

SKRIPSI

**PERBANDINGAN ALGORITMA *NAÏVE BAYES* DENGAN
ALGORITMA *RANDOM FOREST* UNTUK ANALISIS
SENTIMEN PENGUNJUNG WISATA WAI MALINO DI
KABUPATEN ENREKANG**

***COMPARISON OF NAÏVE BAYES ALGORITHM WITH
RANDOM FOREST ALGORITHM FOR SENTIMENT ANALYSIS
OF TOURISTS VISITING WAI MALINO IN ENREKANG
REGENCY***



QALBI ALMUSTIKA M

D0221007

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS SULAWESI BARAT
MAJENE
2025**

SKRIPSI

**PERBANDINGAN ALGORITMA *NAÏVE BAYES* DENGAN
ALGORITMA *RANDOM FOREST* UNTUK ANALISIS
SENTIMEN PENGUNJUNG WISATA WAI MALINO DI
KABUPATEN ENREKANG**

***COMPARISON OF NAÏVE BAYES ALGORITHM WITH
RANDOM FOREST ALGORITHM FOR SENTIMENT ANALYSIS
OF TOURISTS VISITING WAI MALINO IN ENREKANG
REGENCY***

Diajukan untuk memenuhi salah satu syarat memperoleh gelar
Sarjana Komputer



QALBI ALMUSTIKA M

D0221007

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS SULAWESI BARAT
MAJENE
2025**

LEMBAR PERSETUJUAN

SKRIPSI

PERBANDINGAN ALGORITMA *NAÏVE BAYES* DENGAN ALGORITMA *RANDOM FOREST* UNTUK ANALISIS SENTIMEN PENGUNJUNG WISATA WAI MALINO DI KABUPATEN ENREKANG

Telah dipersiapkan dan disusun oleh

**QALBI ALMUSTIKA M
D0221007**

Telah dipertahankan di depan Tim Penguji
Pada tanggal 30 April 2025

Susunan Tim Penguji

Pembimbing I

Penguji I

Dr. Eng. Sulfayanti, S.Si., M.T
NIP: 198903172020122011

Farid Wajidi, S.Kom., M.T
NIP: 198904182019031018

Pembimbing II

Penguji II

Wawan Firgiawan, S.T., M.Kom
NIDK:89480880023

Siti Aulia Rachmini, S.T., M.T
NIP: 198207062008042003

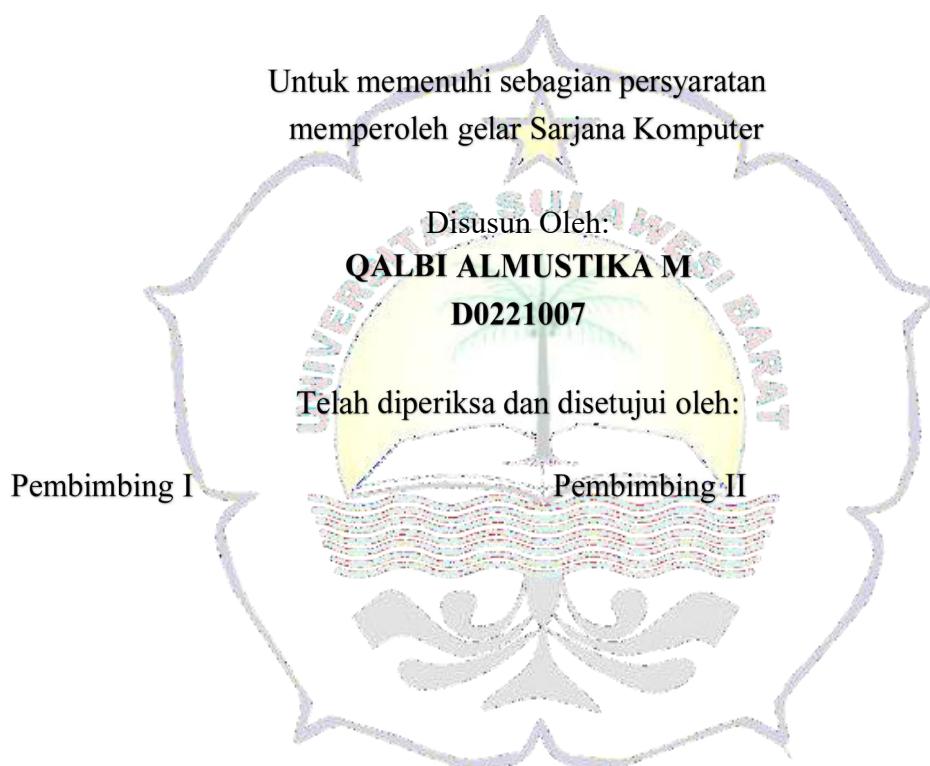
Penguji III

Chairi Nur Insani, S.Kom.,M.T
NIDN: 0027079404

LEMBAR PENGESAHAN

PERBANDINGAN ALGORITMA *NAÏVE BAYES* DENGAN ALGORITMA *RANDOM FOREST* UNTUK ANALISIS SENTIMEN PENGUNJUNG WISATA WAI MALINO DI KABUPATEN ENREKANG

SKRIPSI



Dr. Eng. Sulfayanti, S.Si., M.T
NIP: 198903172020122011

Wawan Firgiawan, S.T., M.Kom
NIDK:89480880023

Dekan Fakultas Teknik

Ketua Program Studi Informatika

Dr. Ir. Hafsa Nirwana, M.T
NIP. 1964040519900322002

Muh Rafli Rasyid, S.Kom., M.T
NIP. 198808182022031006

PERNYATAAN ORISINALITAS

Menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naska skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya, pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naska ini dan disebutkan dalam daftar referensi.

Apabila ternyata disalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundangundangan yang berlaku (**UU NO. 20 TAHUN 2003, pasal 25 ayat 2 dan pasal 70**)

Majene, 17 Februari 2025

Qalbi Almustika M
NIM. D0221007

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh.

Segala puji bagi Allah SWT, yang Maha Pengasih lagi Maha Penyayang dan Pengatur Alam Semesta, dengan Ridho Nya sehingga penulis mampu menyelesaikan penelitian yang berjudul “Perbandingan Algoritma *Naïve Bayes* dengan Algoritma *Random Forest* untuk Analisis Sentimen Pengunjung Wisata Wai Malino Di Kabupaten Enrekang”. Shalawat serta salam tidak lupa tetap tercurahkan kepada baginda Nabi Muhammad SAW, sebagai Uswatun Hasanah dan rahmat bagi seluruh alam.

Skripsi ini di buat untuk memenuhi persyaratan untuk memperoleh gelar Sarjana Komputer (S.Kom) pada Program Sarjana Teknik Informatika, Fakultas Teknik, Universitas Sulawesih Barat. Dalam membuat penelitian ini, walaupun banyak kesulitan dan hambatan yang penulis alami, namun berkat adanya dukungan, dan semangat dari orang terdekat di sekitar, sehingga penulis mampu menyelesaikan penelitian ini. Oleh karena itu penulis mengucapkan terima kasih kepada:

1. Kepada Allah Swt yang telah memberikan limpahan Rahmat dan hidayahnya untuk penulis mempunyai kekuatan dalam proses Pendidikan penulis di Universitas Sulawesi Barat
2. Ibu dan Ayah yang selalu mendoakan dan mendukung penulis dalam menyelesaikan tugas akhir.
3. Ibu Dr.Ir. Hafsa Nirwana, M.T selaku dekan Fakultas Teknik Universitas Sulawesi Barat.
4. Bapak Ir. Sugiarto Cokrowibowo, S.Si., M.T selaku wakil dekan Fakultas Teknik Universitas Sulawesi Barat.
5. Bapak Muh Rafli Rasyid, S.Kom., M.T selaku kaprodi Fakultas Teknik Universitas Sulawesi Barat.

6. Ibu Dr.Eng. Sulfayanti, S.Si., M.T dan Wawan Firgiawan, S.T., M.Kom selaku pembibing pertama dan kedua yang telah meluangkan waktunya untuk membimbing penulis dalam menyelesaikan tugas akhir penulis.
7. Bapak Farid Wajidi, S.Kom., M.T, Ibu Siti Aulia Rachmini, S.T., M.T dan Chairi Nur Insani, S.Kom.,M.T sekaku penguji satu, dua dan tiga yang telah memberikan saran dalam menyelesaikan tugas akhir penulis.
8. Serta teman-teman atas dukungan, motivasi, dan kebersamaan yang selalu memberi semangat kepada penulis dalam penulisan tugas akhir.
9. Seluruh dosen dan staf administrasi Fakultas Teknik yang telah membantu penulis menempu studi di Universitas Sulawesi Barat.
10. Pihak terkait di Kabupaten Enrekang Khususnya para pengunjung wisata Wai Malino, yang telah membantu dalam proses pengumpulan data untuk penelitian ini.

Penulis sadari skripsi ini masih banyak kesalahan dan kekurangan. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan untuk perbaikan di masa mendatang. Semoga karya ini dapat memberikan manfaat bagi pembaca serta menjadi kontribusi kecil dalam pengembangan ilmu pengetahuan.

Majene, 17 Februari 2025

Qalbi Almustika M
NIM. D0221007

ABSTRAK

Pariwisata berkontribusi signifikan terhadap perekonomian daerah, termasuk destinasi unggulan Wai Malino di Kabupaten Enrekang. penelitian ini melakukan analisis sentimen terhadap ulasan pengunjung menggunakan algoritma *Naïve Bayes* dan *Random Forest*. Untuk memahami opini wisatawan. Data dikumpulkan melalui kuesioner, kemudian diproses melalui tahapan preprocessing meliputi *cleansing*, *case folding*, *tokenizing*, *stopword removal*, dan *stemming*. Ulasan diklasifikasikan ke dalam tiga kategori sentimen positif, negatif, dan netral. Evaluasi model dilakukan menggunakan *confusion matrix* untuk mengukur akurasi, *precision*, *recall*, dan *F1-score*. Hasil penelitian menunjukkan bahwa sebelum dilakukan *oversampling*, kedua algoritma mengalami kesulitan dalam mengklasifikasikan kelas minoritas (netral dan negatif), meskipun akurasi keseluruhan terlihat cukup tinggi. *Naïve Bayes* mencatat akurasi sebesar 80,42%, sementara *Random Forest* mencapai 74,07%. Namun, keduanya menunjukkan *F1-score* yang rendah pada kelas minoritas. *Oversampling* adalah teknik dalam data *balancing* yang digunakan untuk mengatasi ketidakseimbangan kelas dengan cara memperbanyak jumlah data pada kelas minoritas, sehingga model tidak bias terhadap kelas *majoritas*. Setelah dilakukan *oversampling* untuk menyeimbangkan distribusi data, performa kedua algoritma berubah. *Naïve Bayes* menunjukkan peningkatan signifikan dengan akurasi 86,27% dan *F1-score* yang seimbang pada semua kelas (85–87%), sedangkan *Random Forest* mengalami penurunan performa dengan akurasi turun menjadi 64,95% dan *F1-score* yang tidak konsisten antar kelas (63–67%). Visualisasi *word cloud* mengungkap bahwa kata-kata seperti "nyaman", "indah", dan "sejuk" mendominasi sentimen positif, sementara kata seperti "kurang", "sampah", dan "fasilitas" lebih sering muncul dalam sentimen negatif. Hasil ini diharapkan membantu pengelola wisata dalam meningkatkan layanan berdasarkan opini pengunjung serta sebagai referensi pemilihan model klasifikasi sentimen.

Kata Kunci: Analisis Sentimen, *Naïve Bayes*, *Random Forest*, Wisata Wai Malino, *Oversampling*

ABSTRACT

Tourism contributes significantly to the regional economy, including the flagship destination of Wai Malino in Enrekang Regency. This study conducts sentiment analysis on visitor reviews using the Naïve Bayes and Random Forest algorithms to understand tourist opinions. Data was collected through questionnaires and then processed through several preprocessing stages, including cleansing, case folding, tokenizing, stopword removal, and stemming. The reviews were classified into three sentiment categories: positive, negative, and neutral. Model evaluation was carried out using a confusion matrix to measure accuracy, precision, recall, and F1-score. The results showed that before applying oversampling, both algorithms had difficulty classifying the minority classes (neutral and negative), even though the overall accuracy appeared relatively high. Naïve Bayes achieved an accuracy of 80.42%, while Random Forest reached 74.07%. However, both showed low F1-scores in the minority classes. Oversampling is a data balancing technique used to address class imbalance by increasing the number of samples in the minority classes so that the model is not biased toward the majority class. After applying oversampling to balance the data distribution, the performance of both algorithms changed. Naïve Bayes showed a significant improvement with an accuracy of 86.27% and balanced F1-scores across all classes (85–87%), while Random Forest experienced a performance decline with accuracy dropping to 64.95% and inconsistent F1-scores between classes (63–67%). The word cloud visualization revealed that words such as “comfortable,” “beautiful,” and “cool” dominated positive sentiments, while words like “lack,” “trash,” and “facilities” appeared more frequently in negative sentiments. These results are expected to help tourism managers improve services based on visitor feedback and serve as a reference for selecting sentiment classification models.

Keywords: Sentiment Analysis, Naïve Bayes, Random Forest, Wai Malino Tourism, Oversampling

DAFTAR ISI

LEMBAR PERSETUJUAN	ii
LEMBAR PENGESAHAN.....	iii
PERNYATAAN ORISINALITAS	iv
KATA PENGANTAR	v
ABSTRAK.....	vii
<i>ABSTRACT</i>	viii
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xi
DAFTAR TABEL.....	xii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	5
1.3 Tujuan Penelitian	5
1.4 Manfaat Penelitian	5
1.5 Batasan Masalah	6
BAB II TINJAUAN PUSTAKA	7
2.1 Landasan Teori	7
2.2 Penelitian Terkait	20
2.3 Kerangka Pikir	26
BAB III METODE PENELITIAN	29
3.1 Jenis Penelitian	29
3.2 Tempat dan Jawal Penelitian	29
3.3 Tahapan Penelitian	29
3.4 Perancangan Program	32
BAB IV HASIL DAN PEMBAHASAN	40
4.1 Implementasi	40
4.2 Pengumpulan Data	40
4.3 Teks <i>Preprocessing</i>	42
4.4 Pelabelan <i>Lexicon</i>	47

4.5 Pembobotan TF-IDF	49
4.6 Pembagian Data	54
4.7 Mekanisme Kerja Algoritma Klasifikasi	55
4.8 Pengujian Akurasi	84
4.9 Visualisasi	97
4.10 Analisis Hasil	103
BAB V KESIMPULAN DAN SARAN	109
5.1 Kesimpulan	109
5.2 Saran	110
DAFTAR PUSTAKA	111
LAMPIRAN	114

DAFTAR GAMBAR

Gambar 2. 1 Ilustrasi <i>oversampling</i>	17
Gambar 2. 2 Kerangka Fikir	27
Gambar 3.1 Alur kerja Penelitian	30
Gambar 3. 2 <i>Flowchart</i> Penggambaran Sistem	32
Gambar 3. 3 Tahapan <i>Preprocessing</i> Data	33
Gambar 3. 4 <i>Flowchart</i> Algoritma <i>Naïve Bayes</i>	36
Gambar 3. 5 <i>Flowchart</i> Algoritma <i>Random Forest</i>	37
Gambar 4. 1 Pohon Pertama	81
Gambar 4. 2 Pohon Kedua	82
Gambar 4. 3 Pohon Ketiga	82
Gambar 4. 4 <i>Confusion Matrix</i> Sebelum <i>Oversampling Naïve Bayes</i>	85
Gambar 4. 5 <i>Confusion Matrix</i> Sebelum <i>Oversampling Random Forest</i>	88
Gambar 4. 6 <i>Confusion Matrix</i> Sesudah <i>Oversampling Naïve Bayes</i>	91
Gambar 4. 7 <i>Confusion Matrix</i> Sesudah <i>Oversampling Random Forest</i>	94
Gambar 4. 8 <i>Word Cloud</i> Sentimen Positif <i>Naïve Bayes</i>	98
Gambar 4. 9 <i>Word Cloud</i> Sentimen Negatif <i>Naïve Bayes</i>	99
Gambar 4. 10 <i>Word Cloud</i> Sentimen Netral <i>Naïve Bayes</i>	99
Gambar 4. 11 <i>Word Cloud</i> Sentimen Positif <i>Random Forest</i>	101
Gambar 4. 12 <i>Word Cloud</i> Sentimen Negatif <i>Random Forest</i>	101
Gambar 4. 13 <i>Word Cloud</i> Sentimen Netral <i>Random Forest</i>	102
Gambar 4. 15 Grafik Perbandingan	103
Gambar 4. 16 Grafik Perbandingan <i>Precision</i>	104
Gambar 4. 17 Grafik Perbandingan <i>Recall</i>	106
Gambar 4. 18 Grafik Perbandingan <i>F1-Score</i>	107

DAFTAR TABEL

Tabel 2. 1 <i>Cofusion Matrix</i>	17
Tabel 2. 2 Rumus <i>Multiclass Confussion Matrix</i>	18
Tabel 2.3 Penelitian Terkait.....	20
Tabel 3. 1 <i>Preprocessing Data</i>	34
Tabel 4. 1 Sampel data yang dikumpulkan	41
Tabel 4. 2 Komentar pengunjung wisata	42
Tabel 4. 3 <i>Cleansing</i>	43
Tabel 4. 4 <i>Case Colding</i> dan Normalisasi Kata	44
Tabel 4. 5 <i>Tokenizing</i>	45
Tabel 4. 6 <i>Stopword Removal</i>	46
Tabel 4. 7 <i>Stemming</i>	47
Tabel 4. 8 Kamus <i>Lexicon</i>	48
Tabel 4. 9 Cara kerja <i>Lexicon-based</i>	48
Tabel 4. 10 Hasil <i>Lexicon</i>	49
Tabel 4. 11 Data Komentar untuk Perhitungan TF-IDF	50
Tabel 4. 12 Distribusi nilai TF	51
Tabel 4. 13 Hasil perhitungan DF dan nilai IDF	52
Tabel 4. 14 Hasil perhitunganTF-IDF	53
Tabel 4. 15 Pembagian Data <i>Training</i> dan <i>Testing</i> Sebelum <i>Oversampling</i>	55
Tabel 4. 16 Pembagian Data <i>Training</i> dan <i>Testing</i> Setelah <i>Oversampling</i>	55
Tabel 4. 17 Contoh <i>dataset</i> untuk Perhitungan <i>Naïve Bayes</i>	56
Tabel 4. 18 Probabilitas Frekuensi setiap sentimen	56
Tabel 4. 19 Probabilitas Data Latih Semua Sentimen	57
Tabel 4. 20 Perubahan Nilai Probabilitas Sentimen Positif	59
Tabel 4. 21 Perubahan Nilai Probabilitas Sentimen Negatif	60
Tabel 4. 22 Perubahan Nilai Probabilitas Sentimen Netral	60
Tabel 4. 23 Dafta <i>Term</i> yang akan diklasifikasi	62
Tabel 4. 24 Probabilitas <i>Term</i> Data Uji pada Sentimen Positif	62
Tabel 4. 25 Probabilitas <i>Term</i> Data Uji pada Sentimen Negatif	62
Tabel 4. 26 Probabilitas <i>Term</i> Data Uji pada Sentimen Netral	63
Tabel 4. 27 <i>Dataset</i> Awal perhitungan <i>Random Forest</i>	63

Tabel 4. 28 <i>Bootstrap</i> Pertama	64
Tabel 4. 29 <i>Bootstrap</i> Kedua	64
Tabel 4. 30 <i>Bootstrap</i> Ketiga	65
Tabel 4. 31 <i>Dataset awal/Node 1 (Decision Tree 1)</i>	65
Tabel 4. 32 Hasil Perhitungan <i>Entropy</i> dan <i>Gini (Decision Tree 1)</i>	68
Tabel 4. 33 <i>Dataset</i> untuk <i>node 1.1 (Decision Tree 1)</i>	68
Tabel 4. 34 Hasil Perhitungan <i>Entropy</i> dan <i>Gini</i> pada <i>Node 1.1</i>	70
Tabel 4. 35 <i>Dataset</i> untuk <i>Node 1.2 (Decision Tree 1)</i>	71
Tabel 4. 36 Hasil Perhitungan <i>Entropy</i> dan <i>Gini</i> pada <i>Node 1.2</i>	71
Tabel 4. 37 <i>Dataset awal perhitungan Node 1 (Decision Tree 2)</i>	72
Tabel 4. 38 Hasil Perhitungan <i>Entropy</i> dan <i>Information</i>	74
Tabel 4. 39 <i>Dataset</i> untuk <i>Node 1.1 (Decision Tree 2)</i>	74
Tabel 4. 40 Hasil Perhitungan <i>Entropy</i> dan <i>Gini</i> pada <i>Node 1.1</i>	75
Tabel 4. 41 <i>Dataset awal perhitungan Node 1 (Decision Tree 3)</i>	76
Tabel 4. 42 Hasil Perhitungan <i>Entropy</i> dan <i>Gini Node 1</i>	78
Tabel 4. 43 <i>Dataset</i> untuk <i>Node 1.1 (Decision Tree 3)</i>	79
Tabel 4. 44 Hasil Perhitungan <i>Entropy</i> dan <i>Gini</i> pada <i>Node 1.1</i>	79
Tabel 4. 45 <i>Dataset</i> untuk <i>Node 1.2 (Decision Tree 3)</i>	80
Tabel 4. 46 Hasil Perhitungan <i>Entropy</i> dan <i>Gini</i> pada <i>Node 1.2</i>	80
Tabel 4. 47 Hasil Pengujian <i>Naïve Bayes</i> Sebelum <i>Oversampling</i>	87
Tabel 4. 48 Hasil Pengujian <i>Random Forest</i> Sebelum <i>Oversampling</i>	90
Tabel 4. 49 Hasil Pengujian <i>Naïve Bayes</i> Sesudah <i>Oversampling</i>	93
Tabel 4. 50 Hasil pengujian <i>Random Forest</i> Sesudah <i>Oversampling</i>	96

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pariwisata merupakan sektor industri dengan kemampuan pertumbuhan yang baik di indonesia. Indonesia dikenal sebagai wilayah yang memiliki keberagaman ekosistem yang melimpah, baik di darat maupun di laut. Selain itu, kekayaan sumber daya alam yang dimiliki Indonesia dapat menjadi potensi besar untuk mendorong sektor pariwisata dan meningkatkan perekonomian. Kekayaan alam tersebut menarik minat berbagai pihak dan instansi untuk mengembangkannya menjadi objek wisata (Utami et. al., 2022). Berdasarkan data Badan Pusat Statistik (BPS), Jumlah kunjungan wisatawan tahun 2023 mencapai 6,31 juta kunjungan, meningkat sebesar 196,8% dibandingkan dengan 5,47 juta kunjungan pada priode yang sama tahun sebelumnya. Peningkatan jumlah wisatawan ini menunjukkan adanya ketertarikan yang tinggi terhadap destinasi wisata di Indonesia, baik dari sisi keindahan alam maupun fasilitas yang tersedia. Hal ini memberikan tantangan bagi pemerintah dan pengelola destinasi wisata untuk terus melakukan inovasi dan pengembangan agar destinasi wisata tetap menarik serta memberikan pengalaman yang memuaskan bagi pengunjung.

Objek wisata merupakan salah satu komponen ekonomi bagi masyarakat dan juga pendapatan pemerintah daerah (Atmadja, 2022). Kabupaten Enrekang, yang terletak di Provinsi Sulawesi Selatan, memiliki potensi besar dalam pengembangan objek wisata, karena letak geografisnya yang strategis, kekayaan alam yang melimpah, serta budaya yang khas. Salah satu wisata di Enrekang adalah wisata Wai Malino, yang memiliki daya tarik alam yang memikat. Berdasarkan data dari Dinas Pariwisata Kabupaten Enrekang, Jumlah kunjungan wisatawan ke Enrekang, mengalami fluktuasi dalam tiga tahun terakhir. Pada tahun 2022, jumlah wisatawan mencapai 63.511 orang, meningkat 156,3% dari tahun sebelumnya. Tren positif berlanjut di tahun 2023, dengan kunjungan naik

menjadi 116.034 orang atau meningkat 82,7%. Namun, pada tahun 2024, jumlah wisatawan turun drastis menjadi 57.058 orang, mengalami penurunan sebesar 50,8% dibandingkan tahun sebelumnya. Penurunan ini menunjukkan adanya tantangan dalam mempertahankan jumlah kunjungan wisatawan.

Meskipun memiliki potensi besar, sektor pariwisata di Enrekang, termasuk wisata Wai Malino masih menghadapi berbagai tantangan seperti kurangnya pemahaman yang mendalam mengenai minat dan kepuasan pengunjung, keterbatasan dalam pengelolaan destinasi. Selain itu, analisis umpan balik pengunjung yang belum optimal menjadi kendala utama dalam meningkatkan kualitas layanan dan pengalaman wisata.

Oleh karena itu, untuk mengatasi permasalahan ini, diperlukan pendekatan yang lebih sistematis dan berbasis data. Salah satu pendekatan yang dapat digunakan adalah analisis sentimen, yang digunakan untuk menggali sentimen atau opini yang terkandung dalam ulasan dari pengunjung dan mampu mengelola umpan balik pengunjung serta efektif untuk memahami preferensi dan kebutuhan pengunjung (Syahla et al., 2023). Sehingga dapat membantu meningkatkan pengelolaan dan pengembangan destinasi wisata di Kabupaten Enrekang khususnya di desa Parombean secara lebih akurat dan efisien.

Analisis sentimen adalah bidang yang mempelajari bagaimana cara menganalisis opini, sentimen, dan emosi seseorang terhadap suatu produk, topik, masalah, organisasi, atau individu. Secara sederhana, bidang ini mengklasifikasikan teks yang terdapat dalam dokumen, kalimat, atau fitur tertentu. Setelah itu hasil klasifikasi tersebut diberikan label atau status, yang biasanya dikelompokkan ke dalam tiga kategori utama positif, netral, dan negatif (Widyarto et al., 2023). Melalui analisis sentimen, pengelola destinasi wisata dapat memperoleh wawasan yang lebih mendalam tentang apa yang disukai atau tidak disukai oleh pengunjung, serta faktor-faktor yang mempengaruhi pengalaman mereka.

Naïve Bayes merupakan salah satu pendekatan pengklasifikasian berbasis probabilitas yang sering digunakan dalam analisis sentimen. Metode ini memiliki kelebihan dalam mengklasifikasikan data dengan akurasi tinggi, meskipun menggunakan sejumlah besar data. Selain itu, pengklasifikasian dengan metode *Naïve Bayes* memiliki kecepatan dan akurasi yang tinggi ketika diterapkan pada banyak set data (Rizal *et al.*, 2024). Oleh karena itu, algoritma ini sangat cocok untuk menganalisis data ulasan pengunjung.

Selain *Naïve Bayes*, algoritma *Random Forest* juga banyak digunakan dalam analisis sentimen karena kemampuannya dalam menangani data yang kompleks. Algoritma ini bekerja dengan menggabungkan sejumlah pohon keputusan untuk meningkatkan akurasi prediksi. Salah satu keunggulan Random mampu memberikan estimasi terhadap pentingnya setiap variabel dalam model, yang memungkinkan analisis data lebih mendalam. Dalam konteks analisis sentimen, algoritma ini telah terbukti mampu menghasilkan klasifikasi dengan tingkat akurasi yang tinggi (Firdaus *et al.*, 2025).

Sebelum dilakukan proses klasifikasi data menggunakan *Naïve Bayes Classifier*, dan *Random Forest* data akan diproses terlebih dahulu pada tahapan *preprocessing* seperti *case folding*, *tokenizing*, dan *stemming*. Proses awal ini diperlukan agar selama tahapan klasifikasi hasil yang diperoleh dapat lebih optimal. Selanjutnya dilakukan pembobotan dimana fitur dibuat untuk memudahkan terjadinya proses *machine learning* melalui metode *Naïve Bayes Classifier* (Khasanah, 2023).

Beberapa penelitian terdahulu telah melakukan penelitian terkait *Supervised Learning* menggunakan metode *Naïve Bayes* dan *Random Forest* dalam mengenali sentimen. Pada Penelitian yang di lakukan oleh (Singgalen, 2022) dengan judul “Analisis Sentimen Wisatawan Melalui Data Ulasan Candi Borobudur di Tripadvisor Menggunakan Algoritma *Naïve Bayes Classifier*” menunjukkan bahwa klasifikasi sentimen positif dan negatif pada data ulasan wisatawan tentang Candi Borobudur dapat dilakukan menggunakan algoritma

Naïve Bayes Classifier dengan bantuan aplikasi RapidMiner. Dari 3.850 ulasan yang dianalisis, algoritma ini mencapai akurasi sebesar 96,36%, presisi 93,23%, dan *recall* 100%. Menunjukkan bahwa algoritma *Naïve Bayes* afektif dalam menganalisis sentimen ulasan candi Borobudur. Pada penelitian yang dilakukan (Indarbensyah et al., 2021) dengan judul “Penerapan N-Gram menggunakan Algoritma *Random Forest* dan pada Analisis Sentimen Kebijakan PPKM 2021” menunjukkan bahwa klasifikasi sentimen terhadap Kebijakan PPKM 2021 dapat dilakukan menggunakan algoritma *Random Forest* dan *Naïve Bayes Classifier* dengan penerapan teknik N-Gram. Dari hasil pengujian yang dilakukan diperoleh akurasi terbaik pada penerapan Unigram dengan algoritma *Random Forest* sebesar 99,5%, serta *precision*, *recall*, dan *f1-score* yang sama, yaitu 100%. Sementara itu, model *Naïve Bayes Classifier* dengan Unigram mencapai akurasi 97,9%, dengan *precision*, *recall*, dan *f1-score* sebesar 98%. Dengan demikian, dapat disimpulkan bahwa baik algoritma *Random Forest* maupun *Naïve Bayes Classifier* memiliki kemampuan tinggi dalam mengklasifikasikan analisis sentimen.

Berdasarkan latar belakang yang telah diuraikan penulis tertarik melakukan penelitian dengan judul, “Perbandingan Algoritma *Naïve Bayes* dengan Algoritma *Random Forest* untuk Analisis Sentimen Pengunjung Wisata Wai Malino di Kabupaten Enrekang”. Dengan perbandingan algoritma *Naïve Bayes* dan *Random Forest* dalam analisis sentimen, diharapkan dapat memberikan informasi yang berguna bagi pengelola wisata dalam meningkatkan pelayanan, fasilitas dan promosi destinasi wisata. Selain itu, dapat memberikan rekomendasi berbasis data yang lebih akurat untuk pengembangan sektor pariwisata dan menciptakan pengalaman wisata yang lebih menyenangkan, yang pada akhirnya dapat mendukung pertumbuhan ekonomi dan kesejahteraan masyarakat lokal secara berkelanjutan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah di bahas sebelumnya maka rumusan masalah pada penelitian ini yaitu:

1. Bagaimana hasil analisis sentimen dapat digunakan untuk meningkatkan kualitas layanan dan pengelolaan wisata Wai Malino?
2. Bagaimana perbandingan antara algoritma *Naïve Bayes* dan *Random Forest* dalam analisis sentimen ulasan pengunjung wisata Wai Malino di Kabupaten Enrekang?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, adapun tujuan dari penelitian ini yaitu:

1. Menganalisis hasil sentimen ulasan pengunjung wisata Wai Malino untuk meningkatkan kualitas layanan dan pengelolaan destinasi wisata.
2. Membandingkan kinerja algoritma *Naïve Bayes* dan *Random Forest* dalam menganalisis sentimen ulasan pengunjung wisata Wai Malino di Kabupaten Enrekang.

1.4 Manfaat Penelitian

Berdasarkan dengan permasalahan dan tujuan penelitian, maka penulis mengharapkan penelitian ini dapat memberikan beberapa manfaat di bawah ini:

1. Menambah wawasan dan pemahaman kepada penulis dalam menganalisis sentimen pengunjung wisata Wai Malino menggunakan algoritma *Naïve Bayes* dan *Random Forest*.
2. Hasil penelitian ini dapat menjadi referensi penulis apabila hendak melanjutkan penelitian di masa yang akan datang.

1.5 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Analisis sentimen hanya difokuskan pada opini terkait pengunjung wisata Wai Malino dengan kategori sentimen positif, netral, dan negatif.
2. Metode yang digunakan untuk analisis sentimen adalah algoritma *Naïve Bayes* dan *Random Forest*.
3. Data yang digunakan pada penelitian ini adalah ulasan pengunjung yang dikumpulkan melalui kuesioner.

BAB II

TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Wisata

Pariwisata merupakan sektor ekonomi yang mengalami pertumbuhan cepat, dipengaruhi oleh kemajuan teknologi yang memudahkan wisatawan dalam merencanakan dan menikmati perjalanan. Sangat penting untuk memahami dampak integrasi teknologi pada keberlanjutan destinasi pariwisata seiring dengan perkembangan ini. Disinilah pentingnya analisis destinasi wisata yang menggabungkan teknologi dan keberlanjutan. Penggunaan teknologi dalam industri pariwisata telah membuka jalan baru untuk pengalaman wisata yang efisien, produktif, dan informatif bagi pengunjung (Prasetyo et al., 2024).

Pariwisata adalah aktifitas perjalanan yang dilakukan oleh individu maupun kelompok dengan tujuan memperoleh kebahagiaan serta mendapatkan pengetahuan baru. Kegiatan wisata ini merupakan salah satu sektor yang berpotensi besar untuk mengembangkan ekonomi memberikan manfaat signifikan bagi suatu negara. Tujuan utama meningkatkan kesejahteraan, sehingga menjadikannya pilar penting bagi pertumbuhan ekonomi dan budaya (Sunardi et al., 2021).

2.1.2 *Sentiment Analysis*

Sentiment Analysis ialah proses pengambilan dan pemrosesan data secara otomatis yang mengumpulkan informasi sentimen dalam kalimat opini dengan menggunakan algoritma tertentu (Ardianto et al., 2020). *Sentiment Analysis* (Opini mining) merupakan salah satu bidang yang luas dari pengolahan bahasa alami, komputasi linguistik dan teks mining dimana memiliki tujuan dalam menganalisa pendapat, sentimen, evaluasi, sikap penilaian dan emosi seseorang apakah pembicara atau penulis berkenan dengan suatu topik, produk, layanan, organisasi, individu, ataupun kegiatan tertentu.

Sentiment Analysis memiliki tugas dasar dalam mengelompokkan teks yang ada pada sebuah kalimat atau dokumen kemudian menentukan pendapat

yang di kemukakan dalam kalimat atau dokumen tersebut apakah bersifat positif, negatif atau netral (Munawaroh et al., 2024). *Sentiment Analysis* adalah proses mengumpulkan informasi dari kumpulan data tidak terstruktur dengan memahami, mengekstraksi, dan megolah data teks secara otomatis untuk mendapatkan informasi tentang pendapat atau perilaku seseorang. Dengan mengekstrak teks ulasan untuk mengetahui emosi pengguna, analisis sentimen berguna untuk menentukan apakah tanggapan pengguna terhadap suatu produk baik atau tidak (Era et al., 2023).

