Optimasi Algoritma K-Nearest Neighbor Menggunakan GridSearchCV Untuk Meningkatkan Akurasi Pada Klasifikasi Curah Hujan Berdasarkan Data Meteorologi

[1st]Fadhilah Wanda Almira
Teknik Informatika
Universitas Telkom Purwokerto
Purwokerto, Indonesia
fadhilahwandaalmira@student.telkomuniversity.ac.id

[2nd]Amalia Beladinna Arifa
Teknik Informatika
Universitas Telkom Purwokerto
Purwokerto, Indonesia
amaliabela@telkomuniversity.ac.id

[3nd]Maie Istighosah
Teknik Informatika
Universitas Telkom Purwokerto
Purwokerto, Indonesia
maieistigh@telkomuniversity.ac.id

Abstrak — *K-Nearest Neighbor* (K-NN) merupakan salah satu algoritma machine learning yang sering digunakan dalam proses klasifikasi karena memiliki cara kerja yang sederhana dan mudah diterapkan. Namun, performa algoritma K-NN dipengaruhi oleh pemilihan parameter, terutama nilai k, sehingga hasil klasifikasi belum optimal. K-NN juga memiliki kelemahan dalam menangani data tidak seimbang karena kelas mayoritas cenderung mendominasi hasil prediksi. Penelitian ini bertujuan untuk meningkatkan performa algoritma K-NN menggunakan metode optimasi *GridSearchCV*. Dataset yang digunakan berupa data meteorologi yang diperoleh dari *Mendeley* Data. Tahapan penelitian meliputi preprocessing data, balancing data, pembagian data menjadi data training dan testing, penerapan model baseline menggunakan k=5, serta optimasi parameter menggunakan *GridSearchCV* dengan rentang k=1–50. Proses balancing data dilakukan agar distribusi kelas menjadi lebih seimbang sehingga model dapat mengenali kelas minoritas dengan lebih baik. Hasil penelitian menunjukkan bahwa optimasi parameter mampu meningkatkan akurasi, yaitu dari akurasi 0.7704 pada model baseline menjadi 0.8487 setelah optimasi. Parameter terbaik yang diperoleh adalah k=1 dengan performa akurasi tertinggi pada seluruh pengujian.

Kata kunci — klasifikasi curah hujan, *k-nearest neighbor*, optimasi parameter, gridsearchcv, data meteorologi

I. PENDAHULUAN

Algoritma *K-Nearest Neighbor* (K-NN) merupakan salah satu metode *machine learning* yang sering digunakan untuk proses klasifikasi karena cara kerjanya sederhana dan mudah diterapkan [1]. K-NN bekerja dengan menentukan kelas data berdasarkan kedekatan dengan data tetangga terdekat. Metode ini cukup baik digunakan untuk mengenali pola data yang kompleks dan sering diterapkan dalam berbagai penelitian klasifikasi [2].

Namun, kinerja K-NN dipengeruhi oleh pemilihan parameter, terutama nilai k atau jumlah tetangga terdekat [3]. Jika nilai k terlalu kecil, model menjadi sensitif terhadap noise sehingga hasil klasifikasi kurang optimal, sedangkan jika nilai k terlalu besar, batas antar kelas menjadi kurang jelas dan akurasi model dapat menurun. K-NN juga memiliki kelemahan dalam proses klasifikasi yang menjadi lebih lambat ketika jumlah data semakin banyak [4]. Selain itu, algoritma ini kurang optimal dalam menangani data tidak seimbang karena kelas mayoritas cenderung mendominasi hasil klasifikasi [5]. Oleh karena itu, diperlukan optimasi parameter agar kinerja dan akurasi algoritma K-NN dapat meningkat.

