,278236	50,993376
29	39,087470	42,556968	45,722286	49,587884	52,335618
30	40,256024	43,772972	46,979242	50,892181	53,671962
31	41,421736	44,985343	48,231890	52,191395	55,002704
32	42,584745	46,194260	49,480438	53,485772	56,328115
33	43,745180	47,399884	50,725080	54,775540	57,648445
34	44,903158	48,602367	51,965995	56,060909	58,963926
35	46,058788	49,801850	53,203349	57,342073	60,274771
36	47,212174	50,998460	54,437294	58,619215	61,581179
37	48,363408	52,192320	55,667973	59,892500	62,883335
38	49,512580	53,383541	56,895521	61,162087	64,181412
39	50,659770	54,572228	58,120060	62,428121	65,475571
40	51,805057	55,758479	59,341707	63,690740	66,765962

RIWAYAT HIDUP



Nurus Syafi'ah, lahir di Kabupaten Jombang pada 14 Desember 2003. Penulis merupakan anak pertama dari dua bersaudara dari pasangan Bapak Hasan dan Ibu Ningsih. Pendidikan formal penulis dimulai di RA Miftahul Huda, yang diselesaikan pada tahun 2010. Selanjutnya, penulis melanjutkan pendidikan di SD Negeri Temuwulan dan lulus pada tahun 2016. Pendidikan tingkat menengah pertama ditempuh di SMP Negeri 1 Perak dan lulus pada tahun 2019. Penulis kemudian menyelesaikan pendidikan menengah atas di SMA Negeri 3 Jombang pada tahun 2022. Pada tahun yang sama, penulis melanjutkan pendidikan tinggi di Universitas Islam Negeri Maulana Malik Ibrahim Malang, dengan memilih Program Studi Matematika di Fakultas Sains dan Teknologi. Selama menempuh pendidikan di universitas, penulis aktif berpartisipasi dalam berbagai kegiatan akademik dan non-akademik. Penulis juga terlibat dalam berbagai kegiatan kemahasiswaan dan acara kampus yang memberikan pengalaman dalam pengembangan kemampuan komunikasi, kerja sama tim, serta kepemimpinan. Selain itu, penulis pernah menjabat dalam beberapa organisasi, baik intra kampus maupun ekstra kampus, salah satunya sebagai Ketua Divisi External Public Relation HMPS “Integral” Matematika Periode 2023-2024. Di bidang akademik, penulis juga pernah mengikuti Riset Kompetisi Mahasiswa sebagai bentuk pengembangan kemampuan penelitian. Selain itu, penulis aktif mengikuti berbagai pelatihan serta kegiatan volunteer di luar kampus yang mendukung peningkatan keterampilan dan pengalaman di berbagai bidang.



KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI

Jl. Gajayana No.50 Dinoyo Malang Telp. Fax. (0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : Nurus Syafi'ah
NIM : 220601110057
Fakultas / Jurusan : Sains dan Teknologi / Matematika
Judul Skripsi : Evaluasi Pengaruh *Chi-Square Feature Selection* dan *Oversampling* LDW-SMOTE terhadap Kinerja *Naïve Bayes* untuk Deteksi Email Spam
Pembimbing I : Hisyam Fahmi, M.Kom.
Pembimbing II : Erna Herawati, M.Pd.

No	Tanggal	Hal	Tanda Tangan
1.	8 September 2025	Konsultasi Bab I	1.
2.	12 September 2025	Konsultasi Bab I. dan II	2.
3.	24 September 2025	Konsultasi Bab I, II. dan III	3.
4.	27 Oktobe 2025	Konsultasi Bab I, II. dan III	4.
5.	20 Oktober 2025	Konsultasi Kajian Agama. Bab I. dan II	5.
6.	28 Oktober 2025	Konsultasi Bab I, II. dan III	6.
7.	24 oktober 2025	Konsultasi Kajian Agama. Bab I, II. dan III	7.
8.	29 Oktober 2025	ACC Bab I, II. dan III	8.
9.	29 Oktober 2025	ACC Kajian Agama Bab I dan II	9.
10.	10 November 2025	ACC Seminar Proposal	10.
11.	12 November 2025	Konsultasi Revisi Seminar Proposal	11.
12.	16 Desember 2025	Konsultasi Bab IV dan V	12.
13.	18 Februari 2026	Konsultasi Bab IV dan V	13.
14.	12 Maret 2026	Konsultasi Bab IV dan V	14.
15.	13 Maret 2026	Konsultasi Bab IV dan V	15.
16.	6 April 2026	Konsultasi Kajian Agama Bab IV	16.
17.	2 April 2026	ACC Bab IV dan V	17.



**KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI**

Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

18.	8 April 2026	ACC Kajian Agama Bab IV	18.
19.	14 April 2026	ACC Seminar Hasil	19.
20.	21 April 2026	Konsultasi Revisi Seminar Hasil	20.
21.	27 April 2026	ACC Sidang Skripsi	21.
22.	12 Mei 2026	ACC Keseluruhan	22.

Malang, 12 Mei 2026

Mengetahui,

Ketua Program Studi Matematika



Dr. H. H. Rozi, M.Si.

NIP. 19800527 200801 1 012