ANALISIS PERBANDINGAN KINERJA ALGORITMA NAIVE BAYES DAN K-NEAREST NEIGHBOR PADA KLASIFIKASI SEKUENS BAKTERI 16S RRNA BERBASIS VARIASI PANJANG FITUR K-MER 4, 5, DAN 6

PROPOSAL SKRIPSI

PANDRA INSANI PUTRA AZWAR
121450137

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
2025

ANALISIS PERBANDINGAN KINERJA ALGORITMA NAIVE BAYES DAN K-NEAREST NEIGHBOR PADA KLASIFIKASI SEKUENS BAKTERI 16S RRNA BERBASIS VARIASI PANJANG FITUR K-MER 4, 5, DAN 6

PROPOSAL SKRIPSI

Diajukan sebagai syarat maju seminar proposal

Pandra Insani Putra Azwar
121450137

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
2025

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Identifikasi dan pemetaan keragaman mikroorganisme menjadi tantangan mendasar dalam dunia medis dan lingkungan, mengingat peran vital bakteri yang memiliki dua sisi yaitu sebagai penyebab penyakit infeksi sekaligus penyeimbang ekosistem. Oleh karena itu, ketepatan dalam mengklasifikasikan jenis bakteri menjadi syarat mutlak bagi keselamatan medis yang mana kegagalan dalam mengenali bakteri berbahaya secara akurat tidak hanya menghambat diagnosis dokter, tetapi juga memicu risiko fatal lainnya seperti percepatan laju resistensi antibiotik (*Antimicrobial Resistance*) serta kegagalan deteksi dini wabah penyakit menular yang kini menjadi ancaman kesehatan global [1], [2]. Namun, tantangan utama muncul dari kompleksitas biologis bakteri itu sendiri di mana ribuan spesies sering kali memiliki tampilan fisik yang sangat mirip namun membawa profil genetik dan sifat penyakit yang jauh berbeda, sehingga metode identifikasi konvensional sering kali tidak memadai untuk membedakan kelompok bakteri yang memiliki hubungan kekerabatan dekat [3].

Pendekatan molekuler berbasis gen 16S rRNA telah menjadi standar emas global karena struktur uniknya yang memuat *conserved regions* sebagai situs pengikatan primer universal dan *hypervariable regions* (V1-V9) yang berfungsi layaknya sidik jari genetik untuk membedakan spesies bakteri satu dengan yang lainnya [4]. Di antara seluruh opsi variasi tersebut, literatur ilmiah terkini secara konsisten menyoroti *hypervariable regions* V3 dan V4 (V3-V4) sebagai segmen yang menawarkan keseimbangan paling optimal antara efisiensi biaya sekuensing dan resolusi taksonomi pada platform berkapasitas tinggi [5]. Data dari *regions* ini dinilai sangat memadai untuk klasifikasi hingga tingkat *Phylum*, yakni tingkatan taksonomi tinggi yang merepresentasikan garis keturunan evolusioner utama. Hal ini krusial mengingat basis data genomik global seperti *Genome Taxonomy Database* (GTDB) kini mencatat keberadaan ratusan *phylum* bakteri

berbeda yang menaungi lebih dari 715.000 genom, menjadikan klasifikasi pada tingkat ini sebagai langkah awal vital dalam memetakan keragaman hayati mikroba [6].

Upaya identifikasi dan pemetaan taksonomi dalam skala luas ini kini difasilitasi oleh kemajuan teknologi *Next Generation Sequencing* (NGS). Teknologi ini merupakan metode sekuensing paralel berkapasitas tinggi yang memungkinkan pembacaan jutaan fragmen DNA secara simultan dengan biaya yang jauh lebih efisien dibandingkan metode konvensional [7]. Namun, volume data masif yang dihasilkan oleh teknologi ini memunculkan tantangan komputasi baru, yakni tuntutan akan metode klasifikasi yang mampu menangani kompleksitas data tanpa bergantung pada proses penyelarasan sekuens atau *alignment* yang rumit dan memakan waktu [8]. Pendekatan *alignment-free* berbasis k-mer menjadi solusi yang tepat karena mekanisme ini meniadakan proses pencocokan posisi, melainkan berfokus pada analisis frekuensi pola sub-sekuens untuk merepresentasikan karakteristik genomik secara matematis [9].

Pendekatan *alignment-free* sangat bergantung pada panjang k sebagai parameter krusial, yang merepresentasikan jumlah nukleotida dalam satu unit segmen genetik. Pemilihan nilai k memunculkan dinamika komputasi yang mana jika nilai k yang kecil hanya mampu menangkap pola komposisi umum yang kurang spesifik, sedangkan nilai k yang besar dapat meningkatkan spesifisitas pengenalan spesies namun berisiko memicu masalah kelangkaan data (*sparsity*) akibat banyaknya kombinasi pola yang tidak muncul [10]. Meskipun beberapa penelitian mengeksplorasi rentang fitur yang luas $k = 3$ hingga 6, penelitian terdahulu secara spesifik menyoroti rentang $k = 5$ hingga $k = 6$ sebagai titik keseimbangan akurasi yang paling optimal [11]. Dari temuan tersebut, penelitian ini secara kritis menguji kembali kinerja pada batas bawah resolusi fitur $k = 4$. Tujuannya untuk mengevaluasi apakah dimensi fitur yang jauh lebih ringkas pada $k = 4$ mampu memberikan informasi diskriminatif yang memadai dengan keuntungan beban komputasi yang lebih rendah.

Pendekatan ekstraksi fitur berbasis frekuensi (*k*-mer *counting*) yang mengadopsi konsep *Bag of Words* (BoW) terbukti efektif dalam memetakan profil sekuens tanpa menghilangkan informasi komposisi [11]. Penelitian ini menerapkan strategi representasi fitur utuh untuk memproses seluruh dimensi fitur yang terbentuk. Langkah ini bertujuan mempertahankan integritas data, memastikan bahwa pola genetik spesifik yang mungkin hilang dalam proses reduksi tetap tersedia untuk membedakan taksonomi secara presisi.

Pemilihan algoritma klasifikasi menjadi aspek krusial, sebagaimana ditunjukkan oleh berbagai penelitian terdahulu yang mengevaluasi efektivitas model *Machine Learning* pada data mikrobioma [12]. Algoritma *K-Nearest Neighbor* (KNN) diketahui unggul dalam menangkap kemiripan lokal berbasis jarak [13], sedangkan *Multinomial Naive Bayes* (MNB) dinilai efektif memodelkan distribusi probabilitas pada fitur frekuensi diskrit [11]. Tantangan utama muncul akibat ketidakseimbangan kelas (*class imbalance*), yang terbukti sering menyebabkan bias prediksi dan kegagalan model dalam mendeteksi kelas minoritas [14]. Mengingat penelitian terdahulu mengenai ketahanan (*robustness*) kedua algoritma ini terhadap isu ketimpangan data khususnya pada fitur *k*-mer tingkat *phylum* masih terbatas, penelitian ini bertujuan untuk mengevaluasi kinerja prediktif kedua model tersebut secara komparatif guna mengisi celah literatur tersebut.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana pengaruh variasi panjang fitur *k*-mer 4, 5, dan 6 terhadap kinerja klasifikasi taksonomi bakteri 16S rRNA, dan manakah nilai *k* yang menghasilkan performa paling optimal?
2. Bagaimana perbandingan performa algoritma *Multinomial Naive Bayes* dan *K-Nearest Neighbor* dalam mengklasifikasikan data sekuens bakteri 16S rRNA untuk menentukan model yang paling optimal?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

1. Mengevaluasi pengaruh variasi panjang fitur *k*-mer 4, 5, dan 6 terhadap kinerja klasifikasi taksonomi bakteri 16S rRNA untuk menentukan nilai *k* yang menghasilkan performa paling optimal.
2. Membandingkan performa algoritma *Multinomial Naive Bayes* dan *K-Nearest Neighbor* dalam mengklasifikasikan data sekuens bakteri 16S rRNA untuk menentukan model yang paling optimal.

1.4 Batasan Masalah

Agar penelitian ini lebih terarah, penulis menetapkan batasan masalah sebagai berikut:

1. Data yang digunakan adalah sekuens gen 16S rRNA yang bersumber dari basis data SILVA, difokuskan pada *hypervariable regions* V3 dan V4 (V3-V4) dan terbatas hanya pada domain *Bacteria*.
2. Fokus analisis klasifikasi dibatasi pada tingkat taksonomi *Phylum*.
3. Algoritma klasifikasi yang dibandingkan adalah *Multinomial Naive Bayes* (MNB) dan *K-Nearest Neighbor* (KNN).
4. Ekstraksi fitur menggunakan pendekatan *alignment-free* berbasis *k*-mer *counting* dengan representasi frekuensi (*Bag of Words*), dibatasi pada variasi panjang $k = 4$, $k = 5$, dan $k = 6$.

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Landasan penelitian terkait metode *alignment-free* berbasis *machine learning* dirangkum secara ringkas dalam Tabel 2.1

Tabel 2.1 Penelitian Terdahulu

No.	Penulis (Tahun)	Judul Penelitian	Dataset	Hasil Penelitian
1	Juneja *et al.* (2022)	*An Approach to DNA Sequence Classification Through ML*	Sekuens DNA sekunder (*Chimpanzee, Dog, Human*).	MNB dengan $k = 6$ mencatat akurasi tertinggi (98.4%) dibanding variasi k lainnya.
2	Alshayeji *et al.* (2023)	*Viral genome prediction from raw human DNA sequence samples by combining NLP and ML techniques*	Metagenomik *Shotgun* (WGS) manusia dari 19 studi independen.	KNN dengan fitur *k-mer* unggul (akurasi 98.6%) mengalahkan SVM dan RF.
3	Wade *et al.* (2024)	*Investigating alignment-free machine learning methods for HIV-1 subtype classification*	20.110 sekuens genom HIV-1 (LANL).	Pada $k = 6$, KNN jauh lebih unggul (*Balanced Acc*: 0.77), sedangkan NB anjlok dengan (*Balanced Acc* 0.38) meski akurasi terlihat tinggi.

4	Wang & Liu (2020)	*Comparative study of classifiers for human microbiome data*	29 dataset 16S rRNA domain *Bacteria* dari mikrobioma manusia.	Membandingkan RF, XGBoost, SVM, dan Elastic Net. Hasil menunjukkan *Random Forest* (RF) paling stabil dan unggul di mayoritas dataset, sedangkan XGBoost menuntut komputasi tinggi untuk *tuning* tanpa jaminan performa lebih baik.

Berdasarkan Tabel 2.1, pendekatan *alignment-free* berbasis *k-mer* telah bagus sebagai standar klasifikasi sekuens yang efisien. Meskipun penelitian Juneja *et al.* dan Alshayeji *et al.* mendemonstrasikan bahwa algoritma klasik (MNB dan KNN) mampu mencapai akurasi di atas 98%, validitas metodologis pada penelitian terdahulu masih menyisakan celah. Wade *et al.* menyoroti bias metrik akurasi konvensional pada data tidak seimbang, sementara Wang dan Liu membuktikan bahwa model kompleks bertipe *ensemble* seringkali membebani komputasi tanpa jaminan superioritas performa. Penelitian ini mengusulkan strategi yang memprioritaskan keseimbangan antara efisiensi dan validitas, yaitu dengan menerapkan algoritma ringan (KNN dan MNB) yang diuji melalui skema *Hold-out Validation* independen. Proses evaluasi selanjutnya diperketat menggunakan metrik evaluasi komprehensif yang mencakup *Balanced Accuracy*, *Macro F1-Score*, serta *AUC-ROC* untuk menjamin objektivitas model pada dataset berskala masif.

2.2 Taksonomi Bakteri

Taksonomi bakteri merupakan disiplin ilmu klasifikasi yang menata organisme ke dalam sistem hierarkis berdasarkan kedekatan evolusioner dan karakteristik biologisnya [15]. Sistem klasifikasi ini

tersusun secara bertingkat mulai dari kategori terluas hingga unit biologis terkecil.

Secara umum, urutan hierarki taksonomi bakteri terdiri atas tujuh tingkatan utama. Tingkatan tertinggi dimulai dari Domain yang mengelompokkan organisme berdasarkan struktur sel dasar, diikuti oleh (*Phylum*), (*Class*), (*Order*), (*Family*), Genus, hingga Spesies sebagai unit klasifikasi paling spesifik. Semakin ke bawah tingkatan hierarkinya, semakin tinggi tingkat kemiripan genetik dan fenotipik antar organisme di dalamnya [16]. Ilustrasi lengkap mengenai susunan hierarki taksonomi bakteri dapat dilihat pada Gambar 2.1.

