OPTIMASI K-NEAREST NEIGHBOR MENGGUNAKAN ALGORITMA SMOTE UNTUK MENGATASI IMBALANCE CLASS PADA KLASIFIKASI ANALISIS SENTIMEN

Andi Surya Firmansyah, Abdul Aziz, Moh. Ahsan.
Teknik Informatika, Fakultas Sains dan Teknologi
Universitas PGRI Kanjuruhan Malang, Jl.S. Supriadi No.48 Malang, Indonesia
andysuryafirmansyah87@gmail.com

ABSTRAK

Penelitian ini menguji kinerja metode K-Nearest Neighbor (KNN) dengan dan tanpa Synthetic Minority Oversampling Technique (SMOTE) dalam klasifikasi analisis sentimen pada data Tweet dengan kata kunci "Jokowi" di Twitter. Pengujian dilakukan dengan berbagai variasi presentase data training dan data testing, menggunakan nilai K=3 untuk KNN tanpa SMOTE, serta K=1 untuk KNN dengan SMOTE. Hasil pengujian menunjukkan bahwa metode KNN dengan SMOTE mencapai akurasi tertinggi sebesar 93%, presisi sebesar 100%, dan recall sebesar 85%, sementara KNN tanpa SMOTE mencapai akurasi 82%, presisi 82%, dan recall 98%. Secara keseluruhan, penggunaan KNN dengan SMOTE memberikan kinerja baik dan akurat dalam mengklasifikasikan sentimen pada data Tweet "Jokowi", khususnya dalam mendeteksi sentimen positif. Penelitian ini menyoroti pentingnya mempertimbangkan metode oversampling seperti SMOTE untuk meningkatkan performa KNN dalam mengatasi ketidakseimbangan data pada klasifikasi analisis sentimen di media sosial Twitter.

***Kata kunci**: Analisis Sentimen, K-Nearest Neighbor (KNN), Synthetic Minority Oversampling Technique (SMOTE), Analisis Sentimen Twitter, Oversampling Data, Evaluasi Kinerja Metode.*

1. PENDAHULUAN

Dalam era digital saat ini, perkembangan pesat media sosial seperti Twitter, Facebook, Instagram, dan Youtube menciptakan berbagai sumber informasi yang memiliki nilai penting dalam berbagai bidang seperti perusahaan, sistem akademik, dan e-commerce. Analisis sentimen atau opinion mining menjadi krusial dalam menggali nilai dari informasi ini.

Analisis sentimen, sebagai teknik pengolahan bahasa alami, digunakan untuk mengidentifikasi opini, sentimen, dan sikap manusia terhadap suatu topik [1]. Salah satu pendekatan yang umum digunakan adalah metode K-Nearest Neighbor (KNN), sebuah teknik klasifikasi yang fokus pada perbedaan jarak di antara contoh-contoh data. Meskipun sederhana, KNN memiliki kelemahan terutama dalam perhitungan jarak yang bisa menjadi lambat untuk data berdimensi tinggi dan dapat dipengaruhi oleh ketidakseimbangan kelas dalam dataset.

Dalam konteks penelitian ini, peneliti merujuk pada kelemahan tersebut dan mengusulkan pengoptimalan metode KNN melalui penerapan algoritma SMOTE. SMOTE telah terbukti efektif mengatasi ketidakseimbangan kelas dalam analisis sentimen [2], meskipun perlu diperhatikan potensi overfitting dalam algoritma tersebut.

Tujuan penelitian ini adalah mengatasi kelemahan metode KNN, terutama terkait ketidakseimbangan data dalam analisis sentimen. Data yang digunakan berupa komentar tweet dengan kata kunci "Jokowi," bertujuan untuk menganalisis opini terhadap Presiden Joko Widodo selama masa jabatannya sebagai Presiden di Indonesia.

Dengan merujuk pada penelitian sebelumnya, seperti yang dilakukan oleh [2], penelitian ini diharapkan memberikan kontribusi dalam pengembangan metode KNN untuk mengatasi masalah ketidakseimbangan kelas dalam klasifikasi analisis sentimen.

2. TINJAUAN PUSTAKA

2.1. Metode K-Nearest Neighbor

Metode K-Nearest Neighbor (KNN) dalam Machine Learning mengelompokkan data melalui pembelajaran dari contoh-contoh yang ada. KNN digunakan untuk mengklasifikasikan data dengan algoritma query instance yang mempertimbangkan jarak antar objek-objek [3].

Salah satu masalah yang muncul dalam KNN adalah pemilihan nilai K yang mempengaruhi hasil klasifikasi dan respons terhadap kebisingan data. KNN tidak membedakan bobot dan fitur-fitur data seperti metode lain, seperti Artificial Neural Networks (ANN).

Ketidakmampuan KNN membedakan tetangga terdekat dan sifat "lazy learning"-nya menyebabkan proses identifikasi tetangga dan nilai K menjadi lambat [4].

Dalam ringkasan, KNN adalah metode yang mengelompokkan data melalui pembelajaran dari contoh-contoh yang ada, tetapi memiliki kelemahan seperti pemilihan nilai K yang sensitif, ketidakmampuan membedakan bobot dan fitur data, serta keterlambatan dalam proses identifikasi tetangga dan nilai K.

2.2. Euclidean Distance

Euclidean distance adalah metode pengukuran jarak antara variabel. Keunggulan utamanya terletak pada kemudahan dan kecepatan penggunaan, menggunakan prinsip teorema Pythagoras pada dimensi 1, 2, dan 3 [5]. Ini digunakan untuk menghitung jarak antara lokasi dengan menerapkan prinsip Teorema Pythagoras dalam dimensi 1, 2, dan 3, mirip dengan pengukuran panjang diagonal dalam segitiga. Formula Euclidean Distance dapat diungkapkan sebagai berikut:

$$n = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (1)$$

- n : jarak
- x1 : koordinat latitudo x satu
- x2 : koordinat latitudo x dua
- y1 : koordinat longtitudo y satu
- y2 : Koordinat longtitudo y dua

2.3. Seleksi Fitur Information Gain

Seleksi fitur Information Gain dalam machine learning memilih fitur relevan dari dataset untuk memangkas dimensi, meningkatkan efisiensi, dan akurasi prediksi model [6]. Dengan mengukur sejauh mana fitur memisahkan kelas-kelas, Information Gain membantu identifikasi fitur penting, mencegah overfitting, dan membangun model yang lebih adaptif.

2.4. SMOTE

SMOTE (*Synthetic Minority Over-sampling Technique*) adalah metode oversampling yang menangani ketidak seimbangan kelas pada dataset dengan menciptakan sampel sintetis dalam kelas minoritas melalui interpolasi [7]. Ketidakseimbangan kelas terjadi saat frekuensi kelas minoritas jauh lebih rendah. Ini dapat menyebabkan bias dalam prediksi model. SMOTE mengatasi ini dengan menciptakan sampel sintetis dalam kelas minoritas [8].

2.5. Confusion Matrix

Proses pengukuran parameter ini mengevaluasi hasil implementasi metode klasifikasi. Dalam kerangka penelitian ini, evaluasi dilakukan melalui Confusion Matrix. Confusion Matrix adalah representasi informasi hasil aktual yang telah diprediksi sebelumnya oleh sistem pengklasifikasi menggunakan data, diorganisir dalam bentuk matriks [9]. Rumus yang digunakan sebagai berikut:

$$\text{accuration } \frac{TP+TN}{TP+FP+TN+FN} \quad (2)$$

$$\text{precission } \frac{TP}{TP+FP} \quad (3)$$

$$\text{recall } \frac{TP}{TP+FN} \quad (4)$$

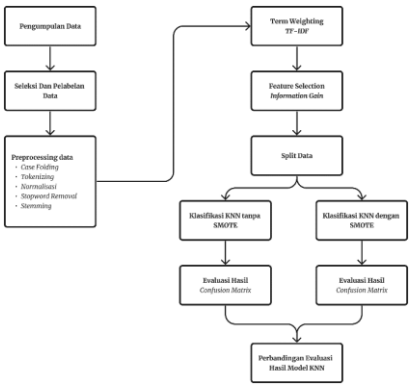
1. TP (True Positive)
2. TN (True Negative)
3. FP (False Positive)
4. FN (False Negative)

2.6. Python

Python merupakan bahasa pemrograman yang diinterpretasikan secara dinamis, berorientasi objek, dan memiliki semantik kuat [10]. Python menawarkan struktur data tingkat lanjut, tipe data dinamis, dan pengikatan dinamis. Sebagai bahasa pemrograman tingkat tinggi, Python memungkinkan penulisan kode yang dikompilasi menjadi bytecode dan dieksekusi, cocok untuk berbagai keperluan seperti scripting, pengembangan aplikasi web, dll. Python mendukung berbagai paradigma pemrograman termasuk orientasi objek, imperatif, dan fungsional. Sebagai bahasa pemrograman dinamis, Python memiliki fitur manajemen memori otomatis yang memudahkan pengelolaan sumber daya.

3. METODE PENELITIAN

Metode penelitian yang digunakan dalam penelitian ini menggunakan metode penelitian kuantitatif. Metode penelitian kuantitatif adalah pendekatan yang berdasarkan pada filsafat positivisme. Ia digunakan untuk menguji hipotesis dengan teknik pengukuran yang cermat terhadap variabel tertentu. Prosesnya melibatkan pengumpulan data dalam bentuk angka dan pengolahan statistik. Berikut rancangan penelitian yang dilakukan dalam penelitian ini.



Gambar 1. Rancangan penelitian

3.1. Pengumpulan Data

Data yang dikumpulkan diperoleh melalui proses pengambilan data dari platform Twitter, suatu proses yang dikenal sebagai "crawling data Twitter". Pengambilan data dilakukan dengan mengambil sejumlah tweet publik yang tersedia di media sosial Twitter. Proses ini diimplementasikan menggunakan bahasa pemrograman Python.

Kata kunci yang dijadikan acuan dalam pengambilan data adalah "Jokowi". Data yang diambil meliputi sebanyak 5000 entri, yang semuanya berupa teks dari tweet yang mengandung kata kunci "Jokowi".

3.2. Seleksi dan Pelabelan Data

Pada tahap seleksi dan pelabelan data, dilakukan secara manual. Data akan disaring untuk menghapus kalimat-kalimat yang bersifat spam, dan hanya akan dipertahankan satu data yang dianggap benar dan relevan. Proses pelabelan data juga dilakukan secara manual dengan tujuan mengategorikan kalimat-kalimat tersebut ke dalam kategori positif atau negatif.

3.3. Preprocessing Data

Preprocessing data merupakan langkah awal untuk membersihkan kalimat dari elemen khusus seperti simbol, emoji, dan hashtag, serta untuk melakukan normalisasi data. Sebelum menjalani tahap pengujian dalam proses klasifikasi, data harus mengalami tahap preprocessing terlebih dahulu. Setelah tahap preprocessing selesai, data siap untuk diproses menggunakan algoritma yang telah ditentukan.

3.4. Term Weighting

Term weighting adalah proses krusial dalam analisis teks yang memberikan bobot pada kata-kata dalam dokumen. Salah satu metode umum dalam term weighting adalah TF-IDF (*Term Frequency-Inverse Document Frequency*). Dalam metode ini, setiap kata dalam kalimat diberi bobot berdasarkan frekuensinya di dokumen serta sejauh mana kata tersebut unik dalam seluruh koleksi dokumen. Proses term Weighting menggunakan dua konsep utama yaitu:

1. Term Frequency (TF): Mengukur seberapa sering kata muncul dalam dokumen. Bobot lebih tinggi diberikan pada kata yang muncul lebih sering.
2. Inverse Document Frequency (IDF): Mengukur seberapa umum atau langka suatu kata di seluruh dokumen. Kata-kata jarang yang muncul mendapat bobot lebih tinggi untuk membedakan dari kata-kata umum.

Gabungan nilai TF dan IDF menghasilkan skor TF-IDF untuk setiap kata dalam dokumen. Skor ini mencerminkan pentingnya kata dalam konteks dokumen. Bobot TF-IDF membantu mengidentifikasi kata kunci yang berdampak signifikan. Dengan menerapkan TF-IDF pada data teks, kalimat-kalimat diolah menjadi representasi angka yang mencerminkan pentingnya kata dalam dokumen. Setelah term weighting, data olahan dapat digunakan untuk analisis lanjutan seperti klasifikasi atau pemahaman makna teks.

Tabel 1. Contoh hasil TF-IDF

NO	Term	TF								DF	IDF Log(n/df)
		D1	D2	D3	D4	D5	D6	D7	D8		
1	Berusaha		1							1	2.07
2	Pak	1	1	1						3	0.98
3	Lebih								1	1	2.07
4	Benci	1								1	2.07
5	Sayang		1							1	2.07

Pada Tabel 1 di atas, terdapat contoh hasil pembobotan kata menggunakan metode TF, DF, dan IDF. Sebagai contoh, kata "berusaha" pada data ke-1 tidak ditemukan dalam konteks "berusaha", sedangkan pada data ke-2 memiliki nilai 1, menunjukkan keberadaan kata "berusaha" dalam dokumen tersebut. Pada data ke-3 hingga ke-8, kata "berusaha" tidak ditemukan, sehingga nilai-nilai pada data ke-3 hingga ke-8 kosong. Hal ini dapat diamati pada perhitungan DF yang hanya bernilai 1.

3.5. Seleksi Fitur Information Gain

Seleksi fitur berdasarkan Information Gain penting dalam memilih atribut relevan dalam penelitian. Tujuannya adalah mengidentifikasi atribut yang berkontribusi signifikan terhadap pemahaman data. Dalam seleksi fitur Information Gain, penting mempertimbangkan nilai atribut setelah tahap TF-IDF. Nilai optimal sekitar 0.5 karena nilai terlalu besar/kecil berpotensi overfitting, menghasilkan model cocok data latihan tapi kurang umum pada data baru.

Proses seleksi ini seimbangkan relevansi fitur dan kompleksitas model untuk menciptakan model baik tanpa mengorbankan generalisasi.

3.6. Split Data

Split data merupakan tahap di mana data dibagi untuk digunakan dalam proses klasifikasi menggunakan metode K-Nearest Neighbors (KNN). Pembagian data ini dilakukan melalui metode random split. Dalam pembagian data, peneliti membagi data menjadi dua bagian, yaitu data latihan (training) dan data uji (testing), dengan presentase berikut: (97% data latihan dan 3% data uji), (90% data latihan dan 10% data uji), (80% data latihan dan 20% data uji), serta (70% data latihan dan 30% data uji).

Pembagian data ini bertujuan untuk menguji performa model KNN dengan variasi pembagian data latihan dan uji, sehingga dapat memahami sejauh mana model tersebut dapat menggeneralisasi data yang belum pernah dilihat sebelumnya. Dengan cara ini, pengaruh dari perbandingan data latihan dan uji terhadap hasil klasifikasi dapat dievaluasi dengan lebih baik.).

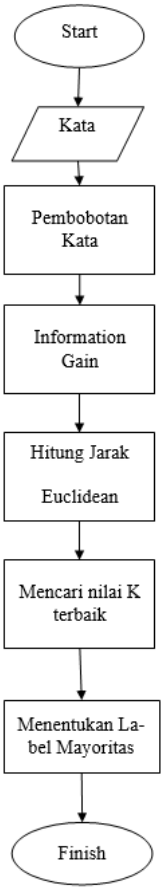
3.7. Klasifikasi K-Nearest Neighbor

Pada tahap ini, metode K-Nearest Neighbors (KNN) diimplementasikan dengan menggunakan pengukuran jarak, seperti Euclidean Distance, untuk mengidentifikasi tetangga terdekat dalam proses

klasifikasi. Berikut adalah langkah-langkah dalam proses klasifikasi KNN:

1. Pemilihan Nilai K.
2. Hitung Jarak.
3. Urutkan Data Pelatihan.
4. Menentukan Label Mayoritas.

Penentuan nilai K melibatkan pengujian beberapa nilai, seperti k = 1, 3, 5, 7, 9, dan 11, guna mencari akurasi terbaik dari setiap nilai K. Setelah mendapatkan akurasi terbaik dari pengujian nilai-nilai K tersebut, nilai K tersebut akan dipilih untuk pengujian lanjutan.



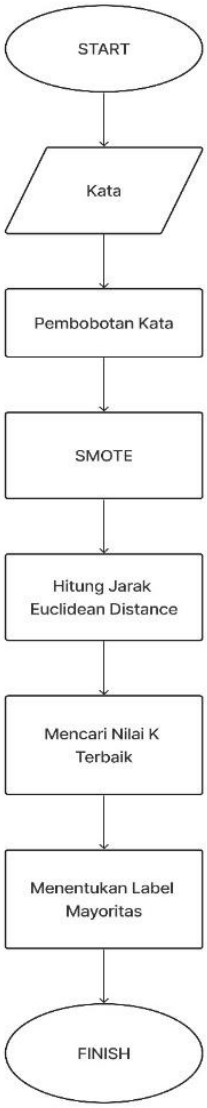
Gambar 2. Tahapan klasifikasi KNN

3.8. Klasifikasi KNN Dengan SMOTE

Dalam klasifikasi KNN menggunakan SMOTE untuk mengatasi ketidakseimbangan kelas pada data analisis sentimen, teknik Synthetic Minority Oversampling Technique (SMOTE) digunakan. SMOTE menciptakan sampel sintetis dari kelas minoritas dengan memanfaatkan tetangga terdekat dan selisih vektor. Dengan SMOTE, jumlah data kelas minoritas dapat ditingkatkan sehingga keseimbangan dengan kelas mayoritas tercapai. Berikut tahapannya:

1. Pemilihan Nilai K.
2. Sintetis Data dengan SMOTE.
3. Perhitungan dengan Euclidean Disntance.
4. Pengurutan Data Pelatihan.
5. Menentukan Label Mayoritas.

Dengan pendekatan ini, SMOTE membantu mengatasi ketidakseimbangan kelas dan meningkatkan performa model KNN pada data yang tidak seimbang secara kelas.



Gambar 3. Tahapan KNN menggunakan SMOTE

3.9. Evaluasi Hasil

Evaluasi hasil yaitu, berupa uji akurasi yang akan dilakukan dengan *confussion matrix* untuk menguji keluaran dari algoritma K-Nearest Neighbor, dengan menggunakan pengukuran jarak Euclidean Distance dan menggunakan teknik Oversampling SMOTE, untuk mengatasi ketidakseimbangan kelas pada data analisis sentimen, dan bertujuan untuk meningkatkan hasil akurasi dari algoritma K-Nearest Neighbor (KNN) pada kasus penelitian ini. Untuk hasil terbaik dilakukan beberapa kali pengujian dengan perbedaan nilai K dan perbedaan jumlah data testing dan data training. Penggunaan confussion matrix selain untuk menghitung accuration juga dipakai untuk menghitung recall, precision.

4. HASIL DAN PEMBAHASAN

4.1. Data

Data untuk penelitian ini diperoleh melalui proses pengambilan data (crawling) dari platform media sosial Twitter. Data yang digunakan dalam penelitian ini berkaitan dengan kata kunci "Jokowi". Hanya teks dari tweet yang diambil, tanpa emoji atau karakter khusus. Data yang diambil termasuk tweet asli (original tweet) dan retweet dari bulan Maret.

Teknik crawling menggunakan API Twitter untuk mengumpulkan data, yang kemudian disimpan dalam format Microsoft Excel. Total data yang terkumpul berjumlah 5000 tweet. Berikut contoh data yang digunakan:

Tabel 2. Contoh Data

No	Text
1.	Keren sekali di era pemerintah Jokowi #JokowiPresidenku #PembangunanUntukRakyat #PresidenJokowiHebat https://t.co/dk931Tj0Z2
2.	RT @bianca125547620: Jokowi, Prabowo, Ganjar berdiri di tengah sawah panen raya, itu pencitraan. Keberpihakan palsu. Pemerintah sudah jadâ€¦
3.	Rakyat Indonesia semakin terbantu dengan kebijakan Presiden Jokowi, kami di Kalimantan Tengah ju ##JokowiHebat Polrestabes Medan Luar Biasa
4.	@jokowi Tanya ke presiden kenapa pupuk langka, oow.... Iya pak, kalau bapak turun kesawah , itu tanda" beras udah mau import ya pak?????
5.	RT @jokowi: Para petani mengeluhkan harga gabah kering panen masih rendah. Lalu, berapa seharusnya? Itulah yang sedang dihitung pemerintahâ€¦

Pada tabel 2, data tersebut akan digunakan untuk proses preprocessing data.

4.2. Seleksi Fitur Information Gain

Selanjutnya, peneliti menerapkan seleksi fitur Information Gain untuk mengidentifikasi 500 fitur paling penting, yang akan mempercepat pembelajaran mesin dengan mengurangi dimensi data. Perhitungan Information Gain dilakukan menggunakan library Scikit-Learn. Mengingat label kelas kami menggunakan nilai 0 dan 1, langkah perhitungan melibatkan tahap TF-IDF terlebih dahulu. Berikut hasil dari seleksi fitur information gain:

Tabel 3 Hasil Information Gain

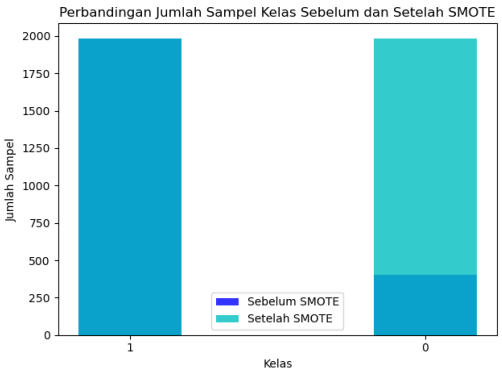
Feature	IG Score
aaamiiin	0.00007771
aabbuoott	0.00007771
Aamiin	0.00097587
Abad	0.00007771

Pada tabel 3, ditampilkan hasil dari seleksi fitur Information Gain, pada hasil tersebut terlihat bahwa, proses seleksi fitur Information Gain, menseleksi kata yang mempunyai nilai dari pembobotan kata yang dianggapnya penting, dan digunakan dalam proses beriktunya dalam proses klasifikasi KNN

4.3. Penerapan SMOTE

SMOTE digunakan untuk menyeimbangkan data tidak seimbang antara kelas label positif dan negatif. Jumlah data yang akan dihasilkan adalah sebanyak 2390, dengan 1985 data positif (label 1) dan 405 data negatif (label 0). Algoritma SMOTE bekerja dengan cara menciptakan sampel sintetis untuk tetangga terdekat yang dipilih, dengan menghitung selisih vektor antara sampel referensi dan tetangga tersebut.

Berikut perbandingan hasil penerapan SMOTE sebelum dan sesudah pada data yang awalnya mengalami ketidakseimbangan kelas positif dan negatif.:



Gambar 5. Hasil Penerapan SMOTE

Pada Gambar 2, terlihat pengujian jumlah sampel kelas sebelum SMOTE, dengan 1985 sampel pada kelas 1 dan 405 sampel pada kelas 0. Setelah pengujian, data menghasilkan sampel sintetis baru untuk menyeimbangkan kelas minoritas 0 hingga mencapai 1985, jumlah yang setara dengan kelas mayoritas. Dengan teknik interpolasi antara sampel referensi dan tetangga terdekat, SMOTE membantu meningkatkan representasi kelas minoritas dalam dataset.

4.4. Klasifikasi K-Nearest Neighbor

Hasil dari pengolahan TF-IDF dan seleksi fitur Information Gain digunakan untuk klasifikasi menggunakan metode K-Nearest Neighbor (KNN). Dalam klasifikasi KNN, data dibagi menjadi data training dan data testing melalui pembagian acak (Random Split). Empat eksperimen dilakukan dengan proporsi yang berbeda: (97% training, 3% testing), (90% training, 10% testing), (80% training, 20% testing), dan (70% training, 30% testing). Perhitungan jarak Euclidean Distance dilakukan pada hasil pembagian data training dan testing. Berikut contoh perhitungan Euclidean Distance menggunakan formula dari persamaan 2.1:

$$n = \sqrt{((0.048640727 - .2200115182221739)^2)} = \sqrt{(0.03218270421969265)} \approx 0.1793568924133371$$

Pada perhitungan diatas menjelaskan tentang, perhitungan jarak euclidean distance, untuk mengukur jarak antara data training dan data testing. Pada tahap

selanjutnya, akan dilakukan proses mencari nilai K terbaik, dengan nilai K yang di uji yaitu, nilai k=1 sampai k=11. Berikut hasil perolehan mencari nilai K terbaik:

Tabel 4. Hasil nilai k terbaik

Nilai k	Akurasi
K=1	0.77
K=3	0.81
K=5	0.81
K=7	0.80
K=9	0.79
K=11	0.77

Dari Tabel 4, diketahui bahwa nilai K terbaik adalah K=3 dengan akurasi sebesar 0.81. Oleh karena itu, nilai K ini akan dijadikan referensi untuk proses penentuan label pada data uji dengan menggunakan metode jarak tetangga terdekat (K=3). Berikut adalah hasil prediksi label pada data uji dengan K=3:

Tabel 5. Hasil Prediksi Label uji nilai K=3

Data uji dan data latih	Jarak	Label Latih	Label Uji
Jarak data uji ke-1 dan latih ke-1386	0.1908758844307998	1	1
Jarak data uji ke-1 dan latih ke-771	0.2825566580839839	0	1
Jarak data uji ke-1 dan latih ke-1223	0.3001072553887473	1	1

Pada Tabel 5, terdapat hasil pengurutan jarak antara data uji dan data latih, serta prediksi label tetangga terdekat dengan nilai k=3. Mayoritas prediksi label data uji adalah positif (nilai 1) dalam urutan tetangga terdekat. Hal ini menunjukkan bahwa berdasarkan kriteria jarak dan prediksi label tetangga terdekat, data uji cenderung diklasifikasikan sebagai positif. Sehingga, dapat disimpulkan bahwa prediksi label data uji menggunakan k=3 adalah positif.

4.5. Klasifikasi KNN dengan SMOTE

Langkah selanjutnya adalah menguji klasifikasi K-Nearest Neighbors (KNN) dengan menerapkan metode SMOTE. Proses pengujian mirip dengan KNN tanpa SMOTE, tetapi pada pengujian KNN dengan SMOTE, data sintetis diciptakan untuk menyeimbangkan ketidakseimbangan kelas. Setelah proses penciptaan data sintetis menggunakan SMOTE, jumlah data pada kedua kelas menjadi seimbang dan data sintetis telah dibuat untuk memperkuat representasi kelas minoritas. Berikut hasil dari sintetis data menggunakan SMOTE:

Tabel 6. Label sebelum dan sesudah SMOTE

Kelas Label	Sebelum	Sesudah	Data Sintetis
Label 1	1397	1397	525
Label 0	276	1397	192

Tabel 6 menunjukkan ketidakseimbangan signifikan antara jumlah sampel pada kelas 1 dan kelas 0 sebelum dilakukan SMOTE. Setelah penerapan SMOTE, dihasilkan 525 data sintetis baru untuk kelas 1 dan 192 data sintetis baru untuk kelas 0, sehingga terjadi penyeimbangan kelas. Pada tahap penentuan nilai K, nilai yang diuji adalah k=1,3,5,7,9,11. Penentuan nilai K terbaik didasarkan pada akurasi tertinggi. Berikut hasil pengujian untuk masing-masing nilai K:

Tabel 7. Hasil nilai k terbaik

Nilai k	Akurasi
K=1	0.9
K=3	0.85
K=5	0.85
K=7	0.85
K=9	0.83
K=11	0.81

Berdasarkan pada tabel 7**,** pengujian akan dilakukan dengan acuan akurasi tertinggi dari nilai k tersebut, nilai k tertinggi pada tabel tersebut yaitu pada nilai k=1 dengan akurasi 0.9. pada tahap selanjutnya akan ditetntukan label data uji dengan tetangga terdekat k=1. Berikut hasil dari prediksi label data uji dari nnilai k=1:

Tabel 8. Hasil prediksi label uji nilai K=1

Data uji dan data latih	Jarak	Label Latih	Label Uji
Jarak data uji ke-1 dan latih ke-1041	0.30517342768571715	1	1

Pada Tabel 8, hasil pengurutan jarak antara data uji dan data latih, serta prediksi label dari tetangga terdekat dengan nilai k=1, menunjukkan mayoritas prediksi label data uji adalah negatif (nilai 0). Hal ini mengindikasikan bahwa berdasarkan kriteria jarak dan prediksi label dari tetangga terdekat, data uji lebih cenderung diklasifikasikan sebagai negatif. Dengan demikian, dapat disimpulkan bahwa prediksi label data uji menggunakan k=1 adalah negatif.

4.6. Evaluasi Hasil

Setelah proses pengklasifikasian menggunakan metode KNN, langkah berikutnya adalah pengujian untuk mengevaluasi performa model. Pengujian ini melibatkan Confusion Matrix dan perbandingan antara metode KNN dengan SMOTE dan KNN tanpa SMOTE. Dari hasil pengujian ini, akan dihitung nilai TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative), serta presisi, akurasi, dan recall untuk analisis lebih lanjut. Berikut hasil dari performa model KNN tanpa SMOTE;

Tabel 9. Hasil performa model KNN tanpa SMOTE

Nilai K	Data Training	Data Testing	Akurasi	Presisi	Recall
3	97%	3%	0.82	0.82	0.98
3	90%	10%	0.79	0.82	0.94
3	80%	20%	0.81	0.83	0.96
3	70%	30%	0.81	0.86	0.92

Tabel 10. Hasil performa model KNN menggunakan SMOTE

Nilai K	Data Training	Data Testing	Akurasi	Presisi	Recall
1	97%	3%	0.93	1.00	0.85
1	90%	10%	0.85	0.93	0.79
1	80%	20%	0.84	0.90	0.78
1	70%	30%	0.85	0.94	0.76

Pada Tabel 9, hasil evaluasi metode KNN tanpa SMOTE menunjukkan akurasi berkisar antara 0.79 hingga 0.82, mencerminkan kemampuan model dalam mengklasifikasikan data secara keseluruhan. Dari evaluasi ini, terlihat bahwa model KNN memiliki akurasi lebih tinggi ketika menggunakan presentase data training yang lebih tinggi, meskipun perbedaannya tidak signifikan.

Sementara itu, tabel 10, menampilkan hasil performa model KNN dengan algoritma SMOTE. Penggunaan SMOTE dalam KNN menghasilkan performa yang baik dengan akurasi antara 0.84 hingga 0.93. Model KNN dengan SMOTE memiliki kecenderungan akurasi lebih tinggi saat presentase data training meningkat, walaupun recall sedikit menurun saat presentase data training menurun.

5. KESIMPULAN DAN SARAN

Kesimpulan dari penelitian ini adalah bahwa penerapan teknik SMOTE pada metode K-Nearest Neighbor (KNN) dalam klasifikasi analisis sentimen berhasil meningkatkan kinerja model dengan mengatasi ketidakseimbangan kelas dalam dataset. Evaluasi model KNN dengan SMOTE menghasilkan akurasi 90%, presisi 100%, dan recall 81%, sedangkan KNN tanpa SMOTE menghasilkan akurasi 82%, presisi 82%, dan recall 98%. Oleh karena itu, penggunaan SMOTE pada KNN dapat meningkatkan performa dalam analisis sentimen. Sebagai saran untuk penelitian selanjutnya, eksplorasi kombinasi dengan metode lain seperti Undersampling dan ADASYN juga bisa dipertimbangkan untuk mengatasi ketidakseimbangan data dalam klasifikasi analisis sentimen menggunakan metode KNN.

DAFTAR PUSTAKA

[1] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining Text Data*, Springer US, 2012, pp. 415–463. doi: 10.1007/978-1-4614-3223-4_13.

[2] S. Maula Chamzah, M. Lestandy, N. Kasan, A. Nugraha, J. Raya Tlogomas No, and J. Timur, "Penerapan Synthetic Minority Oversampling Technique (SMOTE) untuk Imbalance Class pada Data Text Menggunakan KNN," 2022.

[3] Femi Dwi Astuti and Febri Nova Lenti, "Impelementasi SMOTE untuk mengatasi Imbalance Class pada klasifikasi Car Evolution menggunakan K-NN," 2021.

[4] A. Nikmatul Kasanah, U. Pujianto, T. Elektro, F. Teknik, and U. Negeri Malang, "Terakreditasi SINTA Peringkat 2 Penerapan Teknik SMOTE untuk Mengatasi Imbalance Class dalam Klasifikasi Objektivitas Berita Online Menggunakan Algoritma KNN," *masa berlaku mulai*, vol. 1, no. 3, pp. 196–201, 2017.

[5] Yumarlin MZ, Rizqi Mirza Fadilla, and Indra Pratama, "IMPLEMENTASI METODE K-NEARST NEIGHBOR BERBASISEUCLIDEAN DISTANCEUNTUK KLASIFIKASIPENERIMAAN VAKSIN COVID-19," 2021.

[6] Indra Kurniawan, Dwi Retna Sulistyowati, and Yasa Maulana, "Penerapan Seleksi Fitur Median Weighted Information Gain dengan K-NN pada Dataset Label Hours Software Effort Estimation," *Jurnal CoSciTech (Computer Science and Information Technology)*, vol. 3, no. 2, pp. 52–57, Aug. 2022, doi: 10.37859/coscitech.v3i2.3876.

[7] N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," 2002.

[8] Ika Nur Fajri and Femi Dwi Astuti, "PENGARUH SMOTE DAN FORWARD SELECTION DALAM MENANGANI KETIDAKSEIMBANGAN KELAS PADA ALGORITMA KLASIFIKASI," 2022.

[9] Sofia Visa, Atsushi Inoue, and Anca Ralescu, "Proceedings of the Twentysecond Midwest Artificial Intelligence and Cognitive Science Conference," 2011.

[10] L. V. Tulchak, "History of Python."