Beberapa penelitian sebelumnya telah mencoba meningkatkan performa K-NN dalam klasifikasi curah hujan. Salah satu penelitian membandingkan performa algoritma C4.5 dan K-NN dalam mengelompokkan data curah hujan berdasarkan data iklim di Indonesia [3]. Hasilnya menunjukkan bahwa penerapan teknik hyperparameter tuning menggunakan *RandomizedSearchCV* berhasil meningkatkan akurasi K-NN dari 78,10% menjadi 83,37%. Penelitian lain menggunakan algoritma K-NN untuk mengklasifikasikan cuaca menjadi tiga jenis: cerah, berawan, dan hujan [4]. Penelitian sebelumnya menggunakan data dari situs Kaggle yang berisi 96.453 data cuaca, lalu membaginya menjadi 80% untuk pelatihan dan 20% untuk pengujian. Evaluasi dilakukan dengan berbagai nilai k (3, 5, 7, dan 9), dan ditemukan bahwa k = 9 memberikan akurasi tertinggi sebesar 68,07%. Untuk kelas "berawan," *precision* mencapai 72,10%, *recall* 89,29%, dan *f1-score* 79,78% . Hal ini menunjukkan bahwa optimasi parameter memiliki peran penting dalam meningkatkan kinerja algoritma.

Berdasarkan permasalahan tersebut, penelitian ini menggunakan *GridSearchCV* untuk mengoptimasi parameter pada algoritma K-NN dalam klasifikasi curah hujan. *GridSearchCV* dipilih karena mampu menguji seluruh kombinasi parameter yang telah ditentukan sehingga peluang memperoleh parameter terbaik menjadi lebih besar [7]. Berbeda dengan *RandomSearchCV* yang memilih kombinasi parameter secara acak, *GridSearchCV* melakukan pencarian parameter secara menyeluruh sehingga parameter terbaik pada algoritma K-NN dapat diperoleh hasil lebih optimal.

Penelitian ini menggunakan dataset meteorologi yang diperoleh dari platform *Mendeley* Data untuk proses klasifikasi curah hujan. Melalui optimasi parameter menggunakan *GridSearchCV*, penelitian ini bertujuan untuk meningkatkan akurasi klasifikasi sehingga model yang dihasilkan memiliki kinerja yang lebih baik dalam mengklasifikasikan data curah hujan.

II. KAJIAN TEORI

A. *Algoritma K-Nearest Neighbor*

K-Nearest Neighbor (KNN) merupakan salah satu algoritma dalam bidang *Machine Learning* yang sering digunakan untuk proses klasifikasi data. Algoritma ini bekerja berdasarkan prinsip kedekatan antar data, di mana suatu data baru akan diklasifikasikan berdasarkan kemiripannya dengan data yang telah diketahui kelasnya sebelumnya [16].

Cara kerja algoritma *K-Nearest Neighbor* (KNN) dimulai dengan menghitung jarak antara data baru dengan seluruh data latih (*training data*). Setelah seluruh jarak diperoleh, data akan diurutkan dari jarak yang paling dekat hingga yang paling jauh. Selanjutnya, algoritma mengambil sejumlah data terdekat sesuai dengan nilai K yang digunakan. Nilai K menunjukkan jumlah tetangga terdekat yang dijadikan acuan dalam proses klasifikasi. Kelas yang paling banyak muncul dari tetangga terdekat tersebut akan menjadi hasil prediksi data baru [17].

Selain nilai K, performa algoritma KNN juga dipengaruhi oleh fitur atau variabel yang digunakan. Fitur yang tidak relevan atau memiliki perbedaan skala yang besar dapat memengaruhi hasil perhitungan jarak sehingga menyebabkan performa model menurun. Oleh sebab itu, diperlukan proses normalisasi maupun seleksi fitur agar setiap variabel memiliki kontribusi yang lebih seimbang dalam proses klasifikasi [18].

Dalam proses pengukuran kedekatan antar data, algoritma KNN menggunakan metode *Euclidean Distance* dan *Manhattan Distance*. Kedua metode tersebut digunakan untuk menghitung jarak antara dua titik pada ruang berdimensi n sehingga dapat menentukan tingkat kemiripanantar data [19]. Semakin kecil nilai jarak yang dihasilkan, maka semakin tinggi tingkat kemiripan antar data tersebut.