Gambar 2.1 Hierarki tingkatan taksonomi bakteri

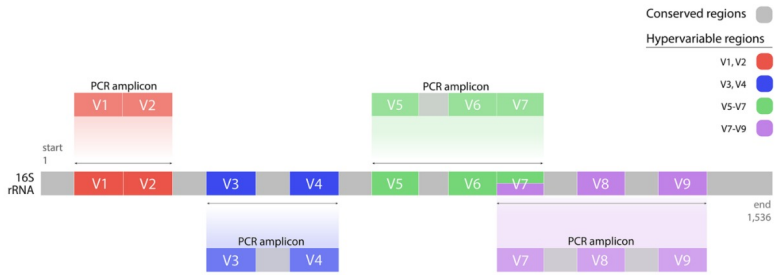
Penetapan setiap tingkatan taksonomi ini banyak mengandalkan pendekatan filogenomik, seperti rekonstruksi pohon berdasarkan gen penanda dan perhitungan *Average Nucleotide Identity* (ANI), untuk memastikan bahwa batasan antar taksa benar-benar mencerminkan hubungan evolusioner yang stabil [17]. Penerapan standar klasifikasi berbasis genom ini telah menyingkap keragaman bakteri yang jauh lebih luas dari perkiraan sebelumnya. Basis data terkini seperti *Genome Taxonomy Database* (GTDB) bahkan mencatat keberadaan ratusan *phylum* bakteri berbeda yang menaungi lebih dari 715.000

genom, menegaskan betapa masifnya keragaman hayati pada tingkatan ini [6].

Analisis komunitas mikroba pada tingkat *Phylum* memiliki makna evolusioner yang sangat penting karena tingkatan ini mewakili garis keturunan besar yang mencerminkan perbedaan strategi hidup, fungsi ekologis dominan, serta adaptasi metabolik yang luas. Analisis pada tingkat ini sering digunakan untuk mengidentifikasi pola umum perubahan komunitas, seperti pergeseran komposisi yang mengindikasikan *dysbiosis* pada kondisi klinis maupun respons adaptif terhadap tekanan lingkungan, sehingga memberikan dasar interpretasi biologis yang kuat sebelum analisis dilanjutkan ke tingkat taksonomi yang lebih spesifik [18].

2.3 Gen 16S rRNA

Gen 16S rRNA (*16S ribosomal RNA*) merupakan komponen utama penyusun ribosom pada sel bakteri yang berfungsi vital dalam sintesis protein. Dalam bioinformatika, gen ini dikenal sebagai identitas biologis atau *barcode* genetik bagi bakteri [4]. Hal ini dikarenakan keberadaannya yang universal pada seluruh bakteri, namun memiliki variasi sekuens yang unik antar spesies. Keunikan gen ini terletak pada kombinasi antara *conserved regions* yang bersifat stabil sebagai situs pengikatan primer universal, dan *hypervariable regions* (V1–V9) yang memiliki laju mutasi tinggi sebagai penanda spesifik untuk membedakan spesies [19]. *Hypervariable regions* inilah yang menjadi target analisis utama dalam identifikasi taksonomi. Di antara seluruh opsi *regions* tersebut, fragmen V3–V4 merupakan segmen yang paling sering ditargetkan karena terbukti memberikan resolusi taksonomi yang optimal hingga tingkat genus serta memiliki panjang sekuens yang kompatibel dengan platform sekuensing modern [20]. Struktur umum *regions* ini dapat dilihat pada Gambar 2.2.



Gambar 2.2 Struktur *hypervariable regions* (V1-V9) pada gen 16S rRNA [19]

Perolehan data sekuens 16S rRNA pada penelitian modern umumnya dilakukan melalui platform *Next Generation Sequencing* (NGS), yang mampu menghasilkan *output* jutaan *reads* dalam satu kali proses. Teknologi ini memungkinkan karakterisasi komunitas mikroba secara lebih komprehensif, baik untuk penelitian ekologi mikroba, diagnosis klinis, maupun pemantauan lingkungan, jauh melampaui kemampuan metode yang lain [7].

Peningkatan drastis volume data yang dihasilkan oleh NGS juga memunculkan tantangan komputasional besar. Metode penyelarasan sekuens berbasis *alignment* tradisional seperti BLAST, meskipun akurat, menjadi tidak lagi efisien ketika diterapkan pada dataset yang mencapai jutaan *reads* karena memerlukan waktu proses dan sumber daya komputasi yang sangat tinggi [21]. Oleh sebab itu, pendekatan *alignment-free* berbasis ekstraksi fitur seperti representasi *k-mer* menjadi alternatif yang semakin diadopsi karena kemampuannya mengolah data dalam skala besar secara lebih cepat dan efisien [10].

2.4 Kode Ambiguitas IUPAC dalam Representasi Sekuens

Informasi genetik DNA tersusun atas empat basa nitrogen yaitu *Adenine* (A), *Cytosine* (C), *Guanine* (G), dan *Thymine* (T), yang dalam format digital direpresentasikan sebagai untaian karakter (*string*). Namun, keterbatasan teknis pada proses sekuensing sering kali menghasilkan ambiguitas pembacaan pada posisi tertentu. Untuk mengatasi hal ini, *International Union of Pure and Applied Chemistry*

(IUPAC) menetapkan standar kode ambiguitas yang memungkinkan satu karakter mewakili beberapa kemungkinan basa kanonikal [22].

Analisis *alignment-free* berbasis *k*-mer, karakter ambigu tersebut berpotensi menjadi *noise* yang mendistorsi perhitungan frekuensi fitur dan menurunkan performa klasifikasi. Oleh karena itu, penanganan sekuens melalui transformasi stokastik pada tahap pra-pemrosesan menjadi langkah krusial. Untuk menjamin reprodisibilitas penelitian, proses acak tersebut harus dikontrol menggunakan parameter *random seed* [23]. Mekanisme ini memastikan hasil pemetaan basa tetap konsisten pada setiap pengulangan eksperimen, sehingga integritas perbandingan antar algoritma tetap terjaga. Daftar lengkap kode IUPAC disajikan pada Tabel 2.2.

Tabel 2.2 Kode Nomenklatur Asam Nukleat IUPAC

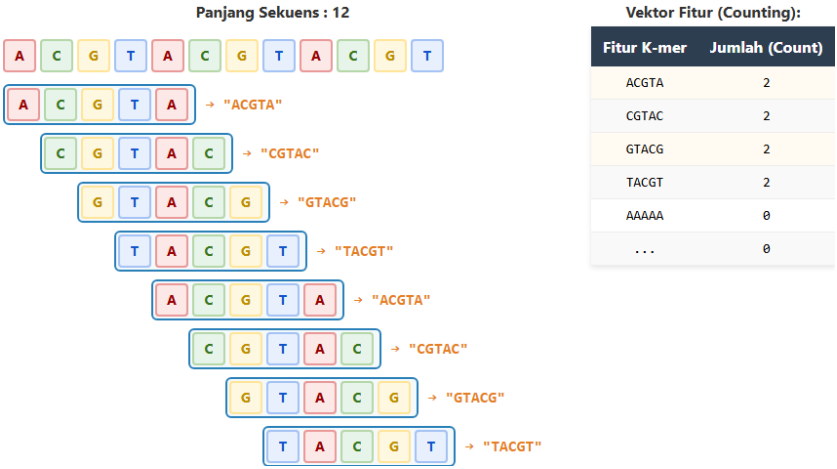
Simbol	***Mnemonic***	***Translation***
A	*Adenine*	A
C	*Cytosine*	C
G	*Guanine*	G
T	*Thymine*	T
R	*puRine*	A atau G
Y	*pYrimidine*	C atau T
M	*aMino group*	A atau C
K	*Keto group*	G atau T
S	*Strong interaction*	C atau G
W	*Weak interaction*	A atau T
H	*not G*	A, C atau T
B	*not A*	C, G atau T
V	*not T*	A, C atau G
D	*not C*	A, G atau T
N	*aNy*	A, C, G atau T

2.5 Ekstraksi Fitur *K*-mer

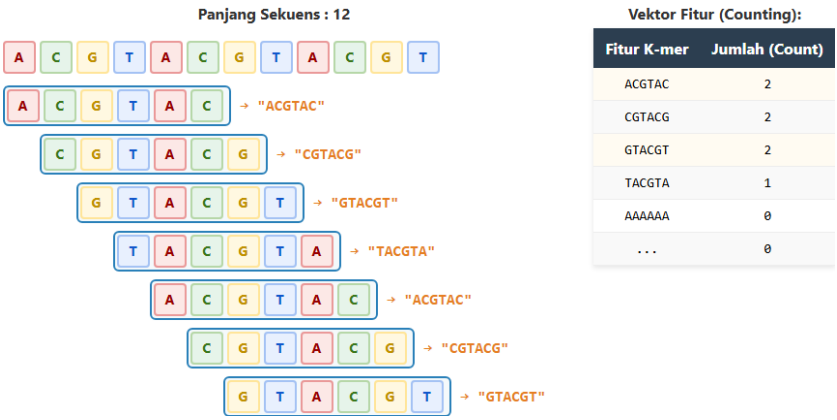
Ekstraksi fitur berbasis *k-mer* merupakan pendekatan *alignment-free* yang merepresentasikan sekuens biologis ke dalam bentuk vektor numerik berdasarkan frekuensi kemunculan *substring*. Berbeda dengan metode berbasis *alignment* yang bergantung pada referensi global dan membutuhkan sumber daya komputasi besar, pendekatan *k*-mer mentransformasi data sekuens menjadi matriks fitur. Transformasi ini memungkinkan algoritma *Machine Learning* untuk mempelajari pola komposisi nukleotida secara efisien, bahkan pada dataset genomik berskala besar [24].

Secara operasional, *k*-mer didefinisikan sebagai *substring* dengan panjang tetap *k* yang diekstraksi dari sekuens DNA menggunakan mekanisme jendela geser (*sliding window*). Jendela ini bergerak sepanjang sekuens satu nukleotida pada satu waktu, memecah sekuens panjang menjadi potongan-potongan pendek yang tumpang tindih (*overlapping*). Ilustrasi proses ekstraksi fitur dengan rentang K=4,5, dan 6 ditunjukkan pada Gambar 2.3, 2.4, dan 2.5.

Gambar 2.3 Ekstraksi Fitur K-mer K=4



Gambar 2.4 Ekstraksi Fitur K-mer K=5



Gambar 2.5 Ekstraksi Fitur K-mer K=6

Jumlah total fitur unik yang terbentuk dalam ruang vektor ditentukan oleh panjang k dengan rumus 4^k, mengingat terdapat empat kemungkinan basa nukleotida $\{A, C, G, T\}$ dalam sekuens DNA [25]. Pemilihan nilai k menjadi parameter krusial dalam klasifikasi sekuens yang mana jika nilai k yang terlalu kecil berisiko menghilangkan informasi urutan yang spesifik, sedangkan nilai k yang terlalu besar dapat menghasilkan matriks fitur yang sangat jarang (*sparse*) serta meningkatkan beban komputasi secara eksponensial [26]. Rentang nilai k antara 3 hingga 7 umumnya dianggap sebagai interval kritis dalam analisis genetik karena menawarkan keseimbangan optimal

antara resolusi informasi dan efisiensi komputasi [11], [14].

Pada batas bawah ($k = 3$), fitur yang terbentuk relatif ringkas yaitu 64 dimensi, yang mana cukup untuk menangkap pola komposisi dasar namun seringkali kurang spesifik untuk membedakan variasi evolusioner yang halus. Sebaliknya, peningkatan nilai k memperluas dimensi fitur secara drastis mulai dari 256 fitur pada $k = 4$, 1.024 fitur pada $k = 5$, hingga mencapai 16.384 fitur pada batas atas $k = 7$ [14]. Meskipun nilai k yang lebih tinggi meningkatkan kemampuan diskriminasi motif genetik, dimensi di atas $k = 7$ cenderung menghasilkan matriks yang sangat jarang (*sparse*) dan membebani sumber daya komputasi secara berlebihan. Oleh karena itu, representasi vektor dalam rentang ini, yang sering dikuantifikasi menggunakan metode frekuensi seperti *Count Vectorizer*, dinilai paling efektif untuk menangkap keragaman biologis pada gen penanda seperti 16S rRNA tanpa terjebak dalam kutukan dimensionalitas (*curse of dimensionality*) [13].

2.6 Stratified Random Sampling dan Pembagian Data

Stratified Random Sampling adalah teknik pengambilan sampel probabilitas dimana populasi heterogen dibagi ke dalam sub-kelompok homogen yang disebut strata, diikuti dengan pengambilan sampel secara independen dari setiap strata tersebut. Metode ini bertujuan untuk meningkatkan presisi estimasi statistik dan menjamin keterwakilan setiap sub-kelompok dalam sampel akhir, sehingga struktur populasi asli dapat dipertahankan secara akurat dalam subset data yang dihasilkan [27].

Pembagian Data *Hold-out* merupakan salah satu pendekatan evaluasi fundamental dalam pengujian model pembelajaran mesin, yang didefinisikan sebagai teknik membagi dataset tunggal menjadi himpunan data yang saling lepas [14]. Pada dataset dengan distribusi kelas yang tidak seimbang (*imbalanced*), penerapan *Hold-out* dengan acak sederhana (*Simple Random Sampling*) berisiko menghilangkan kelas minoritas. Oleh karena itu, integrasi teknik stratifikasi menjadi krusial untuk memastikan bahwa distribusi kelas tetap terjaga secara

proporsional di seluruh subset pembagian, baik pada himpunan data latih, data validasi, maupun data uji. Pendekatan validasi dengan tiga partisi independen ini secara luas diadopsi dalam literatur *Big Data* sebagai alternatif efisien untuk menggantikan metode *Cross-Validation*, yang menurut penelitian sebelumnya menuntut biaya komputasi yang terlalu tinggi pada dataset berskala besar tanpa jaminan peningkatan performa yang signifikan [28]. Perhitungan *stratified random sampling* dirumuskan dalam Persamaan 2.1.

$$n_h = \frac{N_h}{N} \times n \tag{2.1}$$

Keterangan:
n_h : Jumlah sampel yang dialokasikan untuk kelas h.
N_h : Total populasi data dalam kelas h.
N : Total populasi keseluruhan dataset.
n : Ukuran sampel target untuk pembagian yang dituju.

Selain aspek proporsionalitas, validitas evaluasi model juga sangat bergantung pada ukuran sampel dalam setiap strata. Penelitian sebelumnya menekankan bahwa setiap strata, khususnya pada himpunan data uji dan validasi, idealnya memiliki jumlah sampel yang melebihi ambang batas statistik tertentu ($n > 5 - 25$). Ketersediaan jumlah sampel yang memadai ini sangat penting untuk menjamin stabilitas metrik evaluasi komposit seperti *Macro F1-Score* dan *Balanced Accuracy*. Kekurangan sampel pada strata minoritas secara teoritis dapat menghasilkan interval kepercayaan (*confidence interval*) yang terlalu lebar, yang menyebabkan skor performa menjadi bias dan gagal merepresentasikan kemampuan generalisasi model yang sesungguhnya [29].

2.7 Algoritma Naive Bayes

Algoritma *Naive Bayes* merupakan metode klasifikasi probabilistik yang berlandaskan pada Teorema Bayes dengan asumsi independensi yang kuat (*naive*) antar fitur. Dalam literatur bioinformatika, algoritma ini menjadi salah satu pendekatan standar untuk analisis sekuens

biologis karena efisiensi komputasinya yang linear dan kemampuannya menangani dimensi fitur yang besar. Secara spesifik, varian *Multinomial Naive Bayes* (MNB) merupakan pengembangan model yang dirancang khusus untuk data berjenis cacahan (*count data*) atau frekuensi diskrit, menjadikannya model yang relevan untuk klasifikasi berbasis teks maupun representasi *k-mer* pada sekuens DNA [11].

Berbeda dengan varian *Gaussian Naive Bayes* yang mengasumsikan distribusi data kontinu, Multinomial Naive Bayes memodelkan distribusi probabilitas berdasarkan kelimpahan frekuensi kemunculan fitur. Karakteristik ini membuat Multinomial Naive Bayes sangat efektif dalam menangkap pola distribusi motif genetik antar taksonomi bakteri, dimana frekuensi kemunculan *k-mer* tertentu seringkali menjadi penanda biologis yang unik. Penelitian sebelumnya menunjukkan bahwa pendekatan probabilistik ini memiliki kemampuan generalisasi yang baik dan cenderung lebih tahan terhadap *overfitting* pada dataset berdimensi tinggi dibandingkan dengan algoritma yang memiliki kompleksitas model lebih tinggi [30].

Prinsip kerja MNB ini didasarkan pada perhitungan probabilitas posterior maksimum (*Maximum A Posteriori*). Prediksi kelas ditentukan dengan memilih label kelas yang memaksimalkan hasil kali antara probabilitas *prior* dan probabilitas *likelihood*, sebagaimana diformulasikan dalam Persamaan 2.2.

$$\hat{y} = \arg\max_{c \in C} \left[P(c) \prod_{i=1}^{n} P(w_i \mid c) \right] \tag{2.2}$$

Keterangan:

$\hat{y}$: Kelas prediksi dengan probabilitas posterior tertinggi.

$P(c)$: Probabilitas *prior* kemunculan kelas c dalam populasi.

$P(w_i \mid c)$: Probabilitas *likelihood* kondisional munculnya fitur *k-mer* w_i pada kelas c.

Dalam implementasi komputasi, operasi perkalian probabilitas

beruntun seringkali memicu masalah *arithmetic underflow*, dimana nilai hasil perkalian mendekati nol hingga melampaui batas presisi mesin. Untuk mengatasi kendala numerik tersebut, perhitungan secara matematis dikonversi ke dalam domain logaritma (*log-likelihood*). Transformasi ini mengubah operasi perkalian menjadi penjumlahan, yang menjamin stabilitas numerik tanpa mengubah urutan peringkat probabilitas antar kelas. Formulasi logaritma tersebut dinyatakan dalam Persamaan 2.3.

$$\hat{y} = \arg\max_{c \in C} \left[\log P(c) + \sum_{i=1}^{n} x_i \cdot \log P(w_i|c) \right] \tag{2.3}$$

Keterangan:

$\hat{y}$: Kelas prediksi hasil klasifikasi.

x_i : Frekuensi kemunculan fitur *k-mer* ke-i pada sampel observasi.

$P(w_i|c)$: Probabilitas kondisional kemunculan fitur w_i (k-mer ke-i) pada kelas c.

Tantangan mendasar dalam algoritma berbasis frekuensi adalah masalah probabilitas nol (*zero-frequency problem*), yaitu kondisi dimana sebuah fitur valid tidak ditemukan dalam data latih, sehingga menyebabkan nilai probabilitas total menjadi nol. Solusi standar dalam teori MNB adalah penerapan teknik penghalusan *Laplace Smoothing* (penambahan parameter α). Teknik ini memastikan bahwa setiap fitur memiliki peluang kemunculan yang tidak nol, sebagaimana didefinisikan dalam Persamaan 2.4.

$$\hat{P}(w_i|c) = \frac{N_{ic} + \alpha}{N_c + \alpha \cdot |V|} \tag{2.4}$$

Keterangan:

N_{ic} : Akumulasi frekuensi fitur w_i dalam seluruh sampel kelas c.

N_c : Total frekuensi seluruh fitur dalam kelas c.

α : Parameter penghalusan (*Laplace smoothing*).

$|V|$: Total fitur unik dalam kosakata.

2.8 Algoritma K-Nearest Neighbors

K-Nearest Neighbors (KNN) merupakan algoritma pembelajaran berbasis instansi (*instance-based learning*) atau sering dikenal sebagai *lazy learner*. Berbeda dengan algoritma parametrik yang membangun model internal eksplisit selama fase pelatihan, KNN menyimpan seluruh data latih dalam memori dan menunda proses generalisasi hingga tahap klasifikasi dilakukan. Prinsip kerja fundamental algoritma ini didasarkan pada asumsi homofili yang menyatakan bahwa data dengan karakteristik serupa cenderung menempati ruang fitur yang saling berdekatan dalam dimensi vektor [31].

Tahapan fundamental dalam algoritma ini diawali dengan pengukuran jarak antara vektor fitur data baru (*data uji*) terhadap seluruh vektor dalam basis data latih. Pada ruang fitur numerik berdimensi tinggi, metrik standar yang digunakan adalah *euclidean distance*. Penghitungan *euclidean distance* didefinisikan dalam Persamaan 2.5.

$$d(x_{uji}, x_i) = \sqrt{\sum_{j=1}^{n} (x_{uji,j} - x_{i,j})^2} \tag{2.5}$$

Keterangan:

$d(x_{uji}, x_i)$: Jarak Euclidean antara sampel uji dan sampel latih ke-i.
$x_{uji,j}$: Nilai fitur ke-j pada sampel uji.
$x_{i,j}$: Nilai fitur ke-j pada sampel latih ke-i.
n : Total dimensi fitur.

Setelah jarak terhadap seluruh data latih dihitung, algoritma mengidentifikasi sejumlah K tetangga terdekat. Dalam konteks klasifikasi genomik, skema pemungutan suara mayoritas sederhana (*uniform*) seringkali kurang optimal karena menganggap seluruh tetangga memiliki kontribusi setara tanpa memandang kedekatan jarak. Untuk mengatasi keterbatasan tersebut, diterapkan skema pembobotan jarak (*distance weighted voting*) yang memberikan penalti terhadap tetangga yang letaknya jauh [32]. Penghitungan *distance weighted voting* dinyatakan dalam Persamaan 2.6.

$$w_i = \frac{1}{d(x_{uji}, x_i)} \tag{2.6}$$

Keterangan:

w_i : Bobot yang diberikan pada sampel tetangga (neighbor) ke-i.

$d(x_{uji}, x_i)$: Jarak antara data uji dan data latih ke-i.

Bobot tersebut selanjutnya diakumulasikan untuk menentukan kelas prediksi final. Keputusan klasifikasi diambil berdasarkan kelas yang memiliki total bobot tertinggi di antara K tetangga terdekat, seperti diformulasikan pada Persamaan 2.7.

$$\hat{y} = \arg\max_{c \in C} \sum_{i=1}^{K} w_i \cdot I(y_i = c) \tag{2.7}$$

Keterangan:

$\hat{y}$: Kelas prediksi akhir dengan total bobot tertinggi.

C : Himpunan seluruh label kelas yang mungkin.

K : Jumlah tetangga terdekat yang ditinjau.

w_i : Bobot dari tetangga ke-i.

$I(\cdot)$: Fungsi indikator (bernilai 1 jika $y_i = c$, 0 jika tidak).

y_i : Label kelas sebenarnya dari tetangga ke-i.

Aspek krusial dalam optimalisasi algoritma KNN terletak pada penentuan nilai parameter K atau jumlah tetangga. Nilai ini mengendalikan keseimbangan antara bias dan varians model. Penelitian sebelumnya menunjukkan bahwa nilai K yang terlalu kecil membuat model sangat sensitif terhadap *noise* atau data pencilan sehingga menyebabkan *overfitting*. Sebaliknya, nilai K yang terlalu besar cenderung mengaburkan batas keputusan lokal antar kelas yang berpotensi menyebabkan *underfitting*. Oleh karena itu, strategi standar dalam penerapan KNN melibatkan pengujian eksperimental terhadap rentang nilai K yang bervariasi untuk menemukan konfigurasi yang memberikan performa generalisasi terbaik pada data validasi [14].

2.9 Optimasi *Hyperparameter* dan *Grid Search*

Hyperparameter adalah variabel kontrol yang mengatur struktur dan perilaku algoritma selama proses pembelajaran. Berbeda dengan parameter internal model yang dipelajari secara otomatis dari data, *hyperparameter* bersifat eksternal dan harus diinisialisasi sebelum pelatihan dimulai [33]. Konfigurasi nilai ini memegang peranan vital dalam menentukan performa model, karena pengaturan yang tidak tepat dapat menyebabkan kegagalan generalisasi, baik dalam bentuk *overfitting* maupun *underfitting* [34].

Tantangan dalam penentuan konfigurasi nilai tersebut dapat diatasi melalui pendekatan deterministik, salah satunya dengan menerapkan algoritma *Grid Search*. Metode ini bekerja dengan mengeksplorasi seluruh kombinasi nilai dalam ruang pencarian (*grid*) secara sistematis untuk menemukan konfigurasi optimal. Meskipun memiliki kompleksitas komputasi lebih tinggi dibanding *Random Search*, metode ini menawarkan jaminan matematis untuk menemukan parameter terbaik dalam batas yang ditentukan. Algoritma ini umumnya diintegrasikan dengan skema validasi seperti *Hold-out validation* untuk menjamin objektivitas evaluasi kinerja sebelum model diterapkan pada data uji [11].

Kebutuhan optimasi ini menjadi sangat spesifik bergantung pada karakteristik algoritma yang digunakan. Pada algoritma berbasis jarak seperti *K-Nearest Neighbor* (KNN), fokus optimasi terletak pada penentuan jumlah tetangga (K). Secara teoritis, nilai K merepresentasikan keseimbangan *bias variance* yang mana jika nilai yang terlalu kecil membuat model sensitif terhadap *noise* lokal, sedangkan nilai yang terlalu besar dapat mengaburkan batas keputusan antar kelas yang berdekatan [14]. Di sisi lain, pada model probabilistik seperti *Multinomial Naive Bayes* (MNB), parameter kritis yang memerlukan penyetelan adalah faktor penghalusan *Laplace* (α). Parameter ini berfungsi untuk menangani masalah probabilitas nol pada fitur yang jarang muncul, sehingga stabilitas perhitungan probabilitas logaritmik tetap terjaga [11].

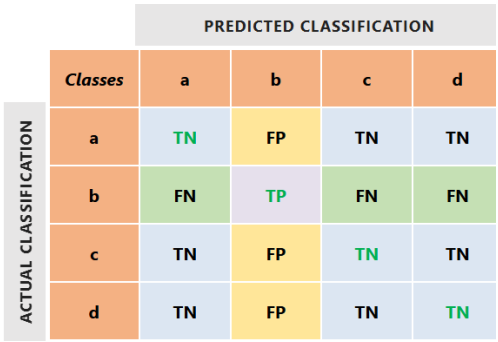
2.10 Evaluasi Model

Evaluasi model merupakan tahapan fundamental untuk memvalidasi kemampuan model klasifikasi dalam mengidentifikasi kelas yang benar. Dalam konteks analisis mikrobioma, data seringkali memiliki karakteristik distribusi kelas yang sangat tidak seimbang (*imbalanced dataset*). Kondisi ini menyebabkan penggunaan metrik akurasi standar menjadi kurang efektif, karena hasilnya cenderung bias terhadap kelas mayoritas dan gagal merepresentasikan kinerja model pada kelas minoritas yang seringkali justru menjadi fokus penelitian [35].

Evaluasi kinerja pada data tidak seimbang memerlukan pendekatan mendalam menggunakan metrik yang tidak sensitif terhadap distribusi kelas. Beberapa instrumen evaluasi utama yang secara teoritis terbukti valid untuk mengukur objektivitas performa meliputi *Confusion Matrix*, *Balanced Accuracy*, *Macro F1-Score*, dan analisis AUC-ROC [14]. Berikut adalah penjabaran teoritis dari masing-masing instrumen tersebut:

1. ***Confusion Matrix***

   Confusion Matrix merupakan tabel representasi visual yang menjadi landasan fundamental dalam evaluasi kinerja algoritma klasifikasi. Secara teknis, matriks ini menyajikan perbandingan langsung antara label kelas yang diprediksi oleh model terhadap label kelas aktual dari data uji. Tujuan utama penerapan matriks ini adalah untuk memberikan gambaran menyeluruh mengenai perilaku model, sehingga dapat mengidentifikasi secara spesifik apakah terdapat kecenderungan bias pada kelas tertentu atau kesulitan model dalam membedakan antar kelas yang memiliki kemiripan karakteristik fitur [35]. Ilustrasi *confusion matrix* disajikan pada Gambar 2.6.



Gambar 2.6 *Confusion Matrix Multikelas*

Confusion matrix ini terbentuk dari empat kemungkinan hasil prediksi yang menjadi dasar perhitungan metrik evaluasi. Definisi ringkas dari keempat komponen tersebut adalah:

a. *True Positive* (TP): Sampel kelas positif yang benar diprediksi sebagai positif.
b. *True Negative* (TN): Sampel kelas negatif yang benar diprediksi sebagai negatif.
c. *False Positive* (FP): Sampel kelas negatif yang salah diprediksi sebagai positif.
d. *False Negative* (FN): Sampel kelas positif yang salah diprediksi sebagai negatif.

2. ***Balanced Accuracy***

Balanced Accuracy merupakan rata-rata aritmatika dari nilai sensitivitas (*Recall*) pada setiap kelas, yang berfungsi untuk menyeimbangkan kontribusi performa antar kategori dengan memberikan bobot yang setara. Metrik ini digunakan secara spesifik dalam penanganan data tidak seimbang untuk mencegah bias penilaian yang seringkali didominasi oleh kelas mayoritas, sehingga memastikan bahwa kemampuan model dalam mendeteksi kelas minoritas tetap terukur secara adil dalam skor akhir [35]. Persamaan matematisnya dinyatakan dalam Persamaan 2.8.

$$\text{Balanced Accuracy} = \frac{1}{K}\sum_{k=1}^{K}\frac{TP_k}{TP_k + FN_k} \tag{2.8}$$

Keterangan:

K : Jumlah total kelas dalam dataset.

TP_k : Jumlah sampel kelas k yang benar diprediksi sesuai kelasnya.

FN_k : Jumlah sampel kelas k yang salah diprediksi sebagai kelas lain.

3. ***F1-Score***

F1-Score adalah rata-rata harmonik antara Presisi dan Recall. Metrik ini berfungsi untuk mengukur kinerja model dengan menyeimbangkan *trade-off* antara tingkat kepercayaan prediksi (Presisi) dan kelengkapan informasi yang terdeteksi (Recall). Penggunaan F1-Score sangat krusial dalam skenario dimana kesalahan positif palsu dan negatif palsu dianggap memiliki bobot konsekuensi yang sama pentingnya [35]. Perhitungan matematis F1-Score untuk setiap kelas spesifik k dihitung menggunakan Persamaan 2.9.

$$\text{F1-Score}_{(k)} = \frac{2 \times TP_k}{2 \times TP_k + FP_k + FN_k} \tag{2.9}$$

Keterangan:

TP_k : Jumlah sampel kelas k yang benar diprediksi sebagai kelas k.

FP_k : Jumlah sampel dari kelas lain yang salah diprediksi sebagai kelas k.

FN_k : Jumlah sampel kelas k yang salah diprediksi sebagai kelas lain.

Gambaran performa model secara global diperoleh dengan mengagregasikan nilai F1-Score seluruh kelas menggunakan metode *Macro-Average*. Pendekatan ini menghitung rata-rata aritmatika dari F1-Score seluruh kelas tanpa memberikan pembobotan berdasarkan frekuensi sampel. Pendekatan ini

bertujuan untuk menjamin bahwa kemampuan model dalam mendeteksi kelas minoritas memiliki kontribusi penilaian yang setara dengan kelas mayoritas, sehingga mencegah bias evaluasi. Formulasi akhir *Macro F1-Score* dinyatakan dalam Persamaan 2.10.

$$\text{F1-Score}_{\text{macro}} = \frac{1}{K} \sum_{k=1}^{K} \text{F1-Score}_{(k)} \tag{2.10}$$

Keterangan:

K : Jumlah total kelas dalam dataset.

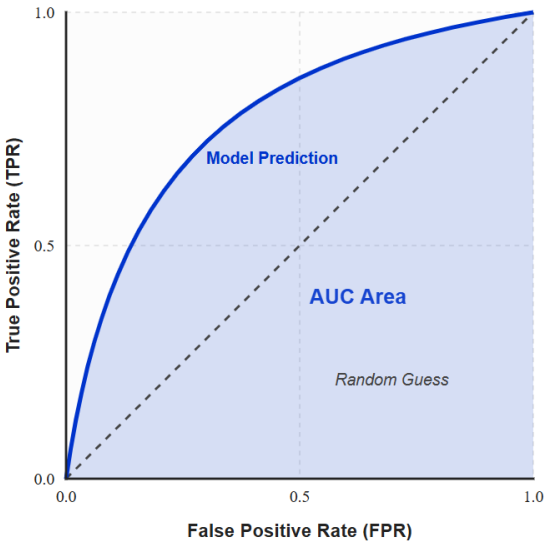
TP_k : Jumlah sampel kelas k yang benar diprediksi sesuai kelasnya.

FP_k : Jumlah sampel negatif yang salah diprediksi sebagai kelas k.

FN_k : Jumlah sampel kelas k yang salah diprediksi sebagai kelas lain.

4. **AUC-ROC**

AUC-ROC adalah metode evaluasi yang digunakan untuk mengukur seberapa baik model klasifikasi dapat membedakan antar kelas. Kurva ROC menggambarkan hubungan dinamis antara kemampuan model mendeteksi data positif secara benar (*True Positive Rate*) dibandingkan dengan laju kesalahannya (*False Positive Rate*). Sementara itu, nilai AUC (*Area Under the Curve*) memberikan skor numerik yang menunjukkan peluang model dalam memberikan nilai probabilitas lebih tinggi pada data positif dibanding data negatif. Tujuan utamanya adalah untuk memastikan bahwa evaluasi kinerja model tetap objektif dan valid pada berbagai kondisi ambang batas (*threshold*), tanpa terpaku pada satu titik keputusan saja [36]. Ilustrasi Kurva AUC-ROC disajikan pada Gambar 2.7.



Gambar 2.7 Kurva ROC dan Area AUC

Evaluasi kinerja pada klasifikasi multikelas umumnya mengadopsi pendekatan *One-vs-Rest* (OvR), sebuah teknik dekomposisi yang mentransformasi masalah kompleks menjadi rangkaian evaluasi biner independen. Untuk mengestimasi nilai AUC secara numerik pada setiap kurva biner tersebut, digunakan metode aturan trapesium (*trapezoidal rule*). Metode ini mendefinisikan luas area total sebagai hasil penjumlahan dari segmen-segmen trapesium yang terbentuk di antara titik-titik koordinat ROC. Perhitungan matematisnya disajikan dalam Persamaan 2.11.

$$AUC_{(k)} = \sum_{i=1}^{n-1} (FPR_{i+1} - FPR_i) \times \frac{(TPR_{i+1} + TPR_i)}{2} \tag{2.11}$$

Keterangan:

n : Jumlah titik ambang batas (*threshold*) yang membentuk kurva.

FPR : Nilai *False Positive Rate*.

TPR : Nilai *True Positive Rate*.

Untuk mendapatkan gambaran performa model secara holistik, nilai AUC dari setiap kelas diagregasikan menggunakan metode *Macro-Average AUC* (MAUC). Metode ini bekerja dengan menghitung rata-rata aritmatika dari seluruh nilai AUC tanpa melakukan pembobotan terhadap frekuensi sampel. Pendekatan ini menjadi standar dalam evaluasi data biologis karena karakteristiknya yang menjamin kesetaraan kontribusi penilaian antara kelas minoritas dan kelas mayoritas dalam skor akhir [37]. Formulasi matematis MAUC dinyatakan dalam Persamaan 2.12.

$$\text{MAUC}_{\text{ovr}} = \frac{1}{K} \sum_{k=1}^{K} AUC_{(k)} \tag{2.12}$$

Keterangan:

K : Jumlah total kelas dalam dataset.

$AUC_{(k)}$: Nilai AUC untuk kelas spesifik k.

BAB III

METODE PENELITIAN

3.1 Deskripsi Data

Dataset yang digunakan dalam penelitian ini merupakan data sekunder sekuens gen 16S rRNA pada daerah *hypervariabel* V3–V4 yang bersumber dari basis data SILVA.Data ini difokuskan pada domain *Bacteria* yang mencakup total 198 kelas taksonomi berbeda pada tingkat *Phylum*. Data mentah awal diperoleh dalam dua berkas terpisah, yaitu berkas sekuens berformat *FASTA* yang memuat 398.508 baris dan berkas taksonomi berformat *TSV* yang memuat 436.679 baris.

Kedua berkas ini memiliki atribut kunci yang sama yaitu *feature_id*. Struktur rinci dari data mentah sekuens (*FASTA*) dan anotasi taksonomi (*TSV*) ditampilkan pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1 Data Sekuens.*fasta*

No	feature_id	Sequence
0	CP013078.2406498.2408039	TGGGGA...CAAACA
1	CP015924.1224168.1225721	TGGGGA...CAAACA
2	CP003278.287486.289015	TGGGGA...CAAACA
...	...	...
398506	Y10757.1.1502	TGGGGA...CAAACA
398507	Z76660.1.1493	TGGGGA...CAAACA

Tabel 3.2 Data Taksonomi.*tsv*

No	feature_id	Taxon
0	CP013078.2406498.2408039	d__Bacteria; p__Proteobacteria; ... s__Bordetella_pertussis
1	CP015924.1224168.1225721	d__Bacteria; p__Proteobacteria; ... s__Salmonella_enterica
2	CP003278.287486.289015	d__Bacteria; p__Proteobacteria; ... s__Salmonella_enterica
...	...	...
436677	Y10757.1.1502	d__Bacteria; p__Proteobacteria; ... s__uncultured_bacterium
436678	Z76660.1.1493	d__Bacteria; p__Proteobacteria; ... s__uncultured_bacterium

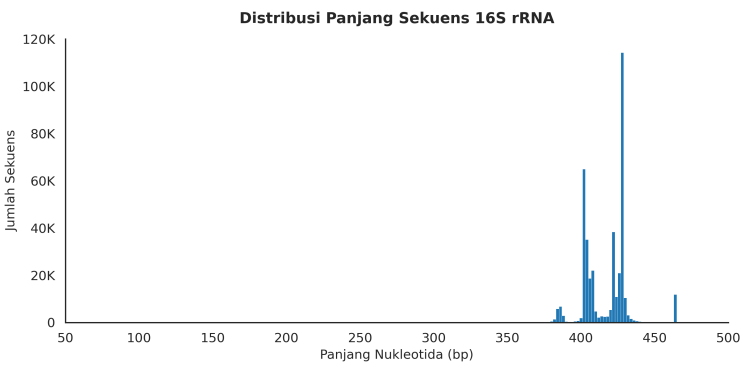
Kedua berkas data tersebut diintegrasikan menggunakan metode *Inner Join* berdasarkan atribut kunci *feature_id*. Kolom taksonomi

selanjutnya dipurai (*parsed*) menjadi tingkatan *Domain* hingga *Species*, serta dilakukan perhitungan panjang sekuens untuk menghasilkan atribut *Seqlength*. Dengan *Phylum* ditetapkan sebagai label target, proses ini menghasilkan total 398.508 data terstruktur sebagaimana ditampilkan pada Tabel 3.3.

Tabel 3.3 Dataset Hasil Integrasi

feature_id	Sequence	Seqlength	Domain	Phylum	Species
CP013078...	TGG...ACA	427	Bacteria	*Proteobacteria*	*Bordetella pertussis*
CP015924...	TGG...ACA	427	Bacteria	*Proteobacteria*	*Salmonella enterica*
CP003278...	TGG...ACA	427	Bacteria	*Proteobacteria*	*Salmonella enterica*
...	...	...	...	...	...
Y10757...	TGG...ACA	427	Bacteria	*Proteobacteria*	*Xanthomonas arboricola*
Z76660...	TGG...ACA	427	Bacteria	*Proteobacteria*	*Pseudomonas coronafaciens*

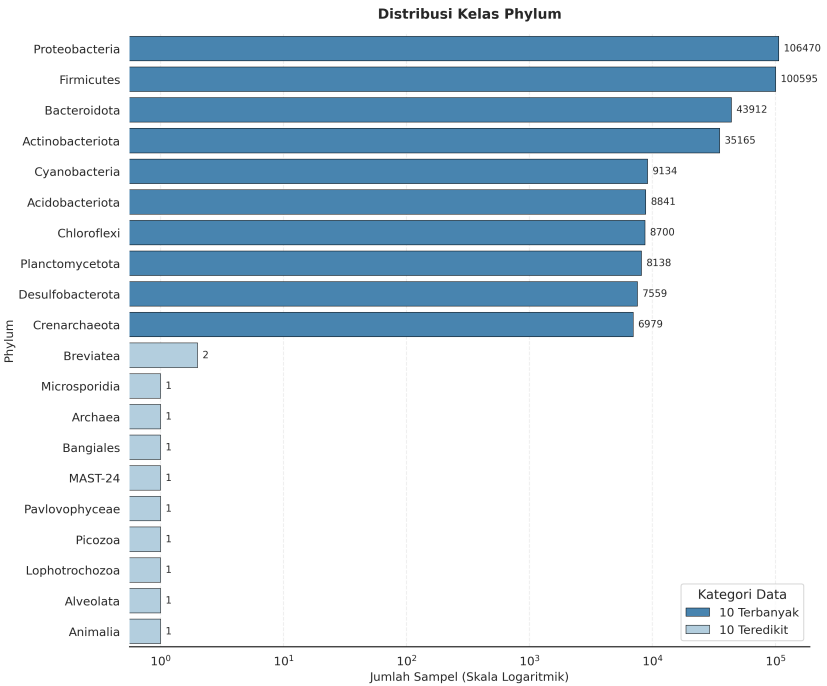
Karakteristik data sekuens menunjukkan bahwa sekuens memiliki rentang panjang antara 51 hingga 464 bp. Meskipun terdapat variasi tersebut, analisis lebih lanjut memperlihatkan bahwa mayoritas data terkonsentrasi secara konsisten pada rentang 400–460 bp, yang mengonfirmasi kesesuaian sekuens dengan target area V3–V4. Profil distribusi panjang sekuens ini divisualisasikan secara rinci pada Gambar 3.1.



Gambar 3.1 Distribusi Panjang Sekuens 16S rRNA (V3-V4)

Analisis lebih lanjut terhadap distribusi variabel target memperlihatkan bahwa proporsi data antar kelas sangat timpang (*imbalanced*). Kondisi ini ditandai dengan kesenjangan jumlah sampel yang ekstrem antara kelompok mayoritas dan minoritas. Visualisasi perbandingan distribusi

kelas tersebut ditampilkan pada Gambar 3.2.



Gambar 3.2 Distribusi Kelas Phylum 10 terbanyak dan 10 terdikit

3.2 Instrumen Penelitian

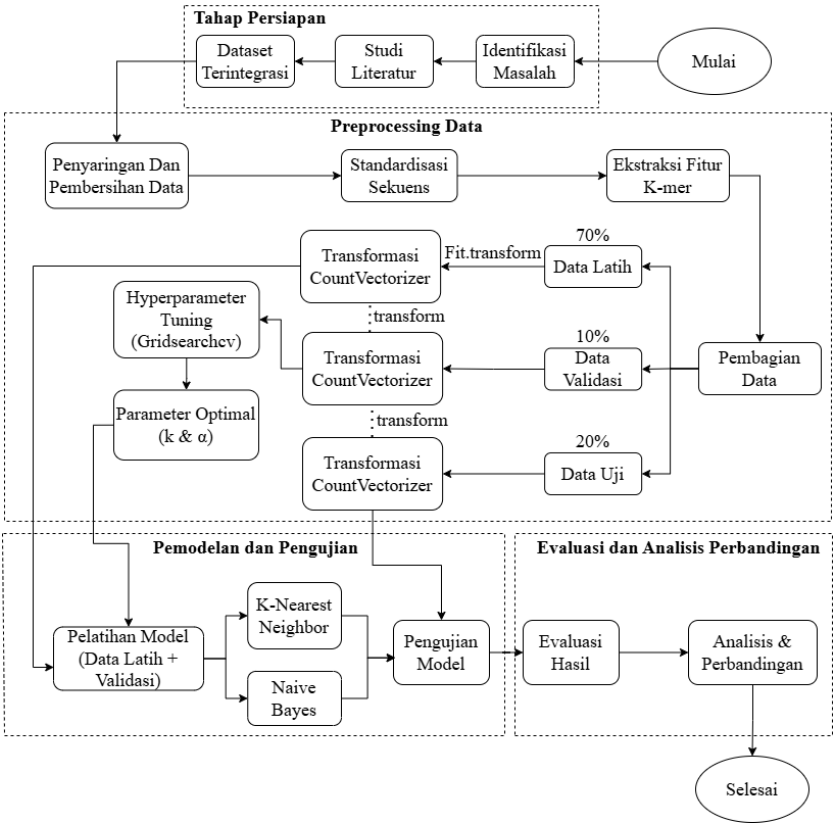
Pelaksanaan eksperimen komputasi dalam penelitian ini didukung oleh integrasi sumber daya perangkat keras (*hardware*) dan perangkat lunak (*software*) yang mumpuni. Mengingat tingginya beban komputasi pada tahap ekstraksi fitur *k-mer*, eksekusi kode utama dilakukan menggunakan lingkungan komputasi awan (*cloud computing*) pada platform Kaggle Notebooks. Platform ini digunakan karena menyediakan infrastruktur berkinerja tinggi yang didukung oleh prosesor 4-core vCPU serta kapasitas memori mencapai 30 GB, sebuah spesifikasi yang melampaui kapasitas komputasi konvensional. Sementara itu, tahap pengembangan kode awal, visualisasi hasil, serta penyusunan naskah dilakukan pada perangkat lokal laptop Asus Vivobook dengan spesifikasi prosesor Intel Core i3-1115G4, memori (RAM) 8 GB, dan sistem operasi Windows 64-bit.

Seluruh implementasi algoritma dibangun menggunakan bahasa pemrograman Python. Untuk mendukung tahapan pra-pemrosesan hingga evaluasi model pembelajaran mesin, penelitian ini memanfaatkan sejumlah pustaka (*library*) spesifik dengan rincian fungsi sebagai berikut:

(a) Python 3.11.13: Lingkungan eksekusi utama integrasi sistem.
(b) Pandas 2.2.3: Manipulasi struktur data tabular (*DataFrame*).
(c) Biopython 1.86: Pemrosesan sekuens biologis format FASTA.
(d) NumPy 1.26.4: Komputasi numerik dan operasi matriks vektor.
(e) Scikit-learn 1.2.2: Implementasi model klasifikasi dan evaluasi.
(f) Matplotlib 3.7.2: Visualisasi dasar distribusi data.
(g) Seaborn 0.12.2: Visualisasi statistik tingkat lanjut (*Heatmap*).

3.3 Alur Penelitian

Penelitian ini dilaksanakan melalui empat tahapan utama yang sistematis untuk mencapai tujuan klasifikasi taksonomi bakteri yang akurat. Rincian langkah-langkah penelitian mulai dari persiapan hingga evaluasi divisualisasikan pada diagram alir di Gambar 3.3.



Gambar 3.3 Diagram Alir Penelitian

1. **Tahap Persiapan**

   Tahap ini merupakan fondasi penelitian yang mencakup identifikasi masalah, studi literatur, serta akuisisi dataset sekunder gen 16S rRNA dari basis data SILVA. Fokus utama fase ini adalah perumusan strategi klasifikasi pendekatan *alignment-free* serta penetapan parameter awal untuk penelitian komparasi algoritma.

2. ***Preprocessing* Data**

   Tahapan ini bertujuan mentransformasi data mentah menjadi vektor numerik bebas *noise*. Proses diawali dengan penyaringan bertingkat meliputi isolasi taksonomi, normalisasi format, penghapusan duplikat (*deduplication*), dan eliminasi kelas minoritas ekstrem ($n < 25$). Selanjutnya, dilakukan standardisasi kode ambiguitas IUPAC melalui metode substitusi acak. Dataset

bersih kemudian dibagi menggunakan teknik *Stratified Random Sampling* (70:10:20). Tahap akhir adalah ekstraksi fitur *K-mer* menggunakan *CountVectorizer* ($k = 4 - 6$) dengan skema ketat yaitu dengan proses pembentukan kosa kata (*fitting*) hanya dilakukan pada data latih untuk mencegah kebocoran data (*data leakage*) ke data validasi dan uji.

3. **Perancangan dan Pelatihan Model**
   Pada tahap ini dibangun dua arsitektur model *Supervised Learning*, yaitu *K-Nearest Neighbors* (KNN) sebagai representasi model berbasis jarak dan *Multinomial Naive Bayes* (MNB) sebagai model probabilistik. Optimasi performa dilakukan melalui *Hyperparameter Tuning* dengan skema *Grid Search* pada data validasi. Fokus pencarian parameter terbaik meliputi jumlah tetangga (K) optimal untuk KNN dengan skema suara mayoritas *(majority voting)*, serta parameter pemulusan *Laplace* (α) untuk MNB.
4. **Evaluasi dan Analisis Perbandingan**
   Tahap akhir mengukur generalisasi model pada data uji independen menggunakan metrik komprehensif, yaitu *Balanced accuracy*, *Macro F1-Score*, dan *AUC-ROC*, serta analisis pola kesalahan melalui *Confusion Matrix*. Hasil evaluasi digunakan untuk menentukan algoritma yang paling efektif dan stabil dalam klasifikasi taksonomi bakteri berbasis fitur *k-mer*.

3.4 Penyaringan dan Pembersihan Data

Tahapan ini bertujuan untuk mentransformasi dataset mentah menjadi sekumpulan data berkualitas tinggi yang valid secara statistik. Proses ini dilakukan melalui mekanisme penyaringan bertingkat (*multi-stage filtering*) untuk menjamin homogenitas data dan mencegah terjadinya kebocoran data (*data leakage*) pada tahap evaluasi model.

Seleksi diawali dengan filter taksonomi untuk mengisolasi domain Bacteria dan mengeleminasi domain lain. Selanjutnya, dilakukan normalisasi format melalui kapitalisasi huruf dan penghapusan spasi sebagai tahap *preprocessing* sekuens. Langkah ini krusial untuk

menjamin keakuratan penghapusan data duplikat, di mana sistem mendeteksi kesamaan konten sekuens secara eksak. Meskipun memiliki ID berbeda, hanya satu sampel unik yang dipertahankan untuk menghilangkan redundansi, sehingga model dapat belajar dari variasi data yang benar-benar berbeda tanpa mengalami bias.

Setelah dataset bersih dari data duplikat, tahap akhir adalah penerapan penyaringan frekuensi kelas dengan menetapkan ambang batas minimal 25 sampel unik per kelas ($n \geq 25$). Batas ini merujuk pada penelitian sebelumnya yang menyatakan bahwa jumlah tersebut merupakan batas toleransi minimal untuk kategori *small sample size* yang masih memungkinkan dilakukan analisis kurva pembelajaran [29]. Kelas *Phylum* dengan jumlah sampel unik di bawah ambang batas ini dikeluarkan untuk menghindari bias model akibat representasi fitur yang tidak memadai.

3.5 Standardisasi Sekuens

Standardisasi sekuens dilakukan untuk menangani karakter non-kanonikal berdasarkan nomenklatur IUPAC yang berpotensi menjadi *noise* dalam ekstraksi fitur [22]. Karakter ambiguitas diselesaikan menggunakan metode substitusi acak (*random replacement*), di mana setiap simbol digantikan secara stokastik oleh basa kanonikal {A, C, G, T} sesuai distribusi peluang biologisnya. Sebagai contoh, simbol 'R' disubstitusi menjadi 'A' atau 'G' dengan probabilitas masing-masing 50%.

Penerapan parameter *random seed* dilakukan pada seluruh proses substitusi untuk menjamin konsistensi dan reprodisibilitas data. Mekanisme ini memastikan konsistensi hasil transformasi sekuens pada setiap iterasi eksperimen sehingga integritas komparasi antar-model tetap terjaga. Detail pemetaan substitusi kode IUPAC dalam penelitian ini disajikan pada Tabel 3.4.

Tabel 3.4 Pemetaan Substitusi Kode Ambiguitas IUPAC

Simbol	Mnemonic (Arti)	Opsi Substitusi Acak
R	*Purine*	A atau G
Y	*Pyrimidine*	C atau T
M	*Amino*	A atau C
K	*Keto*	G atau T
S	*Strong Interaction*	C atau G
W	*Weak Interaction*	A atau T
H	*Not G*	A, C, atau T
B	*Not A*	C, G, atau T
V	*Not T*	A, C, atau G
D	*Not C*	A, G, atau T
N	*Any*	A, C, G, atau T

3.6 Ekstraksi Fitur *k*-mer

Tahap ini mentransformasi sekuens DNA menjadi representasi tekstual melalui metode *k-mer*. Menggunakan mekanisme *sliding window* dengan pergeseran satu basa, setiap sekuens diurai menjadi potongan nukleotida tumpang tindih (*overlapping*). Penelitian ini menyusun variasi korpus dengan panjang $k = 4$ hingga $k = 6$, di mana proses vektorisasi sengaja ditunda hingga setelah pembagian data. Strategi ini diterapkan untuk mencegah kebocoran informasi (*data leakage*) dengan memastikan model hanya terekspos pada pola fitur dari data latih [38].

3.7 Pembagian Data

Penelitian ini menerapkan teknik *Stratified Random Sampling* untuk membagi dataset untuk menangani ketidakseimbangan distribusi kelas. Skema pembagian data menggunakan metode *Hold-Out Validation* dengan rasio 70:10:20, yang terdiri dari 70% data latih (*training set*), 10% data validasi (*validation set*) untuk optimasi *hyperparameter*, dan 20% data uji (*testing set*) independen [28]. Pendekatan pemisahan statis ini dipilih untuk memprioritaskan efisiensi komputasi serta objektivitas evaluasi model pada data sekuens biologis.

3.8 Transformasi *CountVectorizer*

Konversi fitur *k-mer* menjadi matriks fitur numerik dilakukan menggunakan algoritma *CountVectorizer*. Untuk menjaga validitas evaluasi dan mencegah kebocoran data (*data leakage*), proses pembentukan kosa kata (*fitting*) diterapkan secara eksklusif pada data latih. Referensi kosa kata tersebut selanjutnya digunakan untuk memetakan (*transform*) data validasi dan data uji secara konsisten, di mana token asing (*out-of-vocabulary*) diabaikan otomatis. Output akhir proses ini berupa matriks jarang (*sparse matrix*) yang merepresentasikan frekuensi kemunculan fitur sebagai input model klasifikasi.

3.9 Konfigurasi Ruang *Hyperparameter*

Penetapan konfigurasi optimal dilakukan melalui strategi *Grid Search* yang mengeksplorasi ruang pencarian parameter secara sistematis sebagaimana dirincikan pada Tabel 3.5.

Tabel 3.5 Ruang Pencarian *Hyperparameter*

Model	**Parameter**	**Rentang Nilai**
K-Nearest Neighbor	Jumlah Tetangga (K)	1, 3, 5, 7, 9
Multinomial NB	*Smoothing* (α)	0.01, 0.1, 0.5, 1.0, 2.0

Fokus optimasi pada algoritma KNN dibatasi pada penentuan jumlah tetangga (K) dengan metrik jarak tetap *Euclidean* dan pembobotan *invers distance* guna menyeimbangkan *bias-variance* [32]. Sementara itu, optimasi MNB dipusatkan pada parameter *Additive Smoothing* (α) untuk menangani isu *zero-frequency* pada fitur berdimensi tinggi [11]. Setelah nilai parameter terbaik diperoleh dari proses validasi, model final dilatih kembali (*retraining*) dengan menggabungkan himpunan data latih dan data validasi (total 80% data) guna memaksimalkan pola informasi yang dipelajari sebelum dilakukan evaluasi akhir pada data uji independen.

3.10 Algoritma Klasifikasi

Penelitian ini mengimplementasikan dua metode klasifikasi utama, yaitu *Multinomial Naive Bayes* (MNB) dan *Weighted K-Nearest Neighbor* (KNN). Pemilihan kedua metode ini didasarkan pada karakteristik data hasil ekstraksi fitur *k-mer* yang menghasilkan representasi vektor berdimensi tinggi dengan struktur matriks jarang (*sparse matrix*). Kedua algoritma ini dipilih karena terbukti memiliki efisiensi komputasional yang baik dalam menangani karakteristik data genomik tersebut.

Pendekatan pertama yang diterapkan adalah *Multinomial Naive Bayes* (MNB), yang berfungsi sebagai model probabilistik untuk memetakan distribusi frekuensi kemunculan *k-mer*. Algoritma ini dirancang khusus untuk menangani data diskrit. UNtuk menjaga stabilitas numerik akibat perkalian probabilitas yang sangat kecil, penelitian ini menerapkan transformasi *log-likelihood* serta teknik penghalusan *Laplace smoothing* untuk mengatasi masalah frekuensi nol (*zero-frequency*). Langkah-langkah komputasi algoritma MNB tersebut dirumuskan secara rinci dalam Algoritma 1.

Algoritma 1: Multinomial Naive Bayes dengan Log-Likelihood

Data : Data Latih D_{train}, Sampel Uji x_{uji}, Kosakata Fitur V, Parameter Smoothing α

Hasil: Kelas Prediksi $\hat{y}$

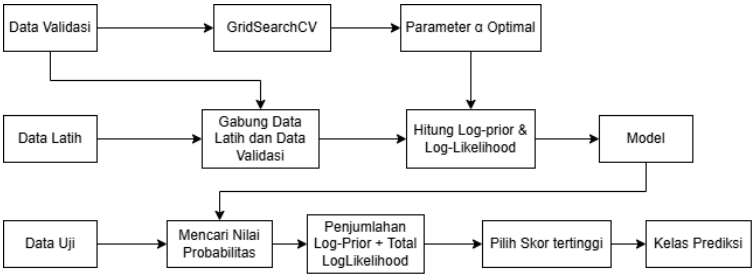
Fase Pelatihan (Training)

for *setiap kelas* $c \in C$ **do**
    $N_c \leftarrow$ jumlah sampel di kelas c;
    $N_{total} \leftarrow$ total sampel di D_{train};
    $log\_prior[c] \leftarrow \ln(N_c/N_{total})$;
    $N_{total,c} \leftarrow$ total frekuensi semua fitur dalam kelas c;
    for *setiap fitur* $t \in V$ **do**
        $N_{t,c} \leftarrow$ frekuensi fitur t di kelas c;
        $P_{val} \leftarrow (N_{t,c} + \alpha)/(N_{total,c} + \alpha \cdot |V|)$;
        $log\_like[t][c] \leftarrow \ln(P_{val})$;

Fase Pengujian (Testing)

$max\_score \leftarrow -\infty$;
for *setiap kelas* $c \in C$ **do**
    $score_c \leftarrow log\_prior[c]$;
    for *setiap fitur t yang muncul di* x_{uji} **do**
        if $t \in V$ **then**
            $freq\_t \leftarrow$ frekuensi fitur t di x_{uji};
            $score_c \leftarrow score_c + (freq\_t \times log\_like[t][c])$;
    if $score_c > max\_score$ **then**
        $max\_score \leftarrow score_c$;
        $\hat{y} \leftarrow c$;
return $\hat{y}$

Sistem MNB memisahkan proses pelatihan (pembentukan tabel probabilitas menggunakan parameter α optimal) dan proses pengujian secara eksplisit. Alur data dan integrasi parameter hasil optimasi pada model MNB diilustrasikan pada Gambar 3.4.



Gambar 3.4 Diagram Alur Proses Algoritma Multinomial Naive Bayes

Algoritma *Weighted K-Nearest Neighbor* (KNN) digunakan sebagai metode pembanding yang merepresentasikan pendekatan berbasis jarak (*distance-based*). Berbeda dengan MNB yang probabilistik, KNN mengidentifikasi kemiripan lokal antar sekuens DNA menggunakan metrik jarak Euclidean. Untuk meningkatkan akurasi pada data biologi yang kompleks, penelitian ini memodifikasi mekanisme *voting* standar dengan skema *inverse distance weighting*, di mana tetangga yang lebih dekat memiliki pengaruh yang lebih besar terhadap keputusan klasifikasi. Prosedur lengkap algoritma ini dijabarkan dalam Algoritma 2.

Algoritma 2: Weighted K-Nearest Neighbor (IDW)

Data : Data Latih D_{train}, Sampel Uji x_{uji}, Jumlah Tetangga K

Hasil: Kelas Prediksi $\hat{y}$

Hitung Jarak Euclidean

for *setiap sampel* $x_i \in D_{train}$ **do**
    $sum\_sq \leftarrow 0$;
    for *setiap dimensi fitur* j **do**
        $diff \leftarrow x_{uji}[j] - x_i[j]$;
        $sum\_sq \leftarrow sum\_sq + (diff)^2$;
    $d[i] \leftarrow \sqrt{sum\_sq}$;

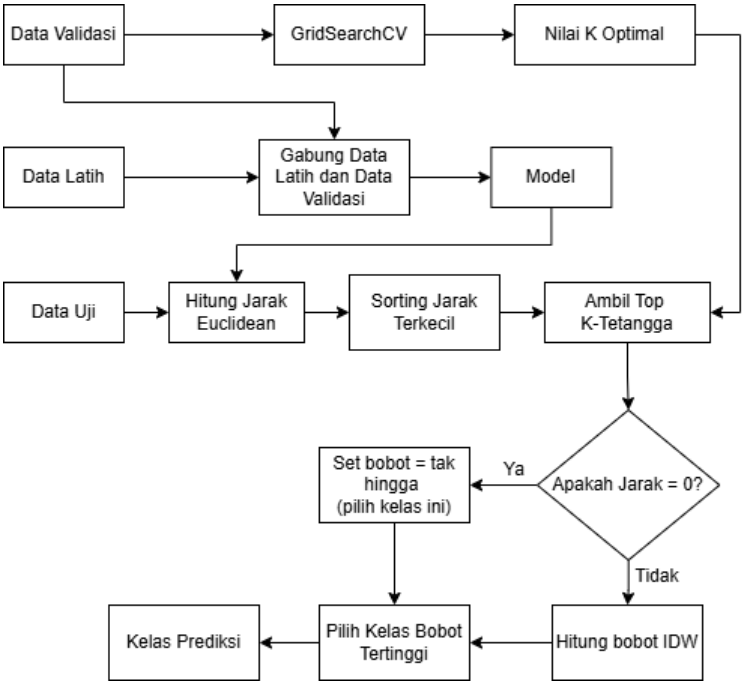
Seleksi Tetangga & Pembobotan IDW

Urutkan d dari terkecil ke terbesar;
Ambil K sampel teratas sebagai set $Neighbors$;
for *setiap* $idx \in Neighbors$ **do**
    $jarak \leftarrow d[idx]$;
    $label \leftarrow$ kelas dari sampel x_{idx};
    if $jarak = 0$ **then**
        return $label$
    else
        $bobot \leftarrow 1/jarak$;
        $TotalVotes[label] \leftarrow TotalVotes[label] + bobot$;

Keputusan Akhir

$\hat{y} \leftarrow$ kelas dengan nilai $TotalVotes$ tertinggi;
return $\hat{y}$

KNN menerapkan paradigma *lazy learning* yang Berbeda dengan MNB yang memiliki fase pelatihan intensif, di mana komputasi utama terjadi saat klasifikasi. Proses ini melibatkan pencarian parameter K optimal dan penggunaan data referensi gabungan, sebagaimana divisualisasikan secara runut pada Gambar 3.5.



Gambar 3.5 Diagram Alur Proses Algoritma Weighted K-Nearest Neighbor

3.11 Evaluasi Model dan Analisis Perbandingan

Evaluasi generalisasi model dilakukan pada himpunan data uji independen. Akibat ketimpangan distribusi kelas (*imbalanced*), penilaian kinerja memprioritaskan penggunaan indikator yang resisten terhadap bias mayoritas dibandingkan sekadar metrik Akurasi standar.

Instrumen evaluasi utama mencakup *Balanced Accuracy* dan *Macro F1-Score* yang merepresentasikan rata-rata harmonik antara presisi dan sensitivitas secara adil pada seluruh kelas. Selain itu, probabilitas diskriminasi model divalidasi menggunakan metrik *Area Under the ROC Curve* (AUC-ROC) dengan pendekatan *One-vs-Rest* (OvR).

Penelitian ini membandingkan performa KNN dan MNB sekaligus mengevaluasi pengaruh panjang fitur *k-mer* ($k = 4, 5, 6$). Pemetaan kesalahan klasifikasi antar *phylum* dilakukan melalui *Confusion Matrix* untuk memvalidasi kecenderungan error model secara spesifik.

DAFTAR PUSTAKA

[1] R. G. Elbaiomy dkk., "Antibiotic resistance: A genetic and physiological perspective", *MedComm*, vol. 6, no. 11, e70447, 2025.

[2] H. Mwanja dkk., "Epidemiology of bloodstream bacterial infections and missed opportunities to detect and respond to possible outbreaks due to drug-resistant bacteria in hospitals in uganda: A review of retrospective national amr surveillance data 2020 to 2023", *Wellcome Open Research*, vol. 10, hlmn. 665, 2025.

[3] J. Nunes Ramos, L. Veloso da Costa, V. Viana Vieira, dan M. L. Lima Brandão, "Challenges in the identification of environmental bacterial isolates from a pharmaceutical industry facility by 16s rrna gene sequences", *DNA*, vol. 5, no. 3, hlmn. 33, 2025.

[4] M.-Q. Yang, Z.-J. Wang, C.-B. Zhai, dan L.-Q. Chen, "Research progress on the application of 16s rrna gene sequencing and machine learning in forensic microbiome individual identification", *Frontiers in Microbiology*, vol. 15, hlmn. 1 360 457, 2024.

[5] K. Hrovat, B. E. Dutilh, M. H. Medema, dan C. Melkonian, "Taxonomic resolution of different 16s rrna variable regions varies strongly across plant-associated bacteria", *ISME communications*, vol. 4, no. 1, ycae034, 2024.

[6] D. H. Parks, P.-A. Chaumeil, A. J. Mussig, C. Rinke, M. Chuvochina, dan P. Hugenholtz, "Gtdb release 10: A complete and systematic taxonomy for 715 230 bacterial and 17 245 archaeal genomes", *Nucleic Acids Research*, gkaf1040, 2025.

[7] H. Satam dkk., "Next-generation sequencing technology: Current trends and advancements", *Biology*, vol. 12, no. 7, hlmn. 997, 2023.

[8] J. Chao, F. Tang, dan L. Xu, "Developments in algorithms for sequence alignment: A review", *Biomolecules*, vol. 12, no. 4, hlmn. 546, 2022.

[9] J. Chu dkk., "Mismatch-tolerant, alignment-free sequence classification using multiple spaced seeds and multiindex bloom filters", *Proceedings of the National Academy of Sciences*, vol. 117, no. 29, hlmn. 16 961–16 968, 2020.

[10] C. Moeckel dkk., "A survey of k-mer methods and applications in bioinformatics", *Computational and Structural Biotechnology Journal*, vol. 23, hlmn. 2289–2303, 2024.

[11] S. Juneja, A. Dhankhar, A. Juneja, dan S. Bali, "An approach to dna sequence classification through machine learning: Dna sequencing, k mer counting, thresholding, sequence analysis", *International Journal of Reliable and Quality E-Healthcare (IJRQEH)*, vol. 11, no. 2, hlmn. 1–15, 2022.

[12] X. Wang dan Y. Liu, *Comparative study of classifiers for human microbiome data. med microecol 4: 100013*, 2020.

[13] M. H. Alshayeji, S. C. Sindhu, dan S. Abed, "Viral genome prediction from raw human dna sequence samples by combining natural language processing and machine learning techniques", *Expert Systems with Applications*, vol. 218, hlmn. 119 641, 2023.

[14] K. E. Wade, L. Chen, C. Deng, G. Zhou, dan P. Hu, "Investigating alignment-free machine learning methods for hiv-1 subtype classification", *Bioinformatics Advances*, vol. 4, no. 1, vbae108, 2024.

[15] L. C. Ferraz Helene, M. S. Klepa, dan M. Hungria, "New insights into the taxonomy of bacteria in the genomic era and a case study with rhizobia", *International journal of microbiology*, vol. 2022, no. 1, hlmn. 4 623 713, 2022.

[16] D. H. Parks dkk., "A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life", *Nature biotechnology*, vol. 36, no. 10, hlmn. 996–1004, 2018.

[17] D. H. Parks, M. Chuvochina, C. Rinke, A. J. Mussig, P.-A. Chaumeil, dan P. Hugenholtz, "Gtdb: An ongoing census of bacterial and archaeal diversity through a

phylogenetically consistent, rank normalized and complete genome-based taxonomy", *Nucleic acids research*, vol. 50, no. D1, hlmn. D785–D794, 2022.

[18] B. W. Stone dkk., "Life history strategies among soil bacteria—dichotomy for few, continuum for many", *The ISME Journal*, vol. 17, no. 4, hlmn. 611–619, 2023.

[19] R. López-Aladid dkk., "Determining the most accurate 16s rrna hypervariable region for taxonomic identification from respiratory samples", *Scientific reports*, vol. 13, no. 1, hlmn. 3974, 2023.

[20] I. Abellan-Schneyder dkk., "Primer, pipelines, parameters: Issues in 16s rrna gene sequencing", *Msphere*, vol. 6, no. 1, hlmn. 10–1128, 2021.

[21] B. Saada, T. Zhang, E. Siga, J. Zhang, dan M. M. Magalhães Muniz, "Whole-genome alignment: Methods, challenges, and future directions", *Applied Sciences*, vol. 14, no. 11, hlmn. 4837, 2024.

[22] A. D. Johnson, "An extended iupac nomenclature code for polymorphic nucleic acids", *Bioinformatics*, vol. 26, no. 10, hlmn. 1386–1389, 2010.

[23] H. Ahmed dan J. Lofstead, "Managing randomness to enable reproducible machine learning", di dalam *Proceedings of the 5th International Workshop on practical reproducible evaluation of computer systems*, 2022, hlmn. 15–20.

[24] E. Rachtman, Y. Jiang, dan S. Mirarab, "Machine learning enables alignment-free distance calculation and phylogenetic placement using k-mer frequencies", *Molecular Ecology Resources*, vol. 25, no. 8, e70055, 2025.

[25] M. Karlicki, S. Antonowicz, dan A. Karnkowska, "Tiara: Deep learning-based classification system for eukaryotic sequences", *Bioinformatics*, vol. 38, no. 2, hlmn. 344–350, 2022.

[26] M. T. Swain dan M. Vickers, "Interpreting alignment-free sequence comparison: What makes a score a good score?", *NAR genomics and bioinformatics*, vol. 4, no. 3, lqac062, 2022.

[27] Y. Chen dkk., "Predicting the role of the human gut microbiome in constipation using machine-learning methods: A meta-analysis", *Microorganisms*, vol. 9, no. 10, hlmn. 2149, 2021.

[28] J. Min dkk., "High-throughput, multiplexed assay platform for integrated analysis of sepsis", 2024.

[29] C. Beleites, U. Neugebauer, T. Bocklitz, C. Krafft, dan J. Popp, "Sample size planning for classification models", *Analytica chimica acta*, vol. 760, hlmn. 25–33, 2013.

[30] M. Ziemski, T. Wisanwanichthan, N. A. Bokulich, dan B. D. Kaehler, "Beating naive bayes at taxonomic classification of 16s rrna gene sequences", *Frontiers in Microbiology*, vol. 12, hlmn. 644 487, 2021.

[31] S. Sangar, P. Kolage, dan P. Chunarkar-Patil, "Species annotation using a k-mer based knn model", *Bioinformation*, vol. 20, no. 9, hlmn. 986, 2024.

[32] S. Orozco-Arias dkk., "K-mer-based machine learning method to classify ltr-retrotransposons in plant genomes", *PeerJ*, vol. 9, e11456, 2021.

[33] W. Nugraha dan A. Sasongko, "Hyperparameter tuning on classification algorithm with grid search", *Sistemasi*, vol. 11, no. 2, hlmn. 391–401, 2022.

[34] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions", *SN computer science*, vol. 2, no. 3, hlmn. 160, 2021.

[35] M. Grandini, E. Bagli, dan G. Visani, "Metrics for multi-class classification: An overview", *arXiv preprint arXiv:2008.05756*, 2020.

[36] F. S. Nahm, "Receiver operating characteristic curve: Overview and practical use for clinicians", *Korean journal of anesthesiology*, vol. 75, no. 1, hlmn. 25–36, 2022.

[37] T. Fawcett, "An introduction to roc analysis", *Pattern recognition letters*, vol. 27, no. 8, hlmn. 861–874, 2006.

[38] L. Sasse dkk., "Overview of leakage scenarios in supervised machine learning", *Journal of Big Data*, vol. 12, no. 1, hlmn. 135, 2025.

LAMPIRAN

Lampiran A
Perhitungan Stratified Random Sampling

Pembagian Data

Diketahui:

- Total Populasi (N) = 1.000 data.
- Proporsi Kelas: Kelas A (N_A = 800), Kelas B (N_B = 150), Kelas C (N_C = 50).
- Target Data Uji (n) = 20%.

Perhitungan:

1. **Total Sampel Uji (n):**

$$n = 0,2 \times 1.000 = 200 \text{ data}$$

2. **Alokasi Per Strata ($n_h = \frac{N_h}{N} \times n$):**

- **Kelas A:** $\frac{800}{1000} \times 200 =$ **160**
- **Kelas B:** $\frac{150}{1000} \times 200 =$ **30**
- **Kelas C:** $\frac{50}{1000} \times 200 =$ **10**

Lampiran B
Ekstraksi Fitur K-mer

Pembentukan Vektor Fitur menggunakan K-mer

Diketahui:

- Sekuens Input (S): `A C G T A C G T A C G T`
- Parameter $k = 4$.

Perhitungan:

1. Daftar Fitur (Sliding Window)

- w_1: `ACGT`
- w_2: `CGTA`
- w_3: `GTAC`
- w_4: `TACG`
- w_5: `ACGT`
- w_6: `CGTA`
- w_7: `GTAC`
- w_8: `TACG`
- w_9: `ACGT`

2. Tabel Frekuensi (Count Vectorizer)

Indeks	Fitur (k-mer)	Window Asal (w_i)	Jumlah
0	`ACGT`	w_1, w_5, w_9	**3**
1	`CGTA`	w_2, w_6	**2**
2	`GTAC`	w_3, w_7	**2**
3	`TACG`	w_4, w_8	**2**

3. Hasil Vektor Akhir

$$V = [3, 2, 2, 2]$$

Lampiran C
Perhitungan Multinomial Naive Bayes

Soal 3: Simulasi Pelatihan dan Klasifikasi

I. FASE PELATIHAN (TRAINING PHASE)

Tujuan: Menghitung estimasi parameter probabilitas $\hat{P}(w_i|c)$.

A. Data Latih Simulasi

Diketahui distribusi frekuensi fitur pada 4 kelas (A, B, C, D) dengan total fitur unik $|V| = 4$ dan parameter *smoothing* $\alpha = 1$.

Tabel Frekuensi Data Latih (N_{ic})

Fitur (w_i)	**Kelas A**	**Kelas B**	**Kelas C**	**Kelas D**
Idx 0 (`ACGT`)	5	0	0	1
Idx 1 (`CGTA`)	2	0	0	1
Idx 2 (`GTAC`)	2	0	1	0
Idx 3 (`TACG`)	2	0	1	0
Total (N_c)	**11**	**0**	**2**	**2**

B. Perhitungan Bobot Log-Likelihood

Estimasi probabilitas fitur dengan *Laplace Smoothing*:

$$\hat{P}(w_i|c) = \frac{N_{ic} + \alpha}{N_c + \alpha \cdot |V|}$$

Konversi ke domain logaritma (*log-likelihood*) untuk menghindari *underflow*:

$$\log \hat{P}(w_i|c) = \ln\left(\frac{N_{ic} + 1}{N_c + 4}\right)$$

Contoh Perhitungan untuk Kelas A:

- **Fitur Idx 0** (w_0)**:**

$$\hat{P}(w_0|A) = \frac{5 + 1}{11 + 4} = \frac{6}{15} = 0.40 \quad \xrightarrow{\ln} \quad \mathbf{-0.916}$$

- **Fitur Idx 1 (w_1):**

$$\hat{P}(w_1|A) = \frac{2+1}{11+4} = \frac{3}{15} = 0.20 \quad \xrightarrow{\ln} \quad \mathbf{-1.609}$$

II. FASE PENGUJIAN (TESTING PHASE)

Tujuan: Klasifikasi data uji dengan vektor frekuensi $x = [3, 2, 2, 2]$ menggunakan fungsi keputusan.

A. Persiapan Parameter

- **Perhitungan Log-Prior** ($\log P(c)$)**:** Diketahui 4 kelas seimbang, maka:

$$P(c) = \frac{1}{\text{Jumlah Kelas}} = \frac{1}{4} = 0.25$$

$$\log P(c) = \ln(0.25) = \mathbf{-1.386}$$

- **Tabel Bobot Log-Likelihood (Hasil Fase I):**

Fitur (w_i)	**Kelas A**	**Kelas B**	**Kelas C**	**Kelas D**
Idx 0 ($x_0 = 3$)	**-0.916**	-1.386	-1.791	-1.098
Idx 1 ($x_1 = 2$)	-1.609	-1.386	-1.791	-1.098
Idx 2 ($x_2 = 2$)	-1.609	-1.386	-1.098	-1.791
Idx 3 ($x_3 = 2$)	-1.609	-1.386	-1.098	-1.791

B. Perhitungan Skor Posterior

Prediksi dilakukan dengan menjumlahkan *log-prior* dan hasil kali frekuensi input (x_i) dengan bobot *log-likelihood*:

$$\text{Score}_c = \log P(c) + \sum_{i=1}^{n} x_i \cdot \log P(w_i|c)$$

1. Kelas A

$$\text{Score}_A = -1.386 + [(3 \cdot -0.916) + (2 \cdot -1.609) + (2 \cdot -1.609) + (2 \cdot -1.609)]$$

$$\text{Score}_A = -1.386 + [-2.748 - 3.218 - 3.218 - 3.218]$$

$$\text{Score}_A = -1.386 + (-12.402) = \mathbf{-13.788}$$

2. Kelas B

$$\text{Score}_B = -1.386 + [(3 \cdot -1.386) + (2 \cdot -1.386) + (2 \cdot -1.386) + (2 \cdot -1.386)]$$

$$\text{Score}_B = -1.386 + [-4.158 - 2.772 - 2.772 - 2.772]$$

$$\text{Score}_B = -1.386 + (-12.474) = \mathbf{-13.860}$$

3. Kelas C

$$\text{Score}_C = -1.386 + [(3 \cdot -1.791) + (2 \cdot -1.791) + (2 \cdot -1.098) + (2 \cdot -1.098)]$$

$$\text{Score}_C = -1.386 + [-5.373 - 3.582 - 2.196 - 2.196]$$

$$\text{Score}_C = -1.386 + (-13.347) = \mathbf{-14.733}$$

4. Kelas D

$$\text{Score}_D = -1.386 + [(3 \cdot -1.098) + (2 \cdot -1.098) + (2 \cdot -1.791) + (2 \cdot -1.791)]$$

$$\text{Score}_D = -1.386 + [-3.294 - 2.196 - 3.582 - 3.582]$$

$$\text{Score}_D = -1.386 + (-12.654) = \mathbf{-14.040}$$

III. KEPUTUSAN AKHIR

$$\hat{y} = \underset{c \in \{A,B,C,D\}}{\arg\max} \{-13.788, -13.860, -14.733, -14.040\}$$

$$\hat{y} = \text{Kelas A}$$

Lampiran D
Perhitungan K-Nearest Neighbor

Klasifikasi Berbasis Jarak (Weighted KNN)

I. Rumus dan Metode

- **Euclidean Distance**

$$d(x_{uji}, x_i) = \sqrt{\sum_{j=1}^{n}(x_{uji,j} - x_{i,j})^2}$$

- **Distance Weighted Voting**

$$w_i = \frac{1}{d(x_{uji}, x_i)}$$

- **Keputusan Klasifikasi Final**

$$\hat{y} = \arg\max_{c \in C} \sum_{i=1}^{K} w_i \cdot I(y_i = c)$$

Dimana $I(y_i = c)$ bernilai 1 jika tetangga i adalah kelas c, dan 0 jika bukan.

II. Diketahui Data Simulasi

- **Vektor Data Uji** (x_{uji}): $[3, 2, 2, 2]$.
- **Parameter**: $K = 3$.
- **Himpunan Data Latih** (x_i)**:**
  - x_1 (ID: D_1, Kelas: A): $[3, 2, 2, 1]$
  - x_2 (ID: D_2, Kelas: B): $[1, 4, 1, 1]$
  - x_3 (ID: D_3, Kelas: C): $[3, 1, 2, 2]$
  - x_5 (ID: D_4, Kelas: A): $[4, 2, 2, 2]$

III. Perhitungan Jarak Euclidean (d)

1. Jarak ke x_1 (D_1)

$$d(x_{uji}, x_1) = \sqrt{(3-3)^2 + (2-2)^2 + (2-2)^2 + (2-1)^2}$$
$$= \sqrt{0+0+0+1} = \mathbf{1.00}$$

2. Jarak ke x_2 (D_2)

$$d(x_{uji}, x_2) = \sqrt{(3-1)^2 + (2-4)^2 + (2-1)^2 + (2-1)^2}$$
$$= \sqrt{4+4+1+1} = \sqrt{10} \approx \mathbf{3.16}$$

3. Jarak ke x_3 (D_3)

$$d(x_{uji}, x_3) = \sqrt{(3-3)^2 + (2-1)^2 + (2-2)^2 + (2-2)^2}$$
$$= \sqrt{0+1+0+0} = \mathbf{1.00}$$

4. Jarak ke x_4 (D_4)

$$d(x_{uji}, x_4) = \sqrt{(3-4)^2 + (2-2)^2 + (2-2)^2 + (2-2)^2}$$
$$= \sqrt{1+0+0+0} = \mathbf{1.00}$$

IV. Seleksi Tetangga dan Pembobotan (w_i)

Dipilih 3 tetangga terdekat (jarak terkecil). Bobot dihitung dengan $w_i = 1/d$.

Rank	**Data Latih (x_i)**	**Jarak (d)**	**Kelas (y_i)**	**Bobot (w_i)**
1	x_1	1.00	A	1.00
2	x_3	1.00	C	1.00
3	x_5	1.00	A	1.00

V. Keputusan Klasifikasi (Voting)

Rumus: $\text{TotalVote}_c = \sum w_i \cdot I(y_i = c)$

1. Total Bobot Kelas A

$$\text{Vote}_A = (w_1 \cdot 1) + (w_3 \cdot 0) + (w_4 \cdot 1)$$

$$\text{Vote}_A = (1.00 \cdot 1) + (1.00 \cdot 0) + (1.00 \cdot 1)$$

$$\text{Vote}_A = 1.00 + 0 + 1.00 = \mathbf{2.00}$$

2. Total Bobot Kelas C

$$\text{Vote}_C = (w_1 \cdot 0) + (w_3 \cdot 1) + (w_4 \cdot 0)$$

$$\text{Vote}_C = (1.00 \cdot 0) + (1.00 \cdot 1) + (1.00 \cdot 0)$$

$$\text{Vote}_C = 0 + 1.00 + 0 = \mathbf{1.00}$$

Hasil Akhir (arg max)**:**

$$\hat{y} = \arg\max\{\text{Vote}_A = 2.00, \text{Vote}_C = 1.00\}$$

$$\hat{y} = \text{Kelas A}$$

Lampiran E
Perhitungan Evaluasi Model

Evaluasi Berbasis Confusion Matrix

I. Data Simulasi (Confusion Matrix) Diketahui hasil prediksi model pada dataset multikelas dengan total sampel **N = 20**.

Aktual \Prediksi	Pred A	Pred B	Pred C	Pred D	Total ($Support$)
Kelas A	**4 (TP)**	1 (FN)	0	0	5
Kelas B	1 (FP)	3	1	0	5
Kelas C	0	0	4	1	5
Kelas D	0	1	0	4	5
Total Prediksi	5	5	5	5	**20**

II. Identifikasi Komponen Matriks (Evaluasi Kelas A)

Tabel Keterangan Komponen Kelas A

Komponen	Nilai	Penjelasan Sumber Data
TP (True Positive)	4	Data Aktual A yang tepat diprediksi A.
FN (False Negative)	1	Data Aktual A yang salah diprediksi ke kelas lain.
FP (False Positive)	1	Data dari kelas lain (Aktual B) yang salah masuk ke Prediksi A.
TN (True Negative)	14	Data selain A yang tepat diprediksi bukan A.

III. Perhitungan Balanced Accuracy

Rumus Persamaan

$$\text{Balanced Accuracy} = \frac{1}{K} \sum_{k=1}^{K} \frac{TP_k}{TP_k + FN_k}$$

Langkah Perhitungan per Kelas:

1. **Kelas A:** $\frac{4}{4+1} = \frac{4}{5} = 0.80$
2. **Kelas B:** $\frac{3}{3+2} = \frac{3}{5} = 0.60$
3. **Kelas C:** $\frac{4}{4+1} = \frac{4}{5} = 0.80$
4. **Kelas D:** $\frac{4}{4+1} = \frac{4}{5} = 0.80$

Hasil Akhir:

$$\text{Bal. Acc} = \frac{0.80 + 0.60 + 0.80 + 0.80}{4} = \frac{3.00}{4} = \mathbf{0.75}$$

IV. Perhitungan Macro F1-Score

Rumus Persamaan

$$\text{F1-Score}_{(k)} = \frac{2 \times TP_k}{2 \times TP_k + FP_k + FN_k}$$

$$\text{F1-Score}_{\text{macro}} = \frac{1}{K} \sum_{k=1}^{K} \text{F1-Score}_{(k)}$$

Langkah Perhitungan per Kelas:

1. **Kelas A** ($TP = 4, FP = 1, FN = 1$)**:**

$$F1_A = \frac{2(4)}{2(4) + 1 + 1} = \frac{8}{10} = \mathbf{0.80}$$

2. **Kelas B** ($TP = 3, FP = 2, FN = 2$)**:**

$$F1_B = \frac{2(3)}{2(3) + 2 + 2} = \frac{6}{10} = \mathbf{0.60}$$

3. **Kelas C** ($TP = 4, FP = 1, FN = 1$)**:**

$$F1_C = \frac{2(4)}{2(4) + 1 + 1} = \frac{8}{10} = \mathbf{0.80}$$

4. **Kelas D** ($TP = 4, FP = 1, FN = 1$)**:**

$$F1_D = \frac{2(4)}{2(4) + 1 + 1} = \frac{8}{10} = \mathbf{0.80}$$

Hasil Akhir (Macro-Average):

$$\text{Macro F1} = \frac{0.80 + 0.60 + 0.80 + 0.80}{4} = \frac{3.00}{4} = \mathbf{0.75}$$

Evaluasi Area Under Curve (AUC-ROC)

I. Data Koordinat ROC (Simulasi Kelas A)

Diketahui 4 titik koordinat (FPR, TPR) untuk kurva ROC **Kelas A**:

Titik (i)	**X** (FPR)	**Y** (TPR)	**Keterangan**
1	0.00	0.00	Titik Awal
2	0.10	0.50	Threshold Ketat
3	0.40	0.70	Threshold Sedang
4	1.00	1.00	Titik Akhir

II. Perhitungan Luas Area (Trapezoidal Rule) Rumus Persamaan

$$AUC_{(k)} = \sum_{i=1}^{n-1} (FPR_{i+1} - FPR_i) \times \frac{(TPR_{i+1} + TPR_i)}{2}$$

Perhitungan Segmen Trapesium:

1. **Segmen 1** ($i = 1 \rightarrow 2$)**:**

$$(0.10 - 0.00) \times \frac{(0.50 + 0.00)}{2} = 0.10 \times 0.25 = \mathbf{0.025}$$

2. **Segmen 2** ($i = 2 \rightarrow 3$)**:**

$$(0.40 - 0.10) \times \frac{(0.70 + 0.50)}{2} = 0.30 \times 0.60 = \mathbf{0.180}$$

3. **Segmen 3** ($i = 3 \rightarrow 4$)**:**

$$(1.00 - 0.40) \times \frac{(1.00 + 0.70)}{2} = 0.60 \times 0.85 = \mathbf{0.510}$$

Total AUC Kelas A:

$$AUC_{(A)} = 0.025 + 0.180 + 0.510 = \mathbf{0.715}$$

III. Perhitungan Macro-Average AUC Rumus Persamaan

$$\text{MAUC}_{\text{ovr}} = \frac{1}{K} \sum_{k=1}^{K} AUC_{(k)}$$

Diketahui nilai AUC kelas lain (Simulasi): $AUC_B = 0.85$, $AUC_C = 0.88$, $AUC_D = 0.82$.

Hasil Akhir:

$$\text{MAUC} = \frac{0.75 + 0.85 + 0.88 + 0.82}{4} = \frac{3.265}{4} = \mathbf{0.816}$$