PERBANDINGAN METODE *NAIVE BAYES CLASSIFIER* DAN *SUPPORT VECTOR MACHINE* DALAM KLASIFIKASI KEASLIAN LOWONGAN PEKERJAAN DI MEDIA SOSIAL

TUGAS AKHIR

Diajukan Oleh:

Maulinatul Hakiki

NIM. 210705015

Mahasiswa Fakultas Sains Dan Teknologi

Program Studi Teknologi Informasi

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI AR-RANIRY
BANDA ACEH
2026 M / 1447 H

LEMBAR PERSETUJUAN

PERBANDINGAN METODE *NAIVE BAYES CLASSIFIER* DAN *SUPPORT VECTOR MACHINE* DALAM KLASIFIKASI KEASLIAN LOWONGAN PEKERJAAN DI MEDIA SOSIAL

TUGAS AKHIR

Diajukan Kepada Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Ar-Raniry Banda Aceh
Sebagai Salah Satu Beban Studi Memperoleh Gelar Sarjana
pada Program Studi Prodi Teknologi Informasi

Oleh:
MAULINATUL HAKIKI
210705015

Mahasiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

Disetujui untuk Dimunaqasyahkan Oleh:

Pembimbing I,	Pembimbing II,
Khairan AR, M.Kom	**Sarini Vita Dewi, M.Eng**
NIP. 198607040201403100 1	**NIP. 198712222022032001**

Mengetahui,
Ketua Program Studi Teknologi Informasi

Malahayati, M.T
NIP. 198301272015032003

LEMBAR PENGESAHAN

PERBANDINGAN METODE *NAIVE BAYES CLASSIFIER* DAN *SUPPORT VECTOR MACHINE* DALAM KLASIFIKASI KEASLIAN LOWONGAN PEKERJAAN DI MEDIA SOSIAL

TUGAS AKHIR

Telah Diuji Oleh Panitia Ujian Munaqasyah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Beban Studi Program Sarjana (S1) Dalam
Program Studi Prodi Teknologi Informasi

Pada Hari/Tanggal: Selasa, 24 Februari 2026
6 Ramadhan 1447 H
Di Darussalam, Banda Aceh

Panitia Ujian Munaqasyah Tugas Akhir:

Ketua,	Sekretaris,
Khairan AR, M.Kom	**Sarini Vita Dewi, M.Eng**
NIP. 198607040201403100l	**NIP. 198712222022032001**
Penguji I,	Penguji II,
Dr. Hendri Ahmadian, S.Si., M.I.M	**Ridha Ilahi, M.T**
NIP. 19831042014031002	**NIP. 198301272015032003**

Mengetahui,
Dekan Fakultas Sains Dan Teknologi
UIN Ar-Raniry Banda Aceh

Prof. Dr. Ir. Muhammad Dirhamsyah, M.T., IPU
NIP. 196210021988111001

LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama	: Maulinatul Hakiki
Nim	: 210705015
Program Studi	: Teknologi Informasi
Fakultas	: Sains dan Teknologi
Judul	: Perbandingan Metode *Naive Bayes Classifier* dan *Support Vector Machine* dalam Klasifikasi Keaslian Lowongan Pekerjaan di Media Sosial

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggungjawabkan;
2. Tidak melakukan plagiasi terhadap naskah karya orang lain:
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya;
4. Tidak memanipulasi dan memalsukan data;
5. Mengerjakan sendiri karya ini dan mampu bertanggungjawab atas karya ini.

Bila dikemudian hari ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat dipertanggungjawabkan dan ternyata memang ditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya siap dikenai sanksi berdasarkan aturan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

Demikian pernyataan ini saya buat dengan sesungguhnya dan tanpa paksaan dari pihak manapun.

Banda Aceh, 24 Februari 2026

Yang Menyatakan

Maulinatul Hakiki

ABSTRAK

Nama : Maulinatul Hakiki

Nim : 210705015

Program Studi : Teknologi Informasi

Judul : Perbandingan Metode *Naive Bayes Classifier* dan *Support Vector Machine* dalam Klasifikasi Keaslian Lowongan Pekerjaan di Media Sosial

Tanggal Sidang : 24 Februari 2026 M/ 1447 H

Jumlah Halaman : 83

Pembimbing I : Khairan AR, M.Kom

Pembimbing II : Sarini Vita Dewi, M.Eng

Kata Kunci : Lowongan Kerja Palsu, *Text Mining*, *Naive Bayes Classifier*, *Support Vector Machine*, Klasifikasi.

Perkembangan media sosial sebagai sarana penyebaran informasi lowongan pekerjaan memberikan kemudahan bagi pencari kerja, namun juga meningkatkan risiko beredarnya lowongan kerja palsu. Penelitian ini bertujuan mengimplementasikan dan membandingkan metode *Naive Bayes Classifier* dan *Support Vector Machine* (SVM) dalam mengklasifikasikan lowongan pekerjaan sebagai asli atau palsu. Dataset yang digunakan berasal dari Kaggle "*Fake Job Posting Prediction*". Proses penelitian meliputi tahapan data mining CRISP-DM, *preprocessing* teks *(cleaning, case folding, tokenizing, filtering, stemming)*, ekstraksi fitur menggunakan TF-IDF, serta pengujian model dengan 10-*Fold Cross Validation*. Evaluasi dilakukan menggunakan metrik akurasi, *precision, recall, F1-score*, dan *confusion matrix*. Hasil penelitian menunjukkan kedua algoritma mampu melakukan klasifikasi dengan baik, namun terdapat perbedaan performa yang menunjukkan metode yang lebih efektif dalam mendeteksi lowongan kerja palsu. Penelitian ini diharapkan dapat mendukung pengembangan sistem deteksi penipuan digital di bidang ketenagakerjaan.

ABSTRACT

Name	: Maulinatul Hakiki
Nim	: 210705015
Departement	: *Information Technology*
Title	: *Comparison of Naive Bayes Classifier and Support Vector Machine Methods in Classifying the Authenticity of Job Vacancies on Social Media*
Date	: *February 24, 2026 M / 1447 H*
Thesis Page	: 83
Supervisor I	: Khairan AR, M.Kom
Supervisor II	: Sarini Vita Dewi, M.Eng
Keyword	: *Fake Job Vacancies, Text Mining, Naive Bayes Classifier, Support Vector Machine, Classification*

The rapid development of social media as a platform for disseminating job vacancy information has provided convenience for job seekers; however, it has also increased the risk of fraudulent job postings. This study aims to implement and compare the Naive Bayes Classifier and Support Vector Machine (SVM) methods in classifying job vacancies as genuine or fraudulent. The dataset used in this research was obtained from Kaggle, entitled "Fake Job Posting Prediction." The research process follows the CRISP-DM data mining framework, including text preprocessing (cleaning, case folding, tokenizing, filtering, and stemming), feature extraction using Term Frequency–Inverse Document Frequency (TF-IDF), and model evaluation using 10-Fold Cross Validation. Performance evaluation was conducted using accuracy, precision, recall, F1-score, and confusion matrix metrics. The results indicate that both algorithms are capable of performing classification effectively; however, differences in performance metrics show that one method is more optimal in detecting fraudulent job postings. This research is expected to contribute to the development of digital fraud detection systems in the employment sector.

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah SWT atas segala rahmat, taufik, dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Perbandingan Metode *Naive Bayes Classifier* dan *Support Vector Machine* dalam Klasifikasi Keaslian Lowongan Pekerjaan di Media Sosial". Shalawat serta salam semoga senantiasa tercurah kepada Nabi Muhammad SAW beserta keluarga, sahabat, dan seluruh umatnya hingga akhir zaman.

Tugas Akhir ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Teknologi Informasi, Fakultas Sains dan Teknologi, Universitas Islam Negeri Ar-Raniry Banda Aceh. Dalam proses penyusunan hingga penyelesaian penelitian ini, penulis memperoleh banyak bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Penulis mengucapkan terima kasih kepada diri sendiri yang telah berusaha, bertahan, dan berkomitmen menyelesaikan setiap tahapan penelitian hingga Tugas Akhir ini dapat diselesaikan.
2. Kedua orang tua dan keluarga tercinta yang senantiasa memberikan doa, dukungan, serta motivasi sehingga penulis dapat menyelesaikan studi hingga tahap akhir.
3. Bapak Nazaruddin Ahmad, M.T selaku dosen pembimbing akademik yang telah memberikan arahan dan motivasi selama masa perkuliahan.
4. Bapak Khairan AR, M.Kom selaku dosen pembimbing I dan Ibu Sarini Vita Dewi, M.Eng selaku dosen pembimbing II yang telah meluangkan waktu, tenaga, dan pemikiran dalam memberikan arahan serta bimbingan selama penyusunan Tugas Akhir.
5. Bapak Dr. Hendri Ahmadian, S.Si., M.I.M selaku dosen penguji I dan Bapak Ridha Ilahi, M.T selaku dosen penguji II yang telah memberikan kritik, saran konstruktif, serta arahan yang sangat membantu dalam penyempurnaan Tugas Akhir.

6. Ibu Malahayati, M.T selaku Ketua Program Studi Teknologi Informasi dan Bapak Khairan AR, M.Kom selaku Sekretaris Program Studi Teknologi Informasi.
7. Ibu Cut Ida Rahmadiana, S.Si selaku operator Program Studi Teknologi Informasi yang telah membantu penulis dalam proses administrasi akademik.
8. Seluruh dosen Program Studi Teknologi Informasi yang telah memberikan ilmu dan pengalaman selama masa perkuliahan.
9. Teman-teman mahasiswa Teknologi Informasi angkatan 2021 yang telah memberikan dukungan, semangat, serta kebersamaan selama proses perkuliahan dan penyusunan Tugas Akhir.
10. Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu penulis baik secara langsung maupun tidak langsung dalam penyelesaian penelitian ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki keterbatasan dan belum sempurna. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan demi perbaikan di masa mendatang. Semoga penelitian ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan khususnya pada bidang klasifikasi teks dan deteksi penipuan digital.

Banda Aceh, 24 Februari 2026

Maulinatul Hakiki

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi digital yang semakin pesat telah membawa dampak besar dalam berbagai aspek kehidupan, termasuk dalam dunia kerja. Salah satu dampaknya terlihat dari perubahan cara individu dalam mencari informasi lowongan pekerjaan. Sebelumnya pencarian kerja dilakukan secara konvensional melalui media cetak atau dari jaringan pertemanan, sedangkan saat ini masyarakat lebih memilih menggunakan *platform* media sosial seperti LinkedIn, Facebook, atau Instagram untuk menemukan peluang kerja. Berdasarkan survei yang diperoleh dari DataIndonesia.id (Rizaty, 2025), Indonesia bahkan menempati peringkat ke-9 dunia untuk durasi penggunaan media sosial terlama pada kuartal III 2024. Hal ini menjadikan media sosial sebagai sarana utama penyebaran informasi, termasuk lowongan pekerjaan.

Kemudahan tersebut tentu memberikan manfaat besar, seperti menghemat waktu dan memperluas akses terhadap peluang kerja. Namun, di sisi lain, peningkatan penggunaan *platform* digital juga diiringi dengan maraknya kasus penipuan lowongan kerja. Banyak pencari kerja menjadi korban lowongan palsu yang seolah-olah terlihat resmi. Penipuan umumnya dilakukan dengan menggunakan nama perusahaan besar, meminta uang administrasi secara tidak resmi, atau menawarkan pekerjaan di luar negeri dengan mekanisme yang meragukan. Menurut data Direktorat Siber Bareskrim Polri, sejak tahun 2022 hingga 2024 terdapat 823 kasus penipuan online dengan total kerugian mencapai Rp 59 miliar (Naibaho, 2024).

Untuk mengatasi permasalahan tersebut, dibutuhkan solusi teknologi berbasis *data mining* yang mampu mengklasifikasikan keaslian lowongan pekerjaan secara otomatis dan akurat. *Data mining* adalah proses menemukan informasi penting dari kumpulan data besar dan kompleks dengan pendekatan dari bidang seperti statistik, ilmu komputer, kecerdasan buatan, dan matematika untuk mengidentifikasi pola tersembunyi, hubungan, serta wawasan yang tidak langsung terlihat dari permukaan data (Sulianta, 2024). Dalam penerapannya, *data mining* memiliki tahapan

sistematis yang dikenal dengan metode *Cross Industry Standard Process for Data Mining* (CRISP-DM), yang mencakup pemahaman bisnis (*business understanding*), pemahaman data (*data understanding*), persiapan data (*data preparation*), pemodelan (*modeling*), evaluasi (*evaluation*), dan implementasi (*deployment*) (Yudiana et al., 2023).

Berbagai metode dalam *data mining* telah banyak digunakan untuk klasifikasi data teks, termasuk dalam konteks identifikasi keaslian lowongan kerja di media sosial. Beberapa algoritma populer yang sering digunakan antara lain *Naive Bayes*, *Bayes Network, K-Nearest Neighbor* (K-NN), *Decision Tree*, dan *Support Vector Machine* (SVM). Dalam penelitian ini, penulis memfokuskan pada dua algoritma *machine learning* yang diterapkan dalam konteks *data mining*, yaitu *Naive Bayes* dan SVM. Keduanya dikenal efektif dalam klasifikasi berbasis teks.

Naive Bayes unggul dalam hal kesederhanaan dan kecepatan pemrosesan pada dataset berukuran besar, sedangkan SVM memiliki kelebihan dalam mengklasifikasikan data berdimensi tinggi serta memisahkan kelas dengan margin maksimal. Penelitian sebelumnya terkait klasifikasi jenis mobil berdasarkan data produksi GAIKINDO tahun 2021 menunjukkan bahwa *Naive Bayes* memberikan performa lebih tinggi dibandingkan SVM. Dari 574 data dengan atribut yang telah disederhanakan, *Naive Bayes* menghasilkan akurasi 97,39% dan precision 97,33%, sementara SVM memperoleh akurasi 96,52% dengan recall lebih tinggi, yaitu 100%. Temuan ini menunjukkan bahwa kedua algoritma sama-sama relevan untuk tugas klasifikasi, namun *Naive Bayes* memiliki keunggulan pada akurasi dan precision, sedangkan SVM unggul dalam hal recall (Paul et al., 2023).

Penulis memanfaatkan dataset *Fake Job Posting Prediction* yang diperoleh dari *platform* Kaggle. Dataset tersebut berisi kumpulan data lowongan pekerjaan yang telah dilabeli sebagai lowongan asli maupun palsu. Atribut yang tersedia meliputi judul pekerjaan, deskripsi, lokasi, perusahaan, serta informasi lain yang relevan. Mengingat data yang digunakan bersifat teks dan tidak terstruktur, maka diterapkan teknik *text mining* agar dapat mengekstraksi informasi penting sebelum proses klasifikasi dilakukan. Implementasi model dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan Visual Studio Code sebagai editor lokal yang mendukung pengembangan dan pengujian kode. Tools tersebut

memudahkan penulis dalam membangun dan membandingkan kinerja metode *Naive Bayes* dan SVM dalam klasifikasi keaslian lowongan pekerjaan di media sosial.

Dengan melakukan perbandingan antara algoritma *Naive Bayes* dan SVM, penelitian ini bertujuan untuk mengetahui metode mana yang lebih efektif dalam mengklasifikasikan keaslian lowongan pekerjaan di media sosial, serta memberikan kontribusi dalam upaya mengurangi risiko penipuan digital di bidang ketenagakerjaan.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang, rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana implementasi metode *Naive Bayes* dan SVM dalam mengklasifikasikan keaslian lowongan pekerjaan yang tersebar melalui media sosial?
2. Bagaimana perbandingan performa algoritma *Naive Bayes* dan SVM berdasarkan metrik evaluasi dalam mengidentifikasi lowongan kerja palsu?

1.3 Tujuan Penelitian

Adapun tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut:

1. Untuk mengetahui bagaimana implementasi algoritma SVM dan *Naive Bayes* dalam melakukan klasifikasi keaslian lowongan pekerjaan yang tersebar di media sosial.
2. Untuk membandingkan performa kedua algoritma berdasarkan metrik evaluasi klasifikasi seperti akurasi, *precision, recall*, dan *F1-score.*

1.4 Batasan Masalah

Agar penelitian ini tetap fokus dan sesuai dengan lingkup yang ditetapkan, maka ditentukan beberapa batasan sebagai berikut:

1. Penelitian ini difokuskan pada data lowongan pekerjaan di media sosial yang diperoleh dari Kaggle, tanpa mencakup *platform* digital lainnya seperti situs pencarian kerja atau aplikasi khusus rekrutmen.
2. Metode analisis yang dibandingkan hanya mencakup dua algoritma klasifikasi, yaitu *Naive Bayes* dan SVM.

3. Data yang digunakan berupa deskripsi lowongan pekerjaan yang telah ditandai sebagai asli atau palsu pada dataset publik.
4. Label data terdiri dari dua kategori yaitu “asli” dan “palsu” pada dataset yang diperoleh dari Kaggle.
5. *Tools* implementasi menggunakan Python dengan Visual Studio Code sebagai lingkungan pengembangan untuk mengolah data serta mengimplementasikan algoritma klasifikasi.

1.5 Manfaat Penelitian

Penelitian ini ditujukan untuk memberikan sejumlah manfaat, di antaranya:

1. Memberikan pemahaman mengenai penerapan algoritma *Naive Bayes* dan SVM dalam proses klasifikasi teks, khususnya dalam konteks identifikasi keaslian informasi lowongan kerja di media sosial.
2. Menyajikan gambaran hasil perbandingan performa kedua metode dalam mendeteksi lowongan pekerjaan palsu, sehingga dapat menjadi referensi dalam pemilihan algoritma yang tepat untuk kasus serupa.
3. Menjadi dasar pengembangan penelitian selanjutnya, seperti eksplorasi algoritma lain atau peningkatan model dengan dataset yang lebih beragam.
4. Memenuhi salah satu syarat kelulusan program studi Strata 1 (S1) di bidang Teknologi Informasi pada Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh, sekaligus berkontribusi pada pengembangan solusi teknologi untuk masalah nyata di masyarakat.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Relevan

Penelitian relevan merujuk pada karya ilmiah terdahulu yang berkaitan dengan topik yang dibahas, digunakan sebagai acuan dalam memahami metode, hasil, dan pendekatan yang telah diterapkan sebelumnya, sekaligus mengidentifikasi celah yang masih dapat dikembangkan lebih lanjut dalam penelitian ini.

Penelitian yang berjudul "Perbandingan Algoritma Naive Bayes dan Support Vector Machine dalam Deteksi Berita Hoax Berbahasa Indonesia" membandingkan kinerja kedua algoritma dalam mengklasifikasikan berita hoax menggunakan 4.599 artikel berlabel dari Twitter dan GitHub, dengan tahap praproses meliputi tokenisasi, *stopword removal*, *stemming*, dan vektorisasi TF-IDF. Hasil evaluasi menunjukkan SVM unggul dalam akurasi (70,87%) dan F1-*score* (82%), sedangkan Naive Bayes memperoleh akurasi 66,52% namun unggul dalam *recall* (99%) dengan waktu pelatihan yang lebih cepat. Temuan ini mengindikasikan bahwa pemilihan algoritma bergantung pada kebutuhan, di mana SVM lebih tepat untuk meminimalkan kesalahan klasifikasi secara keseluruhan, sementara Naive Bayes lebih efektif dalam memaksimalkan deteksi hoaks yang berpotensi terlewat (Mumbunan et al., 2025).

Penelitian lain membandingkan algoritma Naive Bayes Classifier dan *Support Vector Machine* untuk mendeteksi *cyber harassment* di Twitter, menggunakan model *Complement Naive Bayes* dan *Support Vector Classification* (SVC) berbasis Python dengan pembagian data 80% pelatihan dan 20% pengujian. Hasil evaluasi menunjukkan SVM memperoleh akurasi tertinggi sebesar 89,56% dengan *recall* 94,5%, sementara Naive Bayes mencapai akurasi 86,30% dengan *recall* 87,21%. Temuan ini menegaskan bahwa SVM lebih efektif dalam mengidentifikasi kasus pelecehan sehingga lebih sesuai diterapkan pada sistem moderasi konten di media sosial (Saputri et al., 2024).

Studi lain yang relevan membahas deteksi penipuan lowongan kerja yang marak di berbagai platform digital seperti Facebook, email, situs web, dan Google,

yang kerap dimanfaatkan pihak tidak bertanggung jawab sehingga merugikan pencari kerja secara materi maupun non-materi. Penelitian ini menerapkan algoritma Naive Bayes pada dataset berisi 106 data bukan penipuan dan 94 data penipuan, dengan kata atau frasa yang sering muncul pada iklan palsu sebagai fitur utama klasifikasi. Hasil pengujian menunjukkan akurasi sebesar 89%, dengan performa kategori bukan penipuan mencapai *precision* 83%, *recall* 100%, dan F1-*score* 91%, serta kategori penipuan memperoleh *precision* 100%, *recall* 76%, dan F1-*score* 86%. Temuan ini mengonfirmasi efektivitas Naive Bayes dalam mendeteksi penipuan lowongan kerja, khususnya pada dataset yang seimbang (Dzikri et al., 2024).

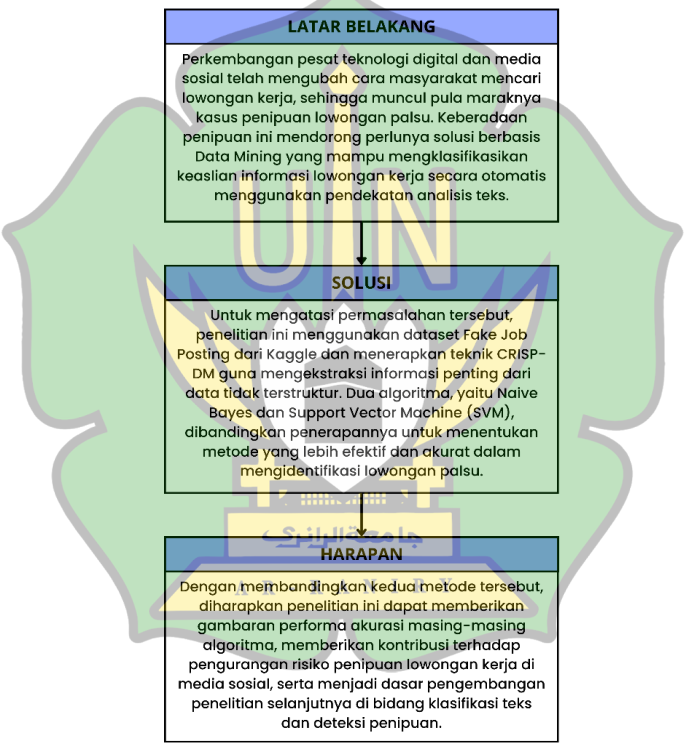
Penelitian berjudul "Perbandingan Implementasi *Machine Learning* Menggunakan Metode KNN, Naive Bayes, dan *Logistic Regression* untuk Mengklasifikasi Penyakit Diabetes" membandingkan kinerja tiga algoritma klasifikasi pada dataset diabetes dari Kaggle dengan dua skema pembagian data, yaitu 70:30 dan 80:20. Penelitian ini menerapkan ekstraksi fitur *Principal Component Analysis* (PCA) dengan *threshold* 80% yang menghasilkan lima fitur utama sebagai masukan model. Hasil evaluasi menunjukkan Naive Bayes memberikan performa terbaik dengan akurasi tertinggi 79% pada pembagian 80:20, diikuti *Logistic Regression* dengan akurasi 73% dan KNN sebesar 71% pada skema yang sama. Temuan ini menegaskan bahwa Naive Bayes merupakan algoritma paling unggul dengan performa yang stabil pada kedua skema pembagian data untuk tugas klasifikasi penyakit diabetes (Nasien et al., 2024).

Penelitian berjudul "Analisis Perbandingan Algoritma *Support Vector Machine*, Naive Bayes, dan Regresi Logistik untuk Memprediksi Donor Darah" membandingkan performa ketiga algoritma dalam memprediksi kemungkinan seorang sukarelawan kembali mendonorkan darah, menggunakan dataset dari Layanan Transfusi Darah Kota Hsin-Chu, Taiwan yang tersedia di UCI Repository. Data diolah melalui normalisasi Z-*score* untuk menyeragamkan skala, kemudian dievaluasi menggunakan metrik akurasi, *precision*, *recall*, dan F1-*score*. Hasil pengujian menunjukkan SVM memberikan performa terbaik dengan akurasi 94,79%, mengungguli Naive Bayes (89,90%) dan *Logistic Regression* (82,59%), sekaligus membuktikan bahwa normalisasi Z-*score* berkontribusi pada peningkatan

kualitas model secara keseluruhan. Temuan ini menegaskan SVM sebagai algoritma paling efektif untuk tugas prediksi serupa dan relevan diterapkan dalam sistem pendukung keputusan guna menjaga ketersediaan stok darah (Hendriyana et al., 2022).

2.2 Kerangka Berpikir

Kerangka berpikir pada penelitian ini dirancang untuk menjelaskan hubungan antar konsep, teori, dan variabel yang relevan. Kerangka ini berfungsi sebagai landasan logis yang memandu seluruh alur penelitian, mulai dari pengumpulan data hingga evaluasi hasil. Hubungan antara variabel-variabel penelitian disajikan dalam Gambar 2.1.



Gambar 2. 1 Kerangka Berpikir

2.3 Landasan Teori

2.3.1 Implementasi

Implementasi dalam penelitian ini merujuk pada tahapan eksekusi dari model yang telah dirancang. Proses ini mencakup langkah-langkah sistematis untuk menerapkan algoritma *Naive Bayes* dan SVM pada dataset lowongan pekerjaan.

Tujuannya adalah untuk menguji dan membandingkan performa kedua model, yang merupakan inti dari penelitian ini.

2.3.2 Klasifikasi

Klasifikasi adalah salah satu metode utama dalam *machine learning* yang bertujuan mengelompokkan data ke dalam kategori yang telah ditentukan. Dalam penelitian ini, klasifikasi digunakan untuk mengidentifikasi apakah suatu lowongan pekerjaan termasuk kategori asli atau palsu berdasarkan data teks yang tersedia. Proses ini melibatkan tahapan *preprocessing data*, pembagian data menjadi data latih dan data uji, serta pelatihan model menggunakan algoritma *Naive Bayes* dan *SVM* untuk menghasilkan model prediksi. Performa model kemudian akan dievaluasi menggunakan metrik seperti *akurasi, presisi, recall*, dan *F1-score*.

2.3.3 Media Sosial

Media sosial telah menjadi salah satu sumber utama informasi lowongan kerja, dengan platform seperti LinkedIn, Facebook, dan Instagram yang banyak dimanfaatkan untuk penyebarannya secara luas. Namun, sifatnya yang terbuka dan minimnya kontrol konten menjadi celah penyebaran informasi palsu, diperparah dengan kemudahan berbagi dan penggunaan akun samaran yang meningkatkan risiko penipuan lowongan kerja. Kondisi ini menjadi dasar pentingnya penelitian mengenai deteksi keaslian lowongan kerja di media sosial.

2.3.4 Lowongan Kerja

Lowongan kerja merupakan peluang yang ditawarkan kepada individu untuk mengisi posisi tertentu di suatu perusahaan atau lembaga, dengan syarat dan kriteria yang disesuaikan dengan kebutuhan pemberi kerja (Fauzi, 2023). Informasi rekrutmen umumnya mencakup nama perusahaan, posisi, deskripsi pekerjaan, kualifikasi, lokasi, dan cara melamar.

Penyebaran lowongan yang kini merambah media sosial membawa kemudahan akses sekaligus risiko penyalahgunaan oleh pihak tidak bertanggung jawab. Lowongan kerja palsu umumnya ditandai dengan bahasa tidak resmi, kualifikasi terlalu umum, tawaran gaji tinggi tanpa rincian jelas, informasi perusahaan tidak lengkap, hingga permintaan transfer uang kepada pelamar. Praktik penipuan ini dapat merugikan pencari kerja dalam berbagai bentuk, mulai dari

kerugian finansial, pencurian data pribadi, hingga risiko eksploitasi tenaga kerja ilegal.

2.3.5 *Machine learning*

Machine learning (ML) merupakan cabang kecerdasan buatan (*Artificial Intelligence*) yang memungkinkan komputer belajar dari data tanpa perlu diprogram secara eksplisit untuk setiap tugas tertentu. Melalui pendekatan ini, sistem dilatih untuk mengenali pola dan menghasilkan keputusan berdasarkan data yang dianalisis, sehingga kinerjanya terus meningkat seiring bertambahnya pengalaman (Bintoro et al., 2024). Tujuan utamanya adalah menghasilkan prediksi yang akurat melalui penyesuaian model terhadap pola dalam data, suatu pendekatan yang berakar dari perkembangan kecerdasan buatan sejak tahun 1950-an (Indra, 2024).

Berdasarkan pendekatan pembelajarannya, *machine learning* dibagi menjadi tiga kategori utama, yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning*. Pemahaman terhadap karakteristik masing-masing kategori menjadi hal yang penting sebelum menentukan metode yang tepat untuk menyelesaikan suatu permasalahan (Wijoyo A et al., 2024).

Supervised learning melatih algoritma menggunakan data berlabel sehingga sistem dapat mempelajari hubungan antara masukan dan keluaran untuk memprediksi label pada data baru. Metode ini mencakup dua tugas utama, yaitu *classification* untuk keluaran berupa kategori dan *regression* untuk prediksi nilai kontinu. Beberapa algoritma yang umum digunakan antara lain Naive Bayes, SVM, *Decision Tree*, *Logistic Regression*, dan *K-Nearest Neighbor*. Penelitian ini berfokus pada *classification* dengan pendekatan *supervised learning*.

Berbeda dengan *supervised learning*, *unsupervised learning* bekerja tanpa data berlabel dan bertujuan menemukan pola tersembunyi atau pengelompokan alami dalam data, umumnya digunakan pada tugas *clustering* dan analisis asosiasi. Sementara itu, *reinforcement learning* mengandalkan interaksi langsung secara dinamis, dimana model belajar memperbaiki keputusan secara berkelanjutan berdasarkan umpan balik berupa *reward* untuk memaksimalkan hasil, pendekatan ini sering diterapkan pada sistem robotika, navigasi, dan pengembangan *game*.

2.3.6 *Data Mining*

Data mining, yang juga dikenal *Knowledge Discovery in Database* (KDD) adalah proses menemukan pola atau informasi bernilai dari kumpulan data yang sangat besar dengan bantuan metode cerdas, termasuk teknik dari statistika dan *machine learning*. Tujuan utamanya adalah untuk memperoleh wawasan baru dari data melalui identifikasi pola, tren, dan korelasi yang tidak tampak secara langsung (Wahyuddin S et al., 2023).

Proses *data mining* seringkali mengikuti kerangka kerja standar, salah satunya adalah *Cross-Industry Standard Process for Data Mining* (CRISP-DM) yang memiliki enam tahap utama, yaitu:

1. *Business Understanding*

   Tahap ini berfokus pada pemahaman masalah. Dalam penelitian ini, tahap ini mencakup pemahaman kasus penipuan lowongan kerja di media sosial dan pentingnya sistem klasifikasi otomatis.

2. *Data Understanding*

   Tahap ini melibatkan pengumpulan dan eksplorasi data awal. Peneliti mempelajari karakteristik data lowongan kerja, seperti judul, deskripsi, perusahaan, dan label asli/palsu.

3. *Data Preparation*

   Pada tahap ini, data disiapkan untuk proses pemodelan. Kegiatannya meliputi pembersihan data (*handling missing values*), transformasi data (seperti tokenisasi), dan pemilihan atribut yang relevan.

4. *Modeling*

   Tahapan ini menerapkan algoritma *data mining* untuk membangun model analisis berdasarkan data yang telah disiapkan. Dalam penelitian ini, algoritma yang digunakan adalah *Naive Bayes* dan SVM untuk klasifikasi.

5. *Evaluation*

   Tahap evaluasi dilakukan untuk menilai performa model dalam menjawab permasalahan yang telah dirumuskan. Evaluasi mencakup *akurasi*, *precision*, *recall*, dan *F1-score* untuk memastikan model sudah optimal.

6. *Deployment*

Tahapan akhir adalah penerapan model yang telah dibangun ke dalam lingkungan nyata. Meskipun penelitian ini bersifat akademik, tahap ini tetap menjadi acuan penting sebagai panduan implementasi model di dunia nyata. Selain tahapan tersebut, proses data mining juga dapat dibedakan ke dalam beberapa jenis berdasarkan fungsi analisis yang ingin dicapai, di mana setiap jenis memiliki peran tersendiri dalam menggali informasi dari data yang tersedia, berikut penjelasan beberapa jenis data mining yang relevan (Mesran et al., 2023):

1. *Clustering*

Clustering merupakan teknik pengelompokan data berdasarkan tingkat kemiripan, di mana data dalam satu klaster memiliki karakteristik yang lebih serupa satu sama lain dibandingkan dengan data pada klaster lain. Pengelompokan dilakukan secara otomatis melalui algoritma seperti K-Means, DBSCAN, AHC, K-Medoids, atau SOM, sehingga pola dan struktur alami dalam data dapat teridentifikasi dengan lebih mudah.

2. Klasifikasi

Klasifikasi merupakan proses pengelompokan data ke dalam kategori tertentu berdasarkan atribut yang dimilikinya, dengan memanfaatkan algoritma *machine learning* yang belajar dari data berlabel untuk menghasilkan model prediktif. Algoritma yang umum digunakan antara lain Naive Bayes, *Decision Tree*, KNN, *Random Forest*, dan SVM. Dalam penelitian ini, klasifikasi menjadi fungsi utama untuk membedakan data lowongan kerja ke dalam dua kelas, yaitu lowongan asli dan lowongan palsu.

3. Asosiasi

Asosiasi merupakan teknik yang digunakan untuk menemukan hubungan antar item dalam sebuah dataset. Metode ini umum digunakan dalam analisis keranjang belanja, seperti menemukan kecenderungan bahwa pelanggan yang membeli produk A juga sering membeli produk B. Pada data mining, teknik asosiasi berguna untuk mengidentifikasi pola keterkaitan yang muncul secara

konsisten. Beberapa algoritma yang sering digunakan antara lain *Apriori, FP-Growth, dan Generalized Sequential Pattern* (GSP).

4. Prediksi

Prediksi digunakan untuk memperkirakan kejadian di masa depan berdasarkan pola data historis, melalui metode statistik atau model *machine learning* yang mempelajari pola terdahulu untuk menghasilkan proyeksi mendatang. Meskipun hasilnya bersifat perkiraan dan tidak selalu akurat, teknik ini sangat membantu dalam pengambilan keputusan strategis di berbagai bidang seperti bisnis, teknologi, dan penelitian ilmiah, dengan contoh metode yang umum digunakan yaitu *Decision Tree* dan KNN.

5. Estimasi

Estimasi merupakan proses memperkirakan nilai suatu variabel atau parameter yang tidak diketahui secara pasti berdasarkan data yang tersedia. Teknik ini banyak diterapkan dalam perencanaan proyek, penganggaran, analisis tren, hingga pengambilan keputusan bisnis, dengan pendekatan seperti *Point Estimation*, *Confidence Interval*, maupun model regresi seperti *Simple Linear Regression* dan *Multiple Regression*. Mengingat hasil estimasi dapat mengandung ketidakpastian, proses validasi menjadi langkah penting untuk memastikan keandalan model yang dihasilkan.

2.3.7 *Text Mining*

Text mining merupakan proses ekstraksi informasi dan pola secara otomatis dari kumpulan teks tidak terstruktur, seperti artikel, ulasan, atau postingan media sosial, yang berbeda dari data terformat dalam basis data konvensional (Nisa, 2024). Hasil ekstraksi tersebut dapat dimanfaatkan untuk keperluan analisis, prediksi, maupun pengambilan keputusan yang lebih efektif.

2.3.8 *Naive Bayes Classifier*

Naive Bayes Classifier merupakan algoritma *machine learning* yang banyak digunakan dalam pemrosesan bahasa alami (*Natural Language Processing*/NLP), didasarkan pada teorema probabilitas yang dikembangkan oleh Thomas Bayes untuk memperkirakan kemungkinan suatu kejadian berdasarkan data historis

(Khoirunnisaa et al., 2024). Cara kerjanya mengukur probabilitas suatu data termasuk ke dalam kelas tertentu berdasarkan nilai fitur yang dimilikinya, dengan asumsi bahwa setiap fitur bersifat independen satu sama lain sehingga distribusi probabilitas masing-masing fitur dapat dihitung secara terpisah. Asumsi independensi tersebut menjadikan proses perhitungan lebih efisien dan sederhana meskipun diterapkan pada data berdimensi tinggi (Sekhar et al., 2023).

Secara umum, langkah-langkah kerja *Naive Bayes* adalah sebagai berikut:

1. Menentukan probabilitas awal (*prior*) dari setiap kelas berdasarkan distribusi data pada data latih.
2. Menghitung peluang munculnya setiap fitur pada masing-masing kelas.
3. Menerapkan teorema Bayes untuk memperoleh probabilitas akhir (*posterior*), yaitu peluang suatu data termasuk ke dalam kelas tertentu berdasarkan fitur yang dimilikinya.
4. Menetapkan kelas dengan nilai probabilitas tertinggi sebagai hasil prediksi.

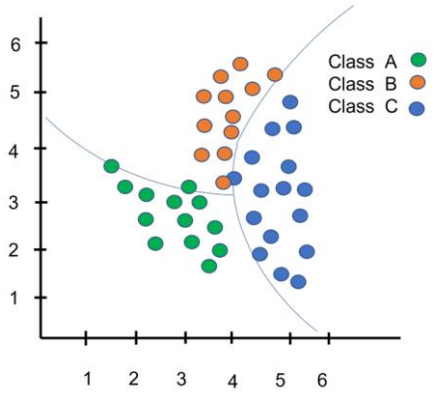
Rumus dasar yang digunakan adalah:

$$P(\frac{y}{X}) = \frac{P\left(\frac{X}{y}\right) P(y)}{P(X)}$$

Penjelasan makna dari masing-masing simbol adalah sebagai berikut:

- P : Probability (peluang), yaitu ukuran seberapa besar kemungkinan suatu kejadian terjadi.
- y : Kelas atau label yang ingin diprediksi.
- x : *Fitur* atau data *input* yang dimiliki oleh suatu data.

Gambaran umum dari struktur kerja *Naive Bayes* dapat dilihat melalui ilustrasi arsitekturnya pada gambar 2.2 berikut.



Gambar 2. 2 Arsitektur *Naive Bayes*

Berdasarkan Gambar 2.2, Naive Bayes mengklasifikasikan data melalui pemetaan probabilitas di mana setiap data lowongan kerja menempati posisi tertentu berdasarkan bobot kata hasil TF-IDF pada koordinat horizontal dan vertikal. Kurva distribusi di atas setiap kelompok kelas menggambarkan wilayah peluang masing-masing kategori, di mana semakin tinggi kurva pada suatu titik maka semakin besar probabilitas data tersebut termasuk dalam kategori terkait. Ketika data baru masuk, algoritma menetapkan label berdasarkan kategori dengan nilai probabilitas tertinggi, sehingga data yang jatuh di wilayah dominan suatu kelas akan secara otomatis memperoleh label kelas tersebut sebagai hasil prediksi akhir.

1. Kelebihan *Naive Bayes*

   Beberapa keunggulan *Naive Bayes* menjadikannya metode yang banyak dimanfaatkan dalam berbagai penelitian klasifikasi, antara lain:

   a. Mudah digunakan karena proses penerapannya sederhana dan tidak memerlukan pelatihan yang rumit.
   b. Mampu mengklasifikasikan baik untuk dua kategori (*biner*) maupun banyak kategori (*multiclass*) dengan hasil yang cukup akurat.
   c. Memiliki kinerja yang cepat dan efisien, bahkan saat mengolah data dalam jumlah besar atau secara *real-time*.
   d. Tidak mudah terpengaruh oleh fitur yang tidak relevan, karena menghitung probabilitas setiap fitur secara terpisah
   e. Sangat efektif untuk pengolahan teks, sehingga sering digunakan dalam klasifikasi dokumen atau penyaringan spam.

f. Dapat mengelola data diskrit maupun data berkelanjutan dengan bantuan distribusi probabilitas seperti *Gaussian* (distribusi normal) atau *Multinomial* (distribusi hitungan kategori).
g. Membutuhkan jumlah data pelatihan yang lebih sedikit dibandingkan metode lain.
h. Dapat menangani data yang hilang dengan pendekatan probabilistik.

2. Kekurangan *Naive Bayes*

Meski memiliki banyak kelebihan, algoritma ini juga memiliki beberapa keterbatasan, di antaranya:

a. Asumsi independensi antar fitur seringkali tidak realistis dalam data dunia nyata.
b. Kurang fleksibel dalam menangkap interaksi antar variabel.
c. Kurang efektif bila terjadi ketidakseimbangan kelas dalam dataset.
d. Cenderung bias terhadap fitur yang sering muncul (frekuensi tinggi).
e. Sensitif terhadap data *outlier* (data menyimpang) yang dapat mempengaruhi perhitungan probabilitas.
f. Kurang mampu menangani fitur berkelanjutan dengan kompleksitas tinggi.
g. Pilihan distribusi probabilitas awal dapat mempengaruhi hasil klasifikasi.
h. Tidak selalu akurat jika data pelatihan terlalu sedikit atau tidak representatif.

2.3.9 *Support Vector Machine*

Support Vector Machine (SVM) adalah algoritma *supervised learning* yang pertama kali diperkenalkan oleh Boser, Guyon, dan Vapnik pada tahun 1992, dikenal efektif untuk menyelesaikan permasalahan klasifikasi terutama pada data berdimensi tinggi (Salsabila, 2022). Cara kerjanya adalah dengan mencari *hyperplane* optimal yang memisahkan dua kelas data secara maksimal, di mana batas pemisah tersebut ditentukan dengan memaksimalkan *margin* atau jarak antara *hyperplane* dengan titik data terdekat dari masing-masing kelas yang disebut

support vector. Dalam representasinya, kelas positif dilambangkan dengan +1 dan kelas negatif dengan -1 (Ayuningtyas & Tantyoko, 2024).

Secara matematis, *hyperplane* pada SVM dinyatakan dalam bentuk persamaan:

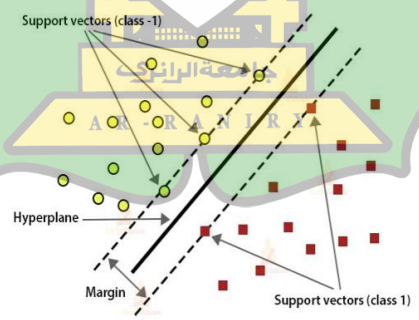
$w.x+b=0$

Dengan keterangan:

- w : parameter atau bobot dari *hyperplane* yang dicari,
- x : data input,
- b : nilai bias.

Dalam proses pemisahan data, SVM memanfaatkan fungsi matematika yang disebut *kernel*. Penelitian ini menggunakan *kernel linear* yang dinilai tepat untuk data teks hasil representasi TF-IDF, mengingat karakteristiknya yang berdimensi tinggi (*high dimensional data*). Dibandingkan jenis *kernel* lainnya, *kernel linear* lebih efisien secara komputasi dan mampu memisahkan kelas secara optimal dengan risiko *overfitting* yang lebih rendah.

Gambaran umum dari struktur kerja SVM dapat dilihat melalui ilustrasi arsitekturnya pada gambar 2.3 berikut.



Gambar 2. 3 Arsitektur SVM

1. Kelebihan SVM

   SVM memiliki beberapa keunggulan dalam penerapannya, di antaranya:

   a. Cocok untuk data berdimensi tinggi, yaitu data yang memiliki banyak fitur.

b. Efisien dalam kasus jumlah fitur lebih banyak dibandingkan jumlah data, seperti pada data teks.
c. Penggunaan memori relatif hemat, karena hanya menggunakan sebagian kecil data (*support vector*) dalam proses pembelajaran.
d. Memberikan hasil yang baik saat terdapat batas yang jelas antara kelas, karena SVM memaksimalkan *margin* pemisah antar kelas.

2. Kekurangan SVM

Meski efektif, SVM juga memiliki keterbatasan sebagai berikut:

a. Kurang cocok untuk dataset berukuran sangat besar, karena proses pelatihannya bisa memakan waktu lama.
b. Kurang optimal saat data mengandung banyak *noise* atau ketika kelas-kelas saling tumpang tindih.
c. Performa menurun saat jumlah fitur per data melebihi jumlah data latih, karena model menjadi sulit untuk digeneralisasi.
d. Tidak memberikan output dalam bentuk probabilitas, sehingga kurang informatif dalam beberapa aplikasi yang memerlukan tingkat keyakinan prediksi.

2.3.10 Kaggle

Kaggle merupakan platform daring yang banyak digunakan oleh praktisi *data science* dan peneliti, menyediakan berbagai dataset dari beragam bidang seperti kesehatan, ekonomi, dan ketenagakerjaan (Angkasa & Pangaribuan, 2022). Dataset yang digunakan dalam penelitian ini adalah *Fake Job Posting Prediction*, yang memuat data lowongan kerja berlabel asli (*genuine*) dan palsu (*fraudulent*).

2.3.11 Python

Python merupakan bahasa pemrograman yang populer berkat fleksibilitas dan kemudahan penggunaannya, dengan sintaks yang sederhana sehingga cocok bagi pemula namun tetap efisien untuk membangun sistem yang kompleks (Gat et al., 2023). Bahasa ini bersifat *open source* dan banyak digunakan di berbagai bidang mulai dari pengembangan aplikasi, analisis data, hingga kecerdasan buatan, didukung oleh komunitas yang luas serta ekosistem *library* yang terus berkembang.

1. Keunggulan Python

    a. Mudah dipelajari karena memiliki sintaks yang sederhana.
    b. Produktivitas tinggi, sehingga memungkinkan pengembangan aplikasi bisa dilakukan lebih cepat.
    c. Serbaguna, sehingga dapat digunakan untuk berbagai keperluan.
    d. Memiliki komunitas yang luas.

2. Keterbatasan Python

    a. Performa relatif lebih lambat dibandingkan dengan bahasa yang dikompilasi seperti C atau C++.
    b. Kurang populer dalam pengembangan *mobile*.
    c. Menggunakan *Global Interpreter Lock* (GIL), yang membatasi eksekusi *multi-threading* tingkat tinggi.

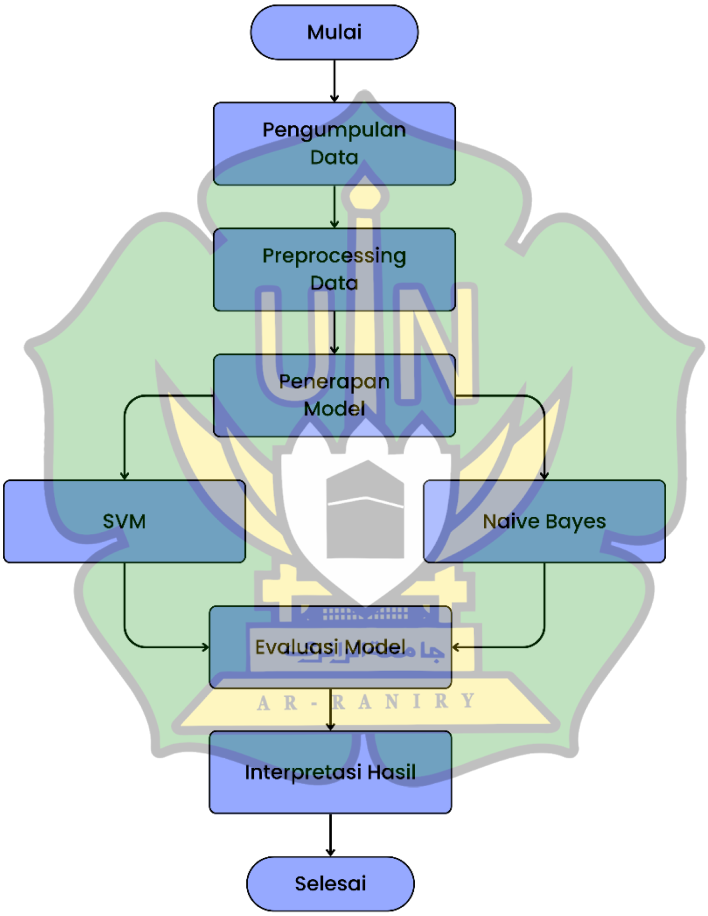
2.3.12 Visual Studio Code

Visual Studio Code adalah editor kode ringan dan gratis dari Microsoft yang mendukung berbagai sistem operasi dan banyak bahasa pemrograman. Dengan performa cepat dan dukungan ekstensi Python (Nurul, 2022), VS Code memudahkan proses pengolahan data dan pembangunan model *machine learning*, termasuk penerapan dan perbandingan metode *Naive Bayes* dan SVM dalam klasifikasi keaslian lowongan pekerjaan di media sosial.

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Bagian ini menjelaskan urutan proses penelitian yang dilakukan untuk mencapai tujuan penelitian. Bagan alur penelitian ditampilkan pada Gambar 3.1.



Gambar 3. 1 Bagan Alur Penelitian

3.2 Pengumpulan Data

Tahap awal penelitian dimulai dengan mengumpulkan sumber ilmiah berupa jurnal, buku, dan penelitian terdahulu yang berkaitan dengan klasifikasi teks menggunakan Naive Bayes dan SVM. Sumber data utama berasal dari repositori publik Kaggle, yaitu dataset *Fake Job Posting Prediction* yang memuat 17.880 data lowongan pekerjaan dengan sekitar 800 di antaranya terindikasi sebagai lowongan palsu, tersedia dalam format CSV dengan 18 atribut berisi deskripsi pekerjaan dan informasi pendukung lainnya. Tidak seluruh atribut yang tersedia digunakan dalam proses klasifikasi. Teknik seleksi atribut (*feature selection*) diterapkan untuk mereduksi dimensi data dan mengeliminasi informasi yang tidak relevan guna mencegah penurunan akurasi model. Rincian fungsi setiap atribut beserta pengolahannya dalam penelitian ini disajikan pada Tabel 3.1.

Tabel 3. 1 Atribut yang akan diterapkan

No	Atribut	Keterangan
1	*job_id*	Nomor identitas unik untuk setiap baris data lowongan.
2	*title*	Nama posisi atau jabatan yang ditawarkan.
3	*location*	Lokasi geografis tempat pekerjaan berada.
4	*department*	Nama divisi atau departemen dalam perusahaan.
5	*salary_range*	Kisaran gaji yang ditawarkan.
6	*company_profile*	Deskripsi singkat profil perusahaan.
7	*description*	Deskripsi pekerjaan secara lengkap.
8	*requirements*	Syarat dan kualifikasi yang harus dimiliki pelamar.
9	*benefits*	Tunjangan atau fasilitas tambahan selain gaji pokok.
10	*telecommuniting*	Indikator apakah pekerjaan dilakukan secara jarak jauh (*remote*).
11	*has_company_logo*	Indikator apakah lowongan memiliki logo perusahaan.
12	*has_questions*	Indikator adanya pertanyaan tambahan saat proses melamar.

13	*employment_type*	Jenis ikatan kerja, seperti kontrak, tetap, atau paruh waktu.
14	*required_experience*	Tingkat pengalaman kerja minimal yang dibutuhkan.
15	*required_education*	Latar belakang pendidikan minimal pelamar.
16	*industry*	Sektor bisnis atau bidang usaha perusahaan tersebut.
17	*function*	Kategori fungsional atau teknis pekerjaan.
18	*fraudulent*	Indikator utama dengan keterangan 0 untuk lowongan asli, 1 untuk palsu.

Berdasarkan Tabel 3.1, seleksi atribut dilakukan dengan menghilangkan data administratif dan atribut dengan banyak nilai kosong (*missing values*). Beberapa atribut teks, yaitu *title*, *company_profile*, *description*, dan *requirements*, digabungkan menjadi satu fitur terpadu agar model dapat mempelajari pola kata secara lebih menyeluruh. Atribut biner seperti *has_company_logo* dan *has_questions* tetap dipertahankan sebagai variabel numerik pendukung yang berpotensi membantu model mengidentifikasi keaslian lowongan. Data hasil seleksi ini selanjutnya digunakan sebagai dasar pembangunan model klasifikasi untuk menentukan apakah suatu lowongan tergolong asli atau palsu.

3.3 Preprocessing Data

Setelah pengumpulan data dan seleksi atribut, tahap selanjutnya adalah *preprocessing*, yaitu proses pembersihan dan persiapan data teks sebelum digunakan dalam klasifikasi. Data mentah dari deskripsi lowongan pekerjaan umumnya masih mengandung elemen tidak relevan seperti simbol, angka, dan variasi penulisan kata yang dapat mengganggu proses analisis, sehingga *preprocessing* diperlukan untuk menghasilkan data yang lebih rapi, terstruktur, dan konsisten. Tahapan *preprocessing* yang diterapkan dalam penelitian ini meliputi beberapa langkah berurutan, di mana setiap tahap memiliki fungsi spesifik untuk memastikan kualitas teks yang digunakan oleh algoritma Naive Bayes dan SVM. Adapun tahapan *preprocessing* dalam penelitian ini meliputi:

3.3.1 Cleaning

Cleaning bertujuan menghilangkan berbagai elemen yang tidak diperlukan dalam teks. Pada dataset lowongan kerja dari Kaggle, sering ditemukan karakter seperti angka, tanda baca, simbol (misalnya $, @, #), hingga tag HTML yang tidak memiliki makna dalam proses klasifikasi. Jika tidak dihapus, elemen-elemen ini dapat menambah noise dan membuat representasi fitur menjadi tidak akurat. Dengan membersihkan seluruh karakter yang tidak relevan tersebut, teks menjadi lebih terfokus pada kata-kata informatif yang benar-benar berkaitan dengan isi lowongan kerja.

3.3.2 Case Folding

Case folding dilakukan untuk menyeragamkan seluruh huruf dalam teks menjadi huruf kecil (*lowercase*). Tujuan dari tahap ini adalah untuk menghindari perbedaan perlakuan terhadap kata yang sama namun ditulis dengan format huruf berbeda. Misalnya, kata "Jakarta", "jakarta", dan "JAKARTA" akan dikenali sebagai kata yang identik setelah melalui proses case folding, yaitu menjadi "jakarta". Penyamaan bentuk ini sangat penting agar model tidak menganggap kata yang sama sebagai entitas berbeda hanya karena perbedaan penulisan.

3.3.3 Tokenizing

Pada tahap *tokenizing*, teks dipecah menjadi unit-unit kecil yang disebut token, biasanya berupa kata. Proses ini membantu algoritma memproses setiap kata secara terpisah, bukan dalam bentuk paragraf atau kalimat utuh. Sebagai contoh, kalimat "Lowongan kerja terbuka" akan diubah menjadi token-token: "Lowongan", "kerja", dan "terbuka". Dengan cara ini, setiap kata dapat dianalisis sebagai variabel tersendiri dalam tahap ekstraksi fitur maupun klasifikasi.

3.3.4 Filtering

Filtering dilakukan untuk menghapus kata-kata umum atau stopword, yaitu kata-kata yang sering muncul dalam teks tetapi tidak memiliki nilai informatif bagi proses klasifikasi. Pada penelitian ini, karena dataset menggunakan bahasa Inggris, filtering dilakukan menggunakan daftar stopword dari *library* NLTK. Kata seperti "and", "the", "is", "at", dan "on" dihapus karena tidak memberikan kontribusi signifikan dalam membedakan lowongan pekerjaan asli dan palsu. Dengan

menghilangkan kata-kata tersebut, analisis dapat lebih fokus pada kata kunci yang memiliki makna penting.

3.3.5 Stemming

Stemming merupakan proses mengubah kata ke bentuk dasarnya dengan menghilangkan imbuhan atau variasi bentuk kata. Tahapan ini penting karena satu kata dasar dapat muncul dalam berbagai bentuk tergantung konteks kalimat. Misalnya, kata "berlari", "melarikan", dan "pelari" semuanya dikembalikan ke bentuk dasar "lari". Proses ini membantu menyatukan variasi kata yang sebenarnya memiliki makna serupa, sehingga model dapat mengidentifikasi pola dengan lebih efisien.

3.4 Penerapan Model

Bagian ini menjelaskan proses penerapan model klasifikasi yang digunakan dalam penelitian untuk membedakan antara lowongan pekerjaan asli dan palsu. Penerapan model meliputi proses pengubahan data teks menjadi fitur yang dapat dipahami oleh algoritma, pemilihan metode klasifikasi, serta pembagian data untuk pelatihan dan evaluasi. Tahapan-tahapan ini dilakukan secara sistematis agar model mampu mengenali pola dalam data secara optimal.

3.4.1 Term Frequency-Invers Document Frequency

Setelah data lowongan pekerjaan melalui berbagai tahap *preprocessing*, teks yang telah dibersihkan tersebut perlu diubah ke dalam bentuk numerik agar dapat diproses oleh algoritma klasifikasi. Hal ini penting karena model seperti *Naive Bayes* dan SVM tidak dapat bekerja langsung dengan data berupa teks mentah. Untuk melakukan konversi ini, penelitian menggunakan metode *Term Frequency–Inverse Document Frequency* (TF-IDF).

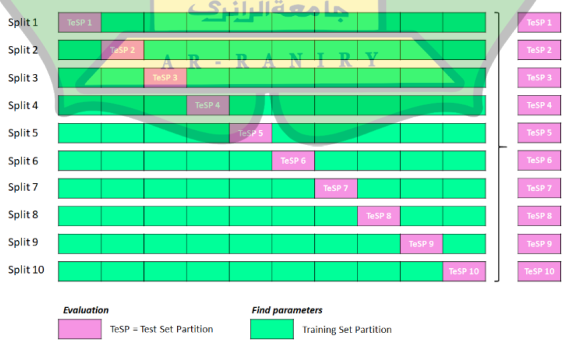
TF-IDF berfungsi untuk mengukur seberapa penting suatu kata dalam sebuah dokumen dibandingkan dengan keseluruhan kumpulan dokumen (corpus). Metode ini tidak hanya menghitung frekuensi kemunculan kata dalam satu deskripsi lowongan (*Term Frequency*), tetapi juga mempertimbangkan tingkat keunikan kata tersebut di antara seluruh data (*Inverse Document Frequency*). Dengan demikian, kata-kata yang memiliki karakteristik khusus seperti istilah tertentu yang lebih sering muncul pada lowongan palsu akan memperoleh bobot yang lebih tinggi.

Sebaliknya, kata-kata umum yang muncul hampir di semua lowongan akan diberi bobot lebih rendah (Dermawan, 2025).

3.4.2 K-Fold Cross Validation

Sebelum model dilatih, dataset perlu dibagi terlebih dahulu menjadi dua kelompok utama, yaitu data latih (*training set*) dan data uji (*testing set*). Data latih digunakan agar algoritma dapat mempelajari pola yang terdapat pada lowongan pekerjaan, sementara data uji berfungsi untuk menilai kemampuan model dalam memprediksi data baru yang belum pernah ditemuinya. Pembagian ini sangat penting untuk mengurangi risiko terjadinya *overfitting*, yaitu kondisi ketika model tampak sangat baik pada data latih tetapi kurang mampu melakukan prediksi pada data baru (Parameswari et al., 2025).

Agar proses evaluasi lebih menyeluruh dan tidak bergantung pada satu kali pembagian data saja, penelitian ini menggunakan metode *k-fold cross validation.* Metode ini membagi dataset menjadi *k* bagian dengan ukuran yang sama. Secara bergiliran, satu bagian dijadikan data uji, sedangkan *k−1* bagian lainnya digunakan sebagai data latih. Setiap putaran menghasilkan nilai akurasi tersendiri; kemudian seluruh nilai tersebut dirata-ratakan untuk memperoleh skor evaluasi akhir yang lebih stabil (Aditya et al., 2023). Proses evaluasi model ini dapat dilihat pada Gambar 3.2.



Gambar 3. 2 Evaluasi Model *K-Fold Cross Validation*

Pada penelitian ini, jumlah lipatan yang digunakan adalah k = 10, sehingga proses evaluasi dilakukan dalam 10 iterasi. Pada tiap iterasi, satu lipatan dijadikan data uji sementara sembilan lipatan lainnya menjadi data latih. Nilai akurasi dari

semua lipatan kemudian dijumlahkan dan dihitung nilai rata-ratanya menggunakan rumus berikut:

Rata-rata Akurasi $= \frac{1}{k} \sum_{i-1}^{k} A_i$

Keterangan:

- k : Jumlah lipatan (*fold*), yaitu 10.
- A_i: Nilai akurasi yang diperoleh pada iterasi ke-i.
- $\sum$: Simbol penjumlahan untuk seluruh nilai akurasi dari iterasi ke-1 sampai ke-10.

Model terbaik ditentukan berdasarkan nilai rata-rata akurasi tertinggi. Namun, pemilihan nilai *k* harus dilakukan dengan hati-hati. Nilai *k* yang terlalu besar dapat meningkatkan beban komputasi dan meningkatkan risiko *overfitting*, sedangkan nilai *k* yang terlalu kecil dapat menghasilkan evaluasi yang kurang stabil. Oleh karena itu, penggunaan nilai k = 10 dipilih karena dianggap memberikan hasil yang paling seimbang dan representatif.

3.5 Evaluasi Model

Tahapan evaluasi merupakan bagian penting dalam proses *data mining* yang bertujuan untuk menilai seberapa efektif model yang dibangun dalam mengklasifikasikan keaslian lowongan pekerjaan.

Dalam penelitian ini, evaluasi dilakukan pada dua model *machine learning*, yaitu *Naive Bayes* dan SVM. Penilaian performa dari setiap model menggunakan pendekatan *Confusion Matrix*, yang memberikan gambaran detail tentang hasil klasifikasi (Priya et al., 2022).

Confusion Matrix menghasilkan empat komponen utama, yaitu:

- *True Positive* (TP): Jumlah lowongan kerja palsu yang berhasil diprediksi dengan benar sebagai palsu.
- *True Negative* (TN): Jumlah lowongan kerja asli yang berhasil diprediksi dengan benar sebagai asli.
- *False Positive* (FP): Jumlah lowongan kerja asli yang salah diprediksi sebagai palsu.
- *False Negative* (FN): Jumlah lowongan kerja palsu yang salah diprediksi sebagai asli.

Dari komponen tersebut, dihitung metrik evaluasi berikut:

1. Akurasi (*Accuracy*) : Mengukur proporsi prediksi yang benar dari total data.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \times 100\%$$

2. Presisi (*Precision*) : Mengambarkan ketepatan model dalam meng-identifikasi lowongan kerja yang benar-benar palsu.

$$Precision = \frac{TP}{TP + FP} \times 100\%$$

3. *Recall* (*Sensitivy*) : Menilai kemampuan model dalam menemukan semua lowongan palsu yang ada.

$$Recall = \frac{TP}{TP + FN} \times 100\%$$

4. *F1-score* : Rata-rata harmonis dari *precision* dan *recall*, yang digunakan untuk menyeimbangkan keduanya dalam satu nilai.

$$\text{F1} - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\%$$

Melalui evaluasi ini, peneliti dapat menentukan model mana yang lebih efektif dalam mendeteksi keaslian lowongan kerja, serta menilai kelayakan model tersebut untuk diterapkan dalam sistem deteksi penipuan berbasis teks di media sosial.

3.6 Interpretasi Hasil

Pada tahap ini, peneliti akan menganalisis hasil evaluasi performa dari dua model klasifikasi, yaitu SVM dan *Naive Bayes*. Analisis didasarkan pada metrik evaluasi seperti *akurasi, presisi, recall*, dan *F1-score* yang telah diperoleh sebelumnya. Tujuannya adalah untuk memahami model mana yang paling optimal dalam mengklasifikasikan lowongan kerja, serta mengidentifikasi kelebihan dan kekurangan masing-masing algoritma. Hasil interpretasi ini akan menjadi landasan untuk menyusun simpulan dan saran.

3.7 Lokasi Penelitian

Laboratorium pada Program Studi Teknologi Informasi, Fakultas Sains dan Teknologi, UIN Ar-Raniry Banda Aceh digunakan sebagai tempat untuk menerapkan model yang telah dibuat.

3.8 Alat dan Bahan

Alat dan bahan yang digunakan dalam penelitian ini terdiri dari perangkat keras (*hardware*), perangkat lunak (*software*), serta sumber data yang mendukung proses pengolahan data dan analisis.

3.8.1 Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini berupa satu unit laptop yang memiliki spesifikasi cukup untuk mendukung proses *data mining* pada klasifikasi berbasis teks. Spesifikasi laptop yang digunakan antara lain sebagai berikut:

- Nama Perangkat : Laptop ASUS VivoBook M1403QA
- Sistem Operasi : Windows 11 Home Single Language 64-bit
- *Processor* : AMD Ryzen 7 5800H with Radeon Graphics (3.20 GHz)
- RAM : 16 GB
- *Storage* : 477 GB

3.8.2 Perangkat Lunak

Adapun perangkat lunak yang digunakan dalam penelitian ini meliputi bahasa pemrograman, lingkungan pengembangan, serta *library* pendukung yang diperlukan untuk menjalankan proses klasifikasi data. Berikut detail perangkat lunak yang digunakan:

- Visual Studio Code : Editor kode untuk menulis dan menjalankan skrip Python.
- Python versi 3.10.11 : Bahasa pemrograman utama.
- Scikit-learn : *Library* untuk implementasi algoritma *Naive Bayes* dan SVM serta evaluasi model.
- Pandas : *Library* untuk manipulasi dan analisis data.
- NumPy : *Library* untuk pengolahan *array* dan operasi numerik.

- NLTK : *Library* untuk *preprocessing* teks seperti *cleaning*, *tokenisasi, filtering, dan stemming.*
- Matplotlib / Seaborn : *Libarary* untuk visualisasi hasil.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengumpulan Data

Tahap pengumpulan data menjadi fondasi utama dalam penelitian ini karena kualitas data sangat menentukan hasil pemodelan klasifikasi. Proses awal penelitian difokuskan pada pengumpulan serta pemahaman mendalam mengenai karakteristik dataset yang digunakan. Dataset bersumber dari repositori publik Kaggle dengan judul *Fake Job Posting Prediction* (Bansal, 2020). Dataset ini dipilih karena telah menyediakan label yang jelas untuk setiap entri lowongan pekerjaan, yaitu asli dan palsu. Label tersebut memungkinkan penerapan pendekatan *supervised learning* secara optimal serta sama dengan tujuan penelitian, yaitu membandingkan performa Naive Bayes dan SVM dalam mengklasifikasikan keaslian lowongan kerja di media sosial.

Secara keseluruhan, dataset terdiri atas 17.880 entri lowongan pekerjaan dari berbagai sektor industri. Masing-masing entri memuat informasi seperti judul pekerjaan, deskripsi, persyaratan, dan atribut pendukung lainnya. Struktur awal dataset sebelum dilakukan pengolahan ditampilkan pada Tabel 4.1 sebagai gambaran atribut-atribut utama.

Tabel 4. 1 Contoh Dataset Awal

Atribut	Data 1	Data 2
job_id	7	103
title	Head of Content (m/f)	Marketing Administrator
location	DE, BE, Berlin	GB, WAR, Coventry
department	ANDROIDPIT	Marketplace
salary_range	20000–28000	15000–18000

company_profile	Founded in 2009, the Fonpit AG rose with its international web portal ANDROIDPIT…	Renewable Energy and Environmental Protection equipment
description	Your Responsibilities: Manage the English-speaking editorial team…	The job is to support the growth of the marketplace project…
requirements	University degree in journalism/media, experience in leading teams, Android passion…	Computer literate, able to work with HTML, enthusiastic…
benefits	Being part of a fast-growing company, creative freedom, flat hierarchies…	Possibility to work remotely depending on motivation
telecommuting	0	1
has_company_logo	1	1
has_questions	1	0
employment_type	Full-time	Full-time
required_experience	Mid-Senior level	Entry level
required_education	Master's Degree	Bachelor's Degree
industry	Online Media	Internet
function	Management	Marketing
fraudulent	0	0

Seluruh data teks pada dataset ini menggunakan bahasa Inggris, baik pada atribut judul pekerjaan, deskripsi, persyaratan, maupun profil perusahaan. Dalam penelitian ini, data tidak diterjemahkan ke dalam bahasa Indonesia dan tetap diproses dalam bahasa aslinya. Hal tersebut dilakukan karena dataset sejak awal dikembangkan, dilabeli, dan divalidasi dalam bahasa Inggris, sehingga pola bahasa

yang menjadi indikator lowongan palsu juga terbentuk dalam konteks tersebut. Proses penerjemahan berpotensi mengubah struktur kalimat maupun makna yang dapat berdampak pada penurunan kualitas fitur teks dan akurasi model.

Selain itu, algoritma Naive Bayes dan SVM bekerja berdasarkan distribusi dan kemunculan data. Proses terjemahan dapat menimbulkan risiko tidak konsistennya representasi kata, seperti perbedaan sinonim, pemenggalan frasa, atau perubahan konteks yang pada akhirnya menurunkan kualitas fitur. Penggunaan bahasa Inggris juga didukung oleh library NLP, seperti NLTK dan Scikit-learn yang memiliki stopword, tokenizer, serta algoritma stemming yang lebih stabil untuk bahasa Inggris dibandingkan bahasa Indonesia. Dengan demikian, mempertahankan bahasa asli dataset diharapkan dapat menghasilkan proses pembelajaran yang lebih optimal dan hasil klasifikasi yang lebih reliabel.

Dataset memiliki 18 atribut yang mencakup atribut teks, kategorikal, biner, serta satu atribut target, yaitu *fraudulent*. Namun tidak semua atribut digunakan dalam proses klasifikasi. Beberapa atribut bersifat administratif, memiliki banyak nilai kosong atau tidak memberikan informasi penting dalam pembedaan lowongan asli dan palsu. Penggunaan seluruh atribut tanpa seleksi dapat meningkatkan dimensi data dan kompleksitas model serta menurunkan efisiensi dan akurasi.

Untuk menyederhanakan dimensi data dan mempertahankan informasi yang relevan, dilakukan seleksi atribut dengan mempertahankan atribut yang memiliki nilai tekstual paling kuat. Atribut *title*, *company_profile*, *description*, dan *requirements* digabungkan menjadi satu variabel teks menggunakan library Pandas dengan cara menggabungkan seluruh isi kolom menjadi satu string untuk tiap entri. Pendekatan ini memungkinkan model mempelajari pola bahasa secara lebih menyeluruh karena indikator penipuan dapat muncul pada berbagai bagian teks, misalnya deskripsi yang terlalu umum, persyaratan yang tidak realistis, atau profil perusahaan yang kurang jelas.

Selain atribut teks, beberapa atribut biner seperti *has_company_logo* dan *has_questions* tetap dipertahankan sebagai fitur tambahan karena memiliki potensi terkait dengan tingkat kredibilitas lowongan kerja. Atribut-atribut tersebut disimpan dalam bentuk numerik agar dapat digabungkan dengan fitur berbasis TF-IDF pada tahap berikutnya. Sementara itu, atribut kategorikal seperti *industry*, *function*, dan

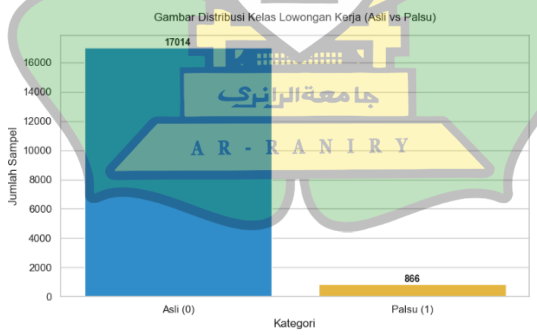
employment_type tidak digunakan karena dapat meningkatkan kompleksitas tanpa memberikan peningkatan performa yang berarti dalam penelitian berbasis teks ini.

Proses seleksi atribut dan persiapan data dilakukan menggunakan beberapa library utama, yaitu Pandas untuk pengelolaan dataset, NumPy untuk pengolahan data numerik, dan Scikit-learn untuk ekstraksi fitur serta pemodelan. Hasil dari proses seleksi atribut ini menghasilkan dataset yang lebih ringkas dan fokus sehingga siap digunakan pada tahap selanjutnya. Visualisasi hasil seleksi atribut ditampilkan pada Gambar 4.1.

	combined_text	has_company_logo	has_questions	fraudulent
0	Marketing Intern We're Food52, and we've creat...	1	0	0
1	Customer Service - Cloud Video Production 90 S...	1	0	0
2	Commissioning Machinery Assistant (CMA) Valor ...	1	0	0
3	Account Executive - Washington DC Our passion ...	1	0	0
4	Bill Review Manager SpotSource Solutions LLC i...	1	1	0

Gambar 4. 1 Hasil Seleksi Atribut

Setelah proses seleksi atribut, dilakukan analisis distribusi kelas pada atribut target palsu untuk mengetahui keseimbangan jumlah data antara lowongan asli dan palsu sebelum model dilatih. Hasil analisis menunjukkan ketidakseimbangan data (*imbalanced data*), dengan 17.014 data lowongan asli (0) dan 866 data lowongan palsu (1) seperti divisualisasikan pada Gambar 4.2.



Gambar 4. 2 Distribusi Keaslian Lowongan Kerja

Ketidakseimbangan data ini memiliki pengaruh penting terhadap proses pelatihan dan evaluasi model. Dominan kelas asli dapat membuat model cenderung memprediksi data sebagai lowongan asli sehingga sulit mendeteksi lowongan palsu. Oleh karena itu, evaluasi model tidak hanya mengandalkan akurasi, tetapi juga mempertimbangkan metrik *precision*, *recall*, dan *F1-score*, untuk menilai sejauh

mana algoritma mampu mendeteksi lowongan palsu secara efektif dalam kondisi data yang tidak seimbang.

4.2 Hasil *Preprocessing* Data

Tahap berikutnya adalah *preprocessing* data. Tahap ini bertujuan untuk mempersiapkan data agar dapat diolah secara optimal oleh algoritma Naive Bayes dan SVM dalam mengenali pola serta membedakan lowongan kerja asli dan palsu. Data teks hasil seleksi atribut masih berupa teks mentah yang mengandung berbagai ketidakteraturan, sehingga perlu dibersihkan agar tidak mengganggu proses pembelajaran model.

Preprocessing dilakukan untuk mengurangi informasi yang tidak relevan, menyeragamkan struktur teks, dan memastikan bahwa kata-kata yang diproses benar-benar merepresentasikan informasi yang relevan untuk tujuan klasifikasi. Seluruh tahapan *preprocessing* dilakukan secara berurutan menggunakan menggunakan bahasa pemrograman Python.

4.2.1 Proses *Cleaning*

Tahap pertama dalam proses *preprocessing* data teks pada penelitian ini adalah *cleaning* atau pembersihan teks. Tahap ini dilakukan untuk menghilangkan berbagai elemen yang tidak memberikan makna penting terhadap konten lowongan pekerjaan. Data teks hasil seleksi atribut pada dasarnya masih mengandung karakter-karakter yang tidak diperlukan, seperti tanda baca, simbol khusus, angka, serta bentuk tulisan lain yang tidak relevan secara makna. Elemen-elemen tersebut, jika tidak dibersihkan, berpotensi menambah gangguan dan membuat jumlah fitur menjadi lebih besar tanpa memberikan kontribusi nyata dalam membedakan lowongan kerja asli dan palsu.

Pada proses *cleaning* ini, seluruh karakter non-alfabet dihapus dari teks. Pembersihan mencakup penghapusan tanda baca seperti titik, koma, tanda seru, tanda tanya, serta berbagai simbol khusus dan angka. Selain itu, dilakukan pula penanganan terhadap singkatan kata dalam bahasa Inggris, misalnya *we're*, *we've*, *doesn't*, dan sebagainya. Singkatan-singkatan tersebut dipisahkan menjadi bentuk kata yang lebih sederhana agar dapat diproses secara konsisten pada tahap berikutnya, seperti *tokenizing* dan *filtering*. Jika singkatan dibiarkan dalam bentuk

aslinya, model dapat menganggap satu kata sebagai dua entitas yang berbeda karena variasi bentuk penulisannya yang dapat menurunkan akurasi representasi teks.

Proses *cleaning* dilakukan dengan menggunakan *regular expression* (regex) melalui library *re* pada Python. Regex memungkinkan sistem mendeteksi pola karakter tertentu dan menghapusnya secara fleksibel dan efisien. Teknik ini kemudian dikombinasikan dengan fungsi-fungsi pemrosesan teks dari library *Natural Language Toolkit* (NLTK) untuk memastikan bahwa hasil pembersihan tetap membentuk struktur kata yang dapat diolah lebih lanjut. Seluruh proses diimplementasikan menggunakan Pandas, sehingga pembersihan dapat diterapkan secara konsisten pada seluruh entri dalam dataset tanpa harus memproses data secara manual.

Hasil dari tahap pembersihan tersebut ditampilkan pada Tabel 4.2. Teks yang awalnya dipenuhi tanda baca, simbol khusus, dan bentuk kontraksi berubah menjadi teks yang lebih bersih dan terstruktur. Meskipun pemisahan singkatan kata menghasilkan token seperti *re*, *ve*, atau *nt*, hal ini tidak menjadi masalah karena token-token tersebut akan dieliminasi pada tahap filtering, sehingga tidak akan mengganggu kualitas fitur akhir yang digunakan dalam pemodelan.

Tabel 4. 2 Proses *Cleaning*

Cleaning	
Teks	Hasil
Marketing Intern We're Food52, and we've created a groundbreaking and award-winning cooking site. We...	Marketing Intern We re Food and we ve created a groundbreaking and award winning cooking site We sup...
Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...	Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...

Tahap *cleaning* memiliki pengaruh besar dalam proses klasifikasi karena algoritma berbasis teks seperti Naive Bayes dan SVM bekerja dengan menghitung

frekuensi kemunculan kata untuk membentuk model. Jika karakter yang tidak memiliki makna dibiarkan, algoritma dapat menganggap simbol atau tanda baca sebagai fitur tersendiri, sehingga memperbanyak fitur yang tidak relevan dan meningkatkan gangguan dalam data. Kondisi ini dapat menurunkan kemampuan model dalam mengenali pola-pola bahasa yang berkaitan dengan indikasi penipuan dalam lowongan pekerjaan. Tahap ini memastikan bahwa model bekerja dengan data yang bersih, relevan, dan bebas dari elemen-elemen yang berpotensi mengganggu proses pembelajaran mesin.

4.2.2 Proses *Case Folding*

Setelah proses *cleaning*, langkah berikutnya adalah *case folding*. Tahap ini dilakukan untuk menyeragamkan seluruh teks dengan mengubah semua huruf menjadi huruf kecil (*lowercase*). Pada teks lowongan pekerjaan, variasi penggunaan huruf kapital umumnya tidak memengaruhi makna suatu kata. Namun, dalam perspektif algoritma pembelajaran mesin, perbedaan huruf kapital membuat kata yang sebenarnya sama dianggap sebagai fitur yang berbeda. Kondisi ini dapat menyebabkan pengulangan fitur, menambah beban komputasi, serta mengganggu proses pembelajaran karena model harus mengolah fitur yang seharusnya tidak perlu dibedakan.

Proses *case folding* diterapkan ke seluruh teks menggunakan fungsi bawaan Python yang diimplementasikan melalui library Pandas, khususnya melalui metode pemrosesan string pada kolom teks. Pendekatan ini memungkinkan proses konversi huruf dilakukan secara otomatis, konsisten, dan efisien pada seluruh dataset. Selain itu, *case folding* tidak mengubah struktur kata maupun makna sebenarnya, sehingga isi teks tetap asli namun dalam format yang seragam. Dengan demikian, kata seperti *Marketing*, *marketing*, dan *MARKETING* akan direpresentasikan sebagai satu bentuk yang sama, yaitu *marketing*.

Hasil transformasi setelah *case folding* dapat dilihat pada Tabel 4.3. Visualisasi sebelum dan sesudah proses ini menjelaskan bahwa tidak ada lagi perbedaan tampilan yang dapat menyebabkan pemecahan fitur pada tahap ekstraksi fitur.

Tabel 4. 3 Proses *Case Folding*

Case Folding	
Teks	Hasil
Marketing Intern We're Food52, and we've created a groundbreaking and award-winning cooking site. We...	marketing intern we re food and we ve created a groundbreaking and award winning cooking site we sup...
Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...	bill review manager spotsource solutions llc is a global human capital management consulting firm he...

Tanpa adanya penyeragaman huruf, TF-IDF akan menghasilkan nilai bobot terpisah untuk kata yang sebenarnya memiliki makna identik, sehingga meningkatkan dimensi data secara tidak perlu dan dapat memperburuk efisiensi pemrosesan. Dimensi data yang terlalu besar berpotensi menghambat performa algoritma, terutama pada model yang sensitif terhadap kompleksitas fitur seperti SVM. Tahap ini berfungsi untuk memastikan bahwa model tidak terganggu oleh perbedaan teknis yang tidak berkaitan dengan makna teks, sehingga dapat berfokus sepenuhnya pada karakteristik bahasa yang relevan untuk mendeteksi indikasi penipuan dalam lowongan pekerjaan.

4.2.3 Proses *Tokenizing*

Proses *tokenizing* bertujuan memecah teks yang semula berbentuk paragraf atau kalimat panjang menjadi unit-unit kata individual yang disebut *token*. Tokenisasi sangat penting karena algoritma Naive Bayes dan SVM, bekerja dengan menganalisis kata per kata dan tidak dapat memproses teks dalam bentuk kalimat utuh. Dengan memecah teks menjadi token, model dapat mengenali pola bahasa secara lebih rinci dan objektif.

Proses tokenizing dalam penelitian ini dilakukan menggunakan fungsi *word_tokenize* dari library NLTK. Fungsi tersebut dibuat untuk menangani struktur bahasa Inggris sehingga mampu memisahkan kata dengan akurat berdasarkan spasi,

tanda baca yang telah dihilangkan pada tahap sebelumnya, serta kaidah umum penulisan teks bahasa Inggris. Penggunaan tokenizer yang sesuai dengan bahasa dataset sangat penting untuk mengurangi kesalahan pemisahan kata, seperti pemecahan kata yang tidak tepat atau penggabungan kata yang seharusnya terpisah.

Hasil dari tahap tokenizing ditampilkan pada Tabel 4.4. Tabel tersebut menunjukkan proses perubahan dari teks berbentuk kalimat menjadi daftar kata yang berdiri sendiri. Setiap token dianggap sebagai satuan informasi yang dapat dianalisis lebih lanjut. Bentuk representasi ini memudahkan sistem dalam memperlakukan setiap kata sebagai elemen terpisah pada proses ekstraksi fitur, sehingga proses klasifikasi dapat berjalan lebih sistematis dan konsisten.

Tabel 4. 4 Proses *Tokenizing*

Tokenizing	
Teks	Hasil
Marketing Intern We're Food52, and we've created a groundbreaking and award-winning cooking site. We...	'marketing', 'intern', 'we', 're', 'food', 'and', 'we'...
Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...	'bill', 'review', 'manager', 'spotsource', 'solutions', 'llc', 'is'...

Setiap token yang tercipta pada tahap ini akan menjadi fitur yang dihitung bobot kemunculannya dalam dokumen. Apabila proses tokenisasi tidak dilakukan dengan benar, misalnya token terbentuk tidak sesuai kata aslinya, terlalu panjang, atau justru terpecah secara keliru, maka representasi fitur menjadi kurang akurat dan dapat menurunkan performa model klasifikasi. Model mungkin gagal menangkap pola asli yang relevan jika tokenisasi menghasilkan struktur kata yang tidak mencerminkan konteks sebenarnya.

Tokenizing berfungsi sebagai penghubung penting antara tahap normalisasi teks sebelumnya dan tahap berikutnya, yaitu *filtering* atau penyaringan kata. Token

yang telah dihasilkan akan melalui proses seleksi lebih lanjut untuk menghilangkan kata-kata yang tidak memiliki nilai informatif. Oleh karena itu, akurasi tokenizing sangat berpengaruh terhadap efektivitas keseluruhan tahap *preprocessing* karena kualitas token menentukan kualitas fitur yang pada akhirnya dipelajari oleh model. Tokenisasi memastikan bahwa model bekerja dengan elemen bahasa yang jelas, rapi, dan konsisten, sehingga mendukung peningkatan performa Naive Bayes dan SVM dalam membedakan lowongan pekerjaan asli dan palsu berdasarkan pola bahasa yang terkandung dalam teks.

4.2.4 Proses *Filtering*

Tahap *filtering* ini memiliki tujuan utama untuk menghilangkan kata-kata yang dianggap tidak memiliki nilai informasi penting dalam proses klasifikasi. Kata-kata umum seperti *and*, *the*, *is*, dan *at* merupakan contoh dari kata-kata yang sering muncul hampir di seluruh dokumen tanpa memberikan makna spesifik yang dapat membantu algoritma dalam membedakan kelas. Keberadaan kata-kata tersebut justru dapat menambah gangguan pada data dan berpotensi menurunkan akurasi model.

Proses *filtering* dilakukan dengan cara membandingkan setiap token hasil tokenizing dengan daftar *stopwords* bahasa Inggris yang disediakan oleh library NLTK. Penggunaan stopword list bahasa Inggris dipilih karena seluruh dataset terdiri dari teks berbahasa Inggris, sehingga proses penyaringan dapat dilakukan secara konsisten dan relevan dengan karakter asli data. Token yang ditemukan dalam daftar stopword akan dihapus karena dianggap tidak membawa informasi penting, sementara token lain yang lebih bermakna tetap dipertahankan untuk diproses pada tahap berikutnya.

Hasil dari proses filtering ditunjukkan pada Tabel 4.5. Tabel tersebut memperlihatkan adanya pengurangan jumlah token secara signifikan setelah proses penyaringan dilakukan. Token-token yang tersisa umumnya merupakan kata-kata yang memiliki potensi lebih besar untuk menggambarkan isi dokumen, seperti istilah terkait jabatan, deskripsi pekerjaan, atau referensi perusahaan. Dengan demikian, token-token tersebut dapat menjadi indikator yang lebih kuat dalam proses identifikasi pola bahasa antara lowongan pekerjaan asli dan palsu.

Tabel 4. 5 Proses *Filtering*

Filtering	
Teks	Hasil
Marketing Intern We're Food52, and we've created a groundbreaking and award-winning cooking site. We...	['marketing', 'intern', 'food', 'created', 'groundbreaking', 'award', 'winning']...
Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...	['bill', 'review', 'manager', 'spotsource', 'solutions', 'llc', 'global']...

Tahap *filtering* memiliki pengaruh penting terhadap kualitas fitur yang akan dihasilkan pada proses ekstraksi menggunakan metode TF-IDF. Dengan menghilangkan kata-kata umum yang tidak relevan, bobot fitur yang dihitung menjadi lebih akurat dan representatif terhadap isi dokumen. Proses ini mencegah dominasi kata-kata dengan frekuensi tinggi tetapi tingkat informasi rendah, sehingga algoritma dapat lebih fokus pada kata-kata kunci yang benar-benar penting dalam membedakan dokumen.

Selain meningkatkan akurasi representasi fitur, *filtering* juga membantu menurunkan dimensi data dan meningkatkan efisiensi komputasi. Dengan jumlah token yang lebih sedikit dan lebih relevan, model dapat menjalani proses pelatihan secara lebih cepat dan stabil tanpa kehilangan informasi kritis.

4.2.5 Proses *Stemming*

Tahap terakhir *preprocessing* data teks pada penelitian ini adalah *stemming*. Proses ini merupakan teknik penyederhanaan kata dengan cara mengembalikan setiap token ke bentuk dasarnya melalui penghilangan imbuhan atau variasi bentuk kata. Dalam konteks data lowongan pekerjaan, satu konsep sering muncul dalam berbagai bentuk kata, misalnya *created*, *creating*, dan *creates*. Apabila variasi bentuk tersebut tidak diseragamkan, algoritma klasifikasi akan menganggap setiap bentuk kata sebagai fitur yang berbeda, meskipun secara makna merujuk pada

makna yang sama. Kondisi ini dapat menyebabkan penumpukan kata dan meningkatkan kompleksitas representasi data secara keseluruhan.

Dalam penelitian ini, proses *stemming* dipilih sebagai metode normalisasi dibandingkan dengan *lemmatization*. Pertimbangan utama pemilihan *stemming* adalah karena metode ini lebih efisien secara komputasi dan sesuai dengan kebutuhan penelitian yang berfokus pada klasifikasi teks dalam skala data yang cukup besar. Penelitian ini tidak memerlukan ketepatan bentuk kata yang baku secara tata bahasa, sebagaimana dihasilkan oleh *lemmatization*. Tujuan utama penelitian adalah mengidentifikasi pola umum dalam teks lowongan pekerjaan asli dan palsu, sehingga penyederhanaan kata ke bentuk dasarnya melalui *stemming* dianggap sudah memadai untuk meningkatkan efektivitas klasifikasi tanpa menambah beban komputasi.

Selain alasan efisiensi, stemming juga berperan dalam mengurangi jumlah fitur yang dihasilkan. Dengan menyatukan berbagai variasi bentuk kata menjadi satu representasi dasar, dimensi data yang digunakan pada tahap ekstraksi TF-IDF dapat ditekan. Pengurangan dimensi ini berdampak positif pada proses pelatihan model karena algoritma dapat mempelajari pola kata dengan lebih stabil, risiko *overfitting* menjadi lebih rendah, dan waktu komputasi dapat diminimalkan.

Pada penelitian ini, proses *stemming* dilakukan menggunakan algoritma *Porter Stemmer* yang tersedia dalam library NLTK. *Porter Stemmer* dipilih karena merupakan salah satu algoritma *stemming* yang paling banyak digunakan dalam penelitian *text mining* berbahasa Inggris. Algoritma ini menerapkan serangkaian aturan pemotongan imbuhan secara bertahap untuk menghasilkan bentuk kata yang lebih sederhana dan konsisten.

Hasil stemming ditampilkan pada Tabel 4.6. Tabel tersebut menunjukkan bahwa sebagian besar token telah berhasil direduksi menjadi bentuk dasar. Walaupun bentuk kata hasil stemming tidak selalu menghasilkan kata yang valid secara aturan bahasa, akar makna dari kata tersebut tetap terjaga. Dalam konteks klasifikasi teks, hal ini tidak menjadi masalah karena yang dibutuhkan oleh algoritma adalah konsistensi representasi, bukan kesempurnaan dari kata tersebut.

Tabel 4. 6 Proses *Stemming*

Stemming	
Teks	Hasil
Marketing Intern We're Food52, and we've created a groundbreaking and award-winning cooking site. We...	'market', 'intern', 'food', 'creat', 'groundbreak', 'award', 'win'...
Bill Review Manager SpotSource Solutions LLC is a Global Human Capital Management Consulting firm he...	'bill', 'review', 'manag', 'spotsourc', 'solut', 'llc', 'global'...

Dengan menyatukan bentuk kata yang memiliki makna serupa, jumlah fitur yang harus dianalisis oleh algoritma menjadi lebih sedikit dan lebih terstruktur. Kompleksitas model menurun, serta kemampuan algoritma dalam mengenali pola bahasa meningkat.

4.3 Penerapan Model

Pada tahap penerapan model, data teks yang telah melalui proses *preprocessing* selanjutnya diubah ke dalam bentuk numerik agar dapat diproses oleh algoritma klasifikasi. Perubahan ini diperlukan karena algoritma Naive Bayes dan SVM tidak dapat mengolah data dalam bentuk teks mentah, melainkan membutuhkan representasi numerik sebagai input. Penerapan model pada penelitian ini bertujuan untuk membangun dan membandingkan performa kedua algoritma dalam mengklasifikasikan keaslian lowongan pekerjaan pada media sosial. Analisis ini tidak hanya berfokus pada kemampuan masing-masing algoritma dalam mengidentifikasi pola bahasa yang membedakan lowongan pekerjaan asli dan palsu, tetapi juga mengevaluasi stabilitas, konsistensi, serta kemampuan generalisasi model ketika dihadapkan pada dataset yang memiliki distribusi kelas tidak seimbang.

Seluruh proses penerapan model dilakukan menggunakan bahasa pemrograman Python dengan dukungan library Scikit-learn sebagai library utama,

karena menyediakan berbagai fungsi untuk ekstraksi fitur, pelatihan model, validasi, serta evaluasi performa. Untuk menjaga objektivitas hasil dan menghindari hasil yang tidak seimbang akibat pembagian data tertentu, proses pelatihan dan pengujian model dilakukan menggunakan skema 10-Fold Cross Validation.

4.3.1 Ekstraksi Fitur menggunakan TF-IDF

Ekstraksi fitur merupakan proses mengonversi teks menjadi bentuk numerik. Tahap ini merupakan bagian yang sangat penting dalam alur klasifikasi, karena algoritma Naive Bayes dan SVM hanya dapat bekerja dengan data yang direpresentasikan dalam bentuk angka. Kualitas representasi numerik yang dihasilkan pada tahap ini akan secara langsung mempengaruhi kemampuan model dalam mempelajari pola bahasa serta menentukan akurasi hasil klasifikasi.

Pada penelitian ini, metode ekstraksi fitur yang digunakan adalah *Term Frequency–Inverse Document Frequency* (TF-IDF). Pemilihan TF-IDF berdasarkan kemampuannya dalam memberikan bobot pada kata secara lebih seimbang. Metode ini tidak hanya memperhitungkan frekuensi kemunculan kata dalam sebuah dokumen (*term frequency*), tetapi juga mempertimbangkan tingkat keumuman kata tersebut dalam seluruh kumpulan dokumen (*inverse document frequency*). Kata-kata yang sifatnya terlalu umum, seperti kata yang muncul pada hampir semua dokumen akan diberikan bobot lebih rendah dibandingkan kata-kata yang spesifik dan berpotensi menjadi indikator penting dalam membedakan dokumen asli dan palsu.

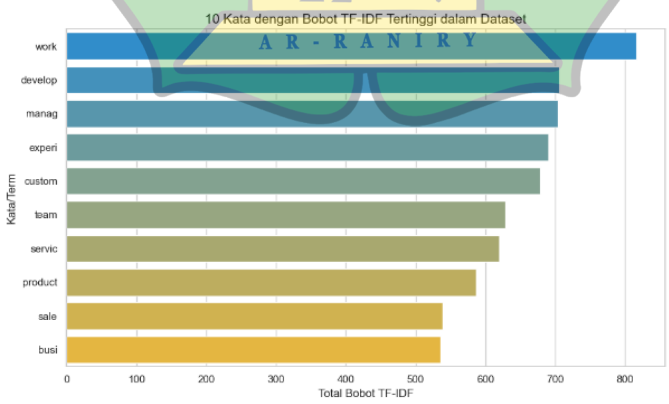
Proses ekstraksi fitur dilakukan melalui kelas *TfidfVectorizer* yang disediakan oleh library Scikit-learn. Library ini dipilih karena menyediakan implementasi TF-IDF yang efisien, mudah digunakan, serta telah terbukti akurat dalam berbagai studi *text mining. TfidfVectorizer* secara otomatis menjalankan dua proses utama, yaitu membangun daftar kosakata (*vocabulary*) dari seluruh dokumen yang telah di *preprocessing* serta menghitung bobot TF-IDF untuk setiap kata dalam kosakata tersebut. Seluruh proses ini berlangsung secara terintegrasi sehingga memastikan konsistensi hasil ekstraksi fitur.

Secara teknis, *TfidfVectorizer* mengubah kumpulan dokumen menjadi sebuah matriks numerik berdimensi tinggi. Pada matriks tersebut, setiap baris merepresentasikan satu dokumen lowongan pekerjaan, sementara setiap kolom

merupakan satu fitur kata unik. Nilai pada setiap sel matriks menunjukkan bobot TF-IDF, yaitu seberapa relevan kata tersebut bagi dokumen tertentu. Matriks yang dihasilkan memiliki bentuk *sparse matrix,* yakni matriks yang sebagian besar nilai elemennya bernilai nol. Struktur seperti ini sangat umum dalam representasi teks, karena satu dokumen hanya mengandung sebagian kecil dari keseluruhan kosakata. Penggunaan *sparse matrix* sangat membantu meningkatkan efisiensi penyimpanan dan pemrosesan data berukuran besar.

Ekstraksi fitur menggunakan TF-IDF menghasilkan ribuan fitur kata dengan bobot yang bervariasi. Kata-kata seperti *work*, *develop*, dan *manag* memperoleh bobot tinggi karena frekuensi kemunculannya yang signifikan dalam deskripsi pekerjaan, sehingga berperan penting sebagai pembeda antar dokumen. Bobot TF-IDF ini selanjutnya menjadi dasar pembelajaran bagi algoritma Naive Bayes dan SVM, di mana kata berbobot tinggi berkontribusi lebih besar terhadap keputusan klasifikasi dibandingkan kata berbobot rendah.

Untuk memberikan gambaran yang lebih jelas, penelitian ini menampilkan visualisasi sepuluh kata dengan bobot TF-IDF tertinggi pada Gambar 4.3. Visualisasi tersebut mencerminkan pola umum yang digunakan model dalam membedakan lowongan pekerjaan asli dan palsu, sekaligus mengonfirmasi bahwa TF-IDF efektif dalam menonjolkan fitur-fitur yang paling informatif dan relevan untuk proses klasifikasi.



Gambar 4. 3 Kata dengan Bobot TF-IDF Tertinggi dalam Dataset

Melalui representasi numerik ini, kedua algoritma dapat mengidentifikasi pola kemunculan kata secara matematis, mempelajari karakteristik bahasa pada

lowongan pekerjaan, dan membangun model klasifikasi yang mampu membedakan dokumen asli dari dokumen palsu secara lebih efektif.

4.3.2 Implementasi 10-*Fold Cross Validation*

Penelitian ini menerapkan 10-Fold Cross Validation sebagai teknik evaluasi model. Dataset sebanyak 17.880 entri dibagi secara acak ke dalam sepuluh *fold* dengan ukuran relatif sama, yaitu sekitar 1.788 data per *fold*. Pada setiap iterasi, satu *fold* berfungsi sebagai data uji sementara sembilan *fold* lainnya sebagai data latih, sehingga seluruh data memperoleh kesempatan diuji satu kali dan dilatih sembilan kali. Implementasi dilakukan menggunakan kelas *KFold* dari library *Scikit-learn* yang membuat pembagian data berlangsung secara sistematis dan terstandarisasi, termasuk pengaturan parameter pengacakan (*shuffle*) untuk memastikan variasi data pada setiap iterasi.

Pendekatan ini dipilih karena karakteristik dataset yang tidak seimbang, di mana jumlah lowongan asli jauh lebih banyak dibandingkan lowongan palsu. Mekanisme *cross validation* memastikan setiap iterasi menghasilkan kombinasi data latih dan data uji yang berbeda, sehingga model tidak hanya mempelajari pola dari kelas dominan dan hasil evaluasi lebih mencerminkan kemampuan generalisasi yang sesungguhnya. Proporsi pembagian data pada setiap iterasi disajikan pada Tabel 4.7.

Tabel 4. 7 Pembagian Data K-Fold Cross Validation

Komponen	Proporsi	Jumlah Baris
Data Latih	9/10	16.092
Data Uji	1/10	1.788
Total	10/10	17.880

Berdasarkan Tabel 4.7, setiap iterasi memanfaatkan 16.092 data untuk membangun model dan 1.788 data untuk mengukur kinerjanya. Komposisi ini memastikan sebagian besar data berkontribusi pada proses pembelajaran, sementara sisanya digunakan sebagai tolok ukur evaluasi yang objektif, sehingga kesimpulan mengenai efektivitas algoritma dapat diperoleh secara valid dan ilmiah.

4.3.3 Implementasi Model *Naive Bayes*

Tahap berikutnya pada penelitian ini adalah melakukan pelatihan dan pengujian model klasifikasi menggunakan algoritma Naive Bayes. Algoritma ini dipilih karena efektif untuk klasifikasi teks berskala besar dan mampu bekerja optimal pada data berdimensi tinggi, termasuk data TF-IDF. Penelitian ini menggunakan varian *Multinomial Naive Bayes* karena sesuai untuk data numerik yang merepresentasikan frekuensi atau bobot kemunculan kata dalam dokumen.

Implementasi model dilakukan melalui kelas MultinomialNB dari Scikit-learn dengan parameter *alpha* = 0.01 sebagai smoothing Laplace untuk mencegah probabilitas nol. Evaluasi model menggunakan metode 10-Fold Cross Validation melalui kelas KFold dengan pengaturan *n_splits=10*, *shuffle=True*, dan *random_state=42*. Pada setiap iterasi, TF-IDF selalu di-fit pada data latih dan kemudian ditransformasikan ke data uji agar tidak terjadi data leakage. Potongan sintaks implementasi tersebut ditunjukkan sebagai berikut:

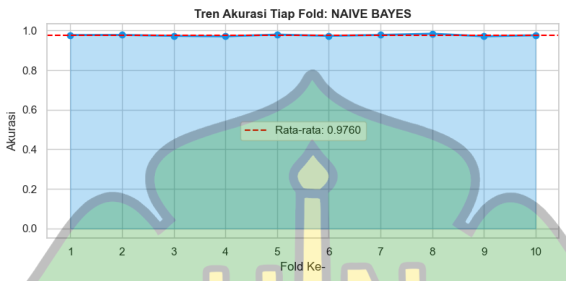
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import KFold
from sklearn.feature_extraction.text import TfidfVectorizer

kf = KFold(n_splits=10, shuffle=True, random_state=42)
tfidf = TfidfVectorizer(max_features=5000)

fold_accuracies = []
for i, (train_idx, test_idx) in enumerate(kf.split(X), 1):
    X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
    y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

    X_train_tfidf = tfidf.fit_transform(X_train)
    X_test_tfidf  = tfidf.transform(X_test)

    model = MultinomialNB(alpha=0.01)
    model.fit(X_train_tfidf, y_train)
    y_pred = model.predict(X_test_tfidf)

    acc = accuracy_score(y_test, y_pred)
    fold_accuracies.append(acc)
```

Hasil pengujian pada setiap fold disajikan pada Gambar 4.4 berikut.

```
================================================================================
---------------------- DETAIL PROSES EVALUASI K-FOLD: NAIVE BAYES ----------------------
================================================================================
Fold    | Data latih dari fold                | Data uji   | Akurasi (%)     | Desimal
--------------------------------------------------------------------------------
1       | 2,3,4,5,6,7,8,9,10                  | 1          |      97.71% |  0.977069351
2       | 1,3,4,5,6,7,8,9,10                  | 2          |      97.76% |  0.977628635
3       | 1,2,4,5,6,7,8,9,10                  | 3          |      97.32% |  0.973154362
4       | 1,2,3,5,6,7,8,9,10                  | 4          |      97.09% |  0.970917226
5       | 1,2,3,4,6,7,8,9,10                  | 5          |      97.93% |  0.979306488
6       | 1,2,3,4,5,7,8,9,10                  | 6          |      97.32% |  0.973154362
7       | 1,2,3,4,5,6,8,9,10                  | 7          |      97.76% |  0.977628635
8       | 1,2,3,4,5,6,7,9,10                  | 8          |      98.38% |  0.983780761
9       | 1,2,3,4,5,6,7,8,10                  | 9          |      97.09% |  0.970917226
10      | 1,2,3,4,5,6,7,8,9                   | 10         |      97.60% |  0.975950783
--------------------------------------------------------------------------------
Rata-rata                                     |      97.60% |  0.975950783
================================================================================
```

Gambar 4. 4 Proses Evaluasi K-Fold Naive Bayes

Selanjutnya, grafik tren akurasi pada Gambar 4.5 menggambarkan distribusi nilai akurasi model Naive Bayes di setiap iterasi pengujian.



Gambar 4. 5 Tren Akurasi Tiap Fold Naive Bayes

Berdasarkan sepuluh pengujian yang dilakukan, model menghasilkan rata-rata akurasi sebesar 97,60%, yang menunjukkan bahwa Naive Bayes bekerja dengan baik dalam mengklasifikasikan lowongan pekerjaan asli dan palsu. Namun, karena dataset bersifat tidak seimbang, akurasi saja belum cukup untuk menggambarkan performa model secara menyeluruh. Oleh sebab itu, interpretasi akurasi perlu dilengkapi dengan analisis metrik evaluasi lain agar kualitas model dapat dinilai secara lebih objektif.

4.3.4 Implementasi Model SVM

Setelah pelatihan dan pengujian menggunakan algoritma Naive Bayes selesai dilakukan, penelitian berlanjut dengan menerapkan algoritma SVM sebagai metode pembanding. SVM bekerja dengan membentuk *hyperplane* optimal yang memisahkan kelas secara maksimal dalam ruang fitur TF-IDF, sehingga mampu meningkatkan kemampuan generalisasi model dalam membedakan lowongan asli dan palsu.

Pada penelitian ini digunakan kernel linear dengan parameter class_weight='balanced'. Kernel linear dipilih karena sesuai dengan karakteristik data teks yang memiliki jumlah fitur sangat besar dan umumnya dapat dipisahkan

secara linear tanpa memerlukan transformasi non-linear. Sementara itu, penggunaan *class_weight='balanced'* bertujuan menyesuaikan bobot kelas agar model tidak bias terhadap kelas mayoritas. Proses evaluasi SVM dilakukan dengan skema yang sama seperti pada Naive Bayes, yaitu 10-Fold Cross Validation menggunakan KFold(n_splits=10, shuffle=True, random_state=42) untuk menjaga konsistensi perbandingan performa. Potongan sintaks implementasi SVM dapat dilihat pada bagian berikutnya sebagai pendukung proses evaluasi.

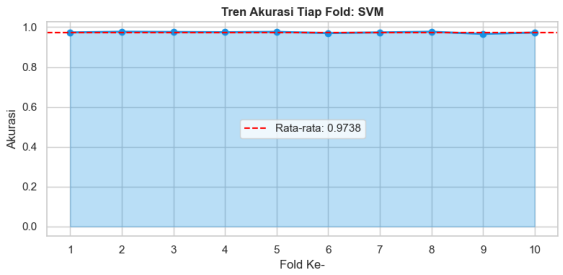
```
from sklearn.svm import SVC
from sklearn.model_selection import KFold
from sklearn.feature_extraction.text import TfidfVectorizer

kf = KFold(n_splits=10, shuffle=True, random_state=42)
tfidf = TfidfVectorizer(max_features=5000)

fold_accuracies = []
for i, (train_idx, test_idx) in enumerate(kf.split(X), 1):
    X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
    y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

    X_train_tfidf = tfidf.fit_transform(X_train)
    X_test_tfidf  = tfidf.transform(X_test)

    model = SVC(kernel='linear', class_weight='balanced')
    model.fit(X_train_tfidf, y_train)
    y_pred = model.predict(X_test_tfidf)

    acc = accuracy_score(y_test, y_pred)
    fold_accuracies.append(acc)
```

Hasil pengujian pada setiap fold disajikan pada Gambar 4.6 berikut.

```
================================================================================
============================ DETAIL PROSES EVALUASI K-FOLD: SVM ============================
================================================================================
Fold   | Data latih dari fold                | Data uji   | Akurasi (%)     | Desimal
--------------------------------------------------------------------------------
1      | 2,3,4,5,6,7,8,9,10                  | 1          |      97.37% |  0.973713647
2      | 1,3,4,5,6,7,8,9,10                  | 2          |      97.71% |  0.977069351
3      | 1,2,4,5,6,7,8,9,10                  | 3          |      97.60% |  0.975950783
4      | 1,2,3,5,6,7,8,9,10                  | 4          |      97.48% |  0.974832215
5      | 1,2,3,4,6,7,8,9,10                  | 5          |      97.65% |  0.976510067
6      | 1,2,3,4,5,7,8,9,10                  | 6          |      97.04% |  0.970357942
7      | 1,2,3,4,5,6,8,9,10                  | 7          |      97.37% |  0.973713647
8      | 1,2,3,4,5,6,7,9,10                  | 8          |      97.71% |  0.977069351
9      | 1,2,3,4,5,6,7,8,10                  | 9          |      96.59% |  0.965883669
10     | 1,2,3,4,5,6,7,8,9                   | 10         |      97.32% |  0.973154362
--------------------------------------------------------------------------------
Rata-rata                                    |      97.38% |  0.973825503
================================================================================
```

Gambar 4. 6 Proses Evaluasi K-Fold SVM

Selanjutnya, grafik tren akurasi pada Gambar 4.7 menggambarkan distribusi nilai akurasi model SVM di setiap iterasi pengujian.



Gambar 4. 7 Tren Akurasi Tiap Fold SVM

Berdasarkan hasil sepuluh kali pengujian, diperoleh nilai rata-rata akurasi sebesar 97,38%, yang menunjukkan bahwa secara keseluruhan algoritma SVM memiliki performa yang sebanding dengan Naive Bayes dalam skema pengujian yang digunakan.

4.4 Evaluasi Model

Tahap evaluasi model pada penelitian ini dilakukan untuk melihat kemampuan algoritma Naive Bayes dan SVM dalam mengelompokkan lowongan pekerjaan asli dan palsu secara akurat dan konsisten. Proses evaluasi menggunakan metode 10-Fold Cross Validation, di mana performa model dihitung berdasarkan rata-rata hasil prediksi dari sepuluh iterasi. Penggunaan nilai rata-rata ini bertujuan mengurangi pengaruh ketidakseimbangan performa antar fold yang mungkin muncul akibat karakteristik data tertentu.

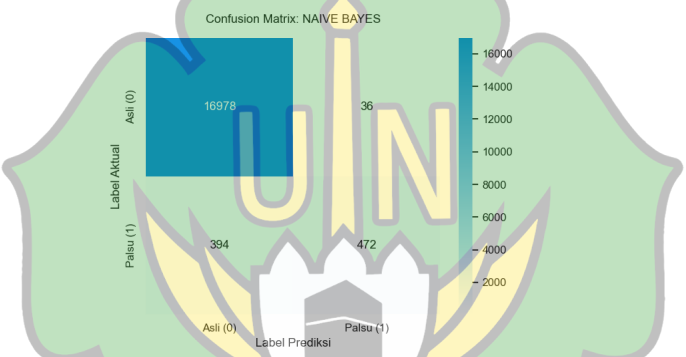
Penelitian ini menilai performa model tidak hanya melalui akurasi, tetapi juga menggunakan Precision, Recall, dan F1-Score. Seluruh metrik dihitung berdasarkan Confusion Matrix untuk mengetahui jumlah prediksi benar dan salah pada tiap kelas. Penggunaan beberapa metrik evaluasi menjadi penting karena dataset memiliki ketidakseimbangan kelas, di mana jumlah lowongan asli jauh lebih banyak dibandingkan lowongan palsu.

Dalam kondisi data yang tidak seimbang, akurasi saja sering tidak cukup menggambarkan performa model secara objektif, sehingga Precision, Recall, dan F1-Score digunakan sebagai indikator yang lebih relevan. Precision mengukur ketepatan model dalam memberi label palsu, sedangkan Recall menilai kemampuan model menemukan seluruh lowongan palsu pada dataset, dan F1-Score memberikan keseimbangan antara keduanya. Proses perhitungan metrik dilakukan menggunakan library Scikit-learn, melalui fungsi *confusion_matrix* dan

classification_report, sehingga evaluasi berlangsung secara konsisten, terstandarisasi, dan minim kesalahan. Hasil evaluasi ini menjadi dasar dalam membandingkan performa Naive Bayes dan SVM untuk menentukan metode yang paling efektif dalam mendeteksi keaslian lowongan pekerjaan di media sosial.

4.4.1 Hasil Evaluasi Naive Bayes

Evaluasi performa algoritma Naive Bayes diawali dengan analisis Confusion Matrix, yang digunakan untuk memberikan gambaran menyeluruh mengenai kualitas prediksi model dengan menunjukkan hubungan langsung antara label asli dan hasil prediksi. Melalui matriks ini, dapat diketahui secara rinci jumlah prediksi yang benar serta kesalahan klasifikasi yang terjadi. Visualisasi Confusion Matrix untuk model Naive Bayes ditampilkan pada Gambar 4.8.



Gambar 4. 8 Confusion Matrix Naive Bayes

Berdasarkan visualisasi pada Gambar 4.8, hasil klasifikasi Naive Bayes dijabarkan ke dalam empat komponen utama sebagai berikut:

- True Negative (TN): Model mampu mengklasifikasikan 16.978 data lowongan asli dengan benar sebagai kelas asli
- True Positive (TP): Model berhasil mendeteksi 472 data lowongan palsu secara tepat.
- False Positive (FP): Terdapat 36 data lowongan asli yang keliru diklasifikasikan sebagai palsu.
- False Negative (FN): Terdapat 394 data lowongan palsu yang gagal terdeteksi dan justru dianggap sebagai lowongan asli oleh model.

Untuk memastikan kesesuaian hasil evaluasi yang diperoleh dari sistem, dilakukan perhitungan manual metrik evaluasi berdasarkan nilai *Confusion Matrix* tersebut.

a) Perhitungan Akurasi

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

$$= \frac{472 + 16.978}{472 + 16.978 + 36 + 394} \times 100\%$$

$$= \frac{17.450}{17.880} \times 100\% = 0.9760 \times 100\% = 97.60\%$$

Sehingga diperoleh nilai akurasi sebesar 97.60%.

b) Untuk kelas palsu

Pada penelitian ini, kelas lowongan palsu diperlakukan sebagai kelas positif, karena tujuan utama model adalah mendeteksi lowongan pekerjaan yang bersifat penipuan. Nilai yang digunakan adalah:

TN	TP	FN	FP
16.978	472	394	36

Proses perhitungan:

- $Precision = \frac{TP}{TP+FP} \times 100\% = \frac{472}{472+36} \times 100\%$

$$= \frac{472}{508} \times 100\% = 0.9291 \times 100\% = 92.91\%$$

- $Recall = \frac{TP}{TP+FN} \times 100\% = \frac{472}{472+394} \times 100\%$

$$= \frac{472}{866} \times 100\% = 0.5450 \times 100\% = 54.50\%$$

- $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\%$

$$= 2 \times \frac{0.9291 \times 0.5450}{0.9291 + 0.5450} \times 100\% = 2 \times \frac{0.5063}{1.4741} \times 100\%$$

$$= 0.6870 \times 100\% = 68.70\%$$

c) Untuk kelas asli

Pada perhitungan ini, kelas lowongan asli diperlakukan sebagai kelas positif, sehingga nilai yang digunakan adalah:

TP	TN	FP	FN
16.978	472	394	36

Proses perhitungan:

- $Precision = \frac{TP}{TP+FP} \times 100\% = \frac{16.978}{16.978+394} \times 100\%$

$$= \frac{16.978}{17.372} \times 100\% = 0.9773 \times 100\% = 97.73\%$$

- $Recall = \frac{TP}{TP+FN} \times 100\% = \frac{16.978}{16.978+36} \times 100\%$

$$= \frac{16.978}{17.014} \times 100\% = 0.9979 \times 100\% = 99.79\%$$

- $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \times 100\%$

$$= 2 \times \frac{0.9773 \times 0.9979}{0.9773 + 0.9979} \times 100\%$$

$$= 2 \times \frac{0.9752}{1.9752} \times 100\% = 0.9875 \times 100\% = 98.75\%$$

Nilai evaluasi yang sangat tinggi pada kelas lowongan asli menunjukkan bahwa model Naive Bayes sangat baik dalam mengenali kelas lowongan asli. Jumlah *False Negative* yang relatif tinggi pada kelas lowongan palsu menunjukkan bahwa model Naive Bayes masih memiliki keterbatasan dalam mendeteksi seluruh lowongan palsu. Kondisi ini dipengaruhi oleh ketidakseimbangan dataset, sehingga model cenderung lebih fokus pada pengenalan kelas lowongan asli.

Untuk memperoleh gambaran performa yang lebih menyeluruh, evaluasi tambahan dilakukan menggunakan fungsi classification_report dari library Scikit-learn. Visualisasi tersebut ditunjukkan pada Gambar 4.9 berikut.

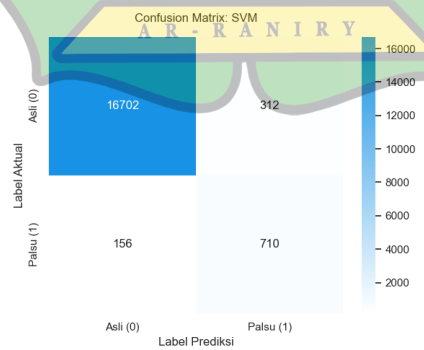
```
============================================================
                HASIL EVALUASI NAIVE BAYES
============================================================
              precision    recall  f1-score   support

           0     0.9773    0.9979    0.9875     17014
           1     0.9291    0.5450    0.6870       866

    accuracy                         0.9760     17880
   macro avg     0.9532    0.7715    0.8373     17880
weighted avg     0.9750    0.9760    0.9729     17880

============================================================
```

Gambar 4. 9 Hasil Evaluasi Naive Bayes

Hasil evaluasi menunjukkan nilai *macro average* dengan *precision* sebesar 0,95, recall sebesar 0,77, dan F1-score sebesar 0,84. Nilai recall yang lebih rendah menegaskan bahwa model belum optimal dalam menangkap seluruh data lowongan palsu. Sementara itu, nilai *weighted average* menunjukkan performa yang sangat tinggi dengan precision 0,97, recall 0,98, dan F1-score 0,97, yang dipengaruhi oleh dominasi kelas asli dalam dataset.

4.4.2 Hasil Evaluasi SVM

Evaluasi selanjutnya dilakukan pada model SVM dengan pendekatan yang sama, yaitu melalui analisis Confusion Matrix. Analisis ini digunakan untuk menilai sejauh mana *hyperplane* yang dibentuk oleh SVM mampu memisahkan data lowongan pekerjaan asli dan palsu secara optimal pada ruang fitur berdimensi tinggi yang dihasilkan melalui ekstraksi TF-IDF. Visualisasi Confusion Matrix untuk model SVM yang ditampilkan pada Gambar 4.10 menjadi dasar utama dalam proses evaluasi performa klasifikasi.



Gambar 4. 10 Confusion Matrix SVM

Berdasarkan hasil pengujian yang divisualisasikan pada Gambar 4.10, model SVM menunjukkan detail ketepatan klasifikasi terhadap label sebagai berikut:

- True Negative (TN): Model SVM berhasil mengklasifikasikan 16.702 data lowongan asli dengan benar.
- True Positive (TP): SVM mampu mendeteksi 710 data lowongan palsu secara tepat, yang menunjukkan peningkatan signifikan dibandingkan model Naive Bayes.
- False Positive (FP): Terdapat 312 data lowongan asli yang salah diklasifikasikan sebagai palsu.
- False Negative (FN): Model SVM hanya memiliki 156 data lowongan palsu yang gagal terdeteksi.

Untuk memverifikasi hasil evaluasi model SVM, dilakukan perhitungan manual metrik evaluasi berdasarkan nilai *Confusion Matrix* tersebut.

a) Nilai akurasi secara umum pada metode SVM, yaitu:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

$$= \frac{710 + 16.702}{710 + 16.702 + 312 + 156} \times 100\%$$

$$= \frac{17.412}{17.880} \times 100\% = 0.9738 \times 100\% = 97.38\%$$

b) Untuk kelas palsu

TN	TP	FN	FP
16.702	710	156	312

Proses perhitungan:

- $Precision = \frac{TP}{TP+FP} \times 100\% = \frac{710}{710+312} \times 100\%$

$$= \frac{710}{1.022} \times 100\% = 0.6947 \times 100\% = 69.47\%$$

- $Recall = \frac{TP}{TP+FN} \times 100\% = \frac{710}{710+156} \times 100\%$

$$= \frac{710}{866} \times 100\% = 0.8199 \times 100\% = 81.99\%$$

- $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\%$

$$= 2 \times \frac{0.6947 \times 0.8199}{0.6947 + 0.8199} \times 100\%$$

$$= 2 \times \frac{0.5695}{1.5146} \times 100\% = 0.7521 \times 100\% = 75.21\%$$

c) Untuk kelas asli

TP	TN	FP	FN
16.702	710	156	312

Proses perhitungan:

- $Precision = \frac{TP}{TP+FP} \times 100\% = \frac{16.702}{16.702+156} \times 100\%$

$$= \frac{16.702}{16.858} \times 100\% = 0.9907 \times 100\% = 99.07\%$$

- $Recall = \frac{TP}{TP+FN} \times 100\% = \frac{16.702}{16.702+312} \times 100\%$

$$= \frac{16.702}{17.014} \times 100\% = 0.9817 \times 100\% = 98.17\%$$

- $F1 - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \times 100\%$

$$= 2 \times \frac{0.9907 \times 0.9817}{0.9907 + 0.9817} \times 100\%$$

$$= 2 \times \frac{0.9725}{1.9724} \times 100\% = 0.9862 \times 100\% = 98.62\%$$

Jumlah *False Negative* yang relatif lebih rendah pada kelas lowongan palsu menunjukkan bahwa SVM memiliki sensitivitas yang lebih baik terhadap kelas minoritas dibandingkan Naive Bayes. Hal ini dipengaruhi oleh penggunaan parameter class_weight='balanced' pada *LinearSVC*, yang meningkatkan bobot pelatihan untuk kelas minoritas. Strategi ini membuat model lebih peka terhadap data lowongan palsu, meskipun efek sampingnya adalah bertambahnya jumlah *False Positive* pada kelas lowongan asli. Dalam konteks deteksi penipuan, pendekatan ini dinilai relevan karena risiko melewatkan kasus penipuan dianggap lebih berbahaya dibandingkan salah memberikan label lowongan asli sebagai palsu.

Untuk memperoleh gambaran performa yang lebih komprehensif, penelitian ini juga menggunakan fungsi classification_report dari Scikit-learn. Visualisasi tersebut ditampilkan pada Gambar 4.11.

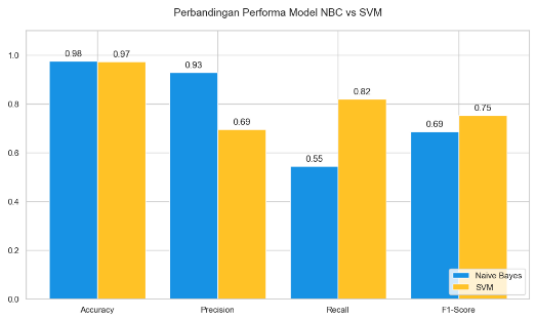
```
============================================================
                    HASIL EVALUASI SVM
============================================================
              precision    recall  f1-score   support

           0     0.9907    0.9817    0.9862     17014
           1     0.6947    0.8199    0.7521       866

    accuracy                         0.9738     17880
   macro avg     0.8427    0.9008    0.8692     17880
weighted avg     0.9764    0.9738    0.9748     17880

============================================================
```

Gambar 4. 11 Hasil Evaluasi SVM

Hasilnya menunjukkan nilai *macro average* dengan *precision* sebesar 0,84, recall sebesar 0,90, dan F1-Score sebesar 0,87. Nilai *macro average* dengan *recall* yang cukup tinggi mengidentifikasi bahwa SVM mampu mendeteksi kedua kelas secara lebih seimbang. Sementara itu, nilai *weighted average* menunjukkan performa yang stabil dengan *precision* sebesar 0,98, *recall* sebesar 0,97, dan *F1-Score* sebesar 0,97, yang menunjukkan konsistensi model terhadap seluruh data dengan mempertimbangkan distribusi kelas yang tidak seimbang.

4.4.3 Perbandingan Performa Model

Perbandingan performa antara algoritma Naive Bayes dan SVM dilakukan untuk memahami karakteristik, keunggulan, serta keterbatasan masing-masing model dalam mengklasifikasikan lowongan pekerjaan asli maupun palsu. Perbandingan ini menggunakan empat metrik evaluasi utama, yaitu *Accuracy*, *Precision*, *Recall*, dan *F1-Score*, yang seluruhnya dihitung berdasarkan hasil pengujian dengan metode 10-Fold Cross Validation. Visualisasi perbandingan performa kedua model ditampilkan pada Gambar 4.12.



Gambar 4. 12 Perbandingan Performa Naive Bayes dan SVM

Berdasarkan hasil analisis, kedua algoritma menunjukkan nilai akurasi yang relatif tinggi. Model Naive Bayes memperoleh akurasi sebesar 0,98, sedangkan model SVM mencapai akurasi 0,97. Perbedaan yang sangat kecil antara kedua nilai ini mengindikasikan bahwa keduanya memiliki kemampuan yang sama-sama baik dalam mengenali pola umum pada dataset. Hal ini menandakan bahwa dari perspektif akurasi total, kedua model dapat bekerja secara efektif dalam tugas klasifikasi lowongan pekerjaan, baik asli maupun palsu.

Namun, hasil pada metrik Precision untuk kelas lowongan palsu, algoritma Naive Bayes menunjukkan kinerja yang jauh lebih unggul dibandingkan SVM. Naive Bayes memiliki precision sebesar 0,93, sedangkan SVM hanya sebesar 0,69. Precision yang tinggi pada Naive Bayes berarti bahwa sebagian besar prediksi palsu yang dihasilkan oleh model ini benar-benar berasal dari lowongan palsu. Naive Bayes lebih minim menghasilkan *False Positive*, sehingga lebih efektif jika tujuan sistem adalah menghindari kesalahan dalam menandai lowongan asli sebagai palsu. Kondisi ini berkaitan dengan kecenderungan Naive Bayes yang lebih hati-hati dalam memberikan label positif pada kelas palsu.

Sebaliknya, pada metrik Recall, SVM unggul secara signifikan dengan nilai 0,82, sementara Naive Bayes hanya mencapai 0,55. Nilai ini menunjukkan bahwa SVM lebih efektif dalam mendeteksi sebagian besar lowongan palsu yang terdapat dalam dataset, sehingga risiko lowongan palsu yang tidak terdeteksi dapat diminimalkan.

Pada metrik F1-Score, yang merepresentasikan keseimbangan antara Precision dan Recall. SVM menghasilkan performa yang lebih baik dengan nilai 0,75, dibandingkan Naive Bayes yang memperoleh nilai 0,69. Hal ini menunjukkan bahwa secara keseluruhan SVM memiliki keseimbangan yang lebih baik antara ketepatan dan sensitivitas dalam mendeteksi penipuan.

Secara keseluruhan, hasil perbandingan memperlihatkan bahwa Naive Bayes lebih unggul dari sisi akurasi dan precision, sehingga lebih sesuai digunakan pada sistem yang membutuhkan ketelitian tinggi dalam memastikan bahwa prediksi palsu benar-benar akurat. Namun, SVM lebih unggul dalam mendeteksi lowongan palsu secara menyeluruh, yang dibuktikan oleh nilai recall dan F1-Score yang lebih

tinggi. Hal ini menjadikan SVM lebih relevan untuk mendeteksi penipuan, di mana tujuan utamanya adalah meminimalkan kemungkinan terlewatnya lowongan palsu.

4.5 Interpretasi Hasil

Hasil penelitian menunjukkan bahwa serangkaian tahapan mulai dari preprocessing hingga evaluasi model berkontribusi signifikan terhadap kualitas model klasifikasi yang dihasilkan. Tahap preprocessing berperan dalam membersihkan dan menyeragamkan teks dengan menghapus elemen tidak informatif serta mereduksi variasi bentuk kata yang berpotensi menimbulkan noise, menggunakan library NLTK dan Pandas. Hasil preprocessing tersebut kemudian direpresentasikan secara numerik melalui metode TF-IDF menggunakan TfidfVectorizer dari Scikit-learn, di mana kata-kata seperti *work*, *experience*, dan *team* memperoleh bobot tinggi karena relevansinya dalam menggambarkan konteks deskripsi lowongan pekerjaan.

Tabel 4. 8 Hasil Evaluasi Model

Lowongan Pekerjaan	Naive Bayes				SVM			
	Acc	*Prec*	*Rec*	*F1*	*Acc*	*Prec*	*Rec*	*F1*
Asli	97.60%	97.73%	99.79%	98.75%	97.38%	99.07%	98.17%	98.62%
Palsu	97.60%	92.91%	54.50%	68.70%	97.38%	69.47%	81.99%	75.21%

Berdasarkan Tabel 4.8, kedua algoritma menunjukkan performa yang sangat baik dalam mengklasifikasikan lowongan pekerjaan asli. Nilai recall lowongan asli yang tinggi yaitu 99,79% pada Naive Bayes dan 98,17% pada SVM mengindikasikan bahwa sebagian besar lowongan asli berhasil teridentifikasi dengan benar, sehingga risiko misklasifikasi pada kelas ini tergolong kecil.

Perbedaan performa kedua algoritma tampak lebih jelas pada kelas lowongan palsu. Naive Bayes menghasilkan precision sebesar 92,91%, namun recall-nya hanya mencapai 54,50% dengan F1-score 68,70%, mengindikasikan bahwa model cenderung kurang sensitif terhadap kelas minoritas dan masih banyak melewatkan sampel lowongan palsu. Kondisi ini sejalan dengan karakteristik Naive Bayes sebagai algoritma probabilistik yang kurang optimal ketika berhadapan dengan distribusi kelas yang tidak seimbang. Sebaliknya, SVM dengan penerapan parameter *class_weight='balanced'* menunjukkan recall yang lebih tinggi sebesar

81,99% dan F1-score 75,21%, meskipun precision-nya lebih rendah di angka 69,47%. Hal ini menunjukkan bahwa SVM memiliki keseimbangan performa yang lebih baik dalam mendeteksi lowongan palsu secara menyeluruh dibandingkan Naive Bayes.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pembahasan dan analisis yang telah dilakukan pada bab sebelumnya, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1. Penelitian ini berhasil menerapkan dua algoritma machine learning, yaitu *Naive Bayes Classifier* dan *Support Vector Machine* (SVM), dalam mengklasifikasikan keaslian lowongan pekerjaan di media sosial. Dataset yang digunakan berjumlah 17.880 data, yang terdiri dari 17.014 lowongan asli dan 866 lowongan palsu, serta telah melalui seluruh tahapan penelitian sesuai dengan metodologi CRISP-DM.
2. Tahapan preprocessing teks yang meliputi cleaning, case folding, tokenizing, filtering dan stemming, dikombinasikan dengan metode *Term Frequency–Inverse Document Frequency* (TF-IDF), terbukti mampu menghasilkan representasi fitur yang efektif. Hal ini terlihat dari kestabilan performa kedua model pada skema 10-Fold Cross Validation dengan nilai akurasi yang konsisten di atas 97%.
3. Hasil evaluasi menunjukkan bahwa algoritma Naive Bayes memperoleh akurasi sebesar 97,60%, dengan precision sebesar 92.91%, recall sebesar 54.50%, dan F1-score sebesar 68.70%. Nilai recall yang relatif rendah menunjukkan bahwa model masih melewatkan sejumlah lowongan palsu, sehingga sensitivitas Naive Bayes terhadap kelas minoritas belum optimal.
4. Algoritma SVM menghasilkan akurasi sebesar 97,38%, dengan precision sebesar 69.47%, recall sebesar 81.99%, dan F1-score sebesar 75.21%. Nilai recall yang lebih tinggi dibandingkan Naive Bayes menunjukkan bahwa SVM lebih efektif dalam mendeteksi lowongan kerja palsu dan mampu menekan jumlah false negative secara signifikan.
5. Berdasarkan perbandingan nilai akurasi, precision, recall, dan F1-score, dapat disimpulkan bahwa SVM merupakan model yang lebih sesuai untuk digunakan dalam sistem deteksi lowongan kerja palsu, karena memiliki

kemampuan deteksi yang lebih baik terhadap kasus penipuan, meskipun harus mengorbankan sebagian nilai presisi.

5.2 Saran

Berdasarkan kesimpulan yang telah diperoleh dari penelitian ini, beberapa saran yang dapat diajukan untuk pengembangan penelitian selanjutnya adalah sebagai berikut:

1. Penelitian selanjutnya disarankan untuk menggunakan dataset dengan jumlah yang lebih besar dan distribusi kelas yang lebih seimbang, khususnya dengan menambah data lowongan palsu, agar model dapat mempelajari pola penipuan secara lebih menyeluruh.
2. Penerapan teknik penanganan data tidak seimbang, seperti oversampling (SMOTE) atau class weighting, dapat dilakukan untuk meningkatkan nilai recall tanpa meningkatkan jumlah kesalahan prediksi secara signifikan.
3. Penelitian lanjutan dapat membandingkan hasil penelitian ini dengan algoritma klasifikasi lain, seperti Random Forest, Gradient Boosting, atau model berbasis deep learning, untuk memperoleh performa yang lebih optimal.
4. Analisis lebih mendalam terhadap kasus kesalahan klasifikasi, khususnya false negative, perlu dilakukan untuk mengidentifikasi karakteristik lowongan palsu yang sulit terdeteksi dan sebagai dasar pengembangan model yang lebih baik.
5. Model klasifikasi yang telah dikembangkan dapat diimplementasikan dalam bentuk sistem berbasis web atau aplikasi, sehingga dapat dimanfaatkan secara langsung oleh masyarakat sebagai alat bantu dalam memverifikasi keaslian lowongan pekerjaan di media sosial.

DAFTAR PUSTAKA

Aditya, A., Wahyuddin, P., Leo, S., Santoso, W., Wahyu, G., Wibowo, N., Khrisna, A., Rahmaddeni, W., Wahidin, A. J., Eka, G., Elisawati, Y., Rizqi, R., & Abdurrasyid, W. (2023). *Machine Learning* (A. Yanto. M.Pd., Ed.; 1st ed.). PT Global Eksekutif Teknologi.

Angkasa, V., & Pangaribuan, J. J. (2022). Komparasi Tingkat Akurasi Random Forest dan KNN Untuk Mendiagnosis Penyakit Kanker Payudara. *Journal Information System Development (ISD)*, *7*(1), 34–41.

Ayuningtyas, P., & Tantyoko, H. (2024). Comparison of the Word2vec Skipgram Model Method Linkaja Application Review using Bidirectional LSTM Algorithm and Support Vector Machine. *Jurnal Sistem Dan Teknologi Informasi (JUSTIN)*, *12*(1), 189–196.

Bansal, S. (2020, February 29). *Real or Fake? Fake Job Posting Prediction*. https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction

Bintoro, P., Ratnasari, Wihardjo, E., Pratiwi Putri, I., & Asari, A. (2024). *Pengantar Machine Learning* (A. Asari, Ed.; 1st ed.). PT Mafy Media Literasi Indonesia.

Dermawan, I. P. (2025). *Analisis Sentimen Komentar Youtube Terhadap Kebijakan Pemerintah Menggunakan Algoritma K-Nearest Neighboar (KNN)*. Universitas Satya Negara Indonesia Jakarta.

Dzikri, M. H., Setiawan, I. R., & Indrayana, D. (2024). Penerapan Algoritma Naive Bayes Untuk Mendeteksi Penipuan Lowongan Pekerjaan. *Jurnal Sistem Informasi Dan Teknik Komputer*, *9*(2), 102–109.

Fauzi, W. (2023). Perancangan User Interface dan User Experience Pada Web Application Pencari Kerja dengan Metode Design Thinking. In *Universitas Pendidikan Indonesia*. Universitas Pendidikan Indonesia.

Gat, Hidayatullah, A., & Berliana, A. (2023). *Workshop Pengenalan Dasar Pemrograman Python Dengan Google Colaboratory*.

Hendriyana, H., Karo Karo, I. M., & Dewi, S. (2022). Analisis perbandingan Algoritma Support Vector Machine, Naive Bayes dan Regresi Logistik untuk Memprediksi Donor Darah. *Jurnal Teknologi Terpadu*, *8*(2).

Indra, W. P. (2024). *Penerapan Metode Sequential Pada Teknologi Artificial Intelligence Chatbot Sederhana Mengenai Depresi Berbasis Web*. Universitas Semarang.

Khoirunnisaa, N., Nabila, K., Kesuma, N., Setiawan, S., Yunizar, A., & Yusuf, P. (2024). Klasifikasi Teks Ulasan Aplikasi Netflix Pada Google Play Store Menggunakan Algoritma Naive Bayes dan SVM. *Sistem Komputer Dan Teknik Informatika (SKANIKA)*, *7*(1), 64–73.

Mesran, Savitri, P., Tundo, Sujarwadi, A., Mahardika, F., Sulistiyanto, Pasnur, Alfarisy, G. A. F., Leidiyana, H., Yuniarti, T., & Suseno, A. T. (2023). *Data Mining* (M. Syahrizal, Ed.; 1st ed.). CV. Graha Mitra Edukasi.

Mumbunan, K., Bawata, M. M. B., Kusen, M. P., Tarigan, V., & Yusupa, A. (2025). Perbandingan Algoritma Naive Bayes Dan Support Vector Machine Dalam Deteksi Berita Hoax Berbahasa Indonesia. *Riau Jurnal Teknik Informatika (RJTI)*, *4*(1), 52–64.

Naibaho, R. (2024). *Sindikat Scam Online Tipu 823 WNI, Modus Tawari Korban Kerja di Dubai*. https://news.detik.com/berita/d-7441517/sindikat-scam-online-tipu-823-wni-modus-tawari-korban-kerja-di-dubai

Nasien, D., Darwin, R., Cia, A., Winata, A. L., Go, J., M.C, R., Wijaya, R. C., & Lo, K. C. (2024). Perbandingan Implementasi Machine Learning Menggunakan Metode KNN, Naive Bayes, dan Logistik Regression Untuk Mengklasifikasi Penyakit Diabetes. *JEKIN - Jurnal Teknik Informatika*, *4*(1).

Nurul, H. (2022). Visual Studio Code: Pengertian, Fitur, Keunggulan dan Jenisnya. *Dewaweb.Com*. https://blog.dewaweb.com/mengenal-visual-studio-code/

Parameswari, S. D., Lubis, M., Suakanto, S., Ramadhan, Y. Z., Amanah, R. N., & Dila, R. A. (2025). Studi Perbandingan Naïve Bayes dan Support Vector

Machine (SVM) dalam Analisis Sentimen Pengguna Metaverse. *Jurnal Teknologi Dan Manajemen Industri Terapan*, *4*(3).

Paul, H., Wiguna, A. S., & Santoso, H. (2023). Penerapan Algoritma Support Vector Machine dan Naive Bayes Untuk Klasifikasi Jenis Mobil Terlaris Berdasarkan Produksi di Indonesia. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(1).

Priya, S., Reddy, V. D., & Balaji, V. (2022). Using AI to detect and classify malicious domain names. *International Journal of Health Sciences*, *6*(1), 9538–9547.

Rizaty, M. A. (2025). Daftar Negara yang Penduduknya Paling Lama Mengakses Medsos pada Kuartal III/2024. In *DataIndonesia.id*. https://dataindonesia.id/internet/detail/negara-dengan-ratarata-waktu-mengakses-internet-terlama-menggunakan-komputer-kuartal-iii2024

Salsabila, N. A. (2022). *Analisis Sentimen pada Media Sosial Twitter Terhadap Tokoh Gus Dur Menggunakan Metode Naive Bayes dan Support Vector Machine (SVM)*. Universitas Islam Negeri Syarif Hidayatullah.

Saputri, R. I., Khomsah, S., & Prasetyo, N. A. (2024). Perbandingan Metode Naive Bayes Classifier dan Support Vector Machine Untuk Klasifikasi Cyber Harassment Pada Twitter. *Jurnal Ilmu Komputer Dan Informatika (Algoritma)*, *8*(1), 10–21.

Sekhar, R., Solke, N., & Shah, P. (2023). Lean Manufacturing Soft Sensors for Automotive Industries. *Applied System Innovation*, *6*(22), 1–30.

Sulianta, F. (2024). *Basic Data Mining From A to Z* (F. Sulianta, Ed.). Universitas Widyatama.

Wahyuddin S, Rismayani, Sihotang, J. I., Aisa, S., Gunawan, H., Tamsir, N., Masturoh, S., Radiyah, U., Gustiana, Z., Harlina, S., & Muslihi, M. T. (2023). *Data Warehouse dan Data Mining* (J. Simarmata & R. Watrianthos, Eds.; 1st ed.). Yayasan Kita Menulis.

Wijoyo A, Saputra A, Ristanti S, Sya'ban S, Amalia M, & Febriansyah R. (2024). Pembelajaran Machine Learning. *OKTAL (Jurnal Ilmu Komputer Dan Science)*, *3*(2).

Yudiana, Y., Agustina, A. Y., & Khofifah, N. (2023). Prediksi Customer Churn Menggunakan Metode CRISP-DM Pada Industri Telekomunikasi Sebagai Implementasi Mempertahankan Pelanggan. *Indonesian Journal of Islamic Economics and Business (IJIEB)*, *8*(1), 1–20.

LAMPIRAN

Adapun sintaks lengkap hasil penelitian ini dapat dilihat pada lampiran berikut:

Lampiran 1. Sintaks Pengumpulan Data dan Seleksi Atribut

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('fake_job_postings.csv')

print("--- Hasil Pengumpulan Data ---")
print(f"Total baris data: {len(df)}")
print(f"Jumlah atribut awal: {len(df.columns)}")

distribusi = df['fraudulent'].value_counts()
print(f"Lowongan Asli: {distribusi[0]}")
print(f"Lowongan Palsu: {distribusi[1]}")

df.drop(columns=['job_id'], inplace=True)

kolom_teks = ['title', 'company_profile', 'description',
'requirements']
for col in kolom_teks:
    df[col] = df[col].fillna('')

df['combined_text'] = (
    df['title'] + " " +
    df['company_profile'] + " " +
    df['description'] + " " +
    df['requirements']
)

atribut_pilihan = ['combined_text', 'has_company_logo',
'has_questions', 'fraudulent']
df_final = df[atribut_pilihan]

df_final.to_csv('data_seleksi.csv', index=False)
```

Lampiran 2. Sintaks Visualisasi Distribusi Kelas Lowongan Kerja

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8, 5))
```

```
sns.set_theme(style="whitegrid")

custom_palette = {0: '#1792e4', 1: '#ffc226'}

ax = sns.countplot(
    x='fraudulent',
    data=df,
    hue='fraudulent',
    palette=custom_palette,
    legend=False
)

plt.title('Distribusi Kelas Lowongan Kerja (Asli vs Palsu)',
fontsize=12, pad=15)
plt.xticks([0, 1], ['Asli (0)', 'Palsu (1)'])
plt.xlabel('Kategori')
plt.ylabel('Jumlah Sampel')

for p in ax.patches:
    ax.annotate(
        f'{int(p.get_height())}',
        (p.get_x() + p.get_width() / 2., p.get_height()),
        ha='center', va='bottom',
        fontsize=10, fontweight='bold'
    )

plt.tight_layout()
plt.show()
```

Lampiran 3. Sintaks Tahapan Pre-Processing Data

```
import pandas as pd
import re
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

nltk.download('punkt')
nltk.download('punkt_tab')
nltk.download('stopwords')

df = pd.read_csv('data_seleksi.csv')

total_kata_awal = df['combined_text'].apply(lambda x:
len(str(x).split())).sum()
```

```
# Cleaning
df['combined_text'] = df['combined_text'].apply(lambda x:
    re.sub(r'[^a-zA-Z\s]', ' ',
    re.sub(r'\d+', '',
    re.sub(r'\S+@\S+', '',
    re.sub(r'http\S+|www\S+|https\S+', '', str(x))))))
)
df['combined_text'] = df['combined_text'].apply(lambda x: "
".join(x.split()))

# Case folding
df['combined_text'] = df['combined_text'].str.lower()

# Tokenizing
df['combined_text'] = df['combined_text'].apply(word_tokenize)

# Stopword removal
stop_words = set(stopwords.words('english'))
df['combined_text'] = df['combined_text'].apply(
    lambda x: [w for w in x if w not in stop_words]
)

# Stemming
stemmer = PorterStemmer()
df['combined_text'] = df['combined_text'].apply(
    lambda x: [stemmer.stem(w) for w in x]
)

df['combined_text'] = df['combined_text'].apply(lambda x: "
".join(x))

df.to_csv('preprocessing_data.csv', index=False)
```

Lampiran 4. Sintaks Ekstraksi Fitur Menggunakan TF-IDF

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.colors import LinearSegmentedColormap
from sklearn.feature_extraction.text import TfidfVectorizer

df = pd.read_csv('preprocessing_data.csv')
df['combined_text'] = df['combined_text'].fillna('')

tfidf_final = TfidfVectorizer(max_features=5000)
X_tfidf = tfidf_final.fit_transform(df['combined_text'])
```

```
words = tfidf_final.get_feature_names_out()
sums = X_tfidf.sum(axis=0)

data_tfidf = [(word, sums[0, idx]) for idx, word in
enumerate(words)]

ranking = pd.DataFrame(data_tfidf, columns=['Kata/Term', 'Total
Bobot TF-IDF'])
ranking = ranking.sort_values('Total Bobot TF-IDF',
ascending=False).head(10)

plt.figure(figsize=(10, 6))
tfidf_colors = sns.color_palette(
    LinearSegmentedColormap.from_list(
        "blue_orange", ["#1792e4", "#ffc226"]
    )(np.linspace(0, 1, 10))
)

sns.barplot(
    x='Total Bobot TF-IDF',
    y='Kata/Term',
    data=ranking,
    palette=tfidf_colors,
    hue='Kata/Term',
    legend=False
)

plt.title('10 Kata dengan Bobot TF-IDF Tertinggi', fontsize=14)
plt.xlabel('Total Bobot TF-IDF')
plt.ylabel('Kata / Term')
plt.tight_layout()
plt.show()
```

Lampiran 5. Sintaks Tahapan Pembagian Data Latih dan Data Uji

```
import pandas as pd
from sklearn.model_selection import KFold

df = pd.read_csv('preprocessing_data.csv')

X = df['combined_text']
y = df['fraudulent']

print(f"Jumlah Total Dataset: {len(df)} baris data.")

jml_latih = int(len(df) * 0.9)
```

```
jml_uji = len(df) - jml_latih

print(f"Data Latih (90%): {jml_latih} baris.")
print(f"Data Uji   (10%): {jml_uji} baris.\n")

kf = KFold(n_splits=10, shuffle=True, random_state=42)
```

Lampiran 6. Sintaks Implementasi Model

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import KFold
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score,
classification_report

df = pd.read_csv('preprocessing_data.csv')
X = df['combined_text']
y = df['fraudulent']

kf = KFold(n_splits=10, shuffle=True, random_state=42)

def evaluasi_model(model_obj, model_name):
    tfidf = TfidfVectorizer(max_features=5000)
    fold_acc = []
    y_true_total, y_pred_total = [], []

    for train_idx, test_idx in kf.split(X):
        X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
        y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

        xt = tfidf.fit_transform(X_train)
        xv = tfidf.transform(X_test)

        model_obj.fit(xt, y_train)
        y_pred = model_obj.predict(xv)

        fold_acc.append(accuracy_score(y_test, y_pred))
        y_true_total.extend(y_test)
        y_pred_total.extend(y_pred)

    print(f"\n=== HASIL EVALUASI: {model_name} ===")
    print(classification_report(y_true_total, y_pred_total))
```

```
    cm = confusion_matrix(y_true_total, y_pred_total)
    return {
        'Model': model_name,
        'Accuracy': np.mean(fold_acc),
        'Confusion Matrix': cm
    }

# MODEL NAIVE BAYES
hasil_nb = evaluasi_model(MultinomialNB(alpha=0.01), "Naive Bayes")

# MODEL SVM
hasil_svm = evaluasi_model(SVC(kernel='linear',
class_weight='balanced'), "SVM")
```

Lampiran 7. Sintaks Perbandingan Performa Model

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

results_df = pd.DataFrame([hasil_nb, hasil_svm])
print(results_df)

metrics = ['Accuracy', 'Precision', 'Recall', 'F1-Score']

nb_values = [hasil_nb.get(m, 0) for m in metrics]
svm_values = [hasil_svm.get(m, 0) for m in metrics]

plt.figure(figsize=(10, 6))
plt.bar(np.arange(len(metrics)) - 0.2, nb_values, 0.4, label='Naive
Bayes')
plt.bar(np.arange(len(metrics)) + 0.2, svm_values, 0.4, label='SVM')

plt.legend()
plt.title('Perbandingan Performa Naive Bayes dan SVM')
plt.show()
```