Secara umum, perhitungan jarak menggunakan metode *Euclidean Distance* dapat dituliskan sebagai berikut:

$$\text{Dist (x,y)} = \sum_{i=1}^{n} \sqrt{(xi - yi)^2} \qquad (1)$$

Keterangan :
d = Jarak antar data
x_i = Nilai data *training* pada variabel ke - i
y_1 = Nilai data *testing* pada variabel ke - i
i = Variabel Data
n = Jumlah variabel

Selain menggunakan *Euclidean Distance*, algoritma KNN juga dapat menggunakan metode *Manhattan Distance*. Metode ini menghitung jarak dengan menjumlahkan selisih nilai dari setiap data. Rumus *Manhattan Distance* dapat dituliskan sebagai berikut:

$$\text{Dist (x,y)} = \sum_{i=1}^{n} |xi - yi| \qquad (2)$$

Keterangan :
d = Jarak antar data
x_i = Nilai data training pada variabel ke - i
y_1 = Nilai data testing pada variabel ke - i
i = Variabel Data
n = Jumlah variable

Berdasarkan rumus tersebut, algoritma KNN menghitung jarak antara data uji dengan seluruh data latih pada setiap variabel yang digunakan. Pada metode *Euclidean Distance*, proses perhitungan dilakukan dengan mencari selisih antara nilai variabel pada data uji dan data latih, kemudian hasil selisih tersebut dipangkatkan dua agar tidak menghasilkan nilai negatif. Setelah itu, seluruh hasil perhitungan dijumlahkan dan diakar untuk memperoleh nilai jarak akhir. Sementara itu, pada metode *Manhattan Distance*, perhitungan dilakukan dengan menjumlahkan nilai dari selisih setiap variabel tanpa proses pemangkatan maupun akar [5].

B. *GridSearchCV*

GridSearchCV merupakan salah satu teknik dalam proses *hyperparameter tuning* yang digunakan untuk menentukan kombinasi parameter terbaik dalam suatu model pada bidang Machine Learning. Metode ini bekerja dengan cara membangun dan mengevaluasi model untuk setiap kombinasi parameter yang telah ditentukan dalam suatu *parameter grid*, sehingga memungkinkan diperolehnya konfigurasi parameter yang optimal [6] dan [7].

Dalam penerapannya, *GridSearchCV* melakukan pengujian terhadap seluruh kombinasi parameter melalui proses *cross-validation* [8]. Setiap kombinasi tersebut kemudian dievaluasi menggunakan metrik performa seperti akurasi, *precision*, *recall*, dan *F1-score* yang diperoleh dari *confusion matrix*. Dengan adanya pembagian data ke dalam beberapa *fold*, proses evaluasi menjadi lebih representatif sehingga model memiliki kemampuan generalisasi yang lebih baik [7].

Penggunaan *GridSearchCV* dinilai efektif karena mampu mengeksplorasi ruang parameter secara sistematis [7]. Beberapa penelitian menunjukkan bahwa pencarian parameter secara menyeluruh dapat meningkatkan performa model secara signifikan, khususnya pada algoritma klasifikasi seperti KNN [6]. Selain itu, metode ini juga banyak digunakan dalam praktik karena konsepnya yang sederhana dan mudah diimplementasikan [8].

C. *Balance Data*

Balance data adalah proses menyeimbangkan jumlah data pada setiap kelas dalam dataset. Ketidakseimbangan data terjadi ketika jumlah data pada satu kelas jauh lebih banyak dibandingkan kelas lainnya. Kondisi ini dapat memengaruhi

hasil klasifikasi karena model cenderung lebih sering mengenali kelas yang jumlah datanya lebih banyak. Akibatnya, kemampuan model dalam mengenali kelas yang jumlah datanya sedikit menjadi kurang optimal.

Untuk mengatasi masalah tersebut, dilakukan proses balancing data agar distribusi data antar kelas menjadi lebih seimbang. Pada penelitian ini, teknik yang digunakan adalah SMOTE (*Synthetic Minority Over-sampling Technique*) dan *Tomek Links*.

1. *Synthetic Minority Over-sampling Technique* (SMOTE)

SMOTE merupakan metode yang digunakan untuk menambah jumlah data pada kelas minoritas dengan cara membuat data baru secara sintetis [9].
Rumus yang digunakan dalam SMOTE adalah sebagai berikut:

$$x_{baru} = \lambda(x_{nn-x_1}) \quad (3)$$

Keterangan :
x_{baru} : data sintetis yang dihasilkan.
x_i : data minoritas yang dipilih.
x_{nn}: tetangga terdekat dari data yang dipilih.
λ : nilai acak antara 0 dan 1.

Cara kerja SMOTE dimulai dengan memilih data dari kelas minoritas. Selanjutnya, sistem mencari beberapa tetangga terdekat menggunakan metode *K-Nearest Neighbor*.Salah satu tetangga dipilih secara acak, kemudian data baru dibuat berdasarkan jarak antara kedua data tersebut menggunakan rumus SMOTE. Proses ini dilakukan berulang hingga jumlah data pada kelas minoritas mencapai jumlah yang diinginkan.

2. *Tomek Links*

Tomek Links merupakan metode yang digunakan untuk membersihkan data setelah proses *balancing*. Metode ini bekerja dengan mencari pasangan data dari kelas yang berbeda tetapi memiliki jarak yang sangat dekat. Pasangan data seperti ini biasanya berada pada batas antar kelas dan dapat menyebabkan kesalahan dalam proses klasifikasi [10].

Secara matematis, dua data x_i dan x_j disebut *Tomek Link* jika keduanya merupakan tetangga terdekat satu sama lain dan berasal dari kelas yang berbeda.

$$\mathrm{d}(x_i, x_j) < (x_i, x_k) \text{ Dan } \mathrm{d}(x_i, x_j) < (x_j, x_k) \quad (4)$$

Keterangan :
$\mathrm{d}(x_i, x_j)$: jarak antara data x_i dan x_j
x_k : data lain dalam dataset
x_i dan x_j : berasal dari kelas yang berbeda

Cara kerja *Tomek Links* dimulai dengan mencari pasangan data dari kelas yang berbeda yang saling bertumpang tindih antar kelas. Pasangan data tersebut menunjukkan adanya data yang letaknya sangat berdekatan meskipun berasal dari kelas yang berbeda. Selanjutnya, data dari kelas *mayoritas* yang berada pada pasangan tersebut dihapus sehingga pemisahan antar kelas menjadi lebih baik dan model dapat mengenali pola setiap kelas dengan lebih akurat.

D. Data Meteorologi

Data meteorologi merupakan kumpulan informasi yang berkaitan dengan kondisi atmosfer, seperti suhu, kelembapan, tekanan udara, kecepatan angin, dan curah hujan [11]. Data ini memiliki peran penting dalam berbagai bidang, antara lain pertanian, penerbangan, serta mitigasi bencana alam. Pengumpulan data meteorologi dilakukan melalui beberapa metode, seperti penggunaan stasiun otomatis, satelit, dan radar. Dengan tersedianya data yang akurat dan terkini, proses analisis dan prediksi curah hujan dapat dilakukan dengan lebih baik sehingga dapat mendukung pengambilan keputusan yang berkaitan dengan kondisi curah hujan [12].

Di Indonesia, pengumpulan dan pengelolaan data meteorologi dilakukan oleh Badan Meteorologi, Klimatologi, dan Geofisika (BMKG). Lembaga ini bertugas melakukan observasi, pengolahan, serta penyebaran informasi meteorologi kepada masyarakat dan instansi terkait. Data yang dihasilkan dimanfaatkan untuk berbagai kebutuhan, seperti peringatan dini bencana, perencanaan kegiatan pertanian, hingga mendukung keselamatan dalam sektor penerbangan [13].

E. Evaluasi Kinerja Model Klasifikasi

Evaluasi kinerja model klasifikasi merupakan tahapan penting untuk mengukur kemampuan model dalam menghasilkan prediksi yang akurat dan dapat diandalkan. Proses evaluasi dilakukan menggunakan metode *confusion matrix*, sedangkan pengukuran performa model dilakukan menggunakan *accuracy, precision, recall,* dan *F1-score*. Metrik-metrik tersebut dihitung berdasarkan hasil klasifikasi terhadap data uji yang telah diketahui label aslinya. Semakin tinggi nilai dari setiap metrik, maka semakin baik performa model dalam melakukan klasifikasi [14]. Berikut merupakan perhitungan pada confusion matrix :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$Recall = \frac{TP}{TP+FN} \quad (7)$$

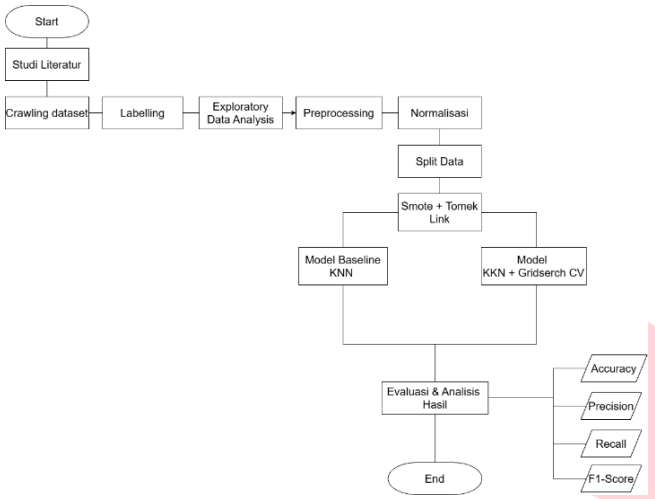
$$F\text{-}1 = 2.\frac{Precision\ x\ Recall}{Precision + Recall} \quad (8)$$

III. METODE

Metodologi penelitian ini menjelaskan tahapan yang dilakukan dalam pengembangan model klasifikasi cuaca menggunakan algoritma K-Nearest Neighbor (K-NN) dengan optimasi parameter menggunakan GridSearchCV serta penanganan ketidakseimbangan data menggunakan SMOTE-Tomek Links. Proses penelitian mencakup tahapan pengumpulan data, prapemrosesan data, penyeimbangan data, pembagian data, optimasi parameter, pelatihan model, dan evaluasi performa.

A. Alat Penelitian

Alur penelitian ini menggambarkan tahapan yang dilakukan untuk membangun model klasifikasi cuaca menggunakan algoritma *K-Nearest Neighbor* (K-NN) yang dioptimasi dengan *GridSearchCV*.Alur penelitian yang digunakan sebagai berikut.



GAMBAR 1
(Alur Penlitian)

B. Studi Literatur

Studi literatur merupakan tahap awal penelitian yang dilakukan dengan mengumpulkan dan mempelajari berbagai referensi ilmiah, seperti jurnal, prosiding, buku, dan artikel penelitian yang berkaitan dengan klasifikasi cuaca, data meteorologi, algoritma *K-Nearest Neighbor* (K-NN), metode penyeimbangan data, serta optimasi parameter menggunakan GridSearchCV. Tahap ini bertujuan untuk memahami konsep dan metode yang digunakan, mengetahui perkembangan penelitian sebelumnya, serta mengidentifikasi pendekatan yang sesuai untuk diterapkan dalam penelitian.

C. Data Collection

Pada penelitian ini digunakan data meteorologi yang diperoleh dari platform Mendeley Data, yang bersumber dari publikasi ilmiah berjudul *A Series of Small-Scale Atmospheric Datasets Observed in South of Java, Pangandaran Bay, Indonesia*. Dataset tersebut merupakan hasil pengamatan langsung menggunakan alat *Automatic Weather Station* (AWS) yang ditempatkan di wilayah Pangandaran, Jawa Barat. Data dikumpulkan dalam rentang waktu Maret 2022 hingga April 2023 dengan jumlah data yang digunakan sebanyak 8.085.

D. Labeling

Pada tahap labeling, data diberi kelas berdasarkan intensitas curah hujan yang mengacu pada klasifikasi berbasis pengukuran radar cuaca (*reflectivity)* yang digunakan oleh Badan Meteorologi, Klimatologi, dan Geofisika [23].

Berikut pengelompokan kelas intensitas curah hujan yang digunakan :

TABEL 1
(Labeling)

Kategori Hujan	Rentang Curah Hujan (mm)
Sangat Ringan	< 1
Ringan	1 – 5
Sedang	5 – 10
Lebat	10 -20
Sangat Lebat	>20

E. Exploratory Data Analysis

Exploratory Data Analysis (EDA) pada penelitian ini dilakukan untuk memahami karakteristik data meteorologi yang digunakan. Analisis dilakukan dengan melihat distribusi masing-masing variabel untuk mengetahui pola sebaran data, nilai dominan, serta kemungkinan adanya ketidakseimbangan data. Proses ini membantu memberikan gambaran awal terkait kondisi dataset sebelum digunakan dalam tahap pemodelan.

F. Data Preprocessing

Pada tahap *preprocessing* data, dilakukan pemeriksaan terhadap dataset untuk mengidentifikasi adanya *missing value* pada setiap variabel. Keberadaan data yang kosong dapat mempengaruhi hasil analisis dan kinerja model, sehingga perlu dilakukan penanganan yang tepat. Dalam penelitian ini, data yang memiliki nilai kosong dihapus agar tidak mengganggu proses pemodelan. Dengan demikian, data yang digunakan telah bersih dan siap untuk digunakan.

G. Normalisasi

Tahap normalisasi dilakukan untuk menyamakan skala nilai antar variabel dalam dataset agar perhitungan jarak pada algoritma *K-Nearest Neighbor* (K-NN) tidak terpengaruh oleh perbedaan skala. Proses ini penting karena K-NN menghitung kedekatan antar data berdasarkan jarak, sehingga variabel dengan skala lebih besar dapat mendominasi hasil perhitungan. Normalisasi dilakukan dengan mengubah nilai setiap variabel ke rentang yang sama, sehingga data menjadi lebih seragam dan siap digunakan dalam proses pemodelan.

H. Split Data

Dalam penelitian ini, pembagian data dilakukan menjadi dua bagian, yaitu data latih (*training* data) dan data uji (*testing* data). Data latih digunakan untuk melatih model agar dapat mempelajari pola dari data, sedangkan data uji digunakan untuk mengevaluasi kinerja model setelah proses pelatihan selesai.

Pembagian data dilakukan menggunakan beberapa split, yaitu 80:20, 70:30, dan 60:40. Penggunaan beberapa variasi pembagian data bertujuan untuk membandingkan kinerja model pada setiap pembagian data yang diterapkan.

I. Permodelan

Pada tahap ini dilakukan pembangunan model untuk melakukan klasifikasi data. Model yang digunakan adalah algoritma *K-Nearest Neighbor* (KNN) dengan dua pendekatan, yaitu model awal (*baseline*) dan model yang dioptimasi.

1. *K-Nearest Neighbor (K-NN) Baseline*

Pada tahap awal pemodelan, digunakan algoritma *K-Nearest Neighbor* (K-NN) sebagai model dasar (*baseline*) dengan nilai k = 5. Pemilihan nilai k ini dilakukan sebagai langkah awal untuk melihat bagaimana performa model tanpa adanya proses optimasi parameter.

2. *K-Nearest Neighbor (K-NN) dengan GridSerchCV*

Setelah diperoleh hasil dari model baseline, tahap selanjutnya adalah melakukan pengembangan model menggunakan GridSearchCV untuk menentukan parameter terbaik. Pada tahap ini, dilakukan pengujian terhadap beberapa variasi nilai k dalam rentang 1 hingga 50, serta parameter lain seperti metrik jarak dan skema pembobotan.

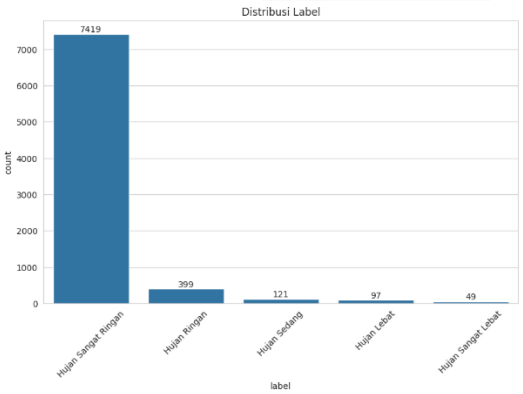
3. Evaluasi Kinerja Model Klasifikasi

Tahap evaluasi model dilakukan untuk mengetahui seberapa baik model yang telah dibangun dalam melakukan klasifikasi data. Pengukuran kinerja model dilakukan menggunakan beberapa metrik evaluasi, yaitu akurasi, precision, recall, dan f1-score.

IV. HASIL DAN PEMBAHASAN

A. Distribusi Kelas Label

Pelabelan data merupakan tahapan yang dilakukan untuk mengubah data curah hujan yang masih berbentuk nilai numerik menjadi data kategorikal yang digunakan sebagai kelas target dalam proses klasifikasi. Pada penelitian ini, pelabelan dilakukan berdasarkan nilai pada atribut Hour Rainfall (mm) yang menunjukkan jumlah curah hujan dalam satu jam. Atribut tersebut dipilih karena dapat menggambarkan intensitas hujan secara langsung sesuai dengan standar pengukuran meteorologi. Selanjutnya, setiap nilai curah hujan dikelompokkan berdasarkan klasifikasi intensitas hujan yang mengacu pada ketentuan BMKG (Badan Meteorologi, Klimatologi, dan Geofisika), sehingga diperoleh lima kategori kelas, yaitu hujan sangat ringan, hujan ringan, hujan sedang, hujan lebat, dan hujan sangat lebat. Hasil pelabelan ini digunakan sebagai variabel target pada proses pembangunan dan evaluasi model klasifikasi.

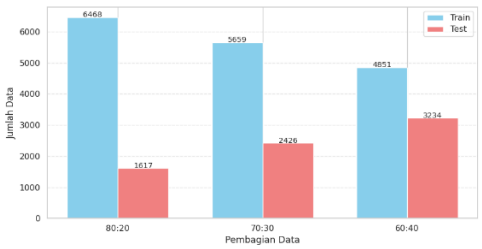


GAMBAR 2
(Distribusi Label)

Hasil dari proses *labeling* menunjukkan bahwa distribusi data tidak merata pada setiap kategori. Sebagian besar data berada pada kategori hujan sangat ringan dengan jumlah 7.419 data, diikuti oleh hujan ringan sebanyak 399 data. Sementara itu, jumlah data untuk kategori hujan sedang, hujan lebat, dan hujan sangat lebat jauh lebih sedikit, masing-masing sebanyak 121, 97, dan 49 data. Kondisi ini menunjukkan bahwa dataset didominasi oleh intensitas hujan rendah, sehingga terjadi ketidakseimbangan data.

B. Pembagian Data

Pada penelitian ini digunakan tiga pembagian data, yaitu 80:20, 70:30, dan 60:40. Variasi pembagian data tersebut digunakan untuk membandingkan performa model pada setiap skenario sehingga dapat diketahui proporsi data yang memberikan hasil klasifikasi terbaik.



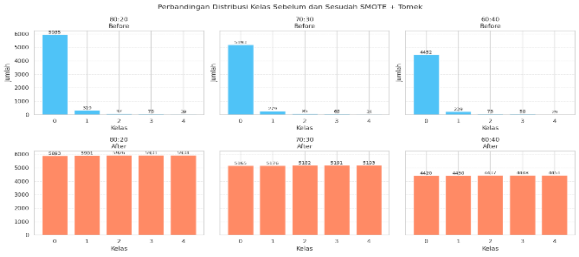
GAMBAR 3
(Pembagian Data)

Hasil menunjukkan bahwa data berhasil dibagi menjadi tiga skenario pembagian data, yaitu 80:20, 70:30, dan 60:40. Pada pembagian 80:20 diperoleh 6.468 data *training* dan 1.617 data *testing*. Pada pembagian 70:30 terdapat 5.659 data *training* dan 2.426 data *testing*. Sementara itu, pada pembagian 60:40 diperoleh 4.851 data *training* dan 3.234 data *testing*.

Berdasarkan hasil tersebut, data training memiliki jumlah lebih besar dibandingkan data *training* karena digunakan untuk melatih model agar dapat mempelajari pola data dengan baik. Sementara itu, data *testing* digunakan untuk menguji kinerja model pada data yang belum pernah digunakan sebelumnya. Variasi pembagian data ini dilakukan untuk membandingkan kinerja model pada setiap proporsi data yang digunakan, sehingga dapat diketahui pembagian data yang memberikan hasil evaluasi terbaik terhadap model klasifikasi.

C. *Balance* Data

Pada tahap *balance* data, dilakukan penyeimbangan jumlah data pada setiap kelas karena sebelumnya distribusi label menunjukkan kondisi yang tidak seimbang, di mana kelas hujan sangat ringan jauh lebih dominan dibandingkan kelas lainnya. Untuk mengatasi hal tersebut, digunakan metode *SMOTE* (*Synthetic Minority Over-sampling Technique*) yang dikombinasikan dengan *Tomek Links*. *SMOTE* berfungsi untuk menambah data pada kelas minoritas dengan membuat sampel sintetis berdasarkan pola data yang ada, sedangkan *Tomek Links* digunakan untuk menghapus data yang saling tumpang tindih antar kelas sehingga batas antar kelas menjadi lebih jelas. Kombinasi kedua metode ini dipilih karena tidak hanya mampu menyeimbangkan jumlah data, tetapi juga meningkatkan kualitas distribusi data.



GAMBAR 4
(Balance Data)

Hasil memperlihatkan perbedaan yang cukup jelas antara kondisi sebelum dan sesudah proses penyeimbangan data pada setiap pembagian data. Sebelum dilakukan penyeimbangan data, jumlah data pada masing-masing kelas tidak merata, dengan satu kelas mendominasi secara signifikan. Namun, setelah diterapkan metode SMOTE dan *Tomek Links*, jumlah data pada setiap kelas menjadi seimbang di semua pembagian data.

D. Model *K-Nearest Neighbor* (K-NN) *Baseline*

Model Baseline merupakan model awal KNN yang dibangun tanpa proses optimasi parameter menggunakan *GridSearchCV*. Pada tahap ini, parameter seperti jumlah tetangga (k) menggunakan nilai default, yaitu k = 5, sehingga performa yang dihasilkan mencerminkan kinerja dasar model. *Baseline* ini digunakan sebagai pembanding untuk melihat peningkatan performa setelah dilakukan *tuning* parameter.

TABEL 2
(Akurasi *Baseline)*

Split Data	*K*	*Accuracy*	*Matrix*	*Weight*
80:20	5	0.7767	*Euclidean*	*Distance*
70:30	5	0.7704	*Euclidean*	*Distance*
60:40	5	0.7696	*Euclidean*	*Distance*

Hasil pengujian model *baseline* menunjukkan akurasi sebesar 0.7767 pada pembagian data 80:20, 0.7704 pada pembagian data 70:30, dan 0.7696 pada pembagian data 60:40 dengan menggunakan jarak *Euclidean* dan pembobotan *Distance*. Hasil tersebut menunjukkan bahwