BAB II

TINJAUAN PUSTAKA

II.1 Landasan Teori

II.1.1 Analisis Sentimen

Analisis sentimen merupakan cabang dari *Natural Language Processing* (NLP) yang berfokus pada pengenalan dan klasifikasi opini, emosi, atau sikap dalam bentuk teks. Teknik ini bertujuan mengelompokkan ekspresi pengguna menjadi polaritas tertentu, yaitu positif, negatif, atau netral (Tarigan, 2024). Konteks digital saat ini, analisis sentimen memainkan peran penting dalam memahami persepsi publik terhadap produk, layanan, atau isu sosial.

II.1.2 *Natural Language Toolkit*(NLTK)

Natural Language Toolkit (NLTK) adalah kumpulan pustaka dan program yang dikembangkan untuk mendukung pemrosesan bahasa alami (*Natural Language Processing*/NLP) menggunakan bahasa Python. NLTK menyediakan antarmuka yang intuitif untuk berbagai tugas pengolahan teks, seperti *Case Folding, Tokenizing, spelling correction, filtering,* hingga *Stemming*. Alat ini sangat populer dalam bidang pengajaran dan penelitian linguistik komputasional karena dilengkapi dengan dokumentasi API yang lengkap serta panduan praktis yang mengajarkan dasar-dasar pemrograman dan konsep linguistik komputasional (Umunyana dkk., 2024).

a. *Case Folding*

Case Folding adalah proses standarisasi teks dengan mengubah seluruh huruf menjadi huruf kecil. Karakter yang bukan huruf alfabet, seperti angka dan tanda baca, biasanya dihapus karena dianggap sebagai pemisah atau delimiter (Alparisi, 2024).

b. *Tokenizing*

Tokenizing adalah teknik pemecahan kalimat menjadi unit-unit kata. Umumnya, proses ini dilakukan dengan memisahkan kata berdasarkan spasi atau *white space* (Alparisi, 2024).

c. *Spelling correction*

Spelling correction merupakan tahapan untuk memperbaiki kesalahan penulisan kata dalam teks. Sering kali pengguna menulis dengan singkatan, salah ketik, atau menggunakan bahasa tidak baku, langkah ini penting untuk memastikan kualitas data sebelum dianalisis (Alparisi, 2024).

d. *Filtering*

Filtering, yang juga dikenal sebagai penghapusan stopword, merupakan proses eliminasi kata-kata umum yang dianggap tidak memberikan makna penting dalam analisis, seperti kata sambung atau kata kerja bantu (Alparisi, 2024).

e. *Stemming*

Stemming adalah teknik untuk mengembalikan kata ke bentuk dasarnya dengan menghapus imbuhan seperti awalan, akhiran, atau gabungan keduanya. Proses ini membantu menyederhanakan variasi kata menjadi satu bentuk dasar (Alparisi, 2024).

f. *Wordcloud*

Wordcloud merupakan teknik visualisasi frekuensi kata yang digunakan untuk menggambarkan kata-kata yang paling dominan dalam sebuah korpus teks. Visualisasi ini, semakin besar ukuran suatu kata maka semakin sering kata tersebut muncul dalam data. Wordcloud digunakan untuk mengevaluasi representasi semantik data sentimen positif dan negatif, serta memastikan bahwa hasil preprocessing telah menghasilkan kata-kata yang relevan dan bermakna. Visualisasi ini memperkuat keyakinan bahwa hasil tokenisasi, filtering, dan stemming telah berhasil mengekstrak kata-kata utama yang mencerminkan sentimen aktual dari pengguna (S. Sharma, 2024).

g. Analisis N-gram

Analisis N-gram adalah teknik eksplorasi teks yang digunakan untuk mengidentifikasi frekuensi kemunculan sekuens kata (urutan kata) dalam korpus. Penelitian ini menggunakan unigram untuk mengekstrak kata penting yang sering muncul dalam setiap kelas sentimen. N-gram berfungsi tidak hanya sebagai alat bantu visualisasi, tetapi juga sebagai metode eksploratif dalam pengujian kualitas data sebelum dan sesudah proses balancing. Hasil dari tahap ini turut berkontribusi dalam peningkatan akurasi pemodelan sentimen, terutama dalam konteks data tidak seimbang (imbalanced data) (Trueman dkk., 2022).

II.1.3 *Term Frequency – Inverse Document Frequency* (TF-IDF)

Algoritma ekstraksi yang paling umum digunakan adalah TF-IDF. Nilai frekuensi kemunculan suatu kata dalam suatu dataset dikenal dengan *term frequency* (TF). Sementara itu, *Inverse Document Frequency* (IDF) digunakan untuk menentukan seberapa penting kata tersebut dalam dataset. Suatu kata akan memiliki bobot yang rendah jika sering muncul dalam setiap dokumen dalam dataset. Jika tidak, maka bobotnya akan lebih besar. Rumus untuk menghitung TF-IDF adalah Persamaan (1), di mana nilai tf adalah frekuensi kemunculan kata t dalam dokumen word, df adalah jumlah dokumen yang memuat kata t, dan N adalah jumlah total dokumen dalam dataset (Nurhaliza Agustina dkk., 2024).

$$TF - IDF(word) = TF(word) * log\left(\frac{N}{df(word)}\right) \quad (1)$$

II.1.4 *Machine learning* (ML)

Machine learning merupakan cabang dari kecerdasan buatan *(Artificial Intelligence/*AI*)* yang memungkinkan sistem komputer belajar dari data tanpa harus diprogram secara eksplisit (P. Sharma dkk., 2024). Teknologi ini memungkinkan sistem untuk secara otomatis memperoleh pengetahuan dari data dan membuat keputusan secara mandiri tanpa campur tangan manusia. Kemampuan ini, komputer menjadi semakin cerdas karena dapat menyimpulkan informasi dari data yang tersedia. Proses pembelajarannya, *machine learning* menggunakan data sebagai bahan utama untuk melakukan pelatihan guna menghasilkan prediksi atau analisis yang akurat. Umumnya, data dalam *machine learning* dibagi menjadi dua kelompok: data pelatihan (*training* data) yang digunakan untuk melatih model, dan

data pengujian (*testing* data) yang digunakan untuk mengevaluasi kinerja model tersebut. Selain itu, data validasi (*validation* data) juga sering digunakan sebagai acuan tambahan untuk menilai efektivitas algoritma jika hasil dari pelatihan kurang optimal (Tarigan, 2024).

Teknik dalam *machine learning* dikategorikan menjadi beberapa jenis, yaitu *supervised learning, unsupervised learning, reinforcement learning*, serta metode yang lebih kompleks seperti *Neural Network* dan *Deep Learning* (P. Sharma dkk., 2024). *Supervised learning*, data yang digunakan memiliki label sehingga sistem dapat belajar dengan memahami hubungan antara input dan output. Sebaliknya, *unsupervised learning* memanfaatkan data yang tidak berlabel, dan sistem akan mengelompokkan data berdasarkan pola atau kemiripan yang ditemukan. Sementara itu, *reinforcement learning* bekerja dengan mekanisme umpan balik, di mana sistem belajar melalui proses coba-coba (trial and error), dan setiap keputusan yang tepat akan diperkuat. Adapun *Deep Learning* adalah pendekatan lanjutan dari *machine learning* yang menggunakan jaringan saraf tiruan berlapis-lapis (*layered Neural Networks*) untuk menganalisis data secara mendalam dan berulang, terutama efektif dalam menangani data yang kompleks dan tidak terstruktur (Tarigan, 2024).

II.1.5 *Naïve Bayes Classifier* (NBC)

Naïve Bayes Classifier merupakan metode klasifikasi yang didasarkan pada Teorema Bayes. *Naïve Bayes Classifier* bukanlah suatu algoritma tunggal, tetapi serangkaian algoritma klasifikasi berdasarkan Teorema Bayes yang menggambarkan probabilitas suatu peristiwa berdasarkan pengetahuan sebelumnya

atau probabilitas lain yang diketahui dari peristiwa tersebut (Ernianti Hasibuan dkk., 2022). Secara umum rumus *Naïve Bayes Classifier* dapat dilihat pada persamaan (2):

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \quad (2)$$

Keterangan:

X : data dengan kelas yang belum diketahui.

H : hipotesis data yang merupakan suatu kelas spesifik

P (H | X) : probabilitas hipotesis H berdasar kondisi X (posterior)

P (H) : probabilitas hipotesis H. (prior probabilitas)

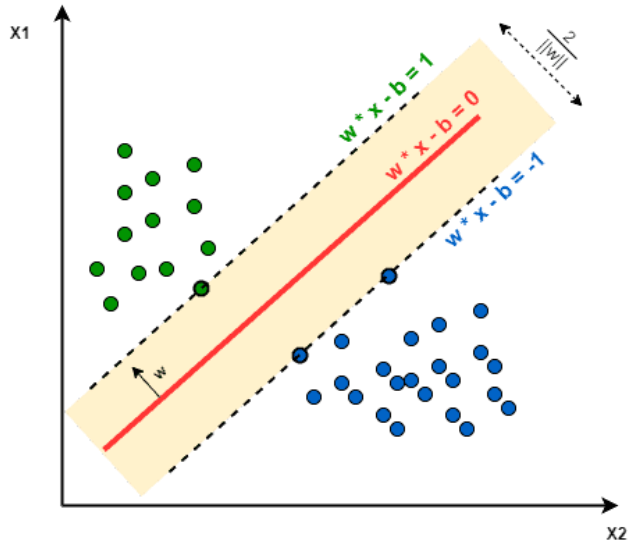
P (X | H) : probabilitas X berdasar kondisipada hipotesis H

P (X) : probabilitas X

Naïve Bayes Classifier menggunakan asumsi independensi antarfitur untuk mempredeksi probabilitas keanggotaan suatu kelas (Tanggraeni dkk., 2022). Penggabungan posterior probabilitas dan prior probabilitas dalam rumus *Naïve Bayes* digunakan untuk menghitung probabilitas dari setiap kemungkinan klasifikasi (Indarso dkk., 2023).

II.1.6 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) adalah algoritma *supervised learning* yang digunakan klasifikasi dengan menentukan hyperplane. Hyperplane yang sesuai terletak tepat di tengah-tengah kedua kelas dan mempunyai jarak terjauh terhadap data terluar kedua kelas (Baita dkk., 2021). Gambar 2.1 merupakan penjelasan kinerja *Support Vector Machine* (SVM).



Gambar 2.1 Kinerja Algoritma *Support Vector Machine* (SVM)

(J. Guo dkk., 2024)

Berdasarkan Gambar 2.1 garis merah merupakan hyperplane yang mengklasifikasikan kedua titik tersebut, dimana titik data pada salah satu sisi diberi label kelas negatif yaitu -1, kelas positif diberi label +1, dan kelas netral diberi label dengan 0 berada di tengah garis merah. Data terdekat terhadap hyperplane disebut dengan support vector. Jarak antara hyperplane dan support vector disebut dengan Margin. Kernel yang digunakan adalah kernel linear. Kernel linear merupakan kernel yang sederhana dan sering digunakan untuk kasus klasifikasi teks (Rahman Isnain dkk., 2021).

Support Vector Machine (SVM) memiliki metode dalam membangun dan melatih model klasifikasi yaitu Sequential *Training*. Tahapan dari metode tersebut dijelaskan sebagai berikut.

a. Inisiasi parameter nilai Complexity (C), epsilon (ε), alpha (αi), lamda (λ), gamma (γ)

b. Menghitung fungsi kernel linear

$$K(x_a, x_b) = (x_a * x_b) \tag{3}$$

(Zy dkk., 2023)

c. Menghitung matriks Hessian

$$D_{ab} = y_a y_b (K(x_a, x_b) + \lambda^2) \tag{4}$$

(Zy dkk., 2023)

Keterangan:

Dab : elemen matriks a dan b

ya : label data a

yb : Label data b

d. Menghitung nilai error, delta alpha dan alpha baru berdasarkan iterasi

$$\begin{gathered} E_i = \sum \alpha_i D_{ab} \\ \delta\alpha_i = \min\{\max[\gamma(1 - E_i), \alpha_i], C - \alpha_i\} \\ \alpha_i = \alpha_i + \delta\alpha_i \end{gathered} \tag{5}$$

(Zy dkk., 2023)

Keterangan:

Ei : nilai error

$\delta\alpha i$: parameter delta alpha

α_i : parameter alpha baru

e. Menghitung bias

$$\begin{aligned} b = -\frac{1}{2} * \sum \alpha_i y_i (K(x_{i,} x^+) + \sum \alpha_i y_i (K(x_{i,} x^0) \\ + \sum \alpha_i y_i (K(x_{i,} x^-) \end{aligned} \tag{6}$$

(Zy dkk., 2023)

f. Menghitung nilai keputusan

$$sign(h(x)) = \sum \alpha_i y_i (K(x_{a,} x_{b,}) + b \quad (7)$$

(Zy dkk., 2023)

II.1.7 *Random Forest* (RF)

Random Forest adalah cara untuk mengklasifikasikan kumpulan pohon keputusan. Metode ini menggunakan data latih dan berbagai fitur acak untuk menghasilkan suara, yang akan menentukan hasil akhir. Untuk menghasilkan prediksi akhir, setiap pohon keputusan dalam kelompok akan menentukan node akar dan node akhir bersama dengan beberapa node daun (Klyueva, 2019). *Random Forest* memiliki tiga komponen penting: (1) sampling bootstrap untuk membuat pohon prediksi; (2) menggunakan prediktor acak untuk setiap pohon keputusan; dan (3) menggabungkan hasil dari setiap pohon keputusan menggunakan metode pemilihan mayoritas untuk klasifikasi atau regresi rata-rata (Ferdita Nugraha dkk., 2022). Keuntungan dari teknik ini adalah kemampuan untuk mengklasifikasikan data yang memiliki fitur yang tidak lengkap (V. A. Fitri dkk., 2019) (Yuda Irawan dkk., 2024):

$$Entropy\ (Y) = -\sum i\ p(c|Y) log^2 p(c|Y) \quad (8)$$

(Asvia, 2023)

Keterangan :

Y : Himpunan kasus

P(c|Y) : Proporsi nilai Y terhadap kelas c

$$information\ Gain\ (Y, a)$$

$$= Entropy(Y) - \sum v\varepsilon Values(a)\frac{|Yv|}{|Ya|}Entropy(Yv) \quad (9)$$

(Asvia, 2023)

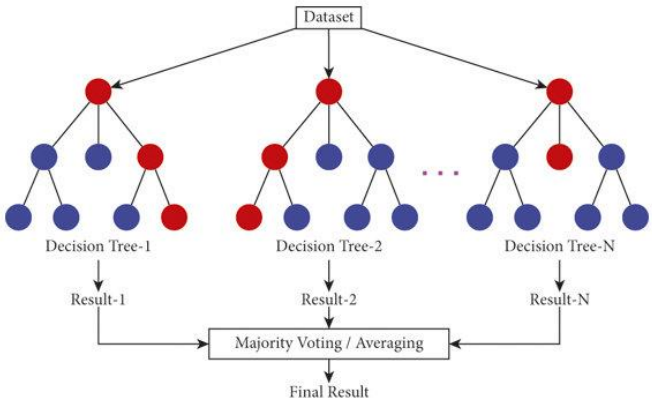
Keterangan :

Values (a): Nilai yang mungkin dalam himpunan kasus a

Yv : Subkelas dari Y dengan kelas v yang berhubungan dengan kelas a

Ya : Semua nilai yang sesuai dengan a

Gambar 2.2 merupakan gambaran sederhana dari proses klasifikasi metode *Random Forest*.



Gambar 2.2 Metode *Random Forest*

(Ridwan dkk., 2024)

II.1.8 *Synthetic Minority Over-sampling Technique* (SMOTE)

Jika kelas data memiliki jumlah objek yang lebih besar daripada kelas data lainnya, itu disebut *Imbalanced* data. Kelas data yang memiliki jumlah objek yang

lebih besar disebut kelas mayor, dan kelas data yang memiliki jumlah objek yang lebih sedikit disebut kelas minor. Akibatnya, pengolahan algoritma yang tidak memperhatikan *Imbalanced* data cenderung meliputi kelas mayor dan mengacuhkan kelas minor. Pengklasifikasian dataset tidak dipresentasikan secara merata, teknik SMOTE membantu menangani masalah ketidakseimbangan tersebut. Cara kerjanya adalah dengan menggunakan teknik pembangkitan data buatan untuk menambah jumlah data kelas minor hingga setara dengan kelas mayor. (Utama, 2023).

$$x_{syn} = x_i + (x_{knn} - x_i) \times \delta \quad (10)$$

(Nikmatul Kasanah dkk., 2019)

Keterangan :

x_{syn} : data sintesis yang akan diciptakan

x_i : data yang akan direplika

x_{knn} : data yang memiliki jarak terdekat dari data yang akan direplikasi

δ : nilai *Random*antara 0 dan 1

II.1.9 *Borderline-SMOTE*

Algoritma *Borderline-SMOTE* melakukan *oversampling* pada instance kelas minoritas di dekat garis batas, mengatasi sampel kelas yang tumpang tindih karena lokasi data kelas mayoritas berdekatan saat mensintesis data kelas minoritas. di mana x adalah sampel minoritas yang ada, y adalah tetangga terdekat, dan δ

adalah parameter yang mengontrol seberapa banyak data sintetis yang dihasilkan (S. Guo dkk., 2020).

$$z = x + \delta(y - x) \tag{11}$$

Keterangan :

z : data titik baru yang terletak di antara x dan y

x : data sampel minoritas yang ada

y : data yang memiliki jarak terdekat

δ : parameter yang mengontrol seberapa banyak data sintetis yang dihasilkan

II.1.10 *Random Oversampling* (ROS)

Teknik *oversampling* terakhir yang digunakan dalam eksperimen ini adalah *Random Oversampling*. Metode ini bekerja dengan menambahkan salinan secara acak dari data pada kelas minoritas hingga jumlahnya seimbang dengan kelas mayoritas. Tujuan dari pendekatan ini adalah untuk meningkatkan representasi kelas minoritas dalam dataset, sehingga algoritma *machine learning* dapat mempelajari pola dari kedua kelas secara lebih seimbang. Diharapkan performa model dalam mengklasifikasikan data dari kelas minoritas dapat meningkat secara signifikan pada saat implementasi (Hayaty dkk., 2021).

II.1.11 *Confusion matrix*

Confusion matrix merupakan matriks untuk mengukur performa model klasifikasi melalui perhitungan *Accuracy*, *Precision*, *Recall*, dan *F1-Score* (Azmi dkk., 2023). Tabel 2.1 merupakan kerangka tabel confusion matriks.

Tabel 2.1 *Confusion matrix*

Confusion matrix	**Prediction Positif**	**Prediction Negatif**
Actual Positif	TP	FN
Actual Negatif	FP	TN

Dimana:

- TP (True Positif), banyaknya data yang kelas aktualnya adalah positif dengan kelas prediksinya merupakan positif.
- FN (False Negatif), banyaknya data yang kelas aktualnya adalah positif dengan kelas prediksinya adalah negatif.
- FP (False Positif), banyaknya data yang kelas aktualnya adalah negatif dengan kelas prediksinya adalah positif.
- TN (True Negatif), banyaknya data yang kelas aktualnya negatif dengan kelas prediksinya adalah negatif

Berdasarkan Tabel 2.1 terdapat 4 indikator yang dihitung untuk menguji performa, yaitu:

a. *Accuracy*

Accuracy, merupakan rasio prediksi benar (positif dan negatif) dengan keseluruhan data. *Accuracy* dapat dihitung dengan persamaan (12):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

b. *Precission*

Precission, merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang di prediksi positif. *Precission* dapat dihitung dengan persamaan (13):

$$Precission = \frac{TP}{TP + FP} \quad (13)$$

c. *Recall*

Recall, merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. *Recall* dapat dihitung dengan persamaan (14):

$$Recall = \frac{TP}{TP + FN} \quad (14)$$

d. *F1-Score*

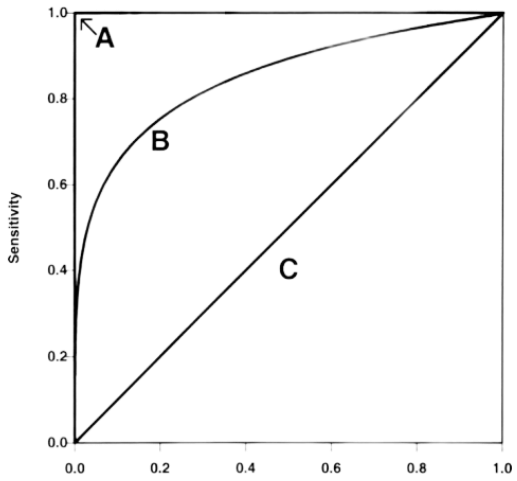
F1- Score, merupakan perbandingan rata-rata *precission* dan *Recall* yang dibobotkan. *F1-Score* dapat dihitung dengan persamaan (15):

$$F1 - Score = \frac{2(Precission \times Recall)}{(Precission + Recall)} \quad (15)$$

II.1.12 AUC-ROC

Hasil akurasinya cenderung tinggi, hal ini dikarenakan model hanya berfokus pada pertimbangan kelas data mayoritas. Penggunaan metode AUC (*Area Under the ROC Curve*) dinilai cocok untuk kasus permasalahan yang *Imbalanced* karena

AUC mampu mengevaluasi prediktor secara komprehensif (Zhang dkk., 2011) dan menilai model mana yang lebih baik secara rata-rata. Evaluasi AUC (*Area Under the ROC*) Klasifikasi *Imbalanced* data dilakukan melalui klasifikasi simetris (Saifudin dkk., 2015). ROC AUC merupakan alat yang juga digunakan untuk menyajikan informasi kinerja algoritma klasifikasi dalam bentuk grafik kurva. Informasi kurva diperoleh berdasarkan hasil perhitungan matriks konfusi, yaitu antara *False Positive Rate* (FPR) dan *True Positive Rate* (TPR) (Carrington dkk., 2023).



Gambar 2. 3 Bentuk kurva ROC

(Carrington dkk., 2023)

Berdasarkan gambar 2.3, terdapat tiga hipotesis kurva yang masing-masing merupakan analisis akurasi. Simbol huruf "A" merupakan area dengan hasil akurasi terbaik (di atas standar), yaitu AUC = 1, "B" merupakan bentuk kurva ROC, di mana AUC = 0,85 dan garis diagonal merupakan pengklasifikasi acak yang sesuai. Indikasi peningkatan hasil analisis adalah ketika AUC mendekati angka 1 atau kurva ROC bergerak ke arah simbol A. (Carrington dkk., 2023). Metrik ini

mengukur sejauh mana model mampu membedakan antara kelas positif dan negatif. Nilai AUC berkisar antara 0 hingga 1, dengan interpretasi sebagai berikut:

- AUC mendekati 1.0 menandakan bahwa model memiliki kemampuan sangat baik dalam mengklasifikasikan antara dua kelas.
- AUC sekitar 0.5 menunjukkan bahwa model tidak lebih baik dari tebakan acak.
- AUC di bawah 0.5 bahkan menunjukkan bahwa model salah mengklasifikasikan lebih sering daripada benar.

II.2 Penelitian Terkait (*State – Of – The – Art*)

Penelitian terkait memuat kajian terhadap studi-studi sebelumnya yang memiliki keterkaitan dengan fokus permasalahan serta hasil penelitian yang relevan. Penelitian ini, dilakukan perbandingan performa tiga metode *oversampling*, yaitu SMOTE, *Borderline-SMOTE*, dan *Random Oversampling*, yang masing-masing dikombinasikan dengan algoritma klasifikasi *Naïve Bayes*, *Support Vector Machine* (SVM), dan *Random Forest*. Rangkuman dari penelitian-penelitian terdahulu yang relevan disajikan dalam Tabel 2.2.

Tabel 2.2 *State of The art*

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
1	(Rifka Noor Ikhsan dkk., 2024)	SKKNI, KNN, SMOTE	Menggunakan Metode SKKNI, algoritma KNN dengan optimasi SMOTE	• Penggunaan SMOTE berhasil meningkatkan akurasi model menjadi 81%, menunjukkan efektivitas metode ini dalam menangani imbalanced data. • KNN cenderung kurang efektif dalam menangani dataset teks yang besar, dibandingkan model lain seperti SVM atau *Random Forest*. • Studi hanya menggunakan SMOTE untuk menangani *Imbalanced* data, tanpa membandingkannya dengan teknik lain seperti *Random Oversampling* atau *Borderline-SMOTE*.

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
2	(Jeffson Sagala dkk., 2024)	SMOTE, RF, SVM, NBC	Menggunakan metode SMOTE dengan algoritma RF, SVM, NBC	• *Random Forest* dan SVM memiliki F1-score tertinggi (90%), menunjukkan keandalan model dalam mengklasifikasikan sentimen positif dan negatif. • Recall untuk kelas negatif masih rendah di semua model (*Random Forest*: 33%, SVM: 36%, Naïve Bayes: 17%), menunjukkan bahwa model kesulitan dalam mengidentifikasi ulasan negatif dengan benar. • Penelitian hanya menggunakan SMOTE untuk menangani *Imbalanced* data, tanpa membandingkannya dengan teknik lain seperti *Random Oversampling* atau *Borderline-SMOTE*, yang mungkin bisa memberikan hasil lebih baik.
3	(Ryanto dkk., 2024)	SMOTE, *Naïve Bayes*, Chi-square	Menggunakan metode SMOTE, algoritma *Naïve Bayes* dengan optimasi Chi-square	• Penerapan Chi-Square meningkatkan akurasi model Naïve Bayes dari 76,1% menjadi 78,5%, membuktikan efektivitas teknik seleksi fitur dalam meningkatkan performa model. • Dataset diambil hanya dari satu video YouTube, yang mungkin tidak cukup untuk merepresentasikan opini publik secara luas terhadap Xiaomi SU7. • Studi ini hanya menggunakan *Naïve Bayes*, tanpa membandingkannya dengan model lain seperti *Random Forest*, SVM.

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
4	(Nur Adhan dkk., 2024)	SMOTE, RF	Menggunakan Metode SMOTE, algoritma RF	• *Random Forest* tanpa SMOTE menghasilkan akurasi 84,05%, dengan nilai AUC 0,9166, yang menunjukkan bahwa model memiliki performa yang sangat baik. • False Positive Rate (FPR) meningkat 3,66%, menunjukkan bahwa model lebih sering salah mengklasifikasikan ulasan positif sebagai negatif setelah SMOTE diterapkan. • Penelitian hanya menggunakan *Random Forest*, tanpa membandingkannya dengan model lain seperti SVM, *Naïve Bayes*
5	(Purnamasari dkk., 2023)	SMOTE, Adaboost, *Naïve Bayes*	Menggunakan metode SMOTE, Adaboost, algoritma *Naïve Bayes*	• Tanpa SMOTE, akurasi model hanya 75.40%, tetapi setelah penerapan SMOTE meningkat menjadi 85.87%, dan ketika dikombinasikan dengan Adaboost mencapai 87.05%. • Dataset hanya terdiri dari 500 tweet dalam rentang waktu Desember 2021 – Maret 2022, yang tergolong kecil untuk analisis sentimen media sosial. • Studi hanya membandingkan kombinasi *Naïve Bayes*, SMOTE, dan Adaboost, tanpa mengeksplorasi model lain seperti SVM, *Random Forest*, dengan teknik *oversampling* lain *Borderline-SMOTE*, atau *Random Oversampling*

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
6	(E. Fitri dkk., 2020)	SMOTE, *Naïve Bayes*, *Random Forest*, *Support Vector Machine*	Menggunakan metode SMOTE, algoritma *Naïve Bayes*, *Random Forest* Dan *Support Vector Machine*	• Hasil menunjukkan bahwa *Random Forest* memiliki akurasi tertinggi (97,16%), diikuti oleh SVM (96,01%) dan Naïve Bayes (94,16%). • Dataset yang digunakan hanya terdiri dari 1.629 ulasan pengguna, yang meskipun cukup representatif, masih bisa diperluas dengan data dari berbagai platform lain seperti Twitter atau YouTube. • Studi ini hanya menggunakan SMOTE untuk menangani *Imbalanced* data, tetapi tidak membandingkannya dengan teknik lain seperti *Random Oversampling* atau *Borderline-SMOTE*, yang mungkin dapat memberikan hasil lebih optimal.
7	(Nikmatul Kasanah dkk., 2019)	KNN, SMOTE	Menggunakan algoritma KNN dengan optimasi SMOTE	• SMOTE terbukti dapat meningkatkan akurasi KNN pada nilai k = 1 dan k = 3, dengan peningkatan rata rata 3,36%. • Dataset hanya terdiri dari 200 berita (176 objektif, 24 subjektif), yang tergolong kecil untuk model klasifikasi teks. • Penelitian hanya menggunakan KNN, tanpa membandingkannya dengan algoritma lain seperti SVM, *Naïve Bayes*, atau *Random Forest*.

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
8	(Sutoyo dkk., 2020)	ANN, SMOTE	Menggunakan Algoritma ANN dengan optimasi SMOTE	• Model ANN+SMOTE menghasilkan peningkatan akurasi hingga 87,06%, dibandingkan dengan ANN tanpa SMOTE yang hanya mencapai 86,35%. • Penelitian hanya menggunakan ANN, tanpa membandingkannya dengan model lain seperti *Random Forest*, SVM, atau Naïve Bayes. • Penelitian hanya menggunakan SMOTE, tanpa membandingkannya dengan teknik lain seperti ADASYN, *Borderline-SMOTE*, atau *Random Oversampling* dan hanya menggunakan ANN, tanpa membandingkannya dengan model lain seperti *Random Forest*, SVM, *Naïve Bayes*
9	(Ridwan dkk., 2024)	RF, SVM, LR, NBC, SMOTE, SVM-SMOTE, Kmeans-SMOTE, dan *Borderline-SMOTE*	Menggunakan Algoritma RF, SVM, LR, NBC dengan optimasi SMOTE, SVM-SMOTE, Kmeans-SMOTE, dan *Borderline-SMOTE*	• *Borderline-SMOTE* + *Random Forest* terbukti memberikan hasil terbaik dalam menangani imbalanced data dengan akurasi 84,09%, recall 85,25%, precision 84,55%, dan F1-score 81,16%. • Penelitian hanya menggunakan dataset statis dari tahun 2018 dan belum menggunakan visualisasi data dengan PCA. • Teknik seperti SHAP (SHapley Additive Explanations) atau Feature Importance dari *Random Forest* bisa digunakan untuk memberikan wawasan lebih dalam.

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
10	(Faisal dkk., 2020)	SVM, NB, PSO, SMOTE	Menggunakan Algoritma SVM, NB dengan optimasi PSO, SMOTE	• Hasil menunjukkan peningkatan akurasi setelah optimasi dengan PSO, di mana SVM dengan PSO mencapai akurasi tertinggi 75,03%. Studi ini membandingkan Support Vector Machine (SVM) dan Naïve Bayes (NB) dalam klasifikasi sentimen dan algoritma dikombinasikan dengan Particle Swarm Optimization (PSO) untuk meningkatkan performa model. • Dataset yang digunakan hanya 604 data setelah preprocessing, yang cukup kecil untuk analisis sentimen skala besar. • Penelitian hanya mengoptimalkan model dengan PSO, tanpa mencoba teknik balancing data seperti *Borderline-SMOTE* atau *Random Oversampling* untuk menangani *Imbalanced* data.
11	(Swana dkk., 2022)	NBC, SVM, KNN, Tomek Link dan SMOTE	Menggunakan Algoritma NBC, SVM, KNN dengan optimasi Tomek Link dan SMOTE	• Ditemukan bahwa kombinasi SMOTE dan Tomek Link memberikan hasil terbaik dalam meningkatkan performa model klasifikasi, terutama untuk k-NN. • Jumlah data yang digunakan tidak disebutkan secara rinci, sehingga sulit menilai sejauh mana hasil ini bisa digeneralisasi ke sistem lain. • Penelitian hanya membandingkan SMOTE dan Tomek Link, tetapi tidak mengeksplorasi teknik lain

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
				seperti *Borderline-SMOTE*, atau *Random Oversampling*.
12	(Rahman dkk., 2024)	NB, *Decision Tree*, RF, Binning dan SMOTE	Menggunakan Algoritma NB, *Decision Tree*, RF dengan optimasi Binning dan SMOTE	• Hasil terbaik diperoleh dari model *Random Forest* dengan kombinasi Binning dan SMOTE, yang menunjukkan akurasi yang lebih baik dibanding model lainnya. • Tidak ada informasi rinci mengenai jumlah data, sumber data, dan distribusi variabel yang digunakan. • Penelitian hanya menggunakan SMOTE untuk menangani ketidakseimbangan data (*Imbalanced* data), tanpa membandingkannya dengan teknik lain seperti *Borderline-SMOTE* atau *Random Oversampling*.
13	(Wicaksono dkk., 2024)	XGBoost *Classifier*, SMOTE, ADASYN, dan *Random Oversampling*	Menggunakan Algoritma XGBoost *Classifier* dengan metode SMOTE, ADASYN, dan *Random Oversampling*	• Hasil menunjukkan bahwa teknik *Random Oversampling* memberikan peningkatan akurasi tertinggi (94,44%), dibandingkan SMOTE (92,59%) dan ADASYN (94,36%). • Dataset yang digunakan hanya terdiri dari 2139 data. • Penelitian hanya menggunakan XGBoost, tanpa membandingkannya dengan algoritma lain seperti *Random Forest*, SVM, atau *Naïve Bayes*.

No	Peneliti	Model	Fokus Penelitian	Gap Penelitian
14	(Ananda dkk., 2024)	SVM, NBC, SMOTE	Menggunakan Algoritma SVM, NBC dengan metode SMOTE	• Hasil menunjukkan bahwa SVM memiliki akurasi lebih baik dibandingkan Naïve Bayes. • Hanya menggunakan 3350 tweet. • Penelitian hanya membandingkan SVM dan *Naïve Bayes*, sementara metode lain seperti *Random Forest* bisa saja memberikan hasil yang lebih baik
15	Penelitian yang dilakukan	*Naïve Bayes*, *Random Forest*, *Support Vector Machine*, SMOTE, *Borderline-SMOTE*, *Random Oversampling*	Menggunakan Algoritma *Naïve Bayes*, *Random Forest*, *Support Vector Machine* dengan metode SMOTE, *Borderline-SMOTE*, *Random Oversampling*	Data yang digunakan berjumlah 10.000 ulasan Tokopedia, dengan membandingkan tiga algoritma *machine learning* dan menerapkan tiga teknik *oversampling* untuk mengatasi *Imbalanced* data. Memperlihatkan visualisasi data sebelum *oversampling* dan sesudah *oversampling* dengan PCA.

II.3 Matriks Penelitian

Matriks penelitian berisikan informasi terkait judul dan ruang lingkup yang berisi metode atau algoritma yang digunakan. Matriks penelitian dapat dilihat pada tabel 2.3.

Tabel 2. 3 Matriks Penelitian

No.	Penulis	Judul	Ruang Lingkup								
			Penerapan Metode					Algoritma			
			SMOTE	***Borderline-SMOTE***	**ROS**	**ADASYN**	**etc**	**NB**	**RF**	**SVM**	**etc**
1	(Wicaksono dkk., 2024)	Peningkatan Performa Model *Machine learning* XGBoost *Classifier* melalui Teknik *Oversampling* dalam Prediksi Penyakit AIDS	√		√	√					√
2	(Ridwan dkk., 2024)	Penerapan Metode SMOTE Untuk Mengatasi *Imbalanced* Data Pada Klasifikasi Ujaran Kebencian	√	√				√	√	√	√