2.1.3 *Text Preprocessing*

1) Pengertian *Text Preprocessing*

Text preprocessing adalah persiapan data sebelum pemodelan. Salah satu teknik data mining adalah *preprocessing*, yang melakukan transformasi pada data mentah agar menjadi format yang lebih mudah dimengerti. Tujuannya adalah untuk menyelesaikan masalah seperti redundansi data, dan data yang hilang (Agustina et al., 2022).

2) Tahapan *Text Preprocessing*

Menurut (Harfian, 2021) tahapan-tahapan *preprocessing text* yang digunakan adalah sebagai berikut:

a) *Case Folding*

Case Folding merupakan tahapan yang mengubah semua buruf dalam dokumen menjadi huruf kecil. Hanya huruf “a” sampai “z” yang di terima.

b) *Cleansing*

Cleansing merupakan tahapan untuk membersikan kata-kata yang tidak di perlukan dengan beberapa teknik seperti memperkecil noise, membetulkan data yang tidak konsisten, *megisi missing value*, mengidentifikasi atau membuang *outlier*. Kata-kata yang dihilangkan pada tahapan ini adalah URL, hashtag(#), username (@username), dan email. Selain itu tanda baca seperti titik(.) koma(,), dan sebuah simbol atau tanda baca dihilangkan.

c) *Tokenizing*

Tokenizing atau tokenisasi adalah proses pemecahan teks menjadi kata-kata atau token. Dengan dilakukan pemecahan teks ini akan memudahkan pada proses *stopword* dan *Stemming*, Karena dua proses tersebut akan mencocokan kata per kata dengan *root* atau kata dasar.

d) *Stopword Removal / Filtering*

Stopword removal merupakan proses penghilangan kata yang tidak penting pada deskripsi melalui pengecekan kata-kata hasil parsing dokumen apakah termasuk di dalam daftar kata yang tidak penting (*stoplist*). Jika termasuk di dalam stoplist maka kata-kata tersebut akan dihilangkan dari deskripsi sehingga kata-kata yang tersisa di dalam deskripsi dianggap sebagai kata-kata penting atau *keywords*).

e) *Stemming*

Stemming adalah proses mengubah berbagai bentuk morfologis dari sebuah kata menjadi bentuk dasar atau akar kata. Proses ini dilakukan untuk menyerderhanakan kata-kata dengan varian morfologi agar menjadi satu bentuk dasar yang konsisten (*root*), sehingga memudahkan analisis dan pemrosesan teks.

2.1.4 Pelabelan/Pembobotan

Pelabelan dilakukan berdasarkan kamus *lexicon* dimana data diberi label positif maupun negatif. Kata-kata yang terdapat dalam kamus *lexicon* akan dihitung skornya berdasarkan jumlah kata yang ada dalam setiap teks atau kalimat. Berikut ini persamaan 2.1 dan 2.2 yang digunakan untuk menghitung skor positif dan negatif (Ismail et al., 2023):

$$S_{positive} = \sum_{i \in t}^n positive\ score_i \quad (2.1)$$

$$S_{negative} = \sum_{i \in t}^n negative\ score_i \quad (2.2)$$

Keterangan:

n = Jumlah total dalam kelompok.

$i \in t$ = Indeks i berada dalam himpunan/kelompok t.

$S_{positive}$ dan $S_{negative}$ = Total untuk kalimat yang diperoleh dari penjumlahan n skor kata positif dan negatif.

Dari persamaan yang mengambarkan nilai sentimen dalam sebuah kalimat, dapat disimpulkan persamaan 2.3 untuk menentukan orientasi sentimen berdasarkan perbandingan antara jumlah nilai positif, negatif, dan netral:

$$Sentiment = \begin{cases} positive & if S_{positive} > S_{negative} \\ neutral & if S_{positive} = S_{negative} \\ negative & if S_{positive} < S_{negative} \end{cases} \quad (2.3)$$

Keterangan:

Positif = Jika kata positif lebih banyak dari kata negatif maka dilabeli positif

Netral = Jika jumlah kata positif sama dengan kata negatif maka dilabeli netral

Negatif = Jika kata positif lebih sedikit dari kata negatif maka dilabeli negatif

Penentuan sentimen dalam analisis ini dilakukan dengan menjumlahkan nilai dari kata kunci positif dan negatif yang terdapat dalam kalimat berdasarkan kamus *lexicon*. Hasil penjumlahan tersebut disebut sebagai skor sentimen. Jika skor sentimen yang dihasilkan lebih dari nol ($score > 0$), maka kalimat diklasifikasikan sebagai sentimen positif. Jika skor kurang dari nol ($score < 0$), maka kalimat termasuk dalam sentimen negatif. Sedangkan jika skor sama dengan nol ($score = 0$), maka kalimat dianggap memiliki sentimen netral (Undap et al., 2021).

Pembobotan kata adalah ketika setiap kata dalam teks atau dokumen diberi nilai atau pertimbangan berdasarkan relevansi dan pentingnya kata tersebut dalam konteks tertentu. Pembobotan kata dilakukan untuk memberikan nilai pada setiap kata dalam dokumen yang bertujuan agar teks tersebut dapat di klasifikasi. Salah satu metode pembobotan yang umum digunakan adalah TF-IDF (*Term Frequency - invers Document Freqency*) (Kosasih et al., 2021).

TF-IDF adalah metode statistika numerik yang mengukur pentingnya suatu kata dalam sebuah dokumen. Metode ini banyak digunakan dalam berbagai

bidang, terutama dalam analisis teks dan klasifikasi. TF-IDF berfungsi sebagai faktor pembobotan dalam *teks mining* yang berarti memberikan hubungan antara kata tertentu dan dokumen (Khasanah, 2023). Pembobotan kata adalah proses untuk menetapkan nilai pada setiap kata dalam data yang sudah melewati tahapan pemrosesan teks (Kosasih et al, 2021).

TF-IDF terdiri dari dua statistik utama, yaitu *Term Frequency* (TF) dan *Inverse Document Frequency* (IDF) (Kosasih et al., 2021):

- 1) *Term Frequency* (TF) adalah istilah yang mengacu pada kemunculan sebuah kata atau istilah dalam suatu dokumen tertentu. Fungsinya untuk menilai seberapa relevan atau singnifikan istilah tersebut dalam konteks dokumen. Dapat dihitung menggunakan persamaan 2.4.

$$tf_{x.d} = \text{count}(x, d) \quad (2.4)$$

Keterangan:

x = Kata

d = Dokumen

$tf_{x.d}$ = Frekuensi banyaknya kata x yang muncul dalam dokumen d

- 2) *Inverse Document Frequency* (IDF) merupakan sebuah bobot ststistik yang secara khusus digunakan untuk mengukur dan menentukan tingkat suatu istilah atau kata dalam sebuah kumpulan dokumen teks. Dalam penerapannya, fitur IDF ini biasanya digabungkan dengan tujuan untuk melakukan pengurangan nilai bobot dari kata yang sering muncul dalam kumpulan dokumen dan juga meningkatkan nilai bobot dari istilah yang jarang atau tidak sering muncul dalam dokumen, dihitung dengan menggunakan persamaan 2.5 berikut:

$$idf_x = \log \left(\frac{N}{d_x} \right) \quad (2.5)$$

Keterangan:

N = Total Dokumen

d_x = Dokumen yang mengandung kata x

idf_x = Frekuensi banyaknya kata x yang muncul di dokumen d

Term Frequency - Inverse Document Frequency (TF-IDF) dihitung setiap kata dengan menggunakan persamaan 2.6 berikut:

$$W_{x,d} = tf_{x,d} * idf_{x,d} \quad (2.6)$$

Keterangan:

$tf_{x,d}$ = Frekuesi banyaknya x kata yang muncul dalam dokumen d.

$idf_{x,d}$ = Frekuensi banyaknya x kata yang muncul di dokumen d.

$W_{x,d}$ = Nilai TF-IDF (bobot dokumen ke-d terhadap kata ke x).

2.1.5 Algoritma *Naïve Bayes*

Naïve Bayes merupakan sebuah metode klasifikasi yang sederhana dan banyak digunakan dalam data mining dan *mechine learning*. Metode ini di dasarkan pada teorema Bayes, dengan asumsi yang sering disebut “naïf” karena menganggap bahwa setiap fitur dalam data bersifat independen satu sama lain (Widyarto et al., 2023).

Algoritma ini pertama kali di usulkan oleh seorang ilmuan Inggris bernama Thomas Bayes. Termasuk dalam kategori algoritma klasifikasi data mining, algoritma ini menggunakan pendekatan *statistic* dan *probabilitas*. Tujuan adalah untuk memprediksi kejadian di masadepan berdasarkan data yang ada saat ini atau dari kejadian-kejadian sebelumnya. Algoritma ini memiliki keterkaitan antara variable-variabel independen dan dikenal sebagai *Naïve Bayes* (Aditya et al., 2022) pada penelitian ini menjelaskan Teorema Bayes memiliki bentuk umum seperti pada persamaan 2.7:

$$P(H|X) = \frac{P(X|H).P(H)}{P(X)} \quad (2.7)$$

Keterangan:

X = Data dengan *class* yang belum diketahui

H = Hipotesis data X merupakan satu *calss* pesifik.

- $P(H|X)$ = Probabilitas *hipotesis H* berdasarkan kondisi X (*probability posteriori*)
 $P(X|H)$ = Probabilitas data X berdasarkan kondisi pada hipotesis H (*likelihood*)
 $P(H)$ = Probabilitas *hipotesis H* (*probability prior*)
 $P(X)$ = Probabilitas awal (*priori*) bukti X tanpa mengundang *hipotesis* yang lain

Naïve Bayes atau bisa disebut sebagai *Multinomial Naïve Bayes* merupakan model penyederhanaan dari Metode Bayes yang cocok dalam pengklasifikasian teks atau dokumen. Pada persamaan (2.7) merupakan persamaan model penyederhanaan dari Metode Bayes.

$$P_{MAP} = \arg \max \frac{P(a_1, a_2, \dots, a_n | v_j) \cdot P(v_j)}{P(a_1, a_2, \dots, a_n)} \quad (2.8)$$

Karena nilai $P(a_1, a_2, \dots, a_n)$ untuk semua v_j besarnya sama maka nilainya dapat diabaikan sehingga persamaan (2.8) menjadi:

$$P_{MAP} = \arg \max P(a_1, a_2, \dots, a_n | v_j) \cdot P(v_j) \quad (2.9)$$

Dengan mengasumsikan bahwa setiap kata dalam $\langle a_1, a_2, \dots, a_n \rangle$ adalah independen, maka $P(a_1, a_2, \dots, a_n | v_j) \cdot P(v_j)$ dalam persamaan (2.9) dapat ditulis:

$$V_{MAP} = \arg \max P(v_j) \prod_i P(a_i | v_j) \quad (2.10)$$

Nilai $P(v_j)$ = ditentukan pada saat pelatihan, yang nilainya di dekati dengan

$$P(v_j) = \frac{|doc_j|}{|Contoh|} \quad (2.11)$$

Keterangan :

- $|doc_j|$ = Banyak dokumen yang memiliki kategori j dalam pelatihan
 $|Contoh|$ = Banyak dolumen dalam contoh yang digunakan untuk pelatihan

Untuk nilai $P(w_k|v_j)$ yaitu probabilitas kata w_k dalam kategori j ditentukan dengan:

$$P(w_k|v_j) = \frac{|n_k+1|}{n+|vocabulary|} \quad (2.12)$$

Keterangan :

n_k = Frekuensi munculnya kata w_k dalam dokumen yang berkategori v_j

n = Seluruh kata dalam dokumen berkategori v_j

$|vocabulary|$ = total kata dalam contoh pelatihan.

2.1.6 Algoritma Random Forest

Random Forest (RF) merupakan salah satu metode dalam *machine learning* yang digunakan untuk mengklasifikasikan data dalam jumlah besar. Metode ini merupakan pengembangan dari teknik *Classification and Regression Tree* (CART) (Firdaus et al., 2025).

Random Forest terdiri dari sekumpulan pohon keputusan yang membentuk suatu hutan, di mana hasil analisis diperoleh dari kombinasi pohon-pohon tersebut. Selain itu, *Random Forest* dapat memberikan perkiraan mengenai tingkat kepentingan setiap variabel dalam model, sehingga membantu pemahaman data dengan lebih mendalam. Dalam penerapan analisis sentimen, algoritma ini terbukti mampu menghasilkan klasifikasi dengan tingkat akurasi yang tinggi (Basar et al., 2022) Persamaan *Random Forest* melibatkan tiga tahapan utama:

- a) Pembuatan Pohon keputusan
 - Persamaan *Gini Impurity* secara umum sebagai berikut:

$$Gini\ Impurity = 1 - \sum_{i=t}^n (pi)^2 \quad (2.13)$$

Keterangan :

pi = Probabilitas atau proporsi kemunculan kelas i di dalam suatu *node* atau *dataset*.

n = Jumlah kelas (kategori) yang diamati.

Semakin besar nilai *Gini Impurity*, semakin tinggi tingkat ketidakmurnian (*impurity*) pada data.

- Persamaan *Entropy*:

$$\text{Entropy} = - \sum_{i=1}^n p_i \log_2(p_i) \quad (2.14)$$

Keterangan :

p_i = Probabilitas atau proporsi kemunculan kelas i .

\log_2 = Penggunaan logaritma basis 2 menunjukkan satuan informasi yang diukur dalam bit

Semakin besar nilai *Entropy*, semakin tinggi tingkat ketidakpastian atau keragaman kelas pada data tersebut

- b) Pemisahan *node* atau *splittin*

Information Gini =

$$\text{Entropy}(\text{parent}) - \sum_{k=1}^m \left(\frac{N_k}{N} \times \text{Entropy}(\text{Child}_k) \right) \quad (2.15)$$

Keterangan :

parent = Nilai *entropy* sebelum dilakukan pemisahan (*split*).

m = Jumlah cabang (*child nodes*) setelah *split*.

N = Jumlah total data pada *parent* (sebelum *split*).

N_k = Jumlah data pada *child* ke- k

Child_k = Nilai *entropy* pada *child* ke- k .

Semakin besar nilai *Information Gini*, semakin baik pemisahan yang dilakukan karena mengurangi ketidakpastian (*entropy*) lebih banyak.

- c) Kombinasi Prediksi (*Ensembling*)

- Kalasifikasi (*Voting Majoritas*)

$$y = \text{mode}\{h_1(x), h_2(X), \dots, h_B(x)\}, \quad (2.16)$$

Keterangan :

$h_1(x)$ = Prediksi dari model (atau pohon) ke- i .

B = Jumlah total model (pohon) dalam *ensemble*.

Prediksi akhir diambil dari kelas yang paling sering dipilih (majoritas) oleh seluruh model.

- *Regresi* (Rata-Rata Prediksi)

$$y = \frac{1}{B} \sum_{i=1}^B h_i(x) \quad (2.17)$$

Keterangan :

$h_i(x)$ = Prediksi nilai dari model (atau pohon) ke- i .

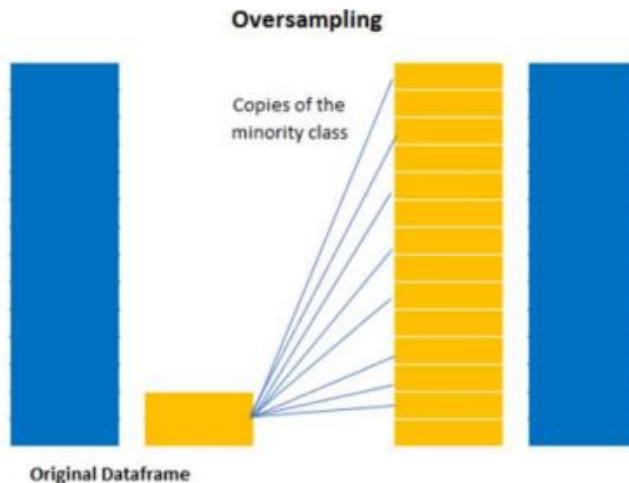
B = Jumlah total model (atau pohon) dalam *ensemble*.

Prediksi akhir diambil dari rata-rata nilai prediksi yang dihasilkan oleh seluruh model.

2.1.7 Penanganan Data Tidak Seimbang (*Oversampling*)

Ketidakseimbangan kelas dalam data merupakan tantangan signifikan dalam *machine learning* dan data *mining*. Kondisi ini terjadi ketika distribusi data tidak merata, di mana jumlah data pada kelas mayoritas jauh melebihi kelas minoritas. Hal ini dapat memicu kesalahan klasifikasi (*misclassification*), karena model cenderung mempelajari pola dari kelas mayoritas dan mengabaikan kelas minoritas. Akibatnya, kinerja model dapat menurun karena data minoritas sering dianggap sebagai anomali atau gangguan (*noise*).

Salah satu metode yang digunakan ketika data tidak seimbang adalah metode *oversampling*. *Oversampling* adalah metode yang digunakan untuk menyamakan jumlah data pada kelas minoritas agar setara dengan jumlah data pada kelas mayoritas. Dalam penelitian ini, digunakan teknik Random *Oversampling* (ROS) sebagai pendekatan *oversampling*. Berikut ini gambar Ilustrasi *oversampling* (Diantika, 2023):



Gambar 2. 1 Ilustrasi *oversampling*

Sumber: Diantika, S. (2023). *JATI (Jurnal Mahasiswa Teknik Informatika)*, 7(1), 19–27.

Pada Gambar 2.1 dapat dilihat bahwa *oversampling* bekerja dengan menambahkan salinan data dari kelas minoritas secara acak ke dalam data pelatihan. Proses ini diulang hingga jumlah data pada kelas minoritas setara dengan kelas mayoritas. Langkah pertama dalam metode ini adalah menghitung selisih jumlah data antara kelas mayoritas dan kelas minoritas. Setelah selisih diketahui, dilakukan penambahan dengan cara memilih data dari kelas minoritas secara acak, kemudian menyalinnya ke dalam dataset pelatihan sebanyak selisih tersebut (Diantika, 2023).

2.1.8 *Confusion Matrix*

Confusion Matrix adalah sebuah metode yang bisa digunakan untuk perhitungan akurasi, *recall*, *precision*, dan *error rate* (Aditya et al., 2022). Rumus untuk *confusion matrix* dapat dilihat pada tabel dibawah ini:

Tabel 2. 1 *Cofusion Matrix*

Klasifikasi	Prediksi	
	Retrieved	Not Retrieved
Retrieved	TP (<i>True Positive</i>)	FN (<i>False Negative</i>)
Not Retrieved	FP (<i>False Positive</i>)	TN (<i>True Negative</i>)

Keterangan:

- TP (*True Positive*) = Jumlah prediksi yang benar dari data yang *relevant*.
 FP (*False Positive*) = Jumlah perprediksi yang salah dari data yang tidak *relevant*.
 FN (*False Negative*) = Jumlah prediksi yang salah dari data yang tidak *relevant*.
 TN (*True Negative*) = Jumlah prediksi yang benar dari data yang *relevant*.

Dalam penelitian ini, Penulis menggunakan *multiclass confusion matrix* 3x3 dikarenakan karena *output sentiment* dari penelitian ini ada 3 *sentiment*, yaitu positif, negatif dan netral sehingga tabel *confusion matrix*-nya adalah sebagai berikut (Khasanah, 2023):

Tabel 2. 2 Rumus *Multiclass Confussion Matrix*

	PREDIKSI		
	POSITIF	NEGATIF	NETRAL
AKTUAL POSITIF	TPos	FPosNeg	FPosNet
AKTUAL NEGATIF	FNegPos	TNeg	FNegNet
AKTUAL NETRAL	FNetPos	FNetNeg	TNet

Keterangan:

- TPos = Jumlah prediksi yang positif dari aktual yang positif.
 FPosNeg = Jumlah prediksi yang negatif dari aktual yang positif.
 FPosNet = Jumlah prediksi yang netral dari aktual yang positif.
 FNegPos = Jumlah prediksi yang positif dari aktual yang negatif.
 TNeg = Jumlah prediksi yang negatif dari aktual yang negatif.
 FNegNet = Jumlah prediksi yang netral dari aktual yang negatif.
 FNegPos = Jumlah prediksi yang positif dari aktual yang netral.
 FNetNeg = Jumlah prediksi yang negatif dari aktual yang netral.
 TNet = Jumlah prediksi yang netral dari aktual yang netral

Melalui *confusion matrix* dapat dilakukan perhitungan berbagai matrik evaluasi kinerja model diantaranya *accuracy*, *precision*, *recall* dan *F1-Score*. Berikut penjelasan dari matrik tersebut (Widyarto et al., 2023):

1) *Accuracy*

Accuracy adalah rasio jumlah total perediksi *valid* (baik prediksi positif yang benar maupun prediksi negatif yang benar) terhadap keseluruhan prediksi yang dilakukan oleh model. Dengan kata lain, *accuracy* mengukur seberapa sering model membuat prediksi yang benar, baik itu untuk kelas positif maupun negatif, dari seluruh data yang diuji, berikut ini rumus persamaan 2.13 yang menujukkan perhitungan *accuracy*:

$$\text{Accuracy} = \frac{TN+TP}{TP+TN+FP+FN} \quad (2.18)$$

2) *Precision*

Precision adalah salah satu matrik evaluasi model pada *supervised learning* yang menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model. *Precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Dari semua kelas positif yang telah di prediksi dengan benar, berapa banyak data yang benar-benar positif. Berikut ini persamaan 2.14 untuk menghitung *precision*:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2.19)$$

3) *Recall*

Recall adalah tingkat keberhasilan sistem dalam menentukan kembali sebuah informasi. Dalam psikologi, *recall* berarti proses mengingat kembali informasi yang telah dipelajari atau dialami sebelumnya. Contohnya adalah ketika seseorang diminta untuk mengingat kembali nama-nama teman sekelas. Berikut ini persamaan 2.15 untuk menghitung *recall*:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (2.20)$$

4) *F1-Score*

Fungsi *F1-Score* adalah memberikan ukuran komprehensif untuk kinerja model klasifikasi dengan mempertimbangkan baik presisi (*precision*) maupun *recall*. *F1-Score* sangat berguna dalam menghadapi ketidakseimbangan jumlah data *training* antara kelas-kelas yang berbeda.

Dalam kasus ketidak seimbangan tersebut, *F1-Score* memberikan evaluasi yang seimbang terhadap kinerja model tanpa terpengaruh oleh dominasi kelas mayoritas atau kelas minoritas. Dengan demikian, *F1-Score* membantu mengatasi bias dan memberikan penilaian yang lebih akurat terhadap model klasifikasi, terlepas dari ketidakseimbangan jumlah data *training* pada setiap kelasnya. Dibawa ini persamaan (2.16) untuk menghitung *F1-Score*:

$$F1\ Score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \quad (2.21)$$

2.2 Penelitian Terkait

Dalam Penelitian ini, penting untuk merujuk pada penelitian sebelumnya sebagai landasan dan wawasan. Hal ini bertujuan untuk memperoleh informasi dan ide-ide yang dapat dikembangkan dalam penelitian ini. Untuk itu, pada penelitian ini telah mengimbau sejumlah referensi dari penelitian-penelitian terdahulu yang relevan dengan permasalahan yang dibahas dalam penelitian ini.

Tabel 2.3 Penelitian Terkait

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
1	Dian Siti Utami dan Adhitia Erfina, 2022	Analisis Sentimen Objek Wisata Bali Di <i>Google Maps</i> Menggunakan Algoritma <i>Naïve Bayes</i>	Hasil akurasi untuk objek wisata Nusa Penida mencapai 94,64%, lebih tinggi dibandingkan objek wisata lainnya. Garuda Wisnu Kencana memiliki akurasi	Penelitian sebelumnya membahas beberapa objek wisata di bali sedangkan penelitian saya berfokus satu objek yaitu wisata Wai Malino.	Persamaanya adalah Sama-sama menggunakan algoritma <i>Naïve Bayes</i> untuk menganalisis sentimen berdasarkan ulasan pengunjung objek wisata.

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
			82,86%, <i>The Edge</i> 80%, Pandawa 90,77, dan Pura Luhur Uluwatu 85,58%.		
2	Edgarsa Bramandyo Widyarto, Jondri, dan Kemas Muslim Lhaksmana, 2023	Implementasi Metode <i>Naïve Bayes Classifier</i> Terhadap Analisis Sentimen Tempat Wista di Nusa Tenggara Barat	Hasil analisis menunjukkan hasil yang baik dengan <i>macro F1-score</i> sebesar 76% (skenario 1), 88% (Skenario 2), dan 82% (skenario 3)	Penelitian sebelumnya hanya menggunakan dua kategori sentimen (positif dan negatif) sedangkan penelitian yang saya lakukan menggunakan tiga kategori sentimen (Positif, negatif dan netral).	Sama-sama menggunakan metode <i>Naïve Bayes</i> untuk klasifikasi sentimen wisata.
3	Nurhaliza Agustina. C.A, Desy Herlina Citra, Wido Purnama, 2022	Implementasi Algoritma <i>Naïve Bayes</i> Untuk Analisis Sentimen Ulasan <i>Shopee</i> pada <i>Google Play</i>	Hasil penelitian dengan teknik <i>Hold Out</i> , akurasi mencapai 83%, lebih baik dari pada <i>10-fold-cross</i>	Perbedaan terletak pada objek yang diteliti Penelitian yang sedang berjalan berfokus pada objek wisata Sedangkan	Keduanya menggunakan Algoritma <i>Naïve Bayes</i> untuk analisis sentimen, dengan tujuan memberikan rekomendasi

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
		<i>Store</i>	validation yang mencapai 82%. Sentimen ulasan yang umumnya positif, dengan <i>precision</i> sebesar 83%, <i>recall</i> 100% dan <i>f1-score</i> 91%.	penelitian sebelumnya berfokus pada ulasan <i>Shopee</i>	berdasarkan ulasan untuk memahami persepsi pengunjung terhadap objek atau layanan.
4	Yogi Harfian, 2021	Klasifikasi Sentimen Aplikasi Digital Dana pada Komentar di Instagram Menggunakan <i>Naïve Bayes Classifier</i>	Hasil pengujian pada penelitian terhadap klasifikasi komentar dalam halaman Instgram menggunakan metode <i>Naïve Bayes Classifier</i> mendapatkan akurasi yang cukup tinggi yaitu sebesar 93,33%.	Bahasa pemrograman yang digunakan pada penelitian yang akan berjalan adalah <i>python</i> sedangkan penelitian sebelumnya menggunakan PHP.	Menggunakan Metode yang sama yaitu <i>Naïve Bayes Classifier</i>
5	Yerik Afrianto Singgalen, 2022	Analisis Sentimen Ulasan Wisatawan terhadap Candi	Hasil penelitian menghasilkan nilai akurasi 96,36%, <i>precision</i>	Penelitian yang akan berjalan menggunakan data kusioner dari	Menggunakan Algoritma <i>Naïve Bayes</i> untuk klasifikasi sentimen

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
		Borobudur Menggunakan Algoritma <i>Naïve Bayes</i>	93,23%, <i>recall</i> 100%, AUC 0,714 kata yang dominan terkait akurasi wisata.	pengunjung langsung sedangkan penelitian terdahulu menggunakan data ulasan dari <i>website Tripadvisor</i> yang di <i>scraping</i> .	
6	Soumya s, Pramod K.V, 2020	Sentimen <i>Analysis of Malayam Tweets using Machine Learning Techniques</i>	Hasil pengklasifikasi an menunjukkan akrasi yang lebih baik dengan fitur RF dan fitur Unigram menggunakan Sentiwordnet yang menyertakan kata-kata negasi mencapai akurasi sebesar 95,6%.	Penelitian yang akan berjalan menggunakan metode <i>Naïve Bayes</i> . Sedangkan peneliti sebelumnya menggunakan metode SVM.	Persamaan penelitian yang akan berjalan dengan penelitian sebelumnya adalah sama ingin mengetahuai analisis sentimen.
7	Moch Saoki Syahlan, Dede Irmayanti dan Syaiful	Analisis Sentimen terhadap Tempat Wisata dari Taman Air	Hasil dari tanggapan masyarakat melalui wisata Taman Air	Penelitian yang akan berjalan menggunakan metode <i>Naïve</i>	Penelitian yang akan berjalan dan penelitian sebelumnya

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
	Alam, 2023	Komentar Pengunjung Dengan Menggunakan Metode <i>Support Vector Machine</i> (SVM) (Studi kasus: Taman Air Mancur Sri Buduga Purwakarta).	Mancur Sri Baduga, Komentar positif 872, negatif 156, dan netral 172. Dengan akurasi 81%, <i>Precision</i> 94% <i>Recall</i> 99%.	<i>Bayes</i> sedangkan penelitian sebelumnya menggunakan metode SVM dan mengambil data di <i>Google Maps</i> .	sama-sama bertujuan untuk analisis sentimen dari komentar pengujung wisata.
8	Raihan Zahran Firdaus, Satrio Hadi Wijoyo dan Welly Purnomo, 2025	Analisis Sentimen Berbasis Aspek Ulasan Pengguna Aplikasi Alfagift Menggunakan Metode <i>Random Forest</i> dan Pemodelan Topik <i>Latent Dirichlet Allocation</i>	Hasil penelitian ini menunjukkan bahwa penggunaan data seimbang menghasilkan akurasi model yang lebih tinggi (89%) dibandingkan dengan data tidak seimbang (84%), menegaskan pentingnya keseimbangan data dalam meningkatkan kinerja model.	Penelitian baru ini berfokus pada analisis sentimen pengunjung membandingkan dua algoritma, <i>Naïve Bayes</i> dan <i>Random Forest</i> . penelitian sebelumnya menganalisis sentimen berbasis aspek ulasan pengguna aplikasi menggunakan	Persamaan dalam tujuan utamanya, yaitu menganalisis sentimen dari data ulasan untuk meningkatkan kualitas layanan.

No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
				<i>Random Forest</i> dan pemodelan topik <i>Latent Dirichlet Allocation</i>	
9	Al Munawaro, Reno Ridhoi, dan Rudiman, 2024	Sentimen Analisis dengan <i>Naïve Bayes</i> Berbasis Orange Terhadap Resiko Pembangunan IKN	Menghasilkan model sentimen terhadap resiko pembangunan IKN dari data media social (Twitter, Instagram, Facebook) dengan akurasi 87% untuk kalasifikasi emosi (<i>joy, surprise, fear</i>) <i>Sentiment negative</i> tertinggi 68% karena kekhawatiran masyarakat tingkat dampak lingkungan.	Penelitian yang akan berjalan menggunakan pengolahan data dengan IDLE <i>python</i> dan pengambilan datanya melalui kiesioner sedangkan penelitian terdahulu menggunakan <i>Orange Teks mining</i> sebagai alat bantu kalasifikasi, dan pengambilan datanya dari media, menggunakan algoritma SVM untuk perbandingan	Sama-sama menggunakan <i>Naïve Bayes</i> untuk analisis sentiment menargetkan opini public terkait isu penting di masyarakat.

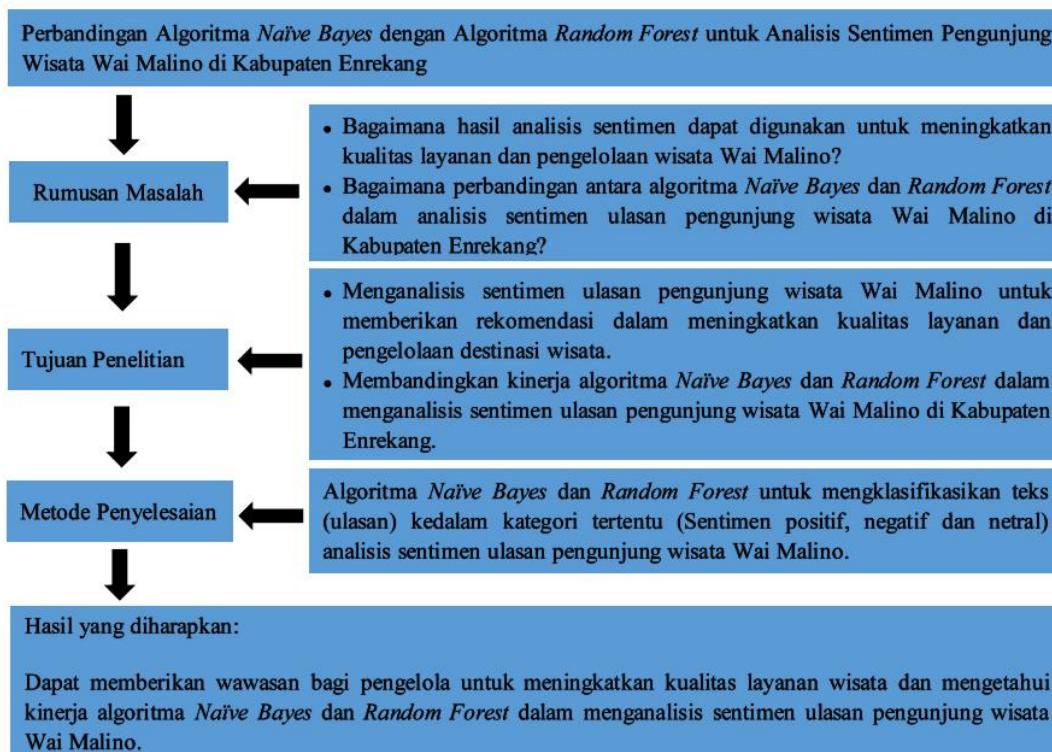
No	Nama/Tahun	Judul	Hasil	Perbedaan	Persamaan
10	Arlan Aditia dan Arief Wibowo, 2022	Analisis Sentimen Mengguanakan Metode <i>Naïve Bayes</i> Berdasarkan Opini Masyarakat Dari Twitter Terhadap Perang Rusia dan Ukraina	Hasil penelitian, menunjukkan bahwa model <i>Naïve Bayes</i> mencapai akurasi 78,261%, yang mengindikasikan bahwa model cukup efektif dalam mengklasifikasi sentimen tweet secara keseluruhan. <i>Precision</i> 93,75% berarti bahwa tweet yang terkласifikasi sebagai negatif memang benar-benar mengundang sentimen negatif.	Perbedaan nya berada pada objek yang diteliti dan juga cara pengambilan datanya yang berbeda serta hasil sentimennya.	Sama-sama menggunakan metode <i>Naïve Bayes</i> untuk analisis sentimen, melalui tahapan seperti <i>pre-processing</i> dan pembobotan kata.

2.3 Kerangka Pikir

Kerangka berpikir, yang juga dikenal sebagai kerangka konseptual, adalah model yang menggambarkan hubungan antara teori dan berbagai faktor yang telah diidentifikasi sebagai isu penting. Kerangka berpikir ini juga berfungsi untuk memberikan penjelasan sementara terhadap fenomena yang menjadi objek penelitian. Alur berpikir dalam kerangka ini disusun berdasarkan teori-teori

sebelumnya serta pengalaman empiris yang menjadi landasan dalam perancangan kerangka berpikir, sehingga bermanfaat untuk mendukung penelitian.

Selain itu, kerangka berpikir mempermudah penyusunan hipotesis penelitian dan memberikan arah yang jelas dalam proses analisis data. Dengan demikian, kerangka berpikir menjadi fondasi penting dalam memastikan bahwa penelitian berjalan secara sistematis. Penyusunan kerangka berpikir yang baik akan membantu peneliti menghindari bias serta menjaga konsistensi antara tujuan penelitian dan metode yang digunakan. Selain itu, kerangka ini juga memberikan gambaran kepada pembaca mengenai arah dan fokus penelitian, sehingga mudah dipahami serta dapat dijadikan acuan dalam pengambilan keputusan selama proses penelitian berlangsung. Berikut ini adalah gambar kerangka berpikir penelitian:



Gambar 2. 2 Kerangka Fikir

Penjelasan dari gambar 2.2 ialah, judul penelitian Perbandingan Algoritma *Naïve Bayes* dengan Algoritma *Random Forest* untuk Analisis Sentimen Pengunjung Wisata Wai Malino di Kabupaten Enrekang. Rumusan masalah berfokus pada dua pertanyaan utama: (1) Bagaimana hasil analisis sentimen dapat digunakan untuk meningkatkan kualitas layanan dan pengelolaan wisata Wai

Malino? (2) Bagaimana perbandingan antara algoritma *Naïve Bayes* dan *Random Forest* dalam menganalisis sentimen ulasan pengunjung wisata wai malino. Tujuan penelitian adalah untuk (1) Menganalisis sentimen ulasan pengunjung wisata Wai Malino untuk memberikan rekomendasi dalam meningkatkan kualitas layanan dan pengelolaan destinasi wisata. (2) Membandingkan kinerja algoritma *Naïve Bayes* dan *Random Forest* dalam menganalisis sentimen pengunjung ke dalam kategori positif, netral, dan negatif. Metode penyelesaian menggunakan algoritma *Naïve Bayes* dan *Random Forest* untuk memproses dan mengelompokkan ulasan pengunjung berdasarkan sentimen yang diekspresikan. Hasil yang di harapkan dari penelitian ini adalah memberikan wawasan berharga bagi pengelola wisata dalam meningkatkan kualitas layanan berdasarkan analisis sentimen pengunjung.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian ini menggunakan pendekatan kuantitatif dengan metode eksperimen untuk mengetahui hasil penerapan algoritma *Naïve Bayes* dan *Random Forest* dalam analisis sentimen pengunjung wisata Wai Malino. Pendekatan ini bertujuan untuk mengukur efektivitas algoritma *Naïve Bayes* dan *Random Forest* berdasarkan kriteria yang terukur secara objektif, seperti akurasi, presisi, *recall*, dan *F1-score*. Metode eksperimen dilakukan dengan menerapkan algoritma *Naïve Bayes* pada dataset ulasan pengunjung yang dikumpulkan melalui kuesioner. *Dataset* berisi ulasan teks untuk mewakili sentimen positif, negatif, dan netral.

Dalam pelaksanaan eksperimen ini, digunakan data ulasan pengunjung wisata Wai Malino melalui kuesioner yang kemudian diproses menggunakan teknik *Tokenizing*, *Stopword Removal*. Data ulasan ini digunakan untuk mengklasifikasikan sentimen menggunakan algoritma *Naïve Bayes* dan *Random Forest*. Evaluasi hasil dilakukan dengan menggunakan *Cofusion Matrix* untuk mengukur akurasi dan efektifitasnya algoritma dalam menganalisis sentimen pengunjung (Kaka et al., 2023).

3.2 Tempat dan Jawal Penelitian

Penelitian ini dilakukan di Kabupaten Enrekang, Provinsi Sulawesi selatan. Fokus penelitian adalah pada objek wisata Wai Malino yang terletak di Kabupaten Enrekang. Penelitian ini dilakukan berdasarkan lama waktu dimulainya penelitian ini dari usulan penelitian sampai dengan perampungan hasil penelitian pada 18 Februari 2025.

3.3 Tahapan Penelitian

Untuk membuat penelitian terstruktur, peneliti membuat sistem alur kerja. Berikut adalah alur kerja yang digunakan dalam penelitian ini.



Gambar 3.1 Alur kerja Penelitian

Alur kerja penelitian ini menggambarkan tahapan-tahapan sistematis yang dilakukan dalam menganalisis sentimen pengunjung wisata Wai Malino menggunakan algoritma *Naïve Bayes* dan *Random Forest*. Berikut ini penjelasan rinci menganai setiap tahapan pada gambar 3.1:

a. Identifikasi Masalah dan Studi Literatur

Langkah awal dalam proses penelitian ini adalah mengidentifikasi masalah yang ingin diteliti, yaitu untuk menganalisis sentimen pengunjung terhadap objek wisata Wai Malino di Kabupaten Enrekang. Kemudian dilakukan studi literatur untuk menggali teori-teori yang relevan dalam analisis sentimen menganai penerapan algoritma *Naïve Bayes* dan *Random Forest* dalam analisis teks.

b. Pengumpulan data

Pada tahapan ini, peneliti mengumpulkan data yang diperlukan untuk penelitian. Data tersebut diperoleh melalui pengisian kuesioner yang disebarluaskan kepada pengunjung objek wisata Wai Malino. Kuesioner ini berisi

pertanyaan yang berkaitan dengan pengalaman pengunjung terhadap objek wisata. Responden diminta untuk memberikan opini mereka dalam bentuk komentar yang kemudian akan di analisis untuk mengetahui sentimen yang terkandung di dalamnya apakah positif, negatif, atau netral. Mencakup beberapa variable utama yaitu Nama Pengunjung sebagai identifikasi responden, Komentar/Opini yang berisi pengalaman dan pendapat pengunjung, Alamat untuk mengetahui lokasi asal pengunjung serta informasi tambahan apakah pengunjung sudah pernah atau belum mengunjungi objek wisata tersebut. Data ini akan dianalisis lebih lanjut menggunakan metode *Naïve Bayes* dan *Random Forest* untuk melakukan klasifikasi sentimen pengunjung objek wisata Wai Malino.

c. Perancangan Program

Pada tahapan ini, penulis melakukan perancangan program dan implementasi dalam bentuk coding dengan menggunakan bahasa pemrograman *Python* untuk mengimplementasikan algoritma *Naïve Bayes* dan *Random Forest* dalam analisis sentimen pengunjung wisata Wai Malino di Kabupaten Enrekang. Perancangan program yang optimal dengan menggunakan *Naïve Bayes* dan *Random Forest* dapat membantu peneliti untuk memperoleh hasil analisis sentimen yang efisien dan memberikan hasil yang baik, penjelasan lebih lengkap dapat dilihat pada 3.3.

d. Implementasi

Pada tahapan ini, implementasi program untuk penelitian analisis sentimen pengunjung wisata Wai Malino menggunakan algoritma *Naïve Bayes* dan *Random Forest* yang melibatkan penerapan modul-modul khusus untuk algoritma, inisialisasi variabel-variabel, dan akses ke data dari sumber eksternal.

e. Pengujian dan Evaluasi

Pada tahapan ini, pengujian dilakukan untuk menilai kinerja algoritma *Naïve Bayes* dan *Random Forest* dalam mengklasifikasikan sentimen pengunjung. Evaluasi dilakukan dengan menggunakan *matrix evaluasi* yang umum digunakan dalam klasifikasi teks, seperti akurasi, *precision*, *recall* dan *F1-score*. *Matrix* ini membantu peneliti untuk mengatahui seberapa baik

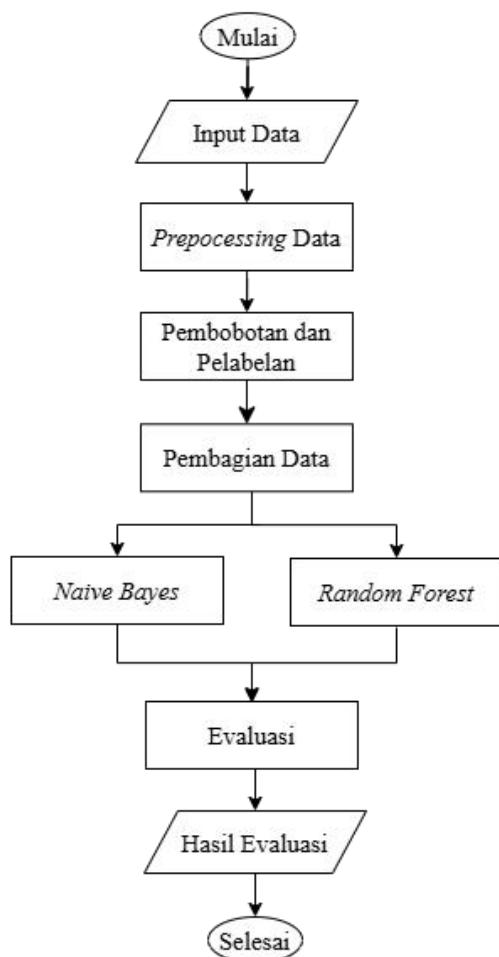
model *Naïve Bayes* dan *Random Forest* dalam mengklasifikasikan komentar sesuai dengan kategori yang di inginkan.

f. Pembuatan Laporan

Pada tahapan pembuatan laporan, peneliti menyusun laporan akhir yang mencakup seluruh tahapan penelitian mulai dari identifikasi masalah dan studi literatur, pengumpulan data, perancangan program, implemantasi, pengujian hingga evaluasi. Laporan ini akan memberikan pemahaman yang komprehensif menganai hasil penelitian serta kontribusinya terhadap bidang analisis sentimen dan pariwisata.

3.4 Perancangan Program

Alur perancangan sistem digambarkan dalam *flowchart* berikut ini:



Gambar 3. 2 *Flowchart* Penggambaran Sistem

Tahapan peracangan sistem ini menggambarkan langka-langkah penting yang dilakukan dalam menganalisis sentimen pengunjung wisata Wai Malino

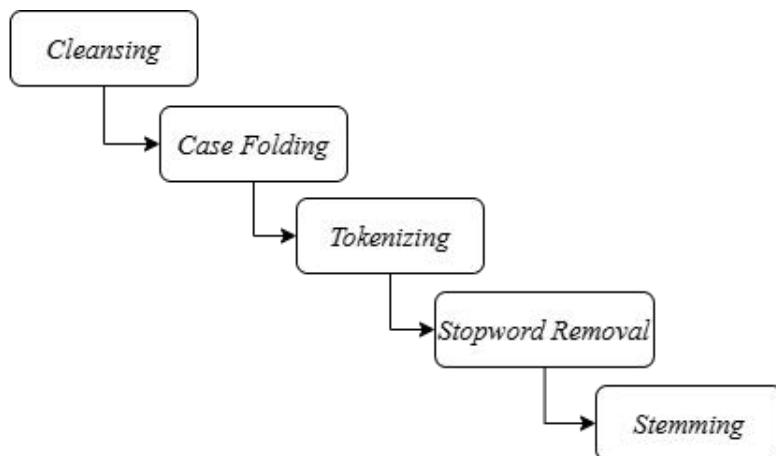
menggunakan algoritma *Naïve Bayes* dan *Random Forest*. Setiap tahapan memiliki peranan penting dalam mempersiapkan data, membangun model, mengevaluasi kinerja, dan menyajikan hasil akhir dari analisis sentimen ini. Penjelasan rinci mengenai setiap tahapan pada gambar 3.2:

a. Input Data

Pada tahapan awal, program melakukan input data yang akan digunakan untuk analisis. Proses *input* data ini melibatkan pengambilan informasi dari file yang berisi hasil kuesioner pengunjung wisata Wai Malino dalam format CSV/XLSX. *Dataset* yang digunakan berisi data sentimen pengunjung terkait pengalaman wisata mereka. Data yang telah diambil akan diakses oleh program dan disimpan dalam variabel yang sesuai. Data kemudian dapat digunakan berbagai keperluan seperti analisis sentimen, perhitungan statistik, atau proses lebih lanjut dalam program.

b. *Preprocessing* Data

Pada tahapan ini data mentah yang diperoleh dilakukan pemrosesan untuk mengubahnya menjadi data yang siap digunakan:



Gambar 3. 3 Tahapan *Preprocessing* Data

Adapun proses yang dilakukan pada tahapan ini adalah membersikan dokumen dari kata yang tidak di perlukan untuk mengurangi *noise*, menyeragamkan bentuk huruf serta penghapusan angka dan tanda baca, mengubah dari kalimat menjadi kata-kata, melakukan proses pemilihan kata yang merupakan kata penghubung dan mengambil kata dasar dari sebuah kata yang memiliki imbuhan. Berikut ini penjelasan dari setiap proses :

1) *Cleansing*

Cleansing yaitu proses membersikan dokumen dari kata yang tidak diperlukan untuk mengurangi *nois*, Kata yang dihilangkan adalah karakter HTML, kata kunci, ikon emosi, dan tanda baca lainnya.

2) *Case Folding*

Case folding yaitu penyeragaman bentuk huruf kapital ke huruf kecil serta normalisasi kata. Dalam hal ini yang digunakan hanya huruf latin antara a sampai dengan z.

3) *Tokenizing*

Tokenizing yaitu proses pengubahan dari kalimat menjadi kata-kata.

4) *Stopword removal*

Stopword removal merupakan proses pemilihan kata penghubung seperti “pada”, “dan”, “yang”, dan lain sebagainya.

5) *Stemming*

Stemming adalah untuk menghapus imbuhan kata, sehingga hanya sisanya yang tetap, yang disebut “*stems*” dan “*roots*”.

Tabel 3. 1 *Preprocessing* Data

No	Proses	Sebelum	Sesudah
1	<i>Case Folding</i>	Tempat yang indah dan sangat nyaman untuk bersantai bersama keluarga!	tempat yang indah dan sangat nyaman untuk bersantai bersama keluarga!
2	<i>Cleansing</i>	tempat yang indah dan sangat nyaman untuk bersantai bersama keluarga!	tempat yang indah dan sangat nyaman untuk bersantai bersama keluarga
3	<i>Tokenizing</i>	tempat yang indah dan sangat nyaman untuk bersantai bersama keluarga	["tempat", "yang", "indah", "dan", "sangat", "nyaman", "untuk", "bersantai", "bersama", "keluarga"]

No	Proses	Sebelum	Sesudah
			"bersama", "keluarga"]
4	<i>Stopword Removal</i>	["tempat", "yang", "indah", "dan", "sangat", "nyaman", "untuk", "bersantai", "bersama", "keluarga"]	["tempat", "indah", "nyaman", "bersantai", "keluarga"]
5	<i>Stemming</i>	["tempat", "indah", "nyaman", "bersantai", "keluarga"]	["tempat", "indah", "nyaman", "santai", "keluarga"]

c. Pembobotan dan Pelabelan

Pembobotan dan pelabelan merupakan salah satu tahapan penting dalam analisis sentimen karena bertujuan untuk memberikan bobot pada kata-kata dalam dataset dengan menggunakan pembobotan TF-IDF dan menetapkan label sentimen, seperti positif, negatif dan netral. Proses pelabelan dilakukan dengan memanfaatkan kamus *lexicon based*. Pendekatan ini data teks berupa kalimat berdasarkan kata-kata yang terdapat dalam kamus *lexicon*, yang berisi daftar kata positif dan negatif beserta nilainya. Setiap kata dalam kalimat yang cocok dengan kamus diberikan nilai sesuai, dan nilai total dihitung dari akumulasi semua kata tersebut. Proses ini memudahkan pengklasifikasian teks ke dalam kategori sentimen tertentu, seperti positif, negatif, atau netral.

d. Pembagian Data

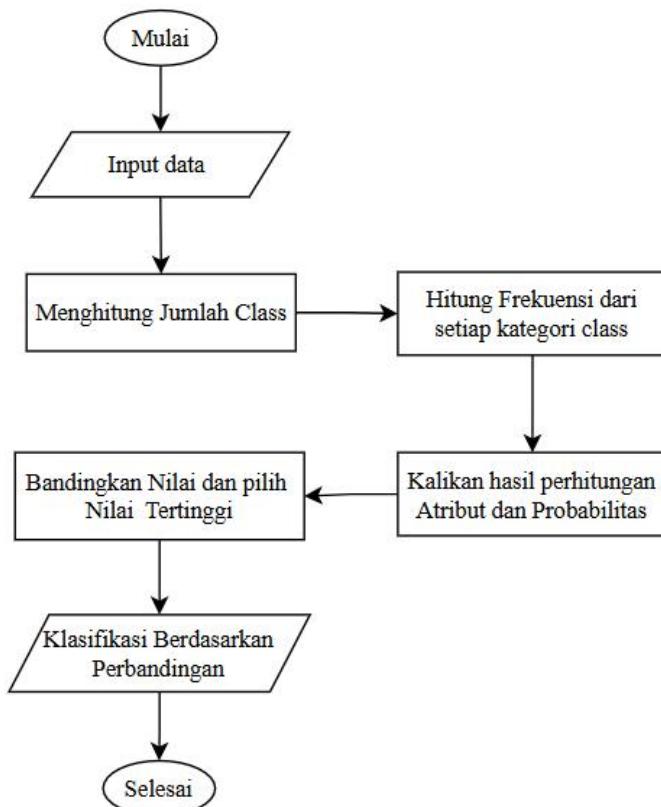
Pada tahapan ini dilakukan pembagian data *training* dan data *testing* untuk membuat model dan menguji model algoritma *Naïve Bayes* dan *Random Forest*. Data dibagi jadi populasi tertentu seperti 80% untuk *training* dan 20% untuk *testing*.

e. Tahapan Pelatihan Algoritma

Berikut ini penjelasan mengenai tahapan algoritma *Naïve Bayes* dan *Random Forest*:

1) Algoritma *Naïve Bayes*

Pada tahapan ini akan dilakukan tahapan latih terhadap opini masyarakat yang sudah di ketahui kasifikasinya sehingga di dapatkan model probabilitas algoritma *Naïve Bayes* yang nantinya akan digunakan pada tahap uji. Berikut ini *flowchart* algoritma *Naïve Bayes*:



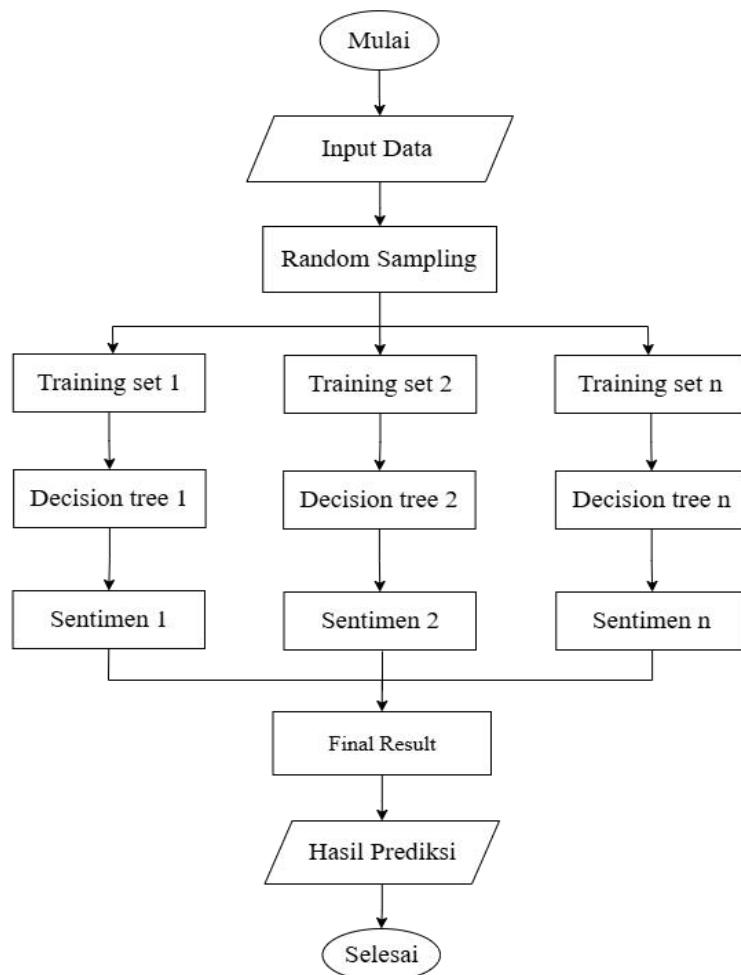
Gambar 3. 4 *Flowchart* Algoritma *Naïve Bayes*

Penjelasan pada gambar *flowchart* diatas yaitu proses klasifikasi dimulai dengan tahap persiapan data, dimana kita memasukkan data *training* dan *testing* sebagai bahan pembelajaran sistem. Setelah data siap, langkah berikutnya adalah melakukan perhitungan jumlah class yang ada, dilanjutkan dengan menghitung frekuensi kemunculan dari setiap kategori *class* tersebut. Proses kemudian berlanjut ke tahapan perhitungan yang lebih mendalam, dimana semua hasil perhitungan atribut dikalikan dengan probabilitasnya masing-masing. Atribut adalah ciri atau informasi yang dimiliki oleh suatu data yang digunakan dalam proses analisis atau klasifikasi. Dimana hasil perhitungan nilai fitur atau atribut merujuk pada probabilitas kemunculan nilai atribut tertentu dalam sebuah kelas. Probabilitas adalah peluang atau gambaran awal seberapa besar kemungkinan data termasuk ke dalam suatu

kelas. Hasil perhitungan ini kemudian dibandingkan untuk menentukan nilai probabilitas tertinggi. Pada tahapan akhir, sistem melakukan klasifikasi berdasarkan hasil perbandingan yang telah dilakukan. Sehingga dapat menentukan kategori yang paling sesuai untuk data yang dianalisis. Keseluruhan proses ini merupakan langkah-langkah sistematis dalam melakukan klasifikasi data menggunakan pendekatan probabilitas.

2) Algoritma *Random Forest*

Pada tahapan ini, dilakukan proses pelatihan menggunakan data yang telah diklasifikasikan sebelumnya untuk membangun model *Random Forest*. Model ini terdiri dari beberapa pohon keputusan yang bekerja secara bersamaan untuk meningkatkan akurasi prediksi. Setelah model terbentuk, model ini akan digunakan pada tahap pengujian. Berikut adalah *flowchart* yang menggambarkan alur kerja algoritma *Random Forest*:



Gambar 3. 5 *Flowchart* Algoritma *Random Forest*

Flowchart di atas menggambarkan proses klasifikasi sentimen menggunakan teknik *ensemble learning*, khususnya *Random Forest*. *Ensemble Learning* adalah teknik *machine learning* yang menggabungkan beberapa model untuk menghasilkan prediksi yang lebih akurat dari pada model tunggal. Proses dimulai dengan langkah *Input Data*, di mana data mentah dimasukkan ke dalam sistem. Selanjutnya, dilakukan *Random Sampling*, yaitu pembagian data ke dalam beberapa subset secara acak untuk membentuk *Training set 1*, *Training set 2*, hingga *Training set n*. Setiap *subset* data ini digunakan untuk membangun *Decision Tree 1*, *Decision Tree 2*, hingga *Decision Tree n*, yang merupakan model pohon keputusan yang dilatih secara independen. Setiap pohon keputusan kemudian menghasilkan prediksi sentimen (Sentimen 1, Sentimen 2, hingga Sentimen n), yang bisa berupa kategori seperti positif, negatif, atau netral. Setelah itu, dilakukan proses *Final Result*, yaitu penggabungan hasil dari seluruh pohon keputusan menggunakan teknik voting mayoritas untuk menentukan hasil akhir yang paling akurat. Akhirnya, hasil klasifikasi ditampilkan dalam bentuk Hasil Prediksi, yang menjadi *output* akhir dari sistem sebelum proses selesai.

f. Evaluasi Performa Klasifikasi

Evaluasi Kinerja algoritma adalah proses untuk mengukur seberapa baik atau buruk kinerja suatu algoritma atau model dalam menyelesaikan tugas tertentu. Tujuan dari evaluasi ini adalah untuk menilai evektivitas model klasifikasi, termasuk menghitung berbagai matrik seperti *accuracy*, *precision*, *recall* dan *F1-Score*. Dalam penelitian ini, algoritma *Naïve Bayes* dan *Random Forest* digunakan untuk analisis sentimen, dimana data diklasifikasikan ke dalam tiga kategori yang telah ditentukan sejak awal. Metode *Confusion Matrix*, dilakukan untuk mengevaluasi hasil prediksi model dengan membandingkannya terhadap nilai sebenarnya. Dari *Confusion Matrix*, beberapa matrik evaluasi dapat dihitung. *Accuracy* mengukur seberapa sering model membuat prediksi yang benar terhadap keseluruhan data. *Precision* menunjukkan seberapa akurat model dalam mengklasifikasikan suatu kelas. Semakin tinggi nilainya, semakin sedikit

kesalahan klasifikasi ke kelas tersebut. *Recall* mengukur seberapa baik model dalam menemukan semua data yang benar, *Recall* tinggi berarti model berhasil menemukan sebagian besar data yang benar dalam suatu kelas. sedangkan *F1-Score*, yang merupakan kombinasi harmonis antara *precision* dan *recall*, sangat berguna dalam situasi dimana data tidak seimbang.

g. Hasil kinerja

Pada tahapan ini akan dilakukan pembahasan dari semua hasil percobaan model yang telah di lakukan pada proses klasifikasi dengan menggunakan algoritma *Naïve Bayes* dan *Random Forest*. Pada tahapan ini akan dilakukan perancangan dan imlementasi visualisasi. *Output* utama dari penelitian ini adalah klasifikasi sentimen ulasan pengunjung wisata Wai Malino, yang dikategorikan ke dalam tiga kelas utama:

- 1) Positif, Komentar yang mencerminkan pengalaman atau pendapat yang baik tentang objek wisata.
- 2) Negatif, komentar yang menunjukkan ketidakpuasan atau kritikan terhadap objek wisata.
- 3) Netral, komentar yang tidak menunjukkan pandangan yang jelas, baik positif maupun negatif, atau memberikan informasi yang lebih objektif.

BAB IV

HASIL DAN PEMBAHASAN

Dalam bab ini akan menjelaskan hasil dan pembahasan tentang penelitian yang telah dilakukan berdasarkan tahapan-tahapan yang telah dilakukan dalam penelitian.

4.1 Implementasi

Pada penelitian ini dilakukan analisis yang dimulai dari tahapan pengumpulan data yang dikumpulkan melalui kuesioner, proses *preprocessing*, melakukan pelabelan, menghitung bobot perkata, selanjutnya pengklasifikasian sehingga mendapatkan *output* dalam penelitian ini. Penelitian analisis sentimen positif, negatif dan netral komentar wisata waimalino di implementasikan menggunakan Google Colab dengan spesifik *Processor 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz, 2995 Mhz, 2 Core(s), 4 Logical Processor(s)* dan *Installed Physical Memory (RAM) 8.00 GB* dengan *OS Name Microsoft Windows 11 Home Single Language*.

4.2 Pengumpulan Data

Pada penelitian ini, data dikumpulkan dalam bentuk komentar atau opini mengenai wisata Wai Malino. Data yang dikumpulkan berasal dari kuesioner yang dibagikan secara online dan survei langsung kepada pengunjung. Kuesioner dibagikan secara terbuka, sehingga responden yang mengisi berasal dari berbagai kalangan. Namun, dalam proses pengolahan data, hanya tanggapan dari responden yang benar-benar pernah berkunjung ke wisata tersebut yang digunakan dalam analisis. Pengambilan data dilakukan selama kurang lebih 2 bulan mulai dari bulan Desember 2024 - Januari 2025, dengan total data yang terkumpul sebanyak 971 tanggapan. Untuk memberikan gambaran lebih jelas mengenai data yang telah dikumpulkan, berikut tabel sampel data yang berisi beberapa komentar/opini dari responden:

Tabel 4. 1 Sampel data yang dikumpulkan

No	Nama	Alamat	Pernah berkunjung?	Kepuasan	Komentar	Dokumentasi
1	Rais	Parombea n	Ya	Puas	Sangat cocok dikunjungi bersama teman dan keluarga	
2	Yusril	Enrekang	Ya	Puas	Pemandangan di sini luar biasa! Udara segar dan suasannya menenangkan	
3	Nahda	Liba	Ya	Puas	Luar biasa	
4	Jihan	Le'tobara	Ya	Netral	Kebersihan dan Ke Indahan Alamnya Sangat Terjaga	

Dalam penelitian ini, pengolahan dan pembersihan data dilakukan untuk memastikan bahwa data yang digunakan dalam analisis memiliki kualitas yang baik dan bebas dari kesalahan. Data yang belum diproses sering kali mengandung informasi yang tidak relevan, duplikasi, atau bahkan nilai kosong yang dapat memengaruhi hasil analisis. Oleh karena itu, dilakukan serangkaian tahapan pembersihan untuk menyaring dan menyiapkan data agar lebih akurat serta siap digunakan dalam proses lebih lanjut. Berikut adalah tahapan pengolahan dan pembersihan data yang dilakukan:

- a) Dataset awal dibaca menggunakan pandas untuk melihat jumlah data yang tersedia. Data mentah yang dikumpulkan berjumlah 971 data.

- b) Penyaringan data berdasarkan responden yang pernah berkunjung, Responden yang menjawab "tidak" pada kolom "Pernah Berkunjung?" dihapus. Setelah menyaring hanya responden yang pernah berkunjung, jumlah data berkurang menjadi 955.
- c) Pemilihan kolom relevan, dari *dataset* yang telah difilter hanya kolom Timestamp, Nama, Alamat, Kepuasan? dan Komentar yang dipilih untuk keperluan analisis lebih lanjut.
- d) Data diperiksa untuk menghindari adanya entri ganda. Duplikasi berdasarkan kolom "Nama", "Alamat" Kepuasan? dan Komentar dihapus agar setiap responden hanya memberikan opini unik. Setelah proses ini, jumlah data yang semula 955 berkurang menjadi 940 data.
- e) Dilakukan pengecekan nilai kosong (*NaN*) dalam *dataset*, namun hasilnya menunjukkan tidak ada nilai kosong pada semua kolom, sehingga jumlah data tetap 940.
- f) *Dataset* dibatasi hanya kolom Komentar yang dipertahankan untuk proses *preprocessing* lebih lanjut, seperti pembersihan teks, tokenisasi, dan analisis sentimen. Berikut ini tabel 4.2 data yang sudah siap diolah sebanyak 940 sampel:

Tabel 4. 2 Komentar pengunjung wisata

No	Komentar
1	Sangat cocok dikunjungi bersama teman dan keluarga
2	Pemandangan di sini luar biasa! Udara segar dan suasannya menenangkan
3	Tempatnya bersih tetapi masih banyak yang masih butuh perbaikan seperti jalanan menuju ke wisata masih sangat rusak
4	Wisata Wai malino tempat yg bersih dan sejuk,,serta nyaman karna tempatnya yg lumayan jauh dari keramaian.

4.3 Teks *Preprocessing*

Tujuan dilakukan *Preprocessing* untuk membersihkan dan mempersiapkan data teks sebelum dilakukan pemodelan. Pada penelitian ini, teks *preprocessing*

dilakukan untuk meningkatkan kualitas data dengan menghilangkan elemen-elemen yang tidak relevan serta menyederhanakan struktur teks agar lebih mudah diproses oleh algoritma klasifikasi.

Proses ini mencakup beberapa langkah utama, seperti *cleansing*, *case folding*, *tokenizing*, *stopword removal*, dan *stemming*:

a) *Cleansing* Data

Cleansing data atau pembersihan data adalah tahap awal dalam analisis sentimen yang bertujuan untuk menghapus atau memperbaiki kesalahan, duplikasi, serta data yang tidak relevan dari dataset yang telah dikumpulkan. Proses ini dilakukan dengan membersihkan dokumen dari kata-kata yang tidak diperlukan guna mengurangi noise. Elemen yang dihilangkan dalam tahap ini meliputi karakter HTML, kata kunci yang tidak relevan, ikon emosi (*emoticon*), serta tanda baca lainnya yang dapat mengganggu proses analisis. Hasil dari *cleansing* data dapat di lihat ada tabel 4.3:

Tabel 4. 3 *Cleansing*

Komentar	<i>Cleansing</i>
Pemandangan di sini luar biasa! Udara segar dan suasannya menenangkan	Pemandangan di sini luar biasa Udara segar dan suasannya menenangkan
Wisata Wai malino menawarkan keindahan yg sangat menakjubkan dengan air yg jernih dan indahnya pepohonan di sekelilingnya namun fasilitas yg tersedia masih terbatas.	Wisata Wai malino menawarkan keindahan yg sangat menakjubkan dengan air yg jernih dan indahnya pepohonan di sekelilingnya namun fasilitas yg tersedia masih terbatas
Wai Malino wisata yg nyaman buat liburan Keluarga/Sahabat??karna memiliki pemandangan yg indah serta berada ditempat yg sejuk dan segar??..+Wisata tempatnya Paling yg Recomend buat camp karna lumayan jauh dari suara2 Berisik????	Wai Malino wisata yg nyaman buat liburan KeluargaSahabatkarna memiliki pemandangan yg indah serta berada ditempat yg sejuk dan segarWisata tempatnya Paling yg Rekomendasi buat camp karna lumayan jauh dari suara Berisik

b) *Case Folding*

Pada tahap *case folding* dilakukan proses *preprocessing* data dengan tujuan untuk menyeragamkan semua huruf menjadi huruf kecil. Hal ini bertujuan agar kata yang sama namun memiliki perbedaan pada penggunaan huruf kapital, seperti "Data" dan "data", dapat dianggap sama oleh sistem dan tidak menimbulkan perbedaan arti. Selain itu, pada tahap ini juga dilakukan proses normalisasi kata, yaitu proses memperbaiki atau mengubah kata-kata tidak baku atau singkatan menjadi bentuk baku sesuai Kamus Besar Bahasa Indonesia (KBBI). Normalisasi bertujuan agar data menjadi lebih konsisten, lebih mudah diproses, dan mengurangi ketidakjelasan makna dalam tahap analisis berikutnya. Contohnya, kata-kata singkatan seperti "yg" diubah menjadi "yang", "utk" menjadi "untuk", dan "tdk" menjadi "tidak". Hasil dari proses *case folding* dan normalisasi kata ini dapat dilihat pada Tabel 4.4:

Tabel 4. 4 *Case Colding* dan Normalisasi Kata

Cleansing	Case Folding
Pemandangan di sini luar biasa Udara segar dan suasannya menenangkan	pemandangan di sini luar biasa udara segar dan suasannya menenangkan
Wisata Wai malino menawarkan keindahan yg sangat menakjubkan dengan air yg jernih dan indahnya pepohonan di sekelilingnya namun fasilitas yg tersedia masih terbatas	wisata wai malino menawarkan keindahan yang sangat menakjukkan dengan air yang jernih dan indahnya pepohonan di sekelilingnya namun fasilitas yang tersedia masih terbatas
Wai Malino wisata yg nyaman buat liburan KeluargaSahabatkarna memiliki pemandangan yg indah serta berada ditempat yg sejuk dan segarWisata tempatnya Paling yg Rekomendasi buat camp karna lumayan jauh dari suara Berisik	wai malino wisata yang nyaman buat liburan keluarga, sahabat, karena memiliki pemandangan yang indah serta berada ditempat yang sejuk dan segar wisata tempatnya paling yang rekomendasi buat kemping karena lumayan jauh dari suara berisik

c) *Tokenizing*

Tokenizing yaitu proses untuk memecah kalimat yang terdapat dalam dokumen *text* dengan cara memisahkan kata berdasarkan kata per-spasi. Potongan kata yang sudah terpisahkan disebut sebagai token. Pemisah kata ini akan dipisahkan dengan tanda koma (,) untuk setiap katanya. Hasil *tokenizing* dapat dilihat pada tabel 4.5:

Tabel 4. 5 *Tokenizing*

<i>Case Folding</i>	<i>Tokenizing</i>
pemandangan di sini luar biasa udara segar dan suasananya menenangkan	['pemandangan', 'di', 'sini', 'luar', 'biasa', 'udara', 'segar', 'dan', 'suasananya', 'menenangkan']
wisata wai malino menawarkan keindahan yang sangat menakjukkan dengan air yang jernih dan indahnya pepohonan di sekelilingnya namun fasilitas yang tersedia masih terbatas	['wisata', 'wai', 'malino', 'menawarkan', 'keindahan', 'yang', 'sangat', 'menakjukkan', 'dengan', 'air', 'yang', 'jernih', 'dan', 'indahnya', 'pepohonan', 'di', 'sekelilingnya', 'namun', 'fasilitas', 'yang', 'tersedia', 'masih', 'terbatas']
wai malino wisata yang nyaman buat liburan keluarga, sahabat, karena memiliki pemandangan yang indah serta berada ditempat yang sejuk dan segar wisata tempatnya paling yang rekomendasi buat kemping karena lumayan jauh dari suara berisik	['wai', 'malino', 'wisata', 'yang', 'nyaman', 'buat', 'liburan', 'keluarga,', 'sahabat,', 'karena', 'memiliki', 'pemandangan', 'yang', 'indah', 'serta', 'berada', 'ditempat', 'yang', 'sejuk', 'dan', 'segar', 'wisata', 'tempatnya', 'paling', 'yang', 'rekomendasi', 'buat', 'kemping', 'karena', 'lumayan', 'jauh', 'dari', 'suara', 'berisik']

d) *Stopword Removal*

Stopword Removal ialah proses membuang kata yang tidak memiliki arti atau pengaruh terhadap analisis sentimen namun seringkali muncul dalam dokumen. Seperti kata penghubung “pada”, “dan”, “yang”, dan lain sebagainya. *Stopword* cenderung memperbesar ukuran data teks tanpa memberikan kontribusi

berarti dalam penentuan makna atau emosi dalam kalimat. Oleh karena itu, proses ini penting untuk dilakukan agar data yang akan dianalisis menjadi lebih bersih, relevan, dan efisien dalam hal pemrosesan. Dalam konteks ini, proses penghapusan *stopword* dilakukan menggunakan *library* Sastrawi, sebuah pustaka pemrosesan bahasa alami yang dirancang khusus untuk Bahasa Indonesia. Sastrawi menyediakan fitur *StopWordRemoverFactory* yang dapat digunakan untuk menghapus kata-kata umum dari sebuah teks. Dengan menggunakan fungsi ini, kata-kata yang tidak penting akan dihilangkan sehingga hanya tersisa kata-kata kunci yang benar-benar dibutuhkan untuk proses analisis lebih lanjut seperti *stemming*. Contoh hasil *stopword removal* dapat dilihat pada tabel 4.6:

Tabel 4. 6 *Stopword Removal*

Tokenizing	Stopword Removal
['pemandangan', 'di', 'sini', 'luar', 'biasa', 'udara', 'segar', 'dan', 'suasananya', 'menenangkan']	['pemandangan', 'sini', 'luar', 'biasa', 'udara', 'segar', 'suasananya', 'menenangkan']
['wisata', 'wai', 'malino', 'menawarkan', 'keindahan', 'yang', 'sangat', 'menakjukkan', 'dengan', 'air', 'yang', 'jernih', 'dan', 'indahnya', 'pepohonan', 'di', 'sekelilingnya', 'namun', 'fasilitas', 'yang', 'tersedia', 'masih', 'terbatas']	['wisata', 'wai', 'malino', 'menawarkan', 'keindahan', 'sangat', 'menakjukkan', 'air', 'jernih', 'indahnya', 'pepohonan', 'sekelilingnya', 'fasilitas', 'tersedia', 'terbatas']
['wai', 'malino', 'wisata', 'yang', 'nyaman', 'buat', 'liburan', 'keluarga', 'sahabat', 'karena', 'memiliki', 'pemandangan', 'yang', 'indah', 'serta', 'berada', 'ditempat', 'yang', 'sejuk', 'dan', 'segar', 'wisata', 'tempatnya', 'paling', 'yang', 'rekomendasi', 'buat', 'kemping', 'karena', 'lumayan', 'jauh', 'dari', 'suara', 'berisik']	['wai', 'malino', 'wisata', 'nyaman', 'buat', 'liburan', 'keluarga', 'sahabat', 'memiliki', 'pemandangan', 'indah', 'berada', 'ditempat', 'sejuk', 'segar', 'wisata', 'tempatnya', 'paling', 'rekomendasi', 'buat', 'kemping', 'lumayan', 'jauh', 'suara', 'berisik']

e) *Stemming*

Stemming adalah suatu teknik dalam *preprocessing* data yang digunakan untuk menghilangkan berbagai variasi kata yang memiliki akar kata yang sama sehingga hanya menyisakan kata dasarnya. *Stemming* dapat digunakan untuk mengurangi jumlah fitur atau kata dalam suatu *dataset* dan memudahkan analisis data. Hasil dari proses *stemming* dapat dilihat pada tabel 4.7 berikut:

Tabel 4. 7 *Stemming*

<i>Stopword Removal</i>	<i>Stemming</i>
['pemandangan', 'sini', 'luar', 'biasa', 'udara', 'segar', 'suasananya', 'menenangkan']	['pandang', 'sini', 'luar', 'biasa', 'udara', 'segar', 'suasana', 'tenang']
['wisata', 'wai', 'malino', 'menawarkan', 'keindahan', 'sangat', 'menakjukkan', 'air', 'jernih', 'indahnya', 'pepohonan', 'sekelilingnya', 'fasilitas', 'tersedia', 'terbatas']	['wisata', 'wai', 'malino', 'tawar', 'indah', 'sangat', 'menakjukkan', 'air', 'jernih', 'indah', 'pohon', 'keliling', 'fasilitas', 'sedia', 'batas']
['wai', 'malino', 'wisata', 'nyaman', 'buat', 'liburan', 'keluarga', 'sahabat', 'memiliki', 'pemandangan', 'indah', 'berada', 'ditempat', 'sejuk', 'segar', 'wisata', 'tempatnya', 'paling', 'rekomendasi', 'buat', 'kemping', 'lumayan', 'jauh', 'suara', 'berisik']	['wai', 'malino', 'wisata', 'nyaman', 'buat', 'libur', 'keluarga', 'sahabat', 'milik', 'pandang', 'indah', 'ada', 'tempat', 'sejuk', 'segar', 'wisata', 'tempat', 'paling', 'rekomen', 'buat', 'kemping', 'lumayan', 'jauh', 'suara', 'berisik']

4.4 Pelabelan Lexicon

Data yang dihasilkan dari teks *preprocessing* akan dilabeli menggunakan *lexicon-based*, pelabelan teks diubah kedalam bentuk numerik agar sesuai dengan kebutuhan sistem.

Kamus *lexicon* yang digunakan dalam penelitian ini bersumber dari repositori *InSet (Indonesia Sentiment Lexicon)* yang tersedia secara terbuka

melalui tautan: <https://github.com/fajri91/InSet>. Kamus ini disimpan dalam bentuk file CSV yang memuat daftar kata dalam bahasa indonesia beserta bobot sentimennya. Proses menentukan sentimen dari suatu teks berdasarkan daftar kata yang telah dikategorikan sebagai positif, negatif, atau netral. Metode ini menggunakan kamus kata (*lexicon*) yang berisi nilai sentimen dari masing-masing kata dalam teks. Berikut ini contoh kamus *lexicon* pada tabel 4.8 :

Tabel 4. 8 Kamus *Lexicon*

No	Kata	Bobot
1	biasa	2
2	pandang	1
3	segar	4
4	suasana	4
5	tenang	5
6	fasilitas	2
7	ramai	3
8	bagus	2
9	kunjung	3
10	sama	3
11	udara	-4
12	kurang	-3
13	batas	-3
14	tidak	-5
15	tempat	-3
16	lumayan	-2
17	matanya	-2

Tabel 4. 9 Cara kerja *Lexicon-based*

No	Komentar							Total Score	Label
1	Pandang 1	Biasa 2	Udara -4	Segar 4	Suasana 4	Tenang 5	12	Positif	

2	Kurang -3	Fasilitas 2	Cukup -5	Batas -3	Ramai 3		-6	Negatif
3	Bagus 2	Kunjung 3	Sama 3	Tidak -5	Tempat -3		0	Netral

Tabel 4. 10 Hasil *Lexicon*

No	Komentar	lexicon-based	Label
1	pandang biasa udara segar suasana tenang	12	1
3	kurang fasilitas cukup batas ramai	-6	-1
2	bagus kunjung sama tidak tempat	0	0

Berdasarkan tabel 4.10 hasil *lexicon* di atas, kalimat “pandang biasa udara segar suasana tenang” memiliki hasil lexicon > 0 yaitu 12 diberi label 1 yang berarti positif. Sementara itu, kalimat “kurang fasilitas cukup batas ramai” dengan nilai *lexicon* < 0 yaitu -6 diberi label -1 yang berarti negatif. Sedangkan kalimat “bagus kunjung sama tidak tempat” memiliki hasil *lexicon* = 0 diberi label 0 yang berarti netral.

Namun, metode ini memiliki keterbatasan, terutama karena pelabelan data sentimen masih menggunakan metode otomatis dari kamus *lexicon* tanpa validasi manual dari manusia. Selain itu, proses *stemming* atau pengembalian kata ke bentuk dasar juga dapat memengaruhi akurasi pelabelan. Misalnya, kata “menyenangkan” setelah melalui *stemming* dapat menjadi “senang”, padahal keduanya memiliki tingkat kekuatan sentimen yang berbeda. Hal ini berpotensi menurunkan akurasi pelabelan.

4.5 Pembobotan TF-IDF

Pembobotan TF-IDF (*Term Frequency-Inverse Document Frequency*) untuk melihat beberapa nilai frekuensi kemunculan terhadap suatu kata. Hasil yang didapatkan dalam pembobotan TF-IDF merupakan hasil perkalian dari TF dikalikan dengan IDF. Pada tahap ini, proses normalisasi data belum dilakukan karena fokus penelitian adalah pada transformasi nilai TF-IDF dari komentar asli.

Namun, proses normalisasi ini menjadi salah satu pertimbangan untuk pengembangan penelitian ke depan guna meningkatkan akurasi model klasifikasi. Berikut ini tabel contoh komentar yang akan digunakan dalam perhitungan TF-IDF:

Tabel 4. 11 Data Komentar untuk Perhitungan TF-IDF

No	Komentar
1	tempat indah puas
2	kecewa tempat kotor terawat
3	tempat biasa istimewa
4	pemandangan luar biasa mengesankan
5	buruk layanan mengecewakan
6	cukup bagus perlu perbaiki
7	tempat cocok liburan keluarga bagus
8	buruk sesuai harapan bagus

Langkah selanjutnya adalah melakukan perhitungan TF-IDF (*Term Frequency–Inverse Document Frequency*) untuk mengetahui bobot atau tingkat kepentingan setiap kata dalam masing-masing komentar. Perhitungan TF-IDF dilakukan melalui tiga tahapan utama yaitu:

1) *Term Frequency* (TF)

TF mengukur seberapa sering suatu kata muncul dalam sebuah dokumen. Dengan menggunakan persamaan.

$$TF_{x,d} = \left(\frac{\text{Jumlah kemunculan trem } t \text{ dalam dokumen } d}{\text{Total jumlah kata dalam dokumen } d} \right) \quad (4.1)$$

Contoh untuk kata "tempat" dalam dokumen "tempat indah puas":

"tempat" muncul 1 kali

Total kata dalam dokumen = 3

TF ("tempat", "tempat indah puas") = 1/3 = 0.33

Berikut ini merupakan hasil perhitungan nilai *Term Frequency* (TF) untuk beberapa kata (*term*) yang muncul dalam masing-masing dokumen. Nilai TF

dihitung berdasarkan rasio kemunculan kata terhadap jumlah total kata dalam dokumen tersebut. Tabel di bawah menunjukkan distribusi nilai TF dari masing-masing *term* pada delapan dokumen:

Tabel 4. 12 Distribusi nilai TF

Term	D1	D2	D3	D4	D5	D6	D7	D8
bagus	0	0	0	0	0	0.25	0.2	0.25
biasa	0	0	0.33	0.25	0	0	0	0
buruk	0	0	0	0	0.33	0	0	0.25
cocok	0	0	0	0	0	0	0.2	0
cukup	0	0	0	0	0	0.25	0	0
harapan	0	0	0	0	0	0	0	0.25
indah	0.33	0	0	0	0	0	0	0
istimewa	0	0	0.33	0	0	0	0	0
kecewa	0	0.25	0	0	0	0	0	0
keluarga	0	0	0	0	0	0	0.2	0
kotor	0	0.25	0	0	0	0	0	0
layanan	0	0	0	0	0.33	0	0	0
liburan	0	0	0	0	0	0	0.2	0
luar	0	0	0	0.25	0	0	0	0
mengesankan	0	0	0	0.25	0	0	0	0
mengecewakan	0	0	0	0	0.33	0	0	0
pemandangan	0	0	0	0.25	0	0	0	0
perbaiki	0	0	0	0	0	0.25	0	0
perlu	0	0	0	0	0	0.25	0	0
puas	0.33	0	0	0	0	0	0	0
sesuai	0	0	0	0	0	0	0	0.25
tempat	0.33	0.25	0.33	0	0	0	0.2	0
terawat	0	0.25	0	0	0	0	0	0

2) *Inverse Document Frequency* (IDF)

Setelah menghitung *Term Frequency* (TF), langkah selanjutnya adalah menghitung *Inverse Document Frequency* (IDF). IDF mengukur seberapa penting sebuah kata dalam seluruh kumpulan dokumen, dengan menggunakan persamaan berikut:

$$IDF_t = \log\left(\frac{N}{DF_t}\right) \quad (4.2)$$

Keterangan:

N = jumlah total dokumen (8)

DF(t) = jumlah dokumen yang mengandung *term t*

Contoh untuk kata "tempat":

Kata "tempat" muncul di 4 dokumen

$$IDF("tempat") = \log(8/4) = \log(2) = 0.301$$

Tabel berikut menyajikan hasil perhitungan *Document Frequency* (DF) dan nilai IDF dari masing-masing *term*:

Tabel 4. 13 Hasil perhitungan DF dan nilai IDF

Term	DF (Document Frequency)	IDF = log(8/DF)
bagus	3	0.426
biasa	1	0.9031
buruk	2	0.6021
cocok	1	0.9031
cukup	1	0.9031
harapan	1	0.9031
indah	1	0.9031
istimewa	1	0.9031
kecewa	1	0.9031
keluarga	1	0.9031
kotor	1	0.9031
layanan	1	0.9031
liburan	1	0.9031
luar	1	0.9031

mengesankan	1	0.9031
mengecewakan	1	0.9031
pemandangan	1	0.9031
perbaiki	1	0.9031
perlu	1	0.9031
puas	1	0.9031
sesuai	1	0.9031
tempat	4	0.301
terawat	1	0.9031

3) TF-IDF

Setelah diperoleh nilai TF dan IDF, tahap selanjutnya menghitung nilai TF-IDF untuk setiap term dalam setiap dokumen. TF-IDF merupakan metode yang menggabungkan frekuensi kemunculan kata dalam dokumen (TF) dan tingkat keunikan kata di antara seluruh dokumen (IDF). Nilai TF-IDF memberikan bobot pentingnya suatu *term* dalam suatu dokumen relatif terhadap keseluruhan koleksi dokumen. TF-IDF dihitung dengan menggunakan persamaan:

$$TF-IDF_{t,d} = TF_{x,d} * IDF_t \quad (4.3)$$

Contoh untuk kata "tempat" dalam dokumen "tempat indah puas":

TF("tempat", "tempat indah puas") = 0.33

IDF("tempat") = 0.301

$$\text{TF-IDF}(\text{"tempat"}, \text{"tempat indah puas"}) = 0.33 \times 0.301 = 0.0993$$

Tabel berikut menunjukkan hasil perhitungan TF-IDF untuk masing-masing term pada delapan dokumen:

Tabel 4. 14 Hasil perhitungan TF-IDF

cukup	0	0	0	0	0	0.2258	0	0
harapan	0	0	0	0	0	0	0	0.2258
indah	0.298	0	0	0	0	0	0	0
istimewa	0	0	0.298	0	0	0	0	0
kecewa	0	0.2258	0	0	0	0	0	0
keluarga	0	0	0	0	0	0	0.1806	0
kotor	0	0.2258	0	0	0	0	0	0
layanan	0	0	0	0	0.298	0	0	0
liburan	0	0	0	0	0	0	0.1806	0
luar	0	0	0	0.2258	0	0	0	0
mengesankan	0	0	0	0.2258	0	0	0	0
mengecewakan	0	0	0	0	0.298	0	0	0
pemandangan	0	0	0	0.2258	0	0	0	0
perbaiki	0	0	0	0	0	0.2258	0	0
perlu	0	0	0	0	0	0.2258	0	0
puas	0.298	0	0	0	0	0	0	0
sesuai	0	0	0	0	0	0	0	0.2258
tempat	0.0993	0.0753	0.0993	0	0	0	0.0602	0
terawat	0	0.2258	0	0	0	0	0	0

Hasil pembobotan TF-IDF merupakan data numerik yang dimana nantinya data tersebut digunakan dalam proses perhitungan klasifikasi.

4.6 Pembagian Data

Pada penelitian ini, jumlah data setelah *preprocessing* ada 940 data. Data dibagi menjadi dua bagian utama: data *training* dan data *testing*. Pembagian dilakukan dengan rasio 80% untuk *training* dan 20% untuk *testing*. Setiap label (positif, negatif, dan netral) dibagi secara proporsional berdasarkan jumlah data yang dimiliki masing-masing. Hal ini bertujuan agar setiap kelas memiliki representasi yang seimbang dalam proses pelatihan dan pengujian model. Strategi ini termasuk ke dalam metode *stratified split*, yaitu teknik pembagian data yang mempertahankan proporsi jumlah data dari setiap kelas agar model tidak bias terhadap salah satu label tertentu. Sebelum dilakukan pembagian data, terdapat ketidak seimbangan jumlah sampel pada setiap kelas adalah sebagai berikut:

Tabel 4. 15 Pembagian Data *Training* dan *Testing* Sebelum *Oversampling*

Label	Jumlah Data	Data <i>Training</i> (80%)	Data <i>Testing</i> (20%)
1 (Positif)	676	540	136
-1(Negatif)	195	156	39
0 (Netral)	69	55	14
Total	940	751	189

Dari tabel 4.15 terlihat bahwa jumlah data tidak seimbang, dengan data berlabel positif memiliki jumlah terbesar, sementara netral memiliki jumlah terkecil. Untuk mengatasi ketidakseimbangan ini, dilakukan *oversampling* pada kelas yang memiliki jumlah data lebih sedikit agar distribusi menjadi merata. Berikut ini hasil pembagian data *oversampling* pada tabel 4.16 :

Tabel 4. 16 Pembagian Data *Training* dan *Testing* Setelah *Oversampling*

Label	Jumlah Data	Data <i>Training</i> (80%)	Data <i>Testing</i> (20%)
1 (Positif)	676	540	136
-1(Negatif)	676	540	136
0 (Netral)	676	540	136
Total	2028	1620	408

Setelah *oversampling*, jumlah data di setiap kelas menjadi sama, yaitu 676 data per kelas, sehingga model dapat dilatih dengan distribusi data yang lebih seimbang. Dari tabel di atas, masing-masing kelas memiliki 540 data untuk *training* dan 136 data untuk *testing*, memastikan bahwa model mendapatkan cukup data untuk belajar serta diuji dengan data yang belum pernah dilihat sebelumnya.

4.7 Mekanisme Kerja Algoritma Klasifikasi

Pada penelitian ini, digunakan dua algoritma klasifikasi yaitu *Naive Bayes* dan *Random Forest*. Keduanya merupakan metode *supervised learning* yang digunakan untuk mengklasifikasikan data berdasarkan fitur yang tersedia. Penjelasan mekanisme kerja dari masing-masing algoritma adalah sebagai berikut:

4.7.1 *Naïve Bayes*

Naïve Bayes adalah salah satu algoritma klasifikasi dalam machine learning yang berdasarkan pada teorema Bayes, dengan asumsi bahwa setiap fitur (atribut) dalam data bersifat independen atau tidak saling bergantung satu sama lain. Algoritma ini menghitung probabilitas dari setiap kelas yang mungkin, dan memilih kelas dengan probabilitas tertinggi sebagai hasil klasifikasinya. Berikut ini cara kerja manual algoritma *Naïve Bayes*:

- a. Dataset yang digunakan

Tabel 4. 17 Contoh *dataset* untuk Perhitungan *Naïve Bayes*

No	Komentar	Kelas
1	"Tempatnya sangat indah, saya puas sekali!"	Positif
2	"Saya kecewa, tempatnya kotor dan tidak terawat"	Negatif
3	"Tempatnya biasa saja, tidak terlalu istimewa"	Netral
4	"Pemandangannya luar biasa, sangat mengesankan"	Positif
5	"Sangat buruk, layanan sangat mengecewakan"	Negatif
6	"Cukup bagus, tapi banyak hal yang perlu diperbaiki"	Netral
7	"Tempat yang sangat cocok untuk liburan keluarga, sangat bagus"	Positif
8	"Sangat buruk, tidak sesuai harapan, tidak bagus sama sekali"	Negatif
9	“Tempat yang bagus”	Uji

- b. Data latih

Langkah pertama menghitung Probabilitas *term* untuk setiap sentimen.

Tabel 4. 18 Probabilitas Frekuensi setiap sentimen

No	Term	Positif (P+)	Negatif (P-)	Netral (P0)
1	sangat	4	1	0
2	indah	1	0	0
3	saya	1	1	0

No	Term	Positif (P+)	Negatif (P-)	Netral (P0)
4	puas	1	0	0
5	sekali	1	1	0
6	pemandangannya	1	0	0
7	luar	1	0	0
8	biasa	1	0	1
9	mengesankan	1	0	0
10	tempat	1	1	1
11	cocok	1	0	0
12	liburan	1	0	0
13	keluarga	1	0	0
14	bagus	1	1	1
15	kecewa	0	1	0
16	tempatnya	0	1	1
17	kotor	0	1	0
18	terawat	0	1	0
19	tidak	0	4	1
20	harapan	0	1	0

Pada Tabel diatas dapat diketahui, Jumlah frekuensi n dari semua sentimen dan ($|vocabulary|$) jumlah kosakata keseluruhan masing-masing $text$, yaitu Positif 14, Negatif 11 dan Netral 5. $P(w_k|v_j)$ probabilitas dari setiap *term* dapat dicari dengan menggunakan persamaan berikut:

$$P(w_k|v_j) = \frac{|n_k+1|}{n+|vocabulary|} \quad (4.4)$$

Contoh data latih sentimen positif $P(\text{bagus}|\text{Positif}) = (1+1)/(4+14) = 15.071$. Berkut ini tabel hasil perhitungan probabilitas dari setiap *term*:

Tabel 4. 19 Probabilitas Data Latih Semua Sentimen

No	Term	Positif (P+)	Negatif (P-)	Netral (P0)
1	sangat	0.17857142	0.090909091	0.1

No	Term	Positif (P+)	Negatif (P-)	Netral (P0)
2	indah	0.07142857	0.045454545	0.1
3	saya	0.07142857	0.090909091	0.1
4	puas	0.07142857	0.045454545	0.1
5	sekali	0.07142857	0.090909091	0.1
6	pemandangannya	0.07142857	0.045454545	0.1
7	luar	0.07142857	0.045454545	0.1
8	Biasa	0.07142857	0.045454545	0.2
9	mengesankan	0.07142857	0.045454545	0.1
10	tempat	0.07142857	0.090909091	0.2
11	cocok	0.07142857	0.045454545	0.1
12	liburan	0.07142857	0.045454545	0.1
13	keluarga	0.07142857	0.045454545	0.1
14	bagus	0.07142857	0.090909091	0.2
15	kecewa	0.03571428	0.090909091	0.1
16	tempatnya	0.03571428	0.090909091	0.2
17	kotor	0.03571428	0.090909091	0.1
18	terawat	0.03571428	0.090909091	0.1
19	tidak	0.03571428	0.227272727	0.2
20	harapan	0.03571428	0.090909091	0.1

Setelah menghitung probabilitas *term* dari setiap sentiment, maka dilakukan perhitungan $P(v_j)$ dari semua sentiment menggunakan persamaan 2.11. Pada tahapan ini diketahui terdapat 8 data latih dengan sentimen 3 positif, 3 negatif, dan 2 netral, sehingga nilainya:

$$P(v_j) = \frac{|doc_j|}{|contoh|}$$

$$P(\text{positif}) = 3/8 = 0.375$$

$$P(\text{negatif}) = 3/8 = 0.375$$

$$P(\text{netral}) = 2/8 = 0.25$$

Setelah menghitung nilai probabilitas dari semua kategori sentimen, terdapat perubahan nilai probabilitas dari setiap *term* dalam semua kategori sentimen. Perubahan dilakukan dengan menggabungkan total kosakata dari semua sentimen. Total (*vocabulary*) dari semua sentimen adalah 20. Sehingga perubahan probabilitas sebagai berikut:

Tabel 4. 20 Perubahan Nilai Probabilitas Sentimen Positif

No	Term	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	sangat	4	$(4+1)/(14+20) = 0.147$
2	indah	1	$(1+1)/(14+20) = 0.058$
3	saya	1	$(1+1)/(14+20) = 0.058$
4	puas	1	$(1+1)/(14+20) = 0.058$
5	sekali	1	$(1+1)/(14+20) = 0.058$
6	pemandangannya	1	$(1+1)/(14+20) = 0.058$
7	luar	1	$(1+1)/(14+20) = 0.058$
8	biasa	1	$(1+1)/(14+20) = 0.058$
9	mengesankan	1	$(1+1)/(14+20) = 0.058$
10	tempat	1	$(1+1)/(14+20) = 0.058$
11	cocok	1	$(1+1)/(14+20) = 0.058$
12	liburan	1	$(1+1)/(14+20) = 0.058$
13	keluarga	1	$(1+1)/(14+20) = 0.058$
14	bagus	1	$(1+1)/(14+20) = 0.058$
15	kecewa	0	$(1+1)/(14+20) = 0.058$
16	tempatnya	0	$(0+1)/(14+20) = 0.029$
17	kotor	0	$(0+1)/(14+20) = 0.029$
18	terawat	0	$(0+1)/(14+20) = 0.029$
19	tidak	0	$(0+1)/(14+20) = 0.029$
20	harapan	0	$(0+1)/(14+20) = 0.029$

Tabel 4. 21 Perubahan Nilai Probabilitas Sentimen Negatif

No	Term	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	sangat	1	$(1+1)/(11+20) = 0.645$
2	indah	0	$(0+1)/(11+20) = 0.032$
3	saya	1	$(1+1)/(11+20) = 0.645$
4	puas	0	$(0+1)/(11+20) = 0.032$
5	sekali	1	$(1+1)/(11+20) = 0.645$
6	pemandangannya	0	$(0+1)/(11+20) = 0.032$
7	luar	0	$(0+1)/(11+20) = 0.032$
8	biasa	0	$(0+1)/(11+20) = 0.032$
9	mengesankan	0	$(0+1)/(11+20) = 0.032$
10	tempat	1	$(1+1)/(11+20) = 0.645$
11	cocok	0	$(0+1)/(11+20) = 0.032$
12	liburan	0	$(0+1)/(11+20) = 0.032$
13	keluarga	0	$(0+1)/(11+20) = 0.032$
14	bagus	1	$(1+1)/(11+20) = 0.645$
15	kecewa	1	$(1+1)/(11+20) = 0.645$
16	tempatnya	1	$(1+1)/(11+20) = 0.645$
17	kotor	1	$(1+1)/(11+20) = 0.645$
18	terawat	1	$(1+1)/(11+20) = 0.645$
19	tidak	4	$(4+1)/(11+20) = 0.161$
20	harapan	1	$(1+1)/(11+20) = 0.645$

Tabel 4. 22 Perubahan Nilai Probabilitas Sentimen Netral

No	Term	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	sangat	0	$(0+1)/(5+20) = 0.04$
2	indah	0	$(0+1)/(5+20) = 0.04$
3	saya	0	$(0+1)/(5+20) = 0.04$
4	puas	0	$(0+1)/(5+20) = 0.04$
5	sekali	0	$(0+1)/(5+20) = 0.04$

No	Term	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
6	pemandangannya	0	$(0+1)/(5+20) = 0.04$
7	luar	0	$(0+1)/(5+20) = 0.04$
8	biasa	1	$(1+1)/(5+20) = 0.08$
9	mengesankan	0	$(0+1)/(5+20) = 0.04$
10	tempat	1	$(1+1)/(5+20) = 0.08$
11	cocok	0	$(0+1)/(5+20) = 0.04$
12	liburan	0	$(0+1)/(5+20) = 0.04$
13	keluarga	0	$(0+1)/(5+20) = 0.04$
14	bagus	1	$(1+1)/(5+20) = 0.08$
15	kecewa	0	$(0+1)/(5+20) = 0.04$
16	tempatnya	1	$(1+1)/(5+20) = 0.08$
17	kotor	0	$(0+1)/(5+20) = 0.04$
18	terawat	0	$(0+1)/(5+20) = 0.04$
19	tidak	1	$(1+1)/(5+20) = 0.08$
20	harapan	0	$(0+1)/(5+20) = 0.04$

c. Klasifikasi Data Uji

Dalam tahap klasifikasi data uji, *term* pada daftar yang akan diklasifikasi dicari nilai probabilitasnya dengan membandingkan *term* yang ada pada tabel *term* data uji dengan tabel nilai probabilitas pada setiap sentimen yang sudah mengalami perubahan. Jika ada kesamaan *term* pada tabel, maka nilai probabilitas yang sama tersebut dijadikan nilai probabilitas pada *term* di tabel data uji. Tetapi, jika *term* tidak sama, maka frekuensinya dihitung nol. Penulis akan menggunakan sampel satu buah komentar yang telah melalui tahap *text preprocessing*. Berikut merupakan perhitungan manualnya.

Tabel 4. 23 Dafta *Term* yang akan diklasifikasi

Term	Positif (P+)	Negatif (P-)	Netral (P0)
tempat	1	1	1
bagus	1	1	1

Setelah mendapatkan nilai probabilitasnya data uji pada masing-masing sentimen, maka di hitung nilai V_{MAP} menggunakan persamaan 2.10.

Tabel 4. 24 Probabilitas *Term* Data Uji pada Sentimen Positif

No	Term	Term Latih	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	tempat	Ada	1	$(1+1)/(14+20) = 0.058$
2	bagus	Ada	1	$(1+1)/(14+20) = 0.058$

Berdasarkan nilai probabilitas data uji sentimen positif sesuai dengan tabel di atas, maka nilai V_{MAP} untuk sentimen positif adalah sebagai berikut

$$\begin{aligned}
 V_{MAP} &= \arg \max P(v_j) \prod_i P(a_1|v_j) \\
 &= ((0,058)(0,058)*(0,375) \\
 &= 0,0012
 \end{aligned}$$

Tabel 4. 25 Probabilitas *Term* Data Uji pada Sentimen Negatif

No	Term	Term Latih	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	tempat	Ada	1	$(1+1)/(11+20) = 0.064$
2	bagus	Ada	1	$(1+1)/(11+20) = 0.064$

Berdasarkan nilai probabilitas data uji sentimen negatif sesuai dengan tabel di atas, maka nilai V_{MAP} untuk sentimen negatif adalah sebagai berikut

$$V_{MAP} = \arg \max P(v_j) \prod_i P(a_1|v_j)$$

$$= ((0,064)(0,064)* (0,375)$$

$$= 0,0015$$

Tabel 4. 26 Probabilitas *Term* Data Uji pada Sentimen Netral

No	Term	Term Latih	Frekuensi (n_k)	Probabilitas $P(w_k v_j)$
1	tempat	Ada	1	$(1+1)/(5+20) = 0.08$
2	bagus	Ada	1	$(1+1)/(5+20) = 0.08$

Berdasarkan nilai probabilitas data uji sentimen negatif sesuai dengan tabel di atas, maka nilai V_{MAP} untuk sentimen negatif adalah sebagai berikut

$$V_{MAP} = \arg \max P(v_j) \prod_i P(a_1|v_j)$$

$$= ((0,08)(0,08)*(0,25)$$

$$= 0,0016$$

Berdasarkan perhitungan manual dengan metode *Naïve Bayes*, komentar dengan term "tempat" dan "bagus" memiliki probabilitas tertinggi pada sentimen netral (0,0016), sehingga komentar tersebut diklasifikasikan sebagai **netral**.

4.7.2 Random Forest

Random Forest adalah algoritma klasifikasi berbasis *ensemble learning* yang membentuk banyak pohon keputusan (*decision tree*) dan menggabungkan hasil dari semua pohon tersebut untuk menghasilkan keputusan akhir berdasarkan voting. Langkah kerja algoritma *Random Forest* adalah sebagai berikut:

Tabel 4. 27 Dataset Awal perhitungan *Random Forest*

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
1	0	0,35	0,35	0	0	0	Positif
2	0	0,31	0	0,31	0	0,1	Negatif

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
3	0	0,33	0	0	0	0	Netral
4	0	0	0	0	0,31	0	Positif
5	0	0	0	0	0,5	0	Negatif
6	0,2	0	0	0	0	0	Netral
7	0,2	0	0	0	0	0	Positif
8	0,23	0	0	0,5	0	0	Negatif
9	0,5	0,65	0	0	0	0	Uji

a. *Bootstrap Sampling*

Berikut ini adalah contoh pengambilan sampel acak dengan pengembalian (*bootstrap sampling*) yang digunakan dalam proses awal pembangunan *decision tree*. Sampel ini diambil dari *dataset* asli dan memungkinkan adanya duplikasi data karena proses pengambilan dilakukan dengan pengembalian. Berikut ini tabel pengambilan sampel acak dengan pengembalian:

Tabel 4. 28 *Bootstrap* Pertama

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
3	0	0,33	0	0	0	0	Netral
4	0	0	0	0	0,31	0	Positif
2	0	0,31	0	0,31	0	0,1	Negatif
7	0,2	0	0	0	0	0	Positif
1	0	0,35	0,35	0	0	0	Positif
6	0,2	0	0	0	0	0	Netral
5	0	0	0	0	0,5	0	Negatif
8	0,23	0	0	0,5	0	0	Negatif

Tabel 4. 29 *Bootstrap* Kedua

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
7	0,2	0	0	0	0	0	Positif
2	0	0,31	0	0,31	0	0,1	Negatif
3	0	0,33	0	0	0	0	Netral

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
2	0	0,31	0	0,31	0	0,1	Negatif
8	0,23	0	0	0,5	0	0	Negatif
1	0	0,35	0,35	0	0	0	Positif
4	0	0	0	0	0,31	0	Positif
7	0,2	0	0	0	0	0	Positif

Tabel 4. 30 Bootstrap Ketiga

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
6	0,2	0	0	0	0	0	Netral
5	0	0	0	0	0,5	0	Negatif
2	0	0,31	0	0,31	0	0,1	Negatif
1	0	0,35	0,35	0	0	0	Positif
3	0	0,33	0	0	0	0	Netral
8	0,23	0	0	0,5	0	0	Negatif
1	0	0,35	0,35	0	0	0	Positif
4	0	0	0	0	0,31	0	Positif

b. Pembangunan *Decision Tree*

Berikut ini merupakan tahapan pembangunan *decision tree*. Proses ini dimulai dari pengambilan sampel acak dengan pengembalian (*bootstrap sampling*), kemudian dilanjutkan dengan perhitungan nilai entropy (*Information Gini*), serta pemilihan fitur terbaik pada setiap *node* berdasarkan nilai gain dalam proses pembentukan pohon keputusan. Berikut ini langkah-langkah dijelaskan secara sistematis untuk setiap *Decision Tree*:

1. Pembangunan *Decision Tree* untuk *Bootstrap* pertama

Berikut ini tabel *dataset* awal untuk perhitungan *node* 1:

Tabel 4. 31 Dataset awal/*Node* 1 (*Decision Tree* 1)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
3	0	0.33	0	0	0	0	Netral
4	0	0	0	0	0.31	0	Positif

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
2	0	0.31	0	0.31	0	0.1	Negatif
7	0.2	0	0	0	0	0	Positif
1	0	0.35	0.35	0	0	0	Positif
6	0.2	0	0	0	0	0	Netral
5	0	0	0	0	0.5	0	Negatif
8	0.23	0	0	0.5	0	0	Negatif

- a) Pemilihan Fitur secara acak, Memilih atribut secara acak contoh disini atribut yang digunakan bagus dan tempat
- b) Perhitungan *Entropy* Awal (*Parent Node*), pada perhitungan *Entropy* awal menggunakan persamaan berikut:

$$\text{Entropy} = - \sum_{i=t}^n p_i \log_2(p_i) \quad (4.5)$$

Untuk *Bootstrap* 1 yang memiliki 8 sampel, 3 Positif, 3 Negatif, dan 2 Netral:

$$\text{Entropy} = - \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{2}{8} \log_2 \frac{2}{8} \right)$$

$$\begin{aligned} \text{Entropy} &= - (0,375 \times -1,415) + (0,375 \times -1,415) + (0,25 \times -2) \\ &= -(-0,530) + (-0,530) + (-0,5) \\ &= 1,561 \end{aligned}$$

- c) Penghitungan *Information Gini* untuk setiap fitur untuk fitur "bagus" dan "tempat":

Fitur "tempat" dengan ambang batas 0:

- Anak kiri (bagus ≤ 0): 5 sampel

2 Positif, 2 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{2}{5} \log_2 \frac{2}{5} \right) + \left(\frac{2}{5} \log_2 \frac{2}{5} \right) + \left(\frac{1}{5} \log_2 \frac{1}{5} \right)$$

$$\text{Entropy} = - (0,4 \times -1,321) + (0,4 \times -1,321) + (0,2 \times -2,321)$$

$$= -(-0,528) + (-0,528) + (-0,464)$$

$$= 1,521$$

- Anak kanan (bagus > 0): 3 sampel

1 Positif, 1 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{1}{3} \log_2 \frac{1}{3} \right) + \left(\frac{1}{3} \log_2 \frac{1}{3} \right) + \left(\frac{1}{3} \log_2 \frac{1}{3} \right)$$

$$\text{Entropy} = - (0,333 \times -1,584) + (0,333 \times -1,584) + (0,333 \times -1,584)$$

$$= -(-0,525) + (-0,525) + (-0,525)$$

$$= 1,5844$$

Untuk menghitung *Gini* menggunakan persamaan berikut:

$$\begin{aligned} \text{Information Gini} &= \\ \text{Entropy}(\text{parent}) - \sum_{k=1}^m & \left(\frac{N_k}{N} \times \text{Entropy}(\text{Child}_k) \right) \end{aligned} \quad (4.6)$$

$$\text{IG} = 1,561 - \left(\frac{5}{8} \times 1,521 \right) + \left(\frac{3}{8} \times 1,584 \right)$$

$$= 1,561 - (0,951) + (0,593)$$

$$= 0,015$$

Lakukan perhitungan untuk semua *Entropy* fitur tempat dan buruk. Berikut ini tabel hasil perhitungan *Entropy* dan perhitungan *Gini* pada fitur (Bagus, Tempat):

Tabel 4. 32 Hasil Perhitungan *Entropy* dan *Gini* (*Decision Tree* 1)

Fitur		Jumlah	Positif (1)	Negatif (-1)	Netral(0)	Entropy	Gini
Total		8	3	3	2	1.5612	
Bagus	≤ 0	5	2	2	1	1.5219	0.0157
	> 0	3	1	1	1	1.5849	
	≤ 0.24	8	3	3	2	1.5612	0
	> 0.24	0	0	0	0	0	
	≤ 0.23	6	2	2	2	1.5849	0.3725
	> 0.23	2	1	0	1	0	
Tempat	≤ 0.33	7	2	3	2	1.5566	0.1992
	> 0.33	1	1	0	0	0	
	≤ 0	5	2	2	1	1.5219	0.0157
	> 0	3	1	1	1	1.5849	
	≤ 0.31	6	2	3	2	1.5566	0.3937
	> 0.31	2	1	0	1	0	
	≤ 0.35	8	3	3	2	1.5612	0
	> 0.35	0	0	0	0	0	

Berdasarkan tabel 4.32 Fitur "tempat" dengan *Entropy* ≤ 0.31 memiliki *Information Gini* tertinggi = 0.3937. Maka, untuk pemisahan pertama dalam pohon keputusan, fitur terbaik adalah tempat dengan *Entropy* ≤ 0.31 . Lanjutkan ke *node* berikutnya (anak kiri dan kanan dari *Entropy* ini), berikut ini tabel tabel untuk perhitungan *node* 1.1:

Tabel 4. 33 Dataset untuk *node* 1.1 (*Decision Tree* 1)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
4	0	0	0	0	0.31	0	Positif
2	0	0.31	0	0.31	0	0.41	Negatif
7	0.24	0	0	0	0	0	Positif
6	0.24	0	0	0	0	0	Netral
5	0	0	0	0	0.45	0	Negatif
8	0.23	0	0	0.54	0	0	Negatif

- 1) Pemilihan Fitur secara acak, fitur selanjutnya adalah buruk dan layanan
- 2) Perhitungan *Entropy* Awal (*Parent Node*), pada perhitungan *Entropy* awal menggunakan persamaan berikut:

$$\text{Entropy} = - \sum_{i=1}^n p_i \log_2(p_i) \quad (4.7)$$

Total sampel 6: 2 Positif, 3 Negatif, dan 1 Netral:

$$\text{Entropy} = - \left(\frac{2}{6} \log_2 \frac{2}{6} \right) + \left(\frac{3}{6} \log_2 \frac{3}{6} \right) + \left(\frac{1}{6} \log_2 \frac{1}{6} \right)$$

$$\begin{aligned} \text{Entropy} &= - (0,333 \times -1,589) + (0,5 \times -1) + (0,166 \times -2,584) \\ &= -(-0,528) + (-0,5) + (-0,430) \\ &= 1,459 \end{aligned}$$

- 3) Penghitungan *Information Gini* untuk fitur "buruk dan layanan:

Fitur "tempat" dengan ambang buruk 0

- Anak kiri (buruk ≤ 0): 4 sampel

2 Positif, 1 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{2}{4} \log_2 \frac{2}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right)$$

$$\begin{aligned} \text{Entropy} &= -(0,5 \times -1) + (0,25 \times -2) + (0,25 \times -2) \\ &= -(-0,5) + (-0,5) + (-0,5) \\ &= 1,5 \end{aligned}$$

- Anak kanan (buruk > 0): 2 sampel

1 Positif, 2 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{2}{2} \log_2 \frac{2}{2} \right)$$

$$\text{Entropy} = \frac{2}{2} = 1 = \log_2 1 = 0$$

$$= - (1 \times 0) = 0$$

Untuk menghitung *Gini* menggunakan persamaan berikut:

$$\text{Information Gini} =$$

$$\text{Entropy}(\text{parent}) - \sum_{k=1}^m \left(\frac{N_k}{N} \times \text{Entropy}(\text{Child}_k) \right) \quad (4.8)$$

$$\text{IG} = 1.561 - \left(\frac{4}{8} \times 1,5 \right) + \left(\frac{2}{8} \times 0 \right)$$

$$= 0,459$$

Berikut ini hasil perhitungan *Entropy* dan *Gini* untuk *node* 1.1

Tabel 4. 34 Hasil Perhitungan *Entropy* dan *Gini* pada *Node* 1.1
(*Decision Tree* 1)

Fitur		Jumlah	positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
Total		6	2	3	1	1.4591	
Buruk	≤ 0	4	2	1	1	1.5	0.4591
	> 0	2	0	2	0	0	
	≤ 0.31	5	2	2	1	1.52192	0.1908
	> 0.31	1	0	1	0	0	
	≤ 0.54	6	2	3	1	1.4591	4.6629
	> 0.54	0	0	0	0	0	
Layanan	≤ 0.31	5	2	2	1	1.52192	0.1908
	> 0.31	1	0	1	0	0	
	≤ 0	4	2	2	1	1.5	0.4591
	> 0	2	1	1	0	0	
	≤ 0.45	6	2	3	1	1.45914	4.6629
	> 0.45	0	0	0	0	0	

Berdasarkan tabel diatas Fitur "buruk" dengan *Entropy* ≤ 0 memiliki *Information Gini* tertinggi = 0,4591. Maka, untuk pemisahan selanjutnya dalam pohon keputusan, fitur terbaik adalah tempat dengan

Entropy ≤ 0 . Lanjutkan ke *node* berikutnya (anak kiri dan kanan dari *Entropy* ini), berikut ini tabel tabel untuk perhitungan *node* 1.2:

Tabel 4. 35 Dataset untuk Node 1.2 (Decision Tree 1)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
4	0	0	0	0	0.31	0	Positif
7	0.24	0	0	0	0	0	Positif
6	0.24	0	0	0	0	0	Netral
5	0	0	0	0	0.45	0	Negatif

- a) Pemilihan fitur secara acak, fitur selanjutnya adalah bagus dan kecewa
- b) Perhitungan *Entropy* Awal (*Parent Node*), untuk menghitung *Entropy* awal menggunakan persamaan berikut:

$$\text{Entropy} = - \sum_{i=1}^n p_i \log_2(p_i) \quad (4.9)$$

Total sampel 4: 2 Positif, 1 Negatif, dan 1 Netral:

$$\text{Entropy} = - \left(\frac{2}{4} \log_2 \frac{2}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right)$$

$$\begin{aligned} \text{Entropy} &= - (0,5 \times -1) + (0,25 \times -2) + (0,25 \times -2) \\ &= -(-0,5) + (-0,5) + (-0,5) \\ &= 1,5 \end{aligned}$$

- c) Penghitungan *Information gini* untuk setiap Fitur "bagus" dan kecewa seperti perhitungan diatas, Berikut ini hasil perhitungan dari pencarian tabel *node* 1.2 :

Tabel 4. 36 Hasil Perhitungan *Entropy* dan *Gini* pada Node 1.2

(Decision Tree 1)

Fitur	Jumlah	Positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
Total	4	2	1	1	1.5	

Fitur		Jumlah	Positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
Bagus	<=0	2	1	1	0	0	1.5
	>0	2	1	0	1	0	
	<=0.24	4	2	1	1	1.5	0
	>0.24	0	0	0	0	0	
Layanan	<=0	4	2	1	1	1.5	0
	>0	0	0	0	0	0	

Proses pada tabel 4.36 berhenti karena semua data dalam *node* sudah homogen atau nilai *Entropy* pada cabang-cabang hasil *split* menjadi 0, sehingga tidak ada lagi informasi baru yang dapat diperoleh dari pemisahan data.

2. Pembangunan *Decision Tree* untuk *Bootstrap* kedua

Berikut ini tabel *dataset* awal untuk perhitungan *node* 1:

Tabel 4. 37 *Dataset* awal perhitungan *Node* 1 (*Decision Tree* 2)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
7	0,2	0	0	0	0	0	Positif
2	0	0,31	0	0,31	0	0,1	Negatif
3	0	0,33	0	0	0	0	Netral
2	0	0,31	0	0,31	0	0,1	Negatif
8	0,23	0	0	0,5	0	0	Negatif
1	0	0,35	0,35	0	0	0	Positif
4	0	0	0	0	0,31	0	Positif
7	0,2	0	0	0	0	0	Positif

- a) Pemilihan Fitur secara acak, Memilih atribut secara acak disini atribut yang digunakan bagus, dan kecewa
- b) Perhitungan *Entropy* Awal (*Parent Node*), *Entropy* dihitung dengan menggunakan persamaan:

$$\text{Entropy} = - \sum_{i=t}^n p_i \log_2(p_i) \quad (4.10)$$

Bootstrap kedua memiliki 8 sampel: 3 Positif, 3 Negatif, dan 2 Netral

$$\text{Entropy} = - \left(\frac{4}{8} \log_2 \frac{4}{8} \right) + \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{1}{8} \log_2 \frac{1}{8} \right)$$

$$\begin{aligned}\text{Entropy} &= - (0,5 \times -1) + (0,375 \times -1,415) + (0,25 \times -3) \\ &= -(-0,5) + (-0,5306) + (-0,375) \\ &= 1,056\end{aligned}$$

c) Penghitungan *Information Gini* untuk setiap fitur "bagus" dan kecewa:

Pemisahan pada tempat dengan ambang batas 0,2:

- Anak kiri (tempat $\leq 0,2$): 8 sampel

Positif, 3 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{4}{8} \log_2 \frac{4}{8} \right) + \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{1}{8} \log_2 \frac{1}{8} \right)$$

$$\begin{aligned}\text{Entropy} &= - (0,5 \times -1) + (0,375 \times -1,415) + (0,25 \times -3) \\ &= -(-0,5) + (-0,5306) + (-0,375) \\ &= 1,056\end{aligned}$$

- Anak kanan (tempat $> 0,2$): 0 sampel

Jadi $\text{Entropy} = 0$

Untuk menghitung *Gini* menggunakan persamaan berikut:

$$\text{Information Gini} =$$

$$\text{Entropy}(\text{parent}) - \sum_{k=1}^m \left(\frac{N_k}{N} \times \text{Entropy}(\text{Child}_k) \right) \quad (4.11)$$

$$\begin{aligned}\text{IG} &= 1,056 - \left(\frac{8}{8} \times 1,4056 \right) + \left(\frac{0}{8} \times 0 \right) \\ &= 0\end{aligned}$$

Lakukan perhitungan untuk semua *Entropy* fitur bagus dan kecewa. Berikut ini tabel perhitungan *Entropy* dan perhitungan *Gini* pada fitur berikut ini:

Tabel 4. 38 Hasil Perhitungan *Entropy* dan *Information Gini* (*Decision Tree 2*)

Fitur		Juml ah	Positif (1)	Negati f (-)	Netral (0)	Entrop y	Gini
Total		8	4	3	1	1.0563	
Bagus	≤ 0.2	8	4	3	1	1.0563	0
	> 0.2	0	0	0	0	0	
	≤ 0	5	2	2	1	1.5219	0.530
	> 0	3	2	1	0	0	
	≤ 0.23	6	2	3	1	1.5917	0.311
	> 0.23	2	2	0	0	0	
Kecewa	≤ 0	6	4	1	1	1.2516	0.669
	> 0	2	0	2	0	0	
	≤ 0.1	8	4	3	1	1.0563	0
	> 0.1	0	0	0	0	0	

Berdasarkan tabel diatas Fitur dengan *Entropy* ≤ 0 memiliki *Information Gini* tertinggi = 0.669. Maka, untuk pemisahan pertama dalam pohon keputusan, fitur terbaik adalah kecewa dengan *Entropy* ≤ 0 . Lanjutkan ke *node* berikutnya (anak kiri dan kanan dari *Entropy* ini), berikut ini tabel *dataset* untuk perhitungan *node* 1.1:

Tabel 4. 39 *Dataset* untuk *Node* 1.1 (*Decision Tree 2*)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
7	0.24	0	0	0	0	0	Positif
3	0	0.33	0	0	0	0	Netral
8	0.23	0	0	0.54	0	0	Negatif
1	0	0.35	0.35	0	0	0	Positif
4	0	0	0	0	0.31	0	Positif

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
7	0.24	0	0	0	0	0	Positif

- 1) Pemilihan Fitur secara acak, fitur selanjutnya adalah tempat dan layanan
- 2) Perhitungan *Entropy* Awal (*Parent Node*), *Entropy* awal dihitung menggunakan persamaan berikut:

$$\text{Entropy} = - \sum_{i=t}^n p_i \log_2(p_i) \quad (4.12)$$

Total sampel 6: 2 Positif, 3 Negatif, dan 1 Netral:

$$\text{Entropy} = - \left(\frac{4}{6} \log_2 \frac{4}{6} \right) + \left(\frac{1}{6} \log_2 \frac{1}{6} \right) + \left(\frac{1}{6} \log_2 \frac{1}{6} \right)$$

$$\text{Entropy} = - (0,666 \times -0,585) + (0,166 \times -2,584) + (0,166 \times -2,584)$$

$$= -(-0,528) + (-0,5) + (-0,430)$$

$$= 1,459$$

- 3) Penghitungan *Information Gini* untuk fitur "tempat dan layanan" sama seperti perhitungan di atas dan berikut hasil perhitungan *node 1.1*:

Tabel 4. 40 Hasil Perhitungan *Entropy* dan *Gini* pada *Node 1.1*

(*Decision Tree 2*)

Fitur		Jumlah	Positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
Total		6	4	1	1	1.2516	
Tempat	<=0	4	3	1	0	0	1.2516
	>0	2	1	0	1	0	
	<=0,33	5	3	1	1	1.3709	0.1091
	>0,33	1	1	0	0	0	
	<=0,35	6	4	1	1	1.2516	0
	>0,35	0	0	0	0	0	
Layanan	<=0	5	3	1	1	1.3709	0.1091
	>0	1	1	0	0		

Fitur		Jumlah	Positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
	<=0,31	6	4	1	1	1.2516	0
	>0,31	0	0	0	0		

Berdasarkan tabel 4.40 karena hasil *Entropy* nya sama-sama 0 (homogen) maka perhitungannya selesai, tidak ada lagi informasi baru.

3. Pembangunan *Decision Tree* untuk *Bootstrap* ketiga

Berikut ini tabel *dataset* awal untuk perhitungan *node* 1:

Tabel 4. 41 *Dataset* awal perhitungan *Node* 1 (*Decision Tree* 3)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
6	0,2	0	0	0	0	0	Netral
5	0	0	0	0	0,5	0	Negatif
2	0	0,31	0	0,31	0	0,1	Negatif
1	0	0,35	0,35	0	0	0	Positif
3	0	0,33	0	0	0	0	Netral
8	0,23	0	0	0,5	0	0	Negatif
1	0	0,35	0,35	0	0	0	Positif
	0	0	0	0	0,31	0	Positif

- a) Pemilihan fitur secara acak, Fitur yang digunakan pada pembuatan pohon ke tiga iyalah fitur tempat dan buruk.
- b) Perhitungan *Entropy* Awal (*Parent Node*) menggunakan persamaan berikut:

$$\text{Entropy} = - \sum_{i=t}^n p_i \log_2(p_i) \quad (4.13)$$

Bootstrap ketiga memiliki 3 Positif, 3 Negatif, dan 2 Netral dengan total 8 sampel:

$$\text{Entropy} = - \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{3}{8} \log_2 \frac{3}{8} \right) + \left(\frac{2}{8} \log_2 \frac{2}{8} \right)$$

$$\text{Entropy} = - (0,375 \times -1,415) + (0,375 \times -1,415) + (0,25 \times -2)$$

$$= - (-0,530) + (- 0,530) + (- 0,5)$$

$$= 1,56$$

c) Penghitungan *Information Gini* untuk setiap fitur tempat dan fitur buruk

Fitur "tempat" dengan ambang batas 0:

- Anak kiri (tempat ≤ 0): 4 sampel

1 Positif, 2 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{1}{4} \log_2 \frac{1}{4} \right) + \left(\frac{2}{4} \log_2 \frac{2}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right)$$

$$\text{Entropy} = - (0,25 \times - 2) + (0,5 \times - 0) + 0,25 \times - 2)$$

$$= - (-0,5) + (- 0) + (- 0,5)$$

$$= 1,5$$

- Anak kanan (tempat > 0): 4 sampel

2 Positif, 1 Negatif, 1 Netral

$$\text{Entropy} = - \left(\frac{2}{4} \log_2 \frac{2}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right) + \left(\frac{1}{4} \log_2 \frac{1}{4} \right)$$

$$\text{Entropy} = - (0,5 \times - 0) + (0,25 \times - 2) + 0,25 \times - 2)$$

$$= - (-0,5) + (- 0) + (- 0,5)$$

$$= 1,5$$

Untuk menghitung *Gini* menggunakan persamaan berikut:

$$\text{Information Gini} =$$

$$\text{Entropy}(\text{parent}) - \sum_{k=1}^m \left(\frac{N_k}{N} \times \text{Entropy}(\text{Child}_k) \right)$$

$$\text{IG} = 1,56 - \left(\frac{4}{8} \times 1,5 \right) + \left(\frac{4}{8} \times 1,5 \right)$$

$$= 1,56 - (0,25 + 0,25)$$

$$= 0,061$$

Lakukan perhitungan untuk semua *Entropy* fitur tempat dan buruk. Berikut ini tabel hasil perhitungan *Entropy* dan perhitungan *Gini* pada fitur (tempat, buruk):

Tabel 4. 42 Hasil Perhitungan *Entropy* dan *Gini Node 1*
(*Decision Tree 3*)

Fitur		Jumlah	Positif (1)	Negatif (-)	Neutral (0)	Entropy	Gini
Total		8	3	3	2	1,5612	
Tempat	≤ 0	4	1	2	1	1,5	0,0612
	> 0	4	2	1	1	1,5	
	$\leq 0,31$	5	1	3	1	1,3709	0,7030
	$> 0,31$	3	2	0	1	0	
	$\leq 0,35$	8	3	3	2	1,5612	0
	$> 0,35$	0	0	0	0	0	
	$\leq 0,33$	6	1	3	2	1,5917	0,6691
	$> 0,33$	2	2	0	0	0	
	≤ 0	6	3	1	2	1,5917	0,6691
Buruk	> 0	2	0	2	0	0	
	$\leq 0,31$	7	3	2	2	1,556	0,1992
	$> 0,31$	1	0	1	0	0	
	$\leq 0,5$	6	3	1	2	1,591	0,6691
	$> 0,5$	0	0	0	0	0	

Berdasarkan tabel 4.42 fitur "tempat" dengan *Entropy* $\leq 0,31$ memiliki Information Gain tertinggi = 0,7030. Maka, untuk pemisahan pertama dalam pohon keputusan, fitur terbaik adalah tempat dengan *Entropy* $\leq 0,31$. Lanjutkan ke *node* berikutnya (anak kiri dan kanan dari *Entropy* ini), berikut ini tabel untuk perhitungan *node* 1.1:

Tabel 4. 43 Dataset untuk Node 1.1 (Decision Tree 3)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
6	0.24	0	0	0	0	0	Netral
5	0	0	0	0	0.45	0	Negatif
2	0	0.31	0	0.31	0	0.41	Negatif
8	0.23	0	0	0.54	0	0	Negatif
4	0	0	0	0	0.31	0	Positif

Fitur acak yang digunakan pada pembuatan pohon ke tiga iyalah fitur buruk dan kecewa, kemudian perhitungan *Entropy* awal. Dengan mengulangi proses atau cara kerja di atas untuk setiap *Bootstrap* pada *Random Forest*, maka hasil perhitungan fitur dengan *node* 1.1 dapat dilihat sebagai berikut:

Tabel 4. 44 Hasil Perhitungan *Entropy* dan *Gini* pada Node 1.1
(Decision Tree 3)

Fitur		Jumlah	positif (1)	Negatif (-)	Netral (0)	Entropy	Gini
Total		5	1	3	1	1.3709	
Buruk	<=0	6	3	1	2	1.4591	-0.3800
	>0	2	0	2	0	0	
	<=0.31	7	3	2	2	1.5566	-0.8083
	>0.31	1	0	1	0	0	
	<=0.54	6	3	1	2	1.4591	-0.3800
	>0.54	0	0	0	0	0	
Kecewa	<=0	4	1	2	1	1.5	0.1709
	>0	1	0	1	0	0	
	<=0.41	5	1	3	1	1.3709	0
	<0.41	0	0	0	0	0	

Berdasarkan tabel 4.44 Fitur "kecewa" dengan *Entropy* ≤ 0 memiliki Information Gain tertinggi = 0.170. Maka, untuk pemisahan selanjutnya dalam pohon keputusan, fitur terbaik adalah kecewa dengan

$Entropy \leq 0$. Lanjutkan ke *node* berikutnya (anak kiri dan kanan dari *Entropy* ini), berikut ini tabel tabel untuk perhitungan *node* 1.2:

Tabel 4. 45 Dataset untuk Node 1.2 (Decision Tree 3)

No	bagus	tempat	sangat	buruk	layanan	kecewa	Label
6	0.24	0	0	0	0	0	Netral
5	0	0	0	0	0.45	0	Negatif
8	0.23	0	0	0.54	0	0	Negatif
4	0	0	0	0	0.31	0	Positif

Penghitungan *Information Gini* untuk fitur acak "bagus dan sangat" sama seperti perhitungan di atas dan hasil perhitungan *node* 1.2:

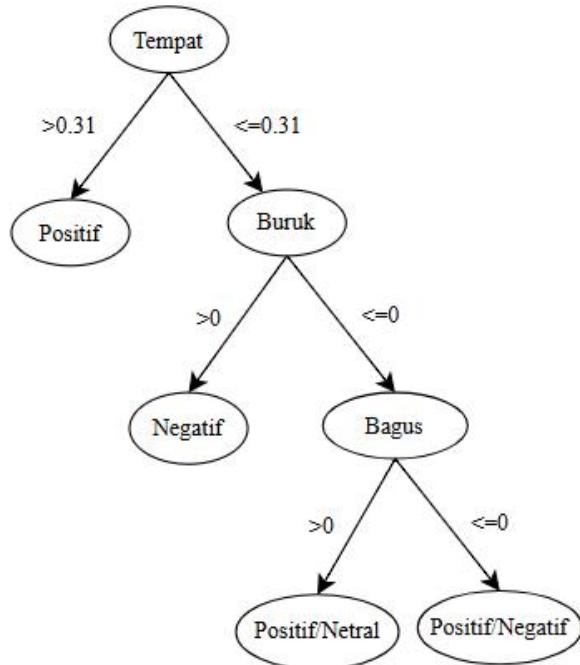
Tabel 4. 46 Hasil Perhitungan *Entropy* dan *Gini* pada Node 1.2
(Decision Tree 3)

Fitur		Jumlah	Positif (1)	Negatif (-)	Netral (0)	Entropy	<i>Gini</i>
Total		4	1	2	1	1.5	
Bagus	≤ 0.24	4	1	2	1	1.5	0
	> 0.24	0	0	0	0	0	
	≤ 0	2	1	1	0	0	1.5
	> 0	2	1	0	1	0	
Sangat	≤ 0.23	3	1	2	0	0	1.5
	> 0.23	1	0	0	1	0	
	≤ 0	4	1	2	1	1.5	0
	> 0	0	0	0	0	0	

Pada tabel 4.36 proses berhenti karena semua simpul sudah mencapai kondisi homogen ($Entropy = 0$) atau tidak ada lagi pemisahan yang dapat meningkatkan kemurnian data secara signifikan. Dengan demikian, pohon keputusan telah selesai dibangun dan siap digunakan untuk klasifikasi.

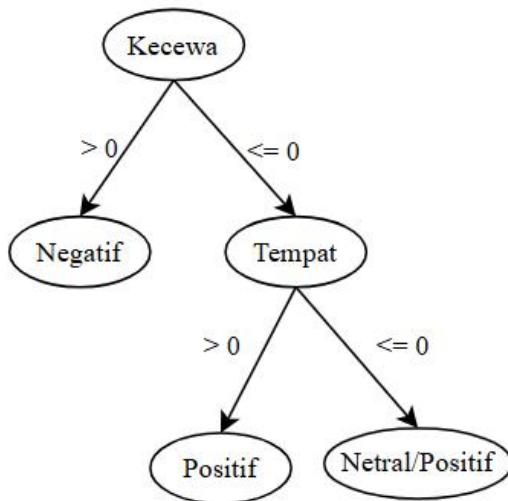
c. Voting (Prediksi dengan *Random Forest*)

Gambar di bawah ini menunjukkan proses prediksi menggunakan metode *Random Forest* dengan pendekatan voting. Dalam metode ini, data uji yang terdiri dari sejumlah fitur seperti "bagus", "tempat", "sangat", "buruk", "layanan", dan "kecewa", akan diklasifikasikan oleh beberapa pohon keputusan (*Decision Tree*) yang telah dibentuk pada tahap pelatihan:



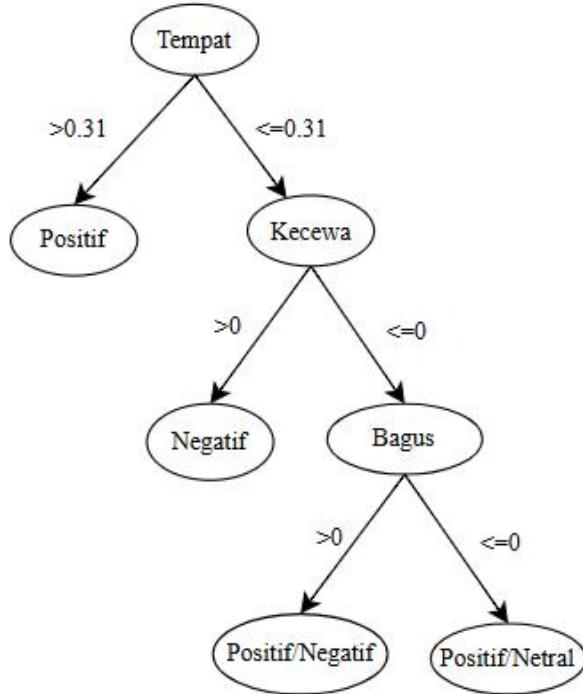
Gambar 4. 1 Pohon Pertama

Gambar 4.1 merupakan pohon keputusan yang mengklasifikasikan sentimen berdasarkan atribut "Tempat", "Buruk", dan "Bagus". Jika nilai "Tempat" > 0.31 , maka hasilnya "Positif". Jika ≤ 0.31 , maka dilanjutkan ke atribut "Buruk". Jika "Buruk" > 0 , maka hasilnya "Negatif", sedangkan jika ≤ 0 , maka diperiksa atribut "Bagus". Jika "Bagus" > 0 maka hasilnya "Positif/Netral", dan jika ≤ 0 maka hasilnya "Positif/Negatif".



Gambar 4. 2 Pohon Kedua

Gambar 4.2 menunjukkan pohon keputusan yang memanfaatkan atribut "Kecewa" dan "Tempat" untuk menentukan sentimen. Jika nilai "Kecewa" lebih dari 0, maka sentimen diklasifikasikan sebagai "Negatif". Jika "Kecewa" kurang dari atau sama dengan 0, maka dilanjutkan ke atribut "Tempat". Jika nilai "Tempat" lebih dari 0, maka hasilnya adalah "Positif", sedangkan jika nilainya kurang dari atau sama dengan 0, maka sentimennya adalah "Netral/Positif".



Gambar 4. 3 Pohon Ketiga

Gambar tersebut adalah pohon keputusan yang mengklasifikasikan sentimen berdasarkan atribut "Tempat", "Kecewa", dan "Bagus". Jika nilai "Tempat" lebih dari 0.31, maka sentimennya "Positif". Jika kurang dari atau sama dengan 0.31, maka dilanjutkan ke atribut "Kecewa". Jika nilai "Kecewa" lebih dari 0, hasilnya "Negatif". Jika nilainya kurang dari atau sama dengan 0, maka diperiksa atribut "Bagus". Jika nilai "Bagus" lebih dari 0, maka hasilnya "Positif/Negatif", dan jika kurang dari atau sama dengan 0, hasilnya "Positif/Netral".

Pada gambar diatas setiap pohon memberikan hasil klasifikasi tersendiri terhadap data uji, dan hasil akhir ditentukan berdasarkan voting mayoritas dari semua pohon. Maka hasil akhir berdasarkan voting dan prinsip penentuan adalah Positif, Perlu analisis pakar lebih lanjut untuk mengklarifikasi jika ingin hasil tunggal dan pasti. Gambar ini mengilustrasikan bagaimana masing-masing pohon memberikan prediksi dan bagaimana sistem menentukan keputusan akhir berdasarkan suara terbanyak.

Pada implementasi dalam penelitian ini, *Random Forest* dibangun secara manual tanpa menggunakan *library*. Adapun parameter-parameter yang digunakan seperti:

- 1) Jumlah pohon yang dibangun pada sistem ada 10 pohon keputusan yang masing-masing dilatih menggunakan *bootstrap* sampling.
- 2) Kedalaman maksimum, untuk mencegah model terlalu menyesuaikan diri dengan data pelatihan, setiap pohon dibatasi hanya sampai kedalaman maksimal 5 level.
- 3) Pemilihan fitur secara acak, dalam setiap pohon pemilihan fitur dilakukan secara acak dari subset fitur yang berukuran n, di mana n adalah jumlah total fitur. Namun, pemilihan acak fitur pada pengimplementasian hanya dilakukan satu kali untuk setiap pohon, bukan pada setiap *node*.
- 4) Pada setiap pohon, pelatihan dilakukan dengan menggunakan teknik pengambilan sampel acak dengan pengembalian, yang memastikan bahwa setiap pohon belajar dari subset data yang unik, meningkatkan keberagaman antar pohon, dan memperkuat akurasi keseluruhan model.

- 5) Voting untuk prediksi klasifikasi, hasil prediksi dari setiap pohon dikombinasikan dengan teknik voting untuk mendapatkan prediksi akhir. Pada klasifikasi, setiap pohon memberikan satu suara, dan kelas yang mendapatkan suara terbanyak dianggap sebagai hasil akhir dari model *Random Forest*.

Dengan menggunakan pendekatan manua, implementasi *Random Forest* bertujuan untuk memberikan pemahaman yang lebih mendalam tentang cara kerja algoritma dan bagaimana parameter-parameter tertentu memengaruhi kinerja model.

4.8 Pengujian Akurasi

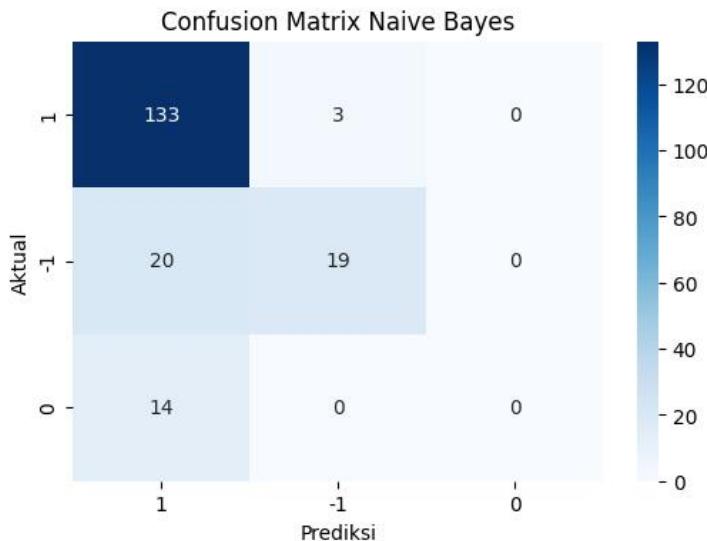
Setelah dilakukan proses tahapan *Preprocessing*, pelabelan, proses klasifikasi *Naïve Bayes* dan *Random Forest* selanjutnya mengukur hasil pengujian *cofusion matrix*. Rumus yang digunakan adalah evaluasi per kelas, yang disesuaikan dengan struktur tabel *Multiclass Confusion Matrix* sebagaimana dijabarkan pada Tabel 2.2.

Evaluasi yang digunakan dalam penelitian ini sesuai dengan persamaan (2.18)-(2.20) yang merupakan adaptasi khusus untuk konteks klasifikasi *multiclass*, yang mempertimbangkan karakteristik data dan kompleksitas (kerumitan) yang dihadapi. Hal ini membedakan evaluasi ini dari rumus evaluasi standar yang umumnya digunakan pada kasus klasifikasi biner. Dengan penerapan metode evaluasi per kelas, penelitian ini dapat memberikan penilaian yang lebih rinci terhadap kinerja model klasifikasi pada masing-masing kelas, sehingga dapat mengidentifikasi performa model secara spesifik baik pada kelas mayoritas maupun minoritas. Evaluasi per kelas juga memungkinkan analisis yang lebih adil dan menyeluruh, terutama dalam kondisi distribusi data yang tidak seimbang.

4.8.1 Akurasi Sebelum *Oversampling*

- a) Hasil Pengujian *Naïve Bayes*

Dengan menggunakan Jumlah data uji mewakili kelas klasifikasi positif, negatif dan netral dapat dilihat pada gambar 4.4:



Gambar 4. 4 *Confusion Matrix Sebelum Oversampling Naïve Bayes*

Pada gambar 4.4 menunjukkan *confusion matrix* dari model *Naïve Bayes*, yang mengevaluasi kinerja klasifikasi. Dari data terlihat bahwa algoritma *Naïve Bayes* memiliki kinerja yang baik dalam mengklasifikasikan sentimen positif, dengan 133 prediksi benar dari total 136 data. Namun, terdapat kesalahan dalam mengklasifikasikan data negatif dan netral. Misalnya, dari 39 data negatif, hanya 19 yang terkласifikasi dengan benar, sedangkan 20 sisanya justru diprediksi sebagai positif. Selain itu, seluruh data netral (14 data) salah diklasifikasikan sebagai positif.

Kesalahan ini terjadi karena *Naïve Bayes* sangat bergantung pada distribusi probabilitas kata dalam masing-masing kelas. Jika kata-kata dalam data negatif dan netral memiliki kemiripan atau kemunculan yang sering juga terdapat pada data positif, maka model cenderung mengklasifikasikannya sebagai positif. Selain itu, jumlah data netral dan negatif yang lebih sedikit dibandingkan data positif juga dapat menyebabkan model bias terhadap kelas mayoritas (positif). Hal ini menunjukkan bahwa *Naïve Bayes* kurang mampu menangkap nuansa atau konteks emosional yang lebih halus, terutama untuk membedakan antara sentimen netral dan negatif. Pada gambar di atas merupakan gambar dari hasil *confusion matrix* yang akan dilakukan perhitungan *accuracy*, *Precision*, *Recall*, dan *F1-Score* sebagai berikut:

$$Accuracy = \frac{133 + 19 + 0}{133 + 3 + 20 + 19 + 14} \times 100\%$$

$$= \frac{152}{189} \times 100\% = 80,42\%$$

1) Perhitungan kelas Positif

$$Precision = \frac{\text{TPos}}{\text{TPos} + \text{FNegPos} + \text{FNetPos}} \times 100\%$$

$$= \frac{133}{133 + 20 + 14} = \frac{133}{167} = 0,80\%$$

$$Recall = \frac{\text{TPos}}{\text{TPos} + \text{FPoSNeg} + \text{FPoSNet}} \times 100\%$$

$$= \frac{133}{133+3+0} = \frac{133}{136} = 0,98\%$$

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

$$= 2 \times \frac{0,80 \times 0,98}{0,80 + 0,98} = 0,88\%$$

2) Perhitungan kelas Negatif

$$Precision = \frac{\text{TNeg}}{\text{TNeg} + \text{FPoSNeg} + \text{FNetNeg}} \times 100\%$$

$$= \frac{19}{19+3+0} = \frac{19}{22} = 0,86\%$$

$$Recall = \frac{\text{TNeg}}{\text{TNeg} + \text{FNegPos} + \text{FNegNet}} \times 100\%$$

$$= \frac{19}{19+20+0} = \frac{19}{39} = 0,49\%$$

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

$$= 2 \times \frac{0,86 \times 0,49}{0,86 + 0,49} = 0,62\%$$

3) Perhitungan kelas Netral

$$\text{Precision} = \frac{\text{TNet}}{\text{TNet} + \text{FPosNet} + \text{FNegNet}} \times 100\%$$

$$= \frac{0}{0} = 0 \%$$

$$\text{Recall} = \frac{\text{TNet}}{\text{TNet} + \text{FNetPos} + \text{FNetNeg}} \times 100\%$$

$$= \frac{0}{0 + 14 + 0} = \frac{0}{14} = 0 \%$$

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

$$= 2 \times \frac{0,1 \times 0}{0,1 + 0} = 0 \%$$

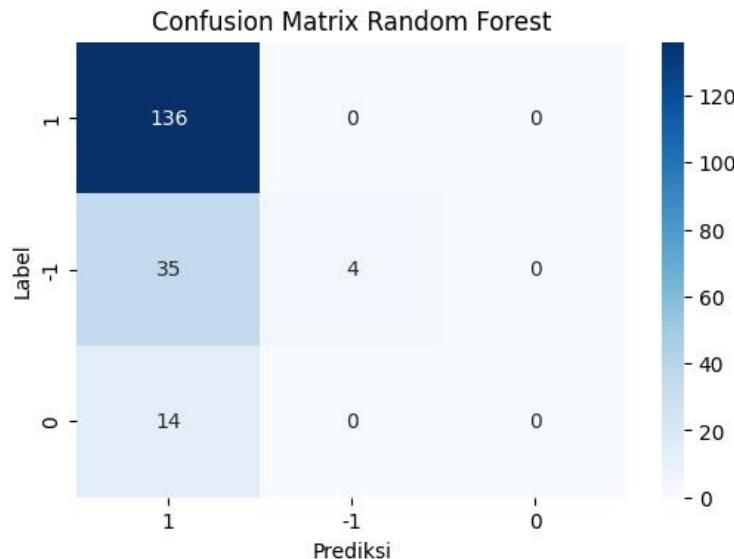
Tabel 4. 47 Hasil Pengujian *Naïve Bayes* Sebelum *Oversampling*

Kelas	Precision	Recall	F1-score
Positif	80 %	98 %	88 %
Negatif	86 %	46 %	62%
Netral	0%	0 %	0 %

Tabel hasil pengujian *Random Forest* menunjukkan bahwa untuk kelas Positif, model memiliki *precision* 80%, yang berarti 80% dari prediksi Positif yang dibuat oleh model benar-benar Positif. *Recall* mencapai 98%, menunjukkan bahwa model dapat mengenali hampir semua data Positif yang ada. *F1-score* untuk kelas ini adalah 88%, mencerminkan keseimbangan yang baik antara *precision* dan *recall*. Pada kelas Negatif, meskipun *precision* tinggi 86%, *recall* sangat rendah, hanya 0,46%, yang berarti model gagal mengenali sebagian besar data Negatif. *F1-score* untuk kelas ini sangat rendah, 0,62%, yang mengindikasikan ketidakseimbangan yang besar antara *precision* dan *Recall*. Sedangkan untuk kelas Netral, semua metrik adalah 0%, yang berarti model tidak mampu memprediksi atau mengenali kelas Netral sama sekali. Ini menunjukkan performa yang buruk dalam mendeteksi data dengan label Netral.

b) Hasil Pengujian *Random Forest*

Dengan menggunakan Jumlah data uji mewakili kelas klasifikasi positif, negatif dan netral dapat dilihat pada gambar 4.5:



Gambar 4. 5 *Confusion Matrix* Sebelum *Oversampling Random Forest*

Pada gambar 4.5 menunjukkan *confusion matrix* dari model *Random Forest*, yang digunakan untuk mengevaluasi kinerja klasifikasi. Model memiliki performa sangat baik dalam mengklasifikasikan kelas Positif, dengan 136 data berhasil diklasifikasikan dengan benar dan tidak ada kesalahan klasifikasi ke kelas lain. Namun, untuk kelas Negatif, hanya 4 data yang berhasil dikenali, sementara 35 lainnya salah diklasifikasikan sebagai Positif. Sedangkan pada kelas Netral, seluruh 14 data diklasifikasikan salah sebagai Positif, tanpa ada satu pun yang terdeteksi dengan benar.

Analisis ini menunjukkan bahwa model *Random Forest* cenderung bias ke kelas positif, karena distribusi data yang tidak seimbang, di mana kelas positif memiliki jumlah data yang jauh lebih banyak dibandingkan kelas negatif dan netral. Selain itu, fitur atau pola yang membedakan kelas negatif dan netral mungkin kurang kuat atau sulit dipelajari oleh model, sehingga menyebabkan kesulitan dalam membedakan kelas-kelas tersebut. Model ini perlu diperbaiki dengan teknik penyeimbangan data atau menggunakan fitur yang lebih informatif agar dapat mengenali kelas negatif dan netral dengan lebih baik. Gambar 4.5

merupakan gambar dari hasil *confusion matrix* yang akan dilakukan perhitungan *accuracy*, *precision*, *recall*, dan *F1-score*:

$$\text{Accuracy} = \frac{136 + 4 + 0}{136 + 35 + 4 + 14 + 14} \times 100\%$$

$$= \frac{39}{408} \times 100\% = 74,07\%$$

1) Perhitungan kelas Positif

$$\text{Precision} = \frac{\text{TPos}}{\text{TPos} + \text{FNegPos} + \text{FNetPos}} \times 100\%$$

$$= \frac{136}{136 + 35 + 14} = \frac{136}{185} = 0,74\%$$

$$\text{Recall} = \frac{\text{TPos}}{\text{TPos} + \text{FPosNeg} + \text{FPosNet}} \times 100\%$$

$$= \frac{136}{136 + 0 + 0} = \frac{136}{136} = 0,1 = 100\%$$

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

$$= 2 \times \frac{0,74 \times 0,1}{0,74 + 0,1} = 0,85\%$$

2) Perhitungan kelas Negatif

$$\text{Precision} = \frac{\text{TNeg}}{\text{TNeg} + \text{FPosNeg} + \text{FNetNeg}} \times 100\%$$

$$= \frac{4}{4 + 0 + 0} = \frac{4}{4} = 0,1 = 100\%$$

$$\text{Recall} = \frac{\text{TNeg}}{\text{TNeg} + \text{FNegPos} + \text{FNegNet}} \times 100\%$$

$$= \frac{4}{4 + 35 + 0} = \frac{4}{39} = 0,10\%$$

$$\begin{aligned}
 F1\text{-score} &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \\
 &= 2 \times \frac{0,1 \times 0,10}{0,1 + 0,10} = 0,19\%
 \end{aligned}$$

3) Perhitungan kelas Netral

$$\begin{aligned}
 Precision &= \frac{TNet}{TNet + FPosNet + FNegNet} \times 100\% \\
 &= \frac{0}{0 + 0 + 0} = 0\%
 \end{aligned}$$

$$\begin{aligned}
 Recall &= \frac{TNet}{TNet + FNetPos + FNetNeg} \times 100\% \\
 &= \frac{0}{0 + 14 + 0} = \frac{0}{14} = 0\%
 \end{aligned}$$

$$\begin{aligned}
 F1\text{-score} &= 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \\
 &= 2 \times \frac{0,1 \times 0}{0,1 + 0} = 0\%
 \end{aligned}$$

Tabel 4. 48 Hasil Pengujian *Random Forest* Sebelum *Oversampling*

Kelas	Precision	Recall	F1-score
Positif	74 %	100 %	85 %
Negatif	100 %	10 %	19 %
Netral	0%	0 %	0 %

Tabel hasil pengujian algoritma *Random Forest* menunjukkan bahwa model memiliki kinerja terbaik pada kelas Positif, dengan *precision* sebesar 74%, *recall* 100%, dan *F1-score* 85%. Hal ini menunjukkan bahwa model sangat baik dalam mengenali data yang termasuk dalam kelas Positif. Pada kelas Negatif, *precision* mencapai 100%, namun *recall* hanya 10%, sehingga *F1-score* tergolong rendah, yaitu 19%. Artinya, meskipun semua prediksi untuk kelas Negatif benar, model hanya mampu mengenali sebagian kecil dari keseluruhan data Negatif yang ada. Untuk kelas Netral, *precision* juga tercatat 0%, namun *recall* 0% dan *F1-*

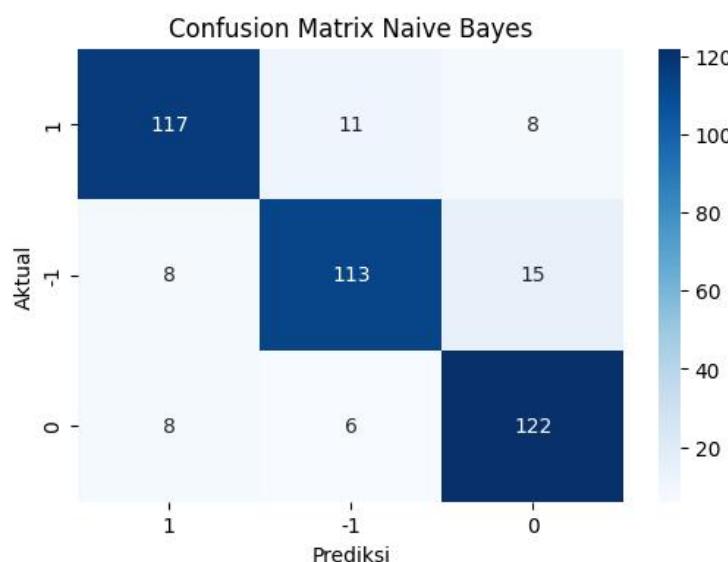
score 0%. Ini menunjukkan bahwa tidak ada data yang diklasifikasikan ke dalam kelas Netral, sehingga kinerja model pada kelas ini dapat dikatakan gagal. Hal ini menunjukkan ketidakseimbangan dalam klasifikasi, di mana model lebih cenderung mengenali kelas mayoritas.

4.8.2 Akurasi Sesudah *Oversampling*

Berikut ini penjelasan hasil akurasi *Naïve Bayes* dan *Random Forest*:

a) Hasil Pengujian *Naïve Bayes*

Untuk mengevaluasi performa model *Naïve Bayes* dalam klasifikasi sentimen ulasan wisatawan, dilakukan pengujian menggunakan *confusion matrix*. Gambar berikut menunjukkan hasil *confusion matrix* yang menggambarkan jumlah prediksi benar dan salah untuk setiap kategori sentimen.



Gambar 4. 6 *Confusion Matrix* Sesudah *Oversampling* *Naïve Bayes*

Gambar 4.6 menunjukkan *Confusion Matrix* dari model *Naïve Bayes* yang digunakan untuk klasifikasi tiga kelas. Model menunjukkan performa yang cukup baik dalam mengklasifikasikan semua kelas, meskipun masih terdapat kesalahan prediksi. Untuk kelas positif, sebanyak 117 data berhasil diklasifikasikan dengan benar, namun masih terdapat 11 data yang salah diprediksi sebagai kelas negatif dan 8 data sebagai kelas netral. Pada kelas negatif, 113 data berhasil dikenali dengan benar, namun 8 data salah diklasifikasikan sebagai positif dan 15 lainnya sebagai netral. Sedangkan untuk kelas netral, model dapat mengklasifikasikan 122

data dengan tepat, namun terdapat 8 kesalahan prediksi ke kelas positif dan 6 ke kelas negatif.

Kesalahan ini menunjukkan bahwa model mengalami kesulitan dalam membedakan antara kelas yang memiliki karakteristik atau fitur yang mirip, khususnya antara kelas negatif dan netral. Secara umum, tidak ada indikasi kuat bahwa model bias terhadap salah satu kelas, yang mengindikasikan bahwa distribusi data relatif seimbang dan fitur yang digunakan cukup representatif. Dari matriks ini, kita dapat mengevaluasi metrik performa seperti akurasi, presisi, *recall*, dan *F1-score* untuk menilai efektivitas model dalam mengklasifikasikan data berikut ini tabel hasil perhitungan:

$$\begin{aligned} \text{Accuracy} &= \frac{117 + 113 + 122}{117 + 11 + 8 + 8 + 113 + 15 + 8 + 6 + 122} \times 100\% \\ &= \frac{352}{408} \times 100\% = 86,27\% \end{aligned}$$

1) Perhitungan kelas Positif

$$\begin{aligned} \text{Precision} &= \frac{\text{TPos}}{\text{TPos} + \text{FNegPos} + \text{FNetPos}} \times 100\% \\ &= \frac{117}{117 + 8 + 8} = \frac{117}{133} = 0,88\% \\ \text{Recall} &= \frac{\text{TPos}}{\text{TPos} + \text{FPoS Neg} + \text{FPoS Net}} \times 100\% \\ &= \frac{113}{113 + 11 + 8} = \frac{113}{127} = 0,86\% \\ \text{F1-score} &= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \times 100\% \\ &= 2 \times \frac{0,88 \times 0,86}{0,88 + 0,86} = 0,87\% \end{aligned}$$

2) Perhitungan kelas Negatif

$$\text{Precision} = \frac{\text{TNeg}}{\text{TNeg} + \text{FPoS Neg} + \text{FNetNeg}} \times 100\%$$

$$= \frac{122}{122 + 8 + 15} = \frac{122}{145} = 0,84\%$$

$$Recall = \frac{T_{Neg}}{T_{Neg} + F_{NegPos} + F_{NegNet}} \times 100\%$$

$$= \frac{113}{113 + 8 + 615} = \frac{113}{636} = 0,83\%$$

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

$$= 2 \times \frac{0,84 \times 0,83}{0,84 + 0,83} = 0,85 \%$$

3) Perhitungan kelas Netral

$$Precision = \frac{T_{Net}}{T_{Net} + F_{PosNet} + F_{NegNet}} \times 100\%$$

$$= \frac{122}{122 + 8 + 15} = \frac{122}{145} = 0,84\%$$

$$Recall = \frac{T_{Net}}{T_{Net} + F_{NetPos} + F_{NetNeg}} \times 100\%$$

$$= \frac{122}{122 + 8 + 6} = \frac{122}{139} = 0,90\%$$

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

$$= 2 \times \frac{0,84 \times 0,90}{0,84 + 0,90} = 0,87\%$$

Tabel 4. 49 Hasil Pengujian *Naïve Bayes* Sesudah *Oversampling*

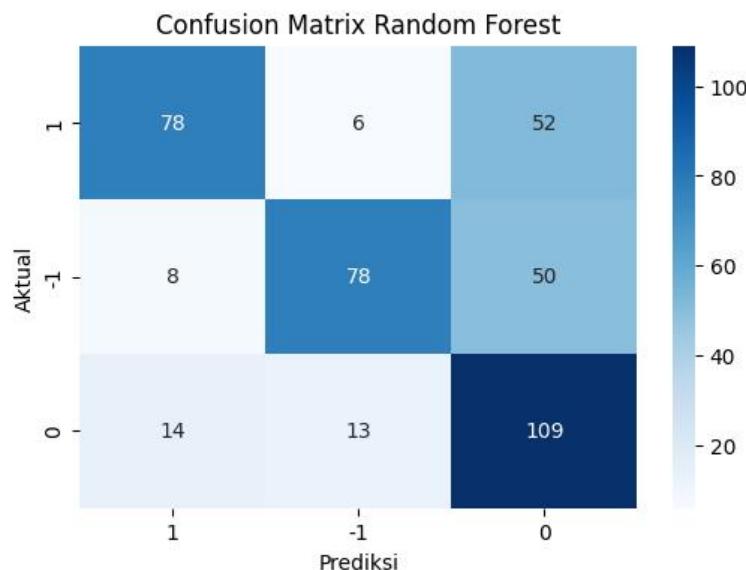
Kelas	Precision	Recall	F1-score
Positif	88%	86%	87%
Negatif	87%	83%	85%
Netral	84%	90%	87%

Hasil evaluasi model menunjukkan kinerja yang cukup konsisten di semua kelas. Untuk kelas Positif, model memiliki *precision* 88% dan *recall* 86%, yang

berarti prediksi positif cukup akurat dan sebagian besar data positif berhasil dikenali, dengan F1-score 87% sebagai keseimbangan antara keduanya. Kelas Negatif memiliki *precision* 87% dan *recall* 83%, menunjukkan performa yang cukup stabil dengan F1-score 85%. Sementara itu, untuk kelas Netral, *precision* 84% dan *recall* 90% menunjukkan bahwa model sangat baik dalam mengenali data netral, dengan F1-score tertinggi yaitu 87%. Secara keseluruhan, model memiliki performa yang cukup baik dan seimbang dalam mengklasifikasikan ketiga kelas.

b) Hasil Pengujian *Random Forest*

Berikut adalah *confusion matrix* dari model *Random Forest* yang digunakan untuk klasifikasi tiga kelas:



Gambar 4.7 *Confusion Matrix Sesudah Oversampling Random Forest*

Gambar 4.7 menggambarkan performa model *Random Forest* menunjukkan kinerja klasifikasi yang cukup baik, namun masih terdapat sejumlah kesalahan yang perlu diperhatikan. Untuk kelas positif, sebanyak 78 data berhasil diklasifikasikan dengan benar, namun 6 data salah diprediksi sebagai negatif dan 52 data lainnya sebagai netral. Pada kelas negatif, juga terdapat 78 data yang terklasifikasi dengan tepat, namun masih ada 8 data yang salah diklasifikasikan sebagai positif dan 50 sebagai netral. Sementara itu, pada kelas netral, model menunjukkan kinerja yang paling tinggi, dengan 109 data berhasil

diklasifikasikan dengan benar, meskipun masih terdapat 14 data yang salah diprediksi sebagai positif dan 13 sebagai negatif.

Hasil ini menunjukkan bahwa model cenderung mengalami kesulitan dalam membedakan data dari kelas positif dan negatif terhadap kelas netral, yang ditandai dengan banyaknya data dari kedua kelas tersebut yang salah diklasifikasikan ke dalam kelas netral. Hal ini dapat menjadi indikasi bahwa pola atau fitur yang membedakan antara kelas-kelas tersebut belum sepenuhnya tertangkap oleh model. Berikut ini tabel hasil perhitungan *accuracy*, *precision*, *recall*, dan *F1-score*:

$$\text{Accuracy} = \frac{78 + 78 + 109}{78 + 6 + 52 + 8 + 78 + 50 + 14 + 13 + 109} \times 100\%$$

$$= \frac{265}{408} \times 100\% = 64,95\%$$

1) Perhitungan kelas Positif

$$\text{Precision} = \frac{\text{TPos}}{\text{TPos} + \text{FNegPos} + \text{FNetPos}} \times 100\%$$

$$= \frac{78}{78 + 8 + 14} = \frac{78}{100} = 0,78\%$$

$$\text{Recall} = \frac{\text{TPos}}{\text{TPos} + \text{FPosNeg} + \text{FPosNet}} \times 100\%$$

$$= \frac{78}{78 + 6 + 52} = \frac{78}{139} = 0,57\%$$

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

$$= 2 \times \frac{0,78 \times 0,57}{0,78 + 0,57} = 0,66\%$$

2) Perhitungan kelas Negatif

$$\text{Precision} = \frac{\text{TNeg}}{\text{TNeg} + \text{FPosNeg} + \text{FNetNeg}} \times 100\%$$

$$= \frac{78}{78 + 6 + 13} = \frac{78}{97} = 0,80\%$$

$$Recall = \frac{T_{Neg}}{T_{Neg} + F_{NegPos} + F_{NegNet}} \times 100\%$$

$$= \frac{78}{78 + 8 + 50} = \frac{78}{136} = 0,57\%$$

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

$$= 2 \times \frac{0,80 \times 0,57}{0,80 + 0,57} = 0,67\%$$

3) Perhitungan kelas Netral

$$Precision = \frac{T_{Net}}{T_{Net} + F_{PosNet} + F_{NegNet}} \times 100\%$$

$$= \frac{109}{109 + 52 + 50} = \frac{109}{211} = 0,52\%$$

$$Recall = \frac{T_{Net}}{T_{Net} + F_{NetPos} + F_{NetNeg}} \times 100\%$$

$$= \frac{109}{109 + 14 + 13} = \frac{109}{136} = 0,80\%$$

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

$$= 2 \times \frac{0,52 \times 0,80}{0,52 + 0,80} = 0,63\%$$

Tabel 4. 50 Hasil pengujian *Random Forest* Sesudah *Oversampling*

Kelas	Precision	Recall	F1-score
Positif	78 %	57%	66%
Negatif	80%	57%	67%
Netral	52%	80%	63%

Hasil evaluasi menunjukkan bahwa Untuk kelas Positif, *precision* 78% berarti dari semua prediksi positif, 78% benar, namun *recall* hanya 57%,

menunjukkan bahwa hanya 57% data positif yang berhasil dikenali, menghasilkan F1-score 66%. Kelas Negatif memiliki *precision* 80% dan *recall* 57%, dengan F1-score 67%, menunjukkan pola serupa. Sementara itu, pada kelas Netral, *recall* cukup tinggi yaitu 80%, artinya sebagian besar data netral berhasil dikenali, meskipun *precision*-nya lebih rendah di angka 52%, menghasilkan F1-score 63%. Secara keseluruhan, model cukup baik dalam mengenali kelas netral, tetapi masih perlu peningkatan dalam ketepatan prediksi terutama pada kelas negatif dan positif.

4.9 Visualisasi

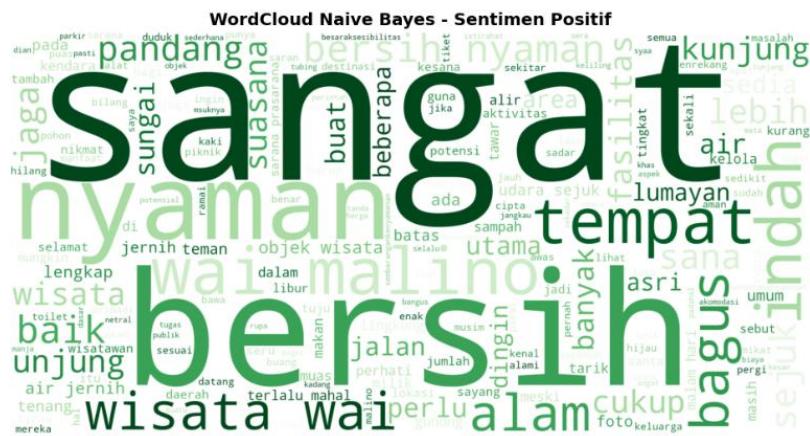
Untuk memahami lebih lanjut hasil analisis menggunakan algoritma *Naïve Bayes* dan *Random Forest*, dilakukan visualisasi data dalam bentuk *Word Cloud*. *Word Cloud* merupakan teknik visualisasi teks yang menampilkan kata-kata dengan ukuran yang bervariasi berdasarkan frekuensi atau tingkat kepentingannya dalam data.

Pada penelitian ini, *Word Cloud* digunakan untuk menggambarkan pola kata yang sering muncul dalam dataset setelah proses klasifikasi. Dengan menggunakan *Word Cloud*, kita dapat melihat kata-kata dominan yang berkontribusi dalam hasil prediksi algoritma, baik pada model *Naïve Bayes* maupun *Random Forest*. Hal ini membantu dalam menganalisis karakteristik data serta faktor yang mempengaruhi klasifikasi. Berikut adalah visualisasi *Word Cloud* untuk kedua algoritma yang digunakan:

4.9.1 *Word Cloud* Algoritma *Naïve Bayes*

Berikut ini adalah *Word Cloud* yang menggambarkan hasil analisis sentimen menggunakan algoritma *Naïve Bayes* berdasarkan tiga kategori positif, negatif, dan netral. Visualisasi ini menunjukkan kata-kata yang paling dominan dalam setiap kategori, memberikan pemahaman yang lebih jelas tentang opini masyarakat terhadap objek wisata Wai Malino:

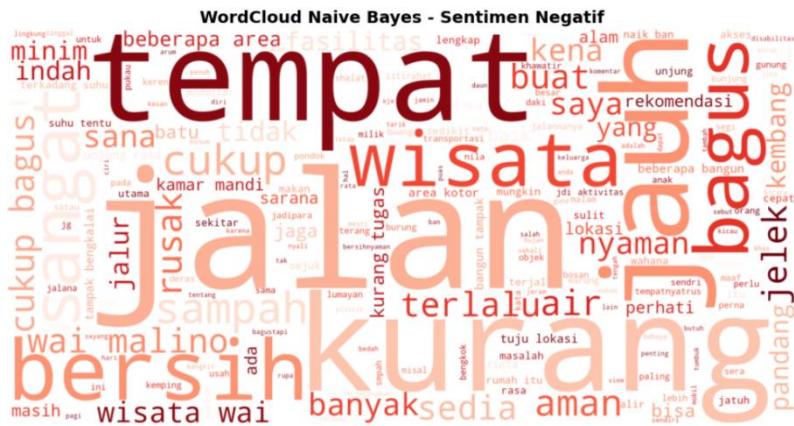
a. *Word Cloud* Sentimen Positif



Gambar 4. 8 *Word Cloud* Sentimen Positif *Naïve Bayes*

Gambar 4.8 merupakan sebuah *Word Cloud* yang menggambarkan hasil analisis sentimen positif menggunakan algoritma *Naïve Bayes* terhadap ulasan wisata. *Word Cloud* ini menampilkan kata-kata yang paling sering muncul dalam ulasan berisi sentimen positif, di mana ukuran dan ketebalan huruf menunjukkan frekuensi atau tingkat kemunculan kata tersebut. Kata-kata seperti "sangat", "bersih", "nyaman", "tempat", dan "bagus" mendominasi, yang menandakan bahwa pengunjung sering mengekspresikan pengalaman positif mereka dengan menggunakan kata-kata tersebut. Selain itu, terdapat juga kata-kata seperti "wisata", "alam", "pemandangan", dan "indah" yang menunjukkan bahwa aspek kebersihan, kenyamanan, dan keindahan alam menjadi faktor utama yang diapresiasi oleh pengunjung. *Word Cloud* ini membantu memvisualisasikan sentimen pengguna secara ringkas dan memberikan wawasan cepat terhadap persepsi positif masyarakat terhadap destinasi wisata yang dianalisis.

b. *Word Cloud* Sentimen Negatif



Gambar 4. 9 *Word Cloud* Sentimen Negatif *Naïve Bayes*

Gambar 4.9 menampilkan kata-kata yang sering muncul dalam komentar negatif tentang tempat wisata. Kata-kata seperti "jelek", "jalanan", dan "rusak" menunjukkan bahwa banyak pengunjung merasa kecewa dengan kondisi lokasi. Kata "jelek" merujuk pada tampilan atau fasilitas yang tidak menarik. "Jalanan" dan "rusak" menggambarkan akses menuju tempat wisata yang tidak nyaman atau sulit dilalui, seperti jalan berlubang, licin, atau tidak terawat. Kata-kata ini memberi gambaran bahwa beberapa pengunjung tidak puas dengan pengalaman mereka, terutama dari segi infrastruktur dan kenyamanan. Informasi ini penting bagi pengelola wisata untuk memperbaiki hal-hal yang sering dikeluhkan, agar pengunjung merasa lebih senang dan aman.

c. *Word Cloud* Sentimen Netral



Gambar 4. 10 Word Cloud Sentimen Netral Naïve Bayes

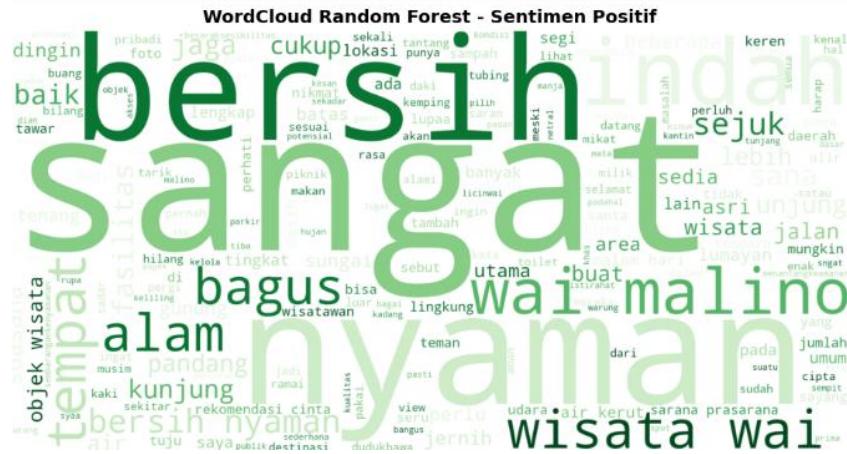
Gambar 4.10 menunjukkan kata-kata yang sering muncul dalam komentar netral tentang tempat wisata, berdasarkan analisis dengan metode *Naïve Bayes*. Kata yang paling menonjol seperti "bersih", "tempat", "sejuk", dan "bagus" menunjukkan bahwa banyak pengunjung menyebutkan hal-hal yang cukup positif, seperti lingkungan yang bersih dan suasana yang sejuk. Namun, muncul juga kata-kata seperti "kurang", "tidak", dan "terlalu", yang menunjukkan bahwa pengunjung juga menyampaikan beberapa catatan atau kekurangan, meskipun tidak terlalu dikeluhkan. Komentar netral ini biasanya berisi pandangan yang seimbang, di mana pengunjung mungkin menyukai beberapa hal, tetapi juga melihat hal lain yang bisa diperbaiki. Misalnya, tempatnya bersih dan sejuk, tapi kurang fasilitas atau pelayanan belum maksimal. *Word cloud* ini bisa membantu pengelola memahami sisi positif yang perlu dipertahankan dan kekurangan yang bisa ditingkatkan.

Secara keseluruhan, kata-kata yang muncul sesuai dengan label sentimen yang dihasilkan oleh model, di mana sebagian besar kata bernada positif sesuai dengan dominasi sentimen positif dalam *dataset*. Tidak terlihat adanya indikasi bias yang signifikan, karena kata-kata yang ditampilkan dalam *Word Cloud* merepresentasikan distribusi sentimen positif, negatif, dan netral yang ditemukan dalam ulasan. Visualisasi ini menjadi alat yang bermanfaat untuk membantu pengelola wisata memahami keunggulan dan kekurangan lokasi dari sudut pandang pengunjung, serta memberikan arahan untuk peningkatan layanan dan fasilitas ke depannya.

4.9.2 *Word Cloud* Algoritma *Random Forest*

Hasil visualisasi *Word Cloud* berdasarkan analisis sentimen yang dilakukan menggunakan algoritma *Random Forest*. *Word cloud* berfungsi untuk memperlihatkan kata-kata yang paling sering muncul dalam masing-masing kategori sentimen, yaitu positif, negatif, dan netral. Berikut ini adalah hasil *Word Cloud* berdasarkan klasifikasi sentimen:

a. *Word Cloud* Sentimen Positif



Gambar 4. 11 *Word Cloud* Sentimen Positif Random Forest

Gambar *word cloud* ini menampilkan kata-kata yang sering muncul dalam komentar positif tentang tempat wisata berdasarkan algoritma *Random Forest*. Kata seperti "sangat", "bersih", "nyaman", "indah", dan "bagus" menunjukkan bahwa pengunjung merasa puas dengan suasana, kebersihan, serta keindahan alam di lokasi tersebut. Kata "sejuk" dan "alam" juga menandakan bahwa lingkungan alami menjadi daya tarik utama. Selain itu, nama tempat seperti "Wai" dan "Malino" sering disebut secara positif. Visualisasi ini menunjukkan bahwa tempat wisata tersebut memiliki kesan yang baik dan memberikan pengalaman menyenangkan bagi pengunjung.

b. *Word Cloud* Sentimen Negatif



Gambar 4. 12 Word Cloud Sentimen Negatif Random Forest

Gambar 4.12 menunjukkan kata-kata yang sering muncul dalam komentar negatif berdasarkan algoritma *Random Forest*. Kata seperti "tempat", "kurang", "bersih", dan "jalan" paling menonjol, menandakan keluhan umum tentang kebersihan dan akses menuju lokasi wisata. Kata-kata seperti "sampah", "rusak", dan "jauh" juga muncul, menunjukkan adanya ketidakpuasan terhadap kondisi lingkungan dan fasilitas. *Word Cloud* ini mencerminkan hal-hal yang dianggap kurang oleh pengunjung dan perlu diperhatikan oleh pengelola.

c. *Word Cloud* Sentimen Netral



Gambar 4. 13 *Word Cloud* Sentimen Netral *Random Forest*

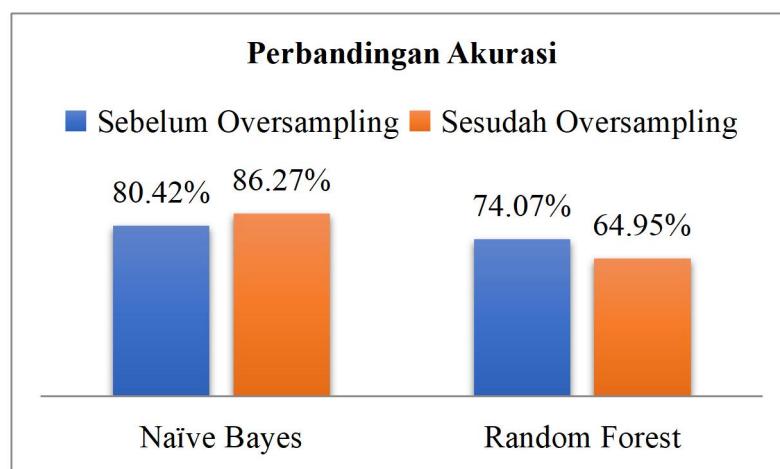
Gambar 4.13 menunjukkan kata-kata yang sering muncul dalam komentar netral berdasarkan algoritma *Random Forest*. Kata seperti "bersih", "tempat", "kurang", "tidak", dan "sangat" mendominasi. Hal ini menunjukkan bahwa pengunjung menyampaikan pendapat yang seimbang ada hal positif seperti kebersihan, tapi juga menyebut kekurangan seperti fasilitas yang kurang memadai. Komentar netral biasanya menggambarkan pengalaman yang biasa, tanpa kesan terlalu puas atau kecewa.

Secara umum, kata-kata yang muncul dalam *Word Cloud* sesuai dengan label sentimen yang telah diklasifikasikan oleh algoritma *Random Forest*. Tidak terlihat adanya bias atau ketidaksesuaian antara frekuensi kata dan distribusi sentimen. Visualisasi ini mendukung hasil klasifikasi dan membantu memberikan gambaran menyeluruh tentang persepsi pengunjung terhadap wisata Wai Malino. Dengan memahami kata-kata yang sering muncul, pengelola dapat fokus

memperkuat keunggulan serta menindaklanjuti masukan negatif yang muncul dari pengunjung.

4.10 Analisis Hasil

Pada bagian ini dilakukan analisis terhadap hasil pengujian model klasifikasi yang telah diterapkan pada data sentimen penilaian wisata di Kabupaten Enrekang. Analisis ini bertujuan untuk memahami sejauh mana efektivitas metode yang digunakan dalam mengklasifikasikan sentimen, serta faktor-faktor yang mempengaruhi performa model. Perbandingan tersebut ditampilkan melalui grafik di bawah ini, yang menggambarkan tingkat akurasi dari setiap algoritma dalam bentuk visual agar lebih mudah dianalisis dan dipahami:

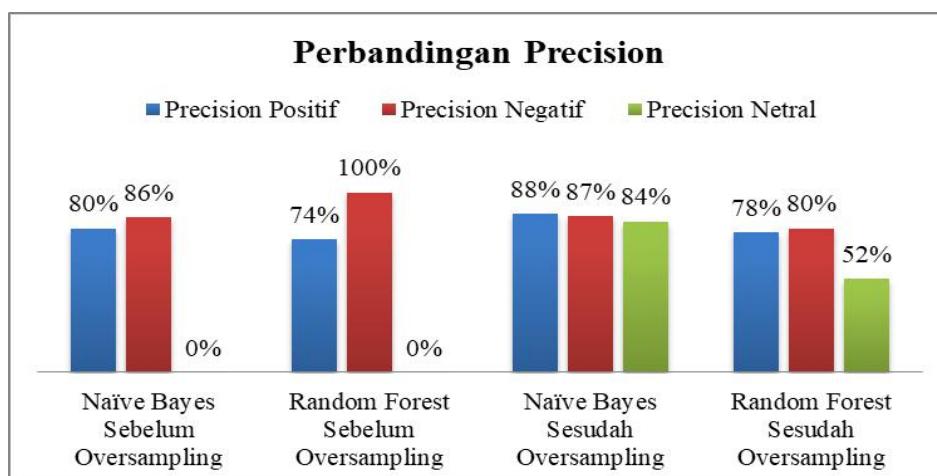


Gambar 4. 15 Grafik Perbandingan

Gambar 4.10 menunjukkan grafik perbandingan akurasi algoritma *Naïve Bayes* dan *Random Forest* sebelum dan sesudah dilakukan *oversampling*. Berdasarkan grafik tersebut, terlihat bahwa kedua algoritma memberikan respons yang berbeda terhadap proses *oversampling*. Algoritma *Naïve Bayes* mengalami peningkatan akurasi dari 80,42% menjadi 86,27%, atau naik sebesar 5,85%. Peningkatan ini menunjukkan bahwa *oversampling* berhasil membantu *Naïve Bayes* dalam mengenali pola data, terutama pada kelas sentimen yang sebelumnya jumlah datanya sedikit, seperti sentimen negatif dan netral. *Naïve Bayes* merupakan algoritma yang berbasis probabilitas, sehingga semakin banyak data

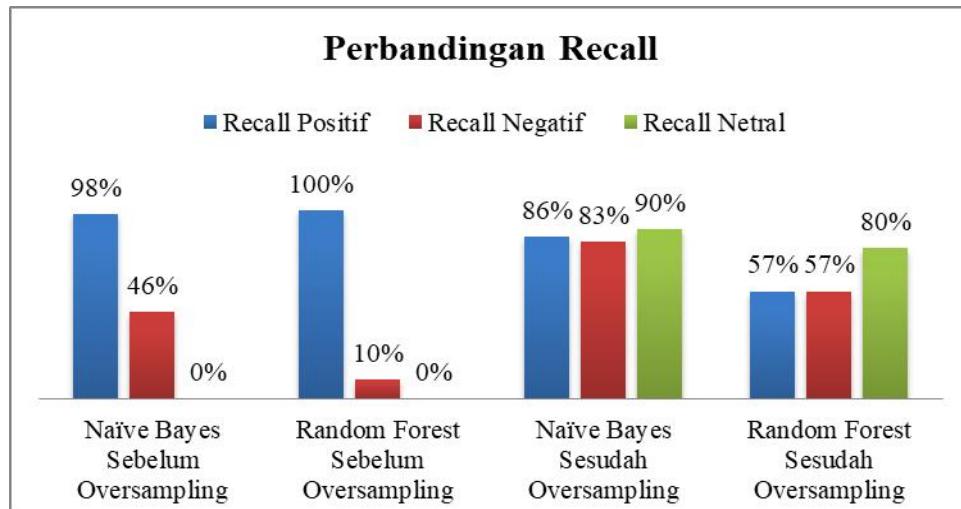
dari kelas minoritas yang tersedia, semakin besar pula peluang algoritma dalam membangun distribusi probabilitas yang akurat. Dengan demikian, proses pembelajaran menjadi lebih seimbang dan hasil klasifikasi menjadi lebih baik. Sebaliknya, algoritma *Random Forest* mengalami penurunan akurasi dari 74,07% menjadi 64,95%, atau turun sebesar 9,12% setelah dilakukan *oversampling*. *Random Forest* justru terganggu oleh pola data yang terlalu mirip akibat duplikasi data. Karena *Random Forest* merupakan algoritma berbasis pohon keputusan, ia sangat sensitif terhadap data yang identik. *Oversampling* sederhana yang hanya memperbanyak data dari kelas minoritas tanpa variasi menyebabkan model terlalu menyesuaikan diri dengan pola data pelatihan, sehingga kinerjanya menurun saat diuji dengan data baru. Secara keseluruhan, hasil ini menunjukkan bahwa metode *oversampling* sederhana lebih cocok digunakan pada algoritma *Naïve Bayes* untuk menangani data yang tidak seimbang dalam analisis sentimen. *Oversampling* mampu meningkatkan akurasi dan memperbaiki performa klasifikasi *Naïve Bayes*, namun justru memberikan efek negatif terhadap kinerja *Random Forest*.

Untuk mengevaluasi kinerja algoritma klasifikasi dalam mendeteksi sentimen pengguna terhadap wisata di Kabupaten Enrekang, dilakukan analisis terhadap hasil evaluasi model klasifikasi yang digunakan pada penelitian ini berdasarkan tiga metrik utama, yaitu *Precision*, *Recall*, dan *F1-Score*. Evaluasi dilakukan terhadap dua model, yaitu *Naïve Bayes* dan *Random Forest*, baik sebelum maupun sesudah dilakukan teknik *oversampling*. Visualisasi dari masing-masing metrik disajikan pada gambar-gambar berikut:



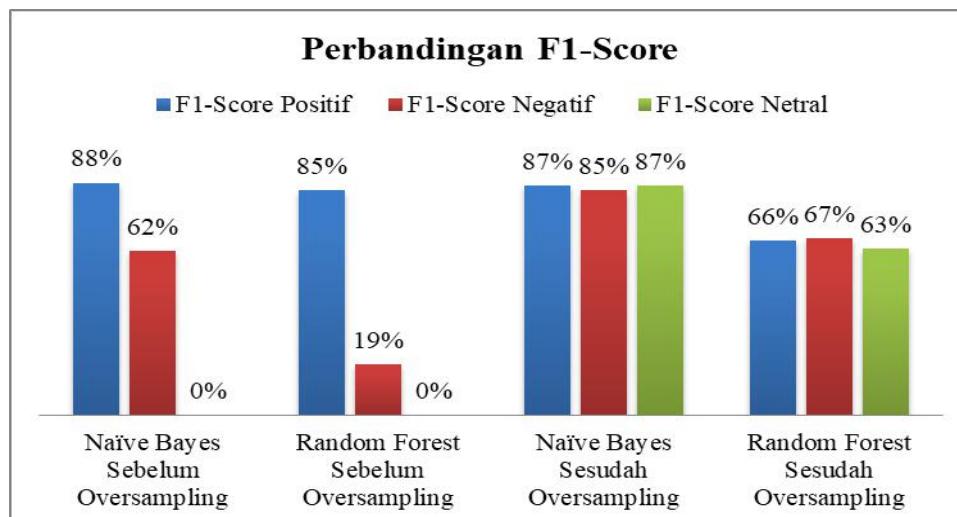
Gambar 4. 16 Grafik Perbandingan *Precision*

Gambar 4.11 menunjukkan perbandingan nilai *precision* antara algoritma *Naïve Bayes* dan *Random Forest* sebelum dan sesudah *oversampling*. *Precision* mengukur seberapa tepat prediksi positif yang dihasilkan. Grafik ini penting untuk mengetahui algoritma mana yang lebih sedikit menghasilkan *false positive* dalam klasifikasi sentimen. Pada algoritma *Naïve Bayes*, *precision* untuk kelas Positif adalah 80% dan Negatif 86%, namun *precision* untuk kelas Netral adalah 0%, menandakan bahwa model tidak mampu mengidentifikasi kelas Netral dengan benar. Begitu pula dengan *Random Forest* yang memiliki *precision* tinggi pada kelas Negatif sebesar 100% dan kelas Positif sebesar 74%, tetapi tidak memiliki *precision* sama sekali pada kelas Netral 0%. Ini mengindikasikan bahwa sebelum *oversampling*, kedua algoritma sangat lemah dalam mengenali kelas Netral yang kemungkinan merupakan kelas minoritas. Setelah dilakukan *oversampling*, terdapat peningkatan yang signifikan pada ketiga kelas untuk kedua algoritma. *Naïve Bayes* menunjukkan performa yang sangat baik dan seimbang, dengan *precision* sebesar 88% untuk kelas Positif, 87% untuk kelas Negatif, dan 84% untuk kelas Netral. Hal ini mencerminkan bahwa model mampu mengenali dan mengklasifikasikan semua kelas secara konsisten. *Random Forest* juga mengalami peningkatan, meskipun tidak sebaik *Naïve Bayes*. *Precision* pada kelas Positif mencapai 78%, Negatif 80%, dan Netral 52%. Nilai *precision* untuk kelas Netral masih tergolong rendah, menunjukkan bahwa *Random Forest* masih mengalami kesulitan dalam mengklasifikasikan kelas tersebut secara akurat. Secara keseluruhan, *oversampling* terbukti efektif dalam meningkatkan *precision* pada seluruh kelas, terutama pada kelas Netral yang sebelumnya tidak terdeteksi sama sekali. *Naïve Bayes* tampil lebih unggul dibanding *Random Forest* setelah *oversampling*, dengan hasil yang lebih seimbang dan konsisten di semua kelas.



Gambar 4. 17 Grafik Perbandingan *Recall*

Gambar 4.12 menunjukkan perbandingan nilai *recall* antara algoritma. *Recall* mengukur kemampuan algoritma dalam menemukan seluruh data positif yang sebenarnya. Dengan grafik ini, dapat diketahui algoritma mana yang lebih sensitif dalam mendekripsi sentimen. *Naïve Bayes* memiliki nilai *recall* yang sangat tinggi pada kelas Positif sebesar 98%, namun hanya 46% pada kelas Negatif dan 0% pada kelas Netral. Sementara itu, *Random Forest* menunjukkan *recall* sempurna pada kelas Positif sebesar 100%, tetapi sangat rendah pada kelas Negatif 10% dan tidak mampu mendekripsi kelas Netral 0%. Hal ini menunjukkan bahwa kedua algoritma cenderung bias terhadap kelas mayoritas, yaitu kelas Positif, dan gagal mengenali kelas minoritas. Setelah dilakukan *oversampling*, performa kedua algoritma meningkat secara signifikan, terutama dalam mendekripsi kelas minoritas. *Naïve Bayes* menunjukkan peningkatan yang sangat baik dengan *recall* sebesar 86% untuk kelas Positif, 83% untuk kelas Negatif, dan 90% untuk kelas Netral. Ini menandakan bahwa *Naïve Bayes* menjadi lebih seimbang dan efektif dalam mengklasifikasikan seluruh kelas. Sementara itu, *Random Forest* juga mengalami peningkatan dengan nilai *recall* sebesar 57% pada kelas Positif dan Negatif, serta 80% pada kelas Netral. Meskipun demikian, performa *Random Forest* setelah *oversampling* masih berada di bawah *Naïve Bayes*, terutama pada kelas Positif dan Negatif.



Gambar 4. 18 Grafik Perbandingan F1-Score

Gambar 4.13 menunjukkan perbandingan nilai F1-Score dari dua algoritma klasifikasi, yaitu *Naïve Bayes* dan *Random Forest*, Sebelum dilakukan *oversampling*, algoritma *Naïve Bayes* menunjukkan performa yang cukup baik pada kelas positif dengan F1-Score sebesar 88%, namun performanya menurun pada kelas negatif 62% dan sama sekali tidak mampu mengklasifikasikan kelas netral 0%. Sementara itu, algoritma *Random Forest* juga memiliki performa yang tinggi pada kelas positif 85%, tetapi sangat rendah pada kelas negatif 19% dan juga gagal mengklasifikasikan kelas netral 0%. Hal ini menunjukkan bahwa kedua algoritma cenderung bias terhadap kelas positif dan tidak mampu mengatasi ketidakseimbangan data pada kelas lainnya, khususnya netral. Setelah dilakukan *oversampling*, performa kedua algoritma mengalami peningkatan yang signifikan dan menjadi lebih seimbang. *Naïve Bayes* berhasil meningkatkan F1-Score pada ketiga kelas menjadi relatif merata, yaitu 87% pada kelas positif, 85% pada kelas negatif, dan 87% pada kelas netral. Hal ini menunjukkan bahwa *Naïve Bayes* mampu mengklasifikasikan semua kelas dengan baik setelah distribusi data diseimbangkan. Sementara itu, *Random Forest* juga mengalami peningkatan performa, meskipun tidak setinggi *Naïve Bayes*. F1-Score pada kelas positif menjadi 66%, kelas negatif 67%, dan kelas netral 63%. Meskipun sudah membaik dibandingkan sebelum *oversampling*, performa *Random Forest* masih lebih rendah dibandingkan *Naïve Bayes* setelah *oversampling*. Secara keseluruhan, proses *oversampling* terbukti efektif dalam meningkatkan kemampuan klasifikasi.

Analisis Kesalahan, Meskipun model menunjukkan performa yang cukup baik secara keseluruhan. Terdapat contoh kasus komentar yang salah/gagal diklasifikasikan sebagai berikut:

- a. "wisata wai malino tempat wisata bagus libur keluarga sama teman tempat puas" yang seharusnya diberi label positif justru diprediksi sebagai negatif.
- b. "sangat kurang fasilitas utama tempat istirahat hujan" yang sebenarnya bernada negatif malah diklasifikasikan sebagai positif.

Kesalahan ini terjadi karena komentar pertama mengandung campuran kata positif dan negatif, namun model lebih menekankan kata-kata negatif yang muncul di akhir kalimat. Sementara itu, pada komentar kedua, model mungkin keliru menganggap kata-kata netral seperti "tempat istirahat" sebagai indikator positif, padahal konteks keseluruhan kalimat bernada negatif. Hal ini menunjukkan bahwa model masih kesulitan dalam memahami konteks kalimat yang kompleks dan bercampur nuansa. Peningkatan dalam teknik pemrosesan teks yang lebih mendalam serta penggunaan model klasifikasi yang lebih sensitif terhadap konteks diperlukan untuk meningkatkan akurasi prediksi, khususnya pada komentar yang memiliki nuansa bercampur antara positif dan negatif.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan untuk membandingkan kinerja algoritma *Naïve Bayes* dan *Random Forest* dalam analisis sentimen, pengunjung wisata Wai Malino di Kabupaten Enrekang, diperoleh beberapa temuan penting. Hasil eksperimen menunjukkan bahwa sebelum dilakukan *oversampling*, kedua algoritma mengalami kesulitan dalam mengklasifikasikan kelas minoritas (netral dan negatif), meskipun akurasi keseluruhan terlihat cukup tinggi. *Naïve Bayes* mencatat akurasi sebesar 80,42%, sementara *Random Forest* mencapai 74,07%. Namun demikian, keduanya menunjukkan F1-score yang sangat rendah pada kelas minoritas. Setelah dilakukan *oversampling* untuk menyeimbangkan distribusi data, performa kedua algoritma mengalami perubahan. *Naïve Bayes* menunjukkan peningkatan signifikan dengan akurasi 86,27% dan F1-score yang seimbang pada semua kelas (85–87%), sedangkan *Random Forest* justru mengalami penurunan performa dengan akurasi turun menjadi 64,95% dan F1-score yang tidak konsisten antar kelas (63–67%). Dengan demikian, dapat disimpulkan bahwa *Naïve Bayes* lebih unggul dalam menangani data sentimen yang tidak seimbang terutama setelah proses *oversampling*, sementara *Random Forest* cenderung kurang stabil dalam kondisi serupa. Namun, keberhasilan model *Naïve Bayes* ini sangat bergantung pada proses penyeimbangan data. Tanpa *balancing*, model ini juga gagal dalam mendekripsi kelas minoritas secara optimal. Oleh karena itu, *balancing* data merupakan langkah penting yang harus diperhatikan dalam analisis sentimen agar hasil klasifikasi menjadi lebih akurat dan merata pada semua kelas. *Naïve Bayes* pun lebih direkomendasikan dibandingkan *Random Forest* untuk kasus seperti ini karena menghasilkan klasifikasi yang lebih stabil dan seimbang pasca *balancing*.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran yang dapat diberikan untuk pengembangan penelitian selanjutnya:

- a) Penelitian ini dapat dikembangkan dengan menggunakan *dataset* yang lebih besar dan lebih beragam untuk meningkatkan akurasi serta validitas model dalam analisis sentimen.
- b) Selain algoritma *Naïve Bayes* dan *Random Forest*, penelitian selanjutnya dapat membandingkan metode lain seperti *Support Vector Machine (SVM)* atau *Deep Learning* untuk mendapatkan hasil yang lebih optimal.
- c) Disarankan untuk menggunakan teknik penyeimbangan data lain yang lebih sesuai agar performanya tidak menurun.
- d) Untuk penelitian selanjutnya, disarankan untuk mempertimbangkan kembali kamus *lexicon-based* yang digunakan, serta menggunakan label *ground-truth* yang di berikan oleh manusia (*manual labeling*) dan memperhatikan pengaruh *stemming* terhadap bobot sentimen. Sehingga evaluasi model menjadi lebih valid dan mencerminkan opini secara lebih tepat.
- e) Implementasi *Random Forest* dalam penelitian ini masih dapat ditingkatkan. Salah satu penyederhanaan yang dilakukan adalah pemilihan fitur secara acak hanya dilakukan satu kali di awal untuk setiap pohon, bukan pada setiap *node*. Untuk penelitian selanjutnya, disarankan agar pemilihan fitur dilakukan secara acak di setiap *node* pembentukan pohon, sesuai dengan prinsip dasar *Random Forest*. Hal ini diharapkan dapat meningkatkan keragaman antar pohon dan mengurangi risiko model terlalu menyesuaikan diri dengan data latih.

DAFTAR PUSTAKA

- Aditya, A., Wibowo, A., 2022. Analisis Sentimen Menggunakan Metode *Naïve Bayes* Berdasarkan Opini Masyarakat Dari Twitter Terhadap Perang Rusia dan Ukraina.
- Agustina, N., Citra, D.H., Purnama, W., Nisa, C., Kurnia, A.R., 2022. Implementasi Algoritma *Naïve Bayes* untuk Analisis Sentimen Ulasan Shopee pada Google Play Store. MALCOM Indones. J. Mach. Learn. Comput. Sci. 2, 47–54. <https://doi.org/10.57152/malcom.v2i1.195>
- Ardianto, R., Rivanie, T., Alkhalfi, Y., Nugraha, F.S., Gata, W., 2020. Sentimen Analisis On E-Sports For Educatioan Curricilum Using *Naïve Bayes* end Support Vector Machine. J. Ilmu Komput. Dan Inf. 13, 109–122. <https://doi.org/10.21609/jiki.v13i2.885>
- Atmadja, B.R., 2022. Analisis Sentimen Bahasa Indonesia Pada Tempat Wisata di Kabupaten Sukabumi Dengan *Naïve Bayes*. J. Ilm. Elektron. DAN Komput. 15, 371–382.
- Basar, T.F., Ratnawati, D.E., Arwani, I., 2022. Analisis Sentimen Pengguna Twitter terhadap Pembayaran Cashless menggunakan Shopeepay dengan Algoritma Random Forest. J. Pengemb. Teknol. Inf. Dan Ilmu Komput. 6, 1426–1433.
- Diantika, S., 2023. Penerapan Teknik Random *Oversampling* Untuk Mengatasi Imbalance Class Dalam Klasifikasi Website Phishing Menggunakan Algoritma Lightgbm. JATI J. Mhs. Tek. Inform. 7, 19–25. <https://doi.org/10.36040/jati.v7i1.6006>.
- Era, D., Andryana, S., Rubhasy, A., 2023. Perbandingan Algoritma *Naïve Bayes* Dan K-Nearest Neighbor pada Analisis Sentimen Pembukaan Pariwisata Di Masa Pandemi Covid 19. J. Sains Komput. Inform. J-SAKTI 7, 263–272.
- Firdaus, R.Z., Wijoyo, S.H., Purnomo, W., 2025. Analisis Sentimen Berbasis Aspek Ulasan Pengguna Aplikasi Alfagift Menggunakan Metode *Random Forest* dan Pemodelan Topik Latent Dirichlet Allocation. J. Pengemb. Teknol. Inf. Dan Ilmu Komput. 9, 1–10.
- Harfian, Y., 2021. Klasifikasi Sentimen Aplikasi Dompet Digital Dana Pada Komentar Di Instagram Menggunakan *Naïve Bayes* Classifier. Fak. Sain Dan Teknol. Iqbal, M., Wiranata, A.D., Suwito, R., Ananda,

- Indarbensyah, P.P.E., Rochmawati, N., 2021. Penerapan N-Gram menggunakan Algoritma *Random Forest* dan *Naïve Bayes Classifier* pada Analisis Sentimen Kebijakan PPKM 2021. *J. Inform. Comput. Sci. JINACS* 2, 235–244. <https://doi.org/10.26740/jinacs.v2n04.p235-244>
- Ismail, A.R., Hakim, R.B.F., 2023. Implementasi Lexicon Based Untuk Analisis Sentimen Dalam Mengetahui Trend Wisata Pantai Di DI Yogyakarta Berdasarkan Data Twitter. *Emerg. Stat. Data Sci. J.* 1, 37–46.
- Kaka, D.L., Pati, G.K., Rato, K.W., 2023. Analisis Sentimen Komentar SIAKAD Menggunakan Metode *Naïve Bayes Classifier*. *J. KRIDATAMA SAINS DAN Teknol.* 5, 266–277. <https://doi.org/10.53863/kst.v5i02.933>
- Khasanah, U., 2023. Analisis Sentimen Terhadap Tempat Wisata Menggunakan Metode *Naïve Bayes Classifier* dan Support Vector Machine. *Fak. Sain Dan Teknol.*
- Kosasih, R., Alberto, A., 2021. Sentiment Analysis Of Game Product on Shopee Using the TF-IDF Method and *Naïve Bayes Classifier*. *Ilk. J. Ilm.* 13, 101–109. <https://doi.org/10.33096/ilkom.v13i2.721.101-109>
- Munawaroh, A., Ridhoi, R., Rudiman, R., 2024. Sentiment Analysis Dengan *Naïve Bayes* Berbasis Orange Terhadap Resiko Pembangunan IKN. *JATI J. Mhs. Tek. Inform.* 8, 587–592. <https://doi.org/10.36040/jati.v8i1.8454>
- Prasetyo, H., Irawati, N., Satriawati, Z., 2024. Menuju Destinasi Wisata Digital Transformasi, Literasi, dan Inovasi.
- R.F., 2023. Perbandingan Algoritma *Naïve Bayes*, KNN, dan Decision Tree terhadap Ulasan Aplikasi Threads dan Twitter. *Kaji. Ilm. Inform. Dan Komput.* 4, 1799–1807.
- Rizal, A.A., Nugraha, G.S., Putra, R.A., Anggraeni, D.P., 2024. Twitter Sentiment Analysis in Tourism with Polynomial *Naïve Bayes Classifier*. *JTIM J. Teknol. Inf. Dan Multimed.* 5, 343–353. <https://doi.org/10.35746/jtim.v5i4.478>
- Singgalen, Y.A., 2022. Analisis Sentimen Wisatawan Melalui Data Ulasan Candi Borobudur di Tripadvisor Menggunakan Algoritma *Naïve Bayes Classifier*. *Build. Inform. Technol. Sci. BITS* 4, 1343–1352. <https://doi.org/10.47065/bits.v4i3.2486>

- Soumya, K.V., P., 2020. Sentiment Analysis Of Malayalam Tweets using machine Learning Techniques. *ICT Express* 6, 300–305. <https://doi.org/10.1016/j.icte.2020.04.003>
- Sunardi, M.H., Nuraeni, S., Nurdin, M.L., 2021. Haltour: Millennial Generation Halal Tourism Literature Media. *J. Halal Prod. Res.* 4, 78. <https://doi.org/10.20473/jhpr.vol.4-issue.2.78-82>
- Syahlan, M.S., Irmayanti, D., Alam, S., 2023. Analisis Sentimen Terhadap Tempat Wisata Dari Komentar Pengunjung Dengan Menggunakan Metode Support Vector Machine (SVM). *Simtek J. Sist. Inf. Dan Tek. Komput.* 8, 315–319. <https://doi.org/10.51876/simtek.v8i2.281>
- Undap, M.G., Rantung, V.P., Rompas, P.T.D., 2021. Analisis Sentimen Situs Pembajak Artikel Penelitian Menggunakan Metode Lexicon-Based. *J. Inform. Eng.* 02.
- Utami, D.S., Erfina, A., 2022. Analisis Sentimen Objek Wisata Bali Di Google Maps Menggunakan Algoritma *Naïve Bayes*. *J. Sains Komput. Inform. J-SAKTI* 6, 418–427.
- Widyarto, E.B., Lhaksmana, K.M., 2023. Implementasi Metode *Naïve Bayes* Classifier Terhadap Analisis Sentimen Tempat Wisata di Nusa Tenggara Barat. *E-Proceeding Eng.* 10, 4934–4941.

LAMPIRAN

Dokumentasi Pengunjung





Dinas Pariwisata Enrekang

Data Sampel

No	Komentar
1	Sangat cocok dikunjungi bersama teman dan keluarga
2	Baik
3	Luar biasa
4	Tempatnya sangat indah, bersih dan sejuk
5	Cukup bersih, nyaman dan lengkap.
6	Kebersihan dan Ke Indahan Alamnya Sangat Terjaga
7	Bersih dan bagus airnya juga jernih
8	Pemandangan di sini luar biasa! Udara segar dan suasannya menenangkan
9	KEAMANAN YANG KURANG TERJAMIN
10	Objek wisata ini dikelilingi oleh pegunungan dan hutan yang indah, menawarkan pemandangan alam yang indah
11	perjalanan menuju ke sana cukup melelahkan dan jalanan kurang terawat
12	Sangat nyaman
13	<p>Wai Malino adalah objek wisata yang menawarkan keindahan alam dan suasana yang asri</p> <p>Kebersihan: Secara umum, area wisata ini terjaga dengan baik. Namun, pengunjung diharapkan untuk selalu menjaga kebersihan dengan tidak membuang sampah sembarangan.</p> <p>Kenyamanan: Suasana yang sejuk dan pemandangan alam yang indah membuat pengunjung merasa nyaman. Fasilitas seperti tempat duduk dan area istirahat tersedia untuk menambah kenyamanan.</p> <p>Ketersediaan Sarana: Fasilitas dasar seperti area parkir tersedia, meskipun mungkin perlu peningkatan dalam hal jumlah dan kualitas untuk mengakomodasi jumlah pengunjung yang lebih besar.</p> <p>Aksesibilitas: Jalan menuju lokasi cukup baik, namun disarankan untuk menggunakan kendaraan yang sesuai mengingat beberapa bagian jalan yang mungkin menantang.</p> <p>Keamanan: Petugas keamanan dan tanda peringatan tersedia di beberapa</p>

No	Komentar
	area untuk memastikan keselamatan pengunjung.
14	Tidak banyak pilihan tempat makan atau penginapan yang layak...
15	Sangat aman
16	Wisata waimalino adalah salah satu tempat wisata yang ada di kecamatan curio tepatnya di desa parombean yang menarik untuk dikunjungi bagi para wisatawan yang ingin menikmati suasana pedesaan,Wisata ini menyediakan area camping yang luas dipinggiran sungai dengan fasilitas yang cukup seperti warung,toilet,ruang ganti dan sebagainya membuat para pengunjung merasa nyaman,selain itu tersedia sarana Arung jeram untuk memacu adrenalin dengan menyusuri sungai sepanjang 200m yang membuat pengunjung merasa puas,namun kurangnya peningkatan fasilitas yang membuat wisata ini hanya bersifat sementara.
17	menurut saya pribadi, dengan mengunjungi wisata tersebut merupakan hal baru buat saya. Karena dengan adanya sarana hiburan (arung jeram) dengan alat yg sederhana dan kreatif mengukir kebahagiaan tersendiri buat sy karena kali pertama sy mencoba hal tersebut. Selain itu keamanannya sangat safety karena berhubung baru pertama kali melakukan arung jeram panitia Wai Malino (pelayanannya) sangat antusias membantu, menjaga, dan memandu dengan cara ikut andil berada diatas arung jeram dan juga pakai baju pelampung. Airnya juga bersih, jernih ditambah pemandangan sawah yg menyegarkan mata????
18	Sangat bagus
19	Wisata Wai malino tempat yg bersih dan sejuk,,serta nyaman karna tempatnya yg lumayan jauh dari keramaian.
20	Wai Malino menawarkan suasana alami yang nyaman, tetapi kebersihan dan kenyamanan sangat tergantung pada kesadaran pengunjung dan pengelola. Fasilitas dasar seperti toilet dan tempat makan perlu ditingkatkan, begitu juga aksesibilitas dan pelayanan untuk menunjang pengalaman wisata. Dengan pengelolaan yang baik, tempat ini bisa menjadi destinasi favorit.
21	Sangat puas

No	Komentar
22	kebersihannya terjaga dan ketersediaan sarana berupa wc dan rumah2 untuk berteduh lumayan lengkap. suasananya juga asri dan sejuk
23	Sangat nyaman
24	Sangat bagus
25	Nyaman dengan suasana yang indah, juga dengan hawa yang sejuk.
26	Makanan dan minuman di dalam area wisata sangat mahal dan tidak sesuai kualitasnya
27	Kebersihan: objek wisata wai malino itu bersih Kenyamanan: Ketersediaan tempat istirahat, kenyamanan fasilitas, dan pelayanan yang ramah. 3. Sarana: Aksesibilitas, parkir, dan ketersediaan fasilitas penunjang lainnya. Kualitas Pelayanan: Keramahan dan kesabaran petugas. Kualitas Fasilitas: Kondisi dan kualitas fasilitas yang ada.
28	Bersih dan nyaman
29	Untuk kenyamanan dan kebersihan sudah terjamin, namun untuk sarana jalanan mungkin kedepannya bisa lebih di kembangkan/perbaiki.
30	sangat nyaman dan susah enak

Adapun Komentar yang lebih lengkap dan data mentah lainnya, seluruh dataset disimpan pada tautan *GitHub* yang telah disediakan di bawah ini:

<https://github.com/QalbiAlmustikaM/DatasetWisata>.

Source code Program Python

```
# Proses pengolahan data
#Import library yang diperlukan untuk analisis data
import pandas as pd
import re
import seaborn as sns
import matplotlib.pyplot as plt
# Hubungkan Google Drive ke Colab
```

```
from google.colab import drive
drive.mount('/content/drive')

# Membaca data CSV
Data = pd.read_csv("/content/drive/MyDrive/Kamus/Data_Wisataaa.csv")
Data.shape #Malihat Jumlah Data

# Filter data yang pernah berkunjung dan simpan ke file CSV baru
Data = pd.read_csv("/content/drive/MyDrive/Kamus/Data_Wisataaa.csv")
Data[Data["Pernah_Berkunjung?"].str.lower() != "tidak"].to_csv("data_output.csv", index=False)

# Baca data hasil filter
Data = pd.read_csv("data_output.csv")

# Ambil kolom tertentu dari data untuk analisis
Data = Data[['Timestamp', 'Nama', 'Alamat', 'Kepuasan', 'Komentar']]

# Hapus baris duplikat berdasarkan Nama, Alamat, Kepuasan, dan Komentar
# komentarnya)
Data = Data.drop_duplicates(subset=['Nama', 'Alamat', 'Kepuasan', 'Komentar'],
keep=False)

# Hapus baris yang memiliki nilai kosong (NaN) & menampilkan
Data = Data.dropna()
Data.isnull().sum()

# Pilih hanya kolom 'Komentar' dari data
Data = Data[['Komentar']]

Data.shape #Malihat Jumlah Data
```

Proses *Cleansing*

```
def clean_wisata_text(text):
    text = re.sub(r'@[A-Za-z0-9_]+', " ", text)
    text = re.sub(r'#\w+', " ", text)
    text = re.sub(r'RT[\s]+', " ", text)
    text = re.sub(r'https?://\S+', " ", text)
    text = re.sub(r'\d+', " ", text) # Hapus angka
    text = re.sub(r'^A-Za-z0-9 ]', " ", text)
    text = re.sub(r'\s+', ' ', text).strip()
    return text
```

```
Data['Cleansing'] = Data['Komentar'].apply(clean_wisata_text)
Data.head(10)
```

```
# Tentukan path output
output_path = "Cleansing.csv"
Data.to_csv(output_path, index=False, encoding='latin1')
Data
```

Proses *Case Folding* dan Normalisasi

```
norm = {
    'bersi': 'bersih',
    'jerni': 'jernih',
    'yg': 'yang',
    'sy': 'saya',
    'safety': 'aman',
    'andil': 'adil',
    'sejukserta': 'sejuk serta',
    'karna': 'karena',
    'keramaian': 'keramaian',
    'sagant': 'sangat',
    'susa': 'sudah',
```

'haruss': 'harus',
'masi': 'masih',
'seruh': 'seru',
'baguslah': 'bagus',
'menakjubkan': 'menakjukkan',
'blm': 'belum',
'wisatax': 'wisata',
'cuku': 'cukup',
'temlat': 'tempat',
'nyapan': 'nyaman',
'seru': 'seruh',
'nyamann': 'nyaman',
'kennyamanan': 'nyaman',
'nnyaman': 'nyaman',
'fresh': 'segar',
'wc': 'toilet',
'adrenalin': "",
'kem': 'kemping',
'rama': 'ramah',
'bengko': 'bengkok',
'enk': 'enak',
'jelektapi': 'jelek tetapi',
'tenahg': 'tenang',
'cam': 'kemping',
'ada pun': 'adapun',
'asrih': 'asri',
'sangattttttttttttt': 'sangat',
'okeeeeeeeeeeeeeee': 'oke',
'oky': 'oke',
'mbaik': 'baik',
'and then': 'dan',
'wishlist': 'daftar',

'bangett': 'banget',
'bangtt': 'banget',
'rusakk': 'rusak',
'jauhh': 'jauh',
'kurangg': 'kurang',
'tpi': 'tapi',
'nyaman nn': 'nyaman',
'jelekkk': 'rusak',
'kayakkk': 'kayak',
'kamii': 'kami',
'jelak': 'rusak',
'jelekk': 'rusak',
'sma': 'sama',
'wisatajya': 'wisatanya',
'sayaa': 'saya',
'nyamn': 'nyaman',
'sejuktetapi': 'sejuk tetapi',
'tdk': 'tidak',
'blum': 'belum',
'malona': 'Malino',
'rekomend': 'rekomendasi',
'aplgi': 'apalagi',
'sekalisangat': 'sangat sekali',
'sulawesi air': 'Sulawesi, air',
'sejuk dan': 'sejuk dan',
'tp': 'tapi',
'baguss': 'bagus',
'sangatt': 'sangat',
'camp': 'kemping',
'nyamann': 'nyaman',
'sngt': 'sangat',
'msi': 'masih',

'bersihh': 'bersih',
'bagusss': 'bagus',
'lengpak': 'lengkap',
'sygg': 'sayang',
'dama': 'sama',
'memuaskn': 'memuaskan',
'sawa': 'sawah',
'karangan': 'wai malino',
'kemp': 'kemping',
'kerenn': 'keren',
'bangat': 'banget',
'dll': 'dan lain-lain',
'lebi': 'lebih',
'yng': 'yang',
'klo': 'kalau',
'Memuaskn': 'Memuaskan',
'cokot': 'cocok',
'wau': 'Wai',
'lumayang': 'lumayan',
'terjaga': 'terjaga',
'salasatu': 'sala satu',
'lingkung': 'lingkungan',
'cuman': 'cuma',
'yaman': 'nyaman',
'pedesaan': 'pedesaan',
'bangettttt': 'banget',
'topp': 'bagus',
'stress': 'stres',
'heppy': 'senang',
'kerennn': 'keren',
'bangett': 'banget',
'sgt': 'sangat',

```

'derass': 'deras',
'wissta': 'wisata',
'wkt': 'waktu',
'bangettt': 'banget',
'tapo': 'tapi',
}

# Fungsi normalisasi dengan regex

def normalisasi(str_text):
    str_text = str_text.lower() # Ubah ke huruf kecil
    words = re.compile(r'\b(' + '|'.join(re.escape(k) for k in norm.keys()) +
r')\b') # Buat pola regex
    return words.sub(lambda x: norm[x.group()], str_text) # Ganti dengan
nilai dari kamus

# Terapkan normalisasi ke DataFrame
Data['Case_Folding'] = Data['Cleansing'].apply(lambda x: normalisasi(x))

Data = Data[['Case_Folding']]
output_path = "Case_Folding.csv"
Data.to_csv(output_path, index=False, encoding='latin1')

# Load data Case_Folding
input_path = "Case_Folding.csv"
Data = pd.read_csv(input_path, encoding='latin1')

```

```

# Proses Tokenizing (Memisahkan kata-kata)
Data['Tokenizing'] = Data['Case_Folding'].apply(lambda x: x.split())
# Simpan hasil Tokenizing ke CSV
output_path = "Tokenizing.csv"
Data.to_csv(output_path, index=False, encoding='latin1')
Data

```

```

# Proses Stopword Removal

!pip install Sastrawi
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory
stop_factory = StopWordRemoverFactory()
stopword = stop_factory.create_stop_word_remover()

def remove_stopwords(token_list):
    text = " ".join(token_list) # Gabungkan kembali ke string
    cleaned_text = stopword.remove(text) # Hapus stopword
    return cleaned_text.split() # Pecah kembali menjadi token list

# Terapkan Stopword Removal
Data['Stopword_Removal'] = Data['Tokenizing'].apply(remove_stopwords)

# Simpan hasil Tokenizing ke CSV
output_path = "Stopword_Removal.csv"
Data.to_csv(output_path, index=False, encoding='latin1')

#Program ini melakukan proses stemming
import pandas as pd
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

# Load Data
file_path = "Stopword_Removal.csv" # Sesuaikan lokasi file
Data = pd.read_csv(file_path)

# Pastikan kolom `Stopword_Removal` sudah dalam bentuk list
def convert_to_list(text):
    return eval(text) if isinstance(text, str) and text.startswith("[") else text

Data['Stopword_Removal'] = Data['Stopword_Removal'].apply(convert_to_list)

```

```
# Buat Stemmer
stemmer_factory = StemmerFactory()
stemmer = stemmer_factory.create_stemmer()

# Fungsi Stemming
def stemming(text_cleaning):
    return [stemmer.stem(word) for word in text_cleaning] if
isinstance(text_cleaning, list) else text_cleaning

# Terapkan Stemming
Data['Stemming'] = Data['Stopword_Removal'].apply(stemming)

# Simpan hasil ke CSV
Data.to_csv("hasil_stemming.csv", index=False)
Data
```

```
# Program Analisis Sentimen Menggunakan Kamus Lexicon
import pandas as pd
from collections import Counter

file_data = "stemming.csv"
Data = pd.read_csv(file_data)

# Perbaiki format teks jika berbentuk list string
Data['Stemming'] = Data['Stemming'].apply(lambda x: " ".join(eval(x)) if
isinstance(x, str) and x.startswith("[") else x)
Data['Stemming'] = Data['Stemming'].astype(str).str.lower()

# Baca kamus lexicon dengan skor
file_positif = "/content/drive/MyDrive/Kamus/positive.csv"
file_negatif = "/content/drive/MyDrive/Kamus/negative.csv"

positive_lexicon = pd.read_csv(file_positif, header=None, names=['word',
```

```

'score'])

negative_lexicon = pd.read_csv(file_negatif, header=None, names=['word',
'score'])

# Simpan dalam dictionary untuk akses cepat
positive_dict = dict(zip(positive_lexicon['word'].str.lower(),
positive_lexicon['score']))

negative_dict = dict(zip(negative_lexicon['word'].str.lower(),
negative_lexicon['score']))

# Fungsi Analisis Sentimen dengan Skor dari Kamus Lexicon
def sentiment_analysis(text):

    words = text.split()
    word_counts = Counter(words) # Hitung jumlah kemunculan kata

    total_score = 0
    words_with_scores = []

    for word, count in word_counts.items():
        if word in positive_dict:
            score = positive_dict[word] * count # Gunakan skor dari
kamus positif
            total_score += score
            words_with_scores.append(f'{word} ({score})')
        elif word in negative_dict:
            score = negative_dict[word] * count # Gunakan skor dari
kamus negatif
            total_score += score
            words_with_scores.append(f'{word} ({score})')

    # Tentukan label berdasarkan total skor
    label = "Positive" if total_score > 0 else "Negative" if total_score < 0 else

```

```

"Neutral"

# Gabungkan kata dan skor dalam satu kolom
combined_text = ", ".join(words_with_scores)
return combined_text, total_score, label

# Terapkan Analisis Sentimen
Data[['Words (Score)', 'Total Score', 'Label']] = \
    Data['Stemming'].apply(sentiment_analysis).apply(pd.Series)

# Cetak Hasil Berjejer ke Samping
print("\n Hasil Analisis Sentimen:")
print(Data.head(10).to_string(index=False)) # Cetak dalam satu baris per data

# Simpan hasil ke file CSV
output_file = "Pelabelan_Lexicon.csv"
Data.to_csv(output_file, index=False)

# Tampilkan Distribusi Sentimen
print("\n Distribusi Sentimen:")
print(Data['Label'].value_counts().to_string())

# Mengonversi label sentimen menjadi angka menggunakan mapping
label_mapping = {'Positive': 1, 'Negative': -1, 'Neutral': 0}

if 'Label' in Data.columns:
    Data['Label'] = Data['Label'].map(label_mapping)

# Menyimpan data yang telah diubah ke dalam file CSV
Data.to_csv("data_convers.csv", index=False)
df_stemming = pd.read_csv("hasil_stemming.csv")
df_convers = pd.read_csv("data_convers.csv")

```

```

df_stemming = df_stemming[['Stemming']] # Mengambil kolom "Stemming"
df_convers = df_convers[['Label']] # Mengambil kolom "Label"

df_merged = pd.concat([df_stemming, df_convers], axis=1)
df_merged.to_csv("hasil_gabungan.csv", index=False)

```

Pembobotan dengan TF-IDF

```

TfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
count_vectorizer = CountVectorizer()
train_counts = count_vectorizer.fit_transform(train_texts)
test_counts = count_vectorizer.transform(test_texts)
fitur = count_vectorizer.get_feature_names_out()

# Hitung TF-IDF
tfidf_transformer = TfidfTransformer()
train_tfidf = tfidf_transformer.fit_transform(train_counts)

# Konversi hasil TF-IDF ke DataFrame
df_tfidf = pd.DataFrame(train_tfidf.toarray(), columns=fitur)
df_tfidf.to_csv("TF_IDF.csv", index=False)

```

Pembagian data *Training* dan *Testing*

```

import random
# Konfigurasi
input_file = "TF_IDF.csv"
train_file = "train_data.csv"
test_file = "test_data.csv"
train_ratio = 0.8 # 80% untuk training, 20% untuk testing
random_seed = 42 # Seed untuk hasil acak yang konsisten

```

```
# Membaca data dari CSV
with open(input_file, 'r', encoding='utf-8') as file:
    csv_reader = csv.reader(file)
    header = next(csv_reader)  # Membaca header
    data = list(csv_reader)    # Membaca semua data

# Menemukan indeks kolom "Label"
label_index = header.index("Label")

# Memisahkan data berdasarkan label
positif_data = [row for row in data if row[label_index] == '1']
negative_data = [row for row in data if row[label_index] == '-1']
neutral_data = [row for row in data if row[label_index] == '0']

# Mengacak data untuk setiap label
random.seed(random_seed)
random.shuffle(positif_data)
random.shuffle(negative_data)
random.shuffle(neutral_data)

# Menentukan titik pembagian untuk setiap label
split_point_positif = int(len(positif_data) * train_ratio)
split_point_negative = int(len(negative_data) * train_ratio)
split_point_neutral = int(len(neutral_data) * train_ratio)

# Membagi data untuk setiap label
train_positif = positif_data[:split_point_positif]
test_positif = positif_data[split_point_positif:]

train_negative = negative_data[:split_point_negative]
test_negative = negative_data[split_point_negative:]
train_neutral = neutral_data[:split_point_neutral]
```

```

test_neutral = neutral_data[split_point_neutral:]

# Menulis data training dan testing untuk setiap label ke file
with open(train_file, 'w', encoding='utf-8', newline="") as file:
    csv_writer = csv.writer(file)
    csv_writer.writerow(header)
    csv_writer.writerows(train_positif)
    csv_writer.writerows(train_negative)
    csv_writer.writerows(train_neutral)

with open(test_file, 'w', encoding='utf-8', newline="") as file:
    csv_writer = csv.writer(file)
    csv_writer.writerow(header)
    csv_writer.writerows(test_positif)
    csv_writer.writerows(test_negative)
    csv_writer.writerows(test_neutral)

# Menampilkan hasil
print(f"\nPembagian data selesai!")
print(f"Total data: {len(data)}")
print(f"Data training: {len(train_positif) + len(train_negative) +
len(train_neutral)} baris ({train_ratio*100}%)")
print(f"Data testing: {len(test_positif) + len(test_negative) + len(test_neutral)} "
baris ({(1-train_ratio)*100}%)")

```

```

# Klasifikasi Naïve Bayes
from collections import defaultdict
class NaiveBayes:
    def __init__(self):
        self.probabilitas_kelas = {} # P(H) = Peluang kelas
        self.probabilitas_kata = defaultdict(lambda: defaultdict(float)) # P(X|H) = Peluang kata dalam kelas

```

```

    self.kata_unik = set() # Semua kata yang muncul dalam dataset

def bersihkan_teks(self, teks):
    return teks.lower().split()

def muat_data_csv(self, nama_file, kolom_teks, kolom_label):
    daftar_teks, daftar_label = [], []
    with open(nama_file, 'r', encoding='utf-8') as file:
        pembaca_csv = csv.DictReader(file)
        for baris in pembaca_csv:
            daftar_teks.append(baris[kolom_teks])
            daftar_label.append(int(baris[kolom_label]))
    return daftar_teks, daftar_label

def latih_model(self, daftar_teks, daftar_label):
    """Melatih model dengan menghitung probabilitas kelas dan kata."""
    jumlah_data = len(daftar_teks) # Total jumlah dokumen
    jumlah_per_kelas = defaultdict(int) # Menghitung jumlah dokumen per kelas

    for label in daftar_label:
        jumlah_per_kelas[label] += 1

    for label, jumlah in jumlah_per_kelas.items():
        self.probabilitas_kelas[label] = jumlah / jumlah_data # P(H) = Jumlah kelas / Total data

    jumlah_kata_per_kelas = defaultdict(lambda: defaultdict(int))
    total_kata_per_kelas = defaultdict(int)

    for teks, label in zip(daftar_teks, daftar_label):
        kata_kata = self.bersihkan_teks(teks)
        for kata in kata_kata:
            jumlah_kata_per_kelas[label][kata] += 1
            total_kata_per_kelas[label] += 1

```

```

        jumlah_kata_per_kelas[label][kata] += 1
        total_kata_per_kelas[label] += 1
        self.kata_unik.add(kata)
        ukuran_kamus = len(self.kata_unik)

    for label in jumlah_per_kelas.keys():
        total_kata = jumlah_per_kelas[label]
        for kata in self.kata_unik:
            jumlah = jumlah_kata_per_kelas[label][kata]
            self.probabilitas_kata[label][kata] = (jumlah + 1) /
            (total_kata + ukuran_kamus) # P(X|H)

    def prediksi(self, teks):
        """Memprediksi kelas berdasarkan teks input."""
        kata_kata = self.bersihkan_teks(teks)
        hasil_kemungkinan = {}

        for label in self.probabilitas_kelas.keys():
            peluang = self.probabilitas_kelas[label] # P(H)
            for kata in kata_kata:
                if kata in self.kata_unik:
                    peluang *= self.probabilitas_kata[label][kata] #
            P(H|X) = P(X|H) * P(H)
            hasil_kemungkinan[label] = peluang

        # Pilih kelas dengan probabilitas tertinggi
        return max(hasil_kemungkinan, key=hasil_kemungkinan.get)

    def evaluasi(self, daftar_teks, daftar_label):
        """Menghitung akurasi model."""
        prediksi = [self.prediksi(teks) for teks in daftar_teks]
        benar = sum(1 for pred, asli in zip(prediksi, daftar_label) if pred ==

```

```

asli)

    akurasi = benar / len(daftar_label)
    return akurasi, prediksi

# Fungsi utama untuk menjalankan program
def main():

    file_latih = "train_data.csv"
    file_uji = "test_data.csv"
    kolom_teks = "Stemming"
    kolom_label = "Label"

    model = NaiveBayes()
    teks_latih, label_latih = model.muat_data_csv(file_latih, kolom_teks,
    kolom_label)
    teks_uji, label_uji = model.muat_data_csv(file_uji, kolom_teks,
    kolom_label)

    model.latih_model(teks_latih, label_latih)
    akurasi, hasil_prediksi = model.evaluasi(teks_uji, label_uji)

    print(f"Akurasi Model: {akurasi:.2%}")

    with open("akurasi.txt", "w") as f:
        f.write(f"{akurasi:.4f}")

    with open("Prediksi_Naive_Bayes.csv", "w", newline="", encoding='utf-8') as f:
        penulis = csv.writer(f)
        penulis.writerow(["Teks", "Label Asli", "Prediksi"])
        for teks, label_asli, label_prediksi in zip(teks_uji, label_uji,
        hasil_prediksi):
            penulis.writerow([teks, label_asli, label_prediksi])

```

```
if __name__ == "__main__":
    main()
```

```
# Program Confusion Matrix untuk Naïve Bayes

import csv
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score

def load_predictions(file_path):
    true_labels = []
    pred_labels = []

    with open(file_path, 'r', encoding='utf-8') as file:
        csv_reader = csv.reader(file)
        next(csv_reader) # Lewati header
        for row in csv_reader:
            if len(row) < 3:
                continue # Lewati baris tidak valid
            true_labels.append(row[1]) # Label Asli
            pred_labels.append(row[2]) # Label Prediksi

    return true_labels, pred_labels

def plot_confusion_matrix(true_labels, pred_labels):
    labels = ["1", "-1", "0"] # Urutan sesuai permintaan
    cm = confusion_matrix(true_labels, pred_labels, labels=labels)

    plt.figure(figsize=(6, 4))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,
```

```

yticklabels=labels)

plt.xlabel("Prediksi")
plt.ylabel("Aktual")
plt.title("Confusion Matrix Naive Bayes")
plt.show()

def main():
    input_file = "Prediksi_Naive_Bayes.csv" # Sesuaikan dengan nama file
hasil prediksi

    try:
        true_labels, pred_labels = load_predictions(input_file)
        print("Confusion Matrix:")
        cm = confusion_matrix(true_labels, pred_labels, labels=["1", "-1",
"0"])
        print(cm)

        # Hitung dan tampilkan akurasi
        accuracy = accuracy_score(true_labels, pred_labels)
        print(f'Accuracy: {accuracy:.2%}')

        # Gunakan `zero_division=1` agar tidak ada peringatan
        print("Classification Report:")
        print(classification_report(true_labels, pred_labels, labels=["1", "-1",
"0"], zero_division=1))

        # Tampilkan plot confusion matrix
        plot_confusion_matrix(true_labels, pred_labels)

    except FileNotFoundError:
        print(f'Error: File {input_file} not found.')
    except Exception as e:
        print(f'Error: {str(e)}')

```

```
if __name__ == "__main__":
    main()
```

```
# Visualisasi WordCloud Naïve Bayes

import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

data = pd.read_csv("Prediksi_Naive_Bayes.csv")
data["Teks"] = data["Teks"].astype(str).fillna("")
all_words = " ".join(data["Teks"])

# Buat WordCloud dari semua kata
wordcloud = WordCloud(
    width=2000, height=1000,
    background_color="white",
    colormap="coolwarm",
    stopwords=STOPWORDS
).generate(all_words)

# Tampilkan WordCloud dalam satu gambar
plt.figure(figsize=(12, 6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("WordCloud Naive Bayes Gabungan Semua Sentimen", fontsize=14,
fontweight="bold")
plt.show()
```

```
# Klasifikasi Random Forest

import csv
import numpy as np
import pandas as pd
```

```

from collections import Counter
import math

# Fungsi untuk memuat data CSV
def load_csv_data(file_path, text_column, label_column):
    texts = []
    labels = []
    with open(file_path, 'r', encoding='utf-8') as file:
        csv_reader = csv.DictReader(file)
        for row in csv_reader:
            if row[text_column] and row[label_column]:
                texts.append(row[text_column])
                labels.append(int(row[label_column])) # Pastikan label berupa integer
    return texts, labels

# Fungsi untuk menghitung entropi
def entropy(y):
    y = np.array(y)
    if len(y) == 0:
        return 0
    _, counts = np.unique(y, return_counts=True)
    probabilities = counts / len(y)
    return -np.sum(probabilities * np.log2(probabilities))

# Fungsi untuk menghitung Information Gain
def information_gain(y, y_left, y_right):
    H_parent = entropy(y)
    w_left = len(y_left) / len(y)
    w_right = len(y_right) / len(y)
    return H_parent - (w_left * entropy(y_left) + w_right * entropy(y_right))

```

```

# Fungsi untuk mencari split terbaik
def find_best_split(X, y, max_features):
    num_features = X.shape[1]
    feature_indices = np.random.choice(num_features, size=max_features,
    replace=False)
    best_feature = None
    best_threshold = None
    best_gain = 0

    for feature in feature_indices:
        thresholds = np.unique(X[:, feature])
        for threshold in thresholds:
            left_indices = X[:, feature] <= threshold
            right_indices = ~left_indices

            if np.sum(left_indices) == 0 or np.sum(right_indices) == 0:
                continue

            gain = information_gain(y, y[left_indices], y[right_indices])
            if gain > best_gain:
                best_gain = gain
                best_feature = feature
                best_threshold = threshold

    return best_feature, best_threshold

# Fungsi untuk membangun Decision Tree secara rekursif dengan pemilihan
fitur acak
def build_decision_tree(X_train, y_train, max_depth=None,
max_features=None):
    tree = {'feature': None, 'threshold': None, 'left': None, 'right': None, 'label':
None}

```

```

if max_depth == 0 or len(set(y_train)) == 1:
    tree['label'] = Counter(y_train).most_common(1)[0][0]
    return tree

if max_features is None:
    max_features = int(np.sqrt(X_train.shape[1])) # Default pemilihan
    fitur

    best_feature, best_threshold = find_best_split(X_train, y_train,
max_features)

    if best_feature is None:
        tree['label'] = Counter(y_train).most_common(1)[0][0]
        return tree

    left_indices = X_train[:, best_feature] <= best_threshold
    right_indices = ~left_indices

    tree['feature'] = best_feature
    tree['threshold'] = best_threshold
    tree['left'] = build_decision_tree(X_train[left_indices],
np.array(y_train)[left_indices],
                                         max_depth - 1 if max_depth is
not None else None,
                                         max_features)

    tree['right'] = build_decision_tree(X_train[right_indices],
np.array(y_train)[right_indices],
                                         max_depth - 1 if max_depth
is not None else None,
                                         max_features)

    return tree

# Fungsi prediksi menggunakan Decision Tree
def tree_predict(tree, x):

```

```

if tree['label'] is not None:
    return tree['label']

if tree['feature'] is None or tree['threshold'] is None:
    return None # Pencegahan error jika tree belum terlatih dengan
baik

if x[tree['feature']] <= tree['threshold']:
    return tree_predict(tree['left'], x)
else:
    return tree_predict(tree['right'], x)

# Fungsi untuk melatih Random Forest
def random_forest_train(X_train, y_train, n_estimators=10, max_depth=None):
    trees = []
    for _ in range(n_estimators):
        # Bootstrap sampling
        indices = np.random.choice(len(X_train), size=len(X_train),
replace=True)
        X_subset = X_train[indices]
        y_subset = np.array(y_train)[indices]
        tree = build_decision_tree(X_subset, y_subset, max_depth)
        trees.append(tree)
    return trees

# Fungsi untuk prediksi menggunakan Random Forest
def random_forest_predict(trees, X_test):
    predictions = []
    for x in X_test:
        tree_preds = [tree_predict(tree, x) for tree in trees]
        majority_pred = Counter(tree_preds).most_common(1)[0][0]
        predictions.append(majority_pred)
    return predictions

# Program Utama

```

```

def main():
    # Nama file data training dan testing
    train_file = "train_data.csv"
    test_file = "test_data.csv"
    text_column = "Stemming"
    label_column = "Label"

    # Melatih Random Forest menggunakan implementasi manual
    trees = random_forest_train(train_data, np.array(train_labels),
n_estimators=10, max_depth=5)

    # Melakukan prediksi pada data testing
    predictions = random_forest_predict(trees, test_data)
    accuracy = np.mean(np.array(predictions) == np.array(test_labels)) * 100
    print(f"Accuracy: {accuracy:.2f}%")

    predictions_df = pd.DataFrame({
        "Komentar": test_texts,
        "Actual": test_labels,
        "Prediksi": predictions
    })
    predictions_df.to_csv("Random_Forest.csv", index=False)

if __name__ == "__main__":
    main()

```

```

# Confusion Matrix untuk Random Forest
import csv
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, classification_report,

```

```

accuracy_score

def load_predictions(file_path):
    true_labels = []
    pred_labels = []

    with open(file_path, 'r', encoding='utf-8') as file:
        csv_reader = csv.reader(file)
        next(csv_reader) # Lewati header
        for row in csv_reader:
            if len(row) < 3:
                continue # Lewati baris tidak valid
            true_labels.append(row[1]) # Label Asli
            pred_labels.append(row[2]) # Label Prediksi
    return true_labels, pred_labels

def plot_confusion_matrix(true_labels, pred_labels):
    labels = ["1", "-1", "0"] # Urutan sesuai permintaan
    cm = confusion_matrix(true_labels, pred_labels, labels=labels)

    plt.figure(figsize=(6, 4))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=labels,
    yticklabels=labels)
    plt.xlabel("Prediksi")
    plt.ylabel("Label")
    plt.title("Confusion Matrix Random Forest")
    plt.show()

def main():
    input_file = "Random_Forest.csv" # Sesuaikan dengan nama file hasil
    prediksi
    try:

```

```

true_labels, pred_labels = load_predictions(input_file)

# Hitung Confusion Matrix
print("Confusion Matrix:")
cm = confusion_matrix(true_labels, pred_labels, labels=["1", "-1",
"0"])
print(cm)

# Hitung dan tampilkan akurasi
accuracy = accuracy_score(true_labels, pred_labels)
print(f'Accuracy: {accuracy:.2%}')

# Gunakan `zero_division=1` agar tidak ada peringatan
print("Classification Report:")
print(classification_report(true_labels, pred_labels, labels=["1", "-1",
"0"], zero_division=1))
plot_confusion_matrix(true_labels, pred_labels)

except FileNotFoundError:
    print(f"Error: File {input_file} not found.")
except Exception as e:
    print(f"Error: {str(e)}")

if __name__ == "__main__":
    main()

```

```

# Visualisasi WordCloud Naïve Bayes
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS
data = pd.read_csv("Random_Forest.csv")
# Pastikan kolom "Stemming" tidak memiliki nilai NaN

```

```

data["Komentar"] = data["Komentar"].astype(str).fillna("")

# Gabungkan semua teks dari semua kategori prediksi
all_words = " ".join(data["Komentar"])

# Buat WordCloud dari semua kata
wordcloud = WordCloud(
    width=2000, height=1000,
    background_color="white",
    colormap="coolwarm",
    stopwords=STOPWORDS
).generate(all_words)

# Tampilkan WordCloud dalam satu gambar
plt.figure(figsize=(12, 6))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.title("WordCloud Random Forest Gabungan Semua Sentimen", fontsize=14,
fontweight="bold")
plt.show()

```

Program Sesudah *oversampling* sama dengan program di atas yang ditambahkan hanya pada proses *oversampling*, di bawah ini:

```

# Program Penyeimbangan data (Oversampling)
import pandas as pd
from imblearn.over_sampling import RandomOverSampler
data = pd.read_csv("TF_IDF.csv")

print("Jumlah data sebelum oversampling:")
print(data['Label'].value_counts())

# Pisahkan fitur (X) dan label (y)

```

```
X = data.drop(columns=['Label'])

y = data['Label']

ros = RandomOverSampler(random_state=42)

X_resampled, y_resampled = ros.fit_resample(X, y)

data_resampled = pd.concat([X_resampled, y_resampled], axis=1)

print("\nJumlah data setelah oversampling:")
print(data_resampled['Label'].value_counts())

data_resampled = data_resampled[['Stemming', 'Label']]

data_resampled.to_csv("Data_Oversampling.csv", index=False)
```

RIWAYAT HIDUP



Nama Qalbi Almustika M, lahir pada tanggal 25 Januari 2003 di Parombean, Kecamatan Curio, Kabupaten Enrekang Sulawesi Selatan. Anak kedua dari sembilan bersaudara, anak dari bapak "Muslimin S" dan ibu "Ratna Sari". Penulis pertama kali menempuh pendidikan tepat pada umur 7 tahun di sekolah TK Aba Desa Parombean, pada tahun 2008 dan selesai pada tahun 2009, dan melanjutkan ke jenjang selanjutnya yaitu Sekolah Dasar (SD) di SDN 30 Parombean tahun 2009 dan lulus pada tahun 2015, dan pada tahun yang sama melanjutkan pendidikan di (MTs) Madrasah Tsanawiyah Swasta Al-Hikmah Parombean dan lulus pada tahun 2018, dan pada tahun yang sama penulis melanjutkan pendidikan di sekolah menengah kejuruan (SMK) pada SMK Negeri 1 Enrekang dan lulus pada tahun 2021. Pada tahun 2021 penulis melanjutkan pendidikan ke jenjang S1 dan diterima di Universitas Sulawesi Barat Fakultas Teknik Prodi Informatika, dan selesai pada tahun 2025.

Berkat petunjuk dan rahmat Allah SWT serta usaha dan doa dari kedua orang tua dan semua keluarga serta orang-orang terdekat dalam menjalani semua aktivitas akademik, alhamdulillah penulis dapat menyelesaikan tugas akhir dengan skripsi yang berjudul "Perbandingan Algoritma *Naïve Bayes* dengan Algoritma *Random Forest* untuk Analisis Sentimen Pengunjung Wisata Wai Malino di Kabupaten Enrekang."

Karena kesempurnaan cuma milik yang maha kuasa, maka penulis mengharapkan kritik dan saran mengenai skripsi yang dibuat ini, kritik dan saran dapat dikirim atau disampaikan pada alamat email: qalbimustika25@gmail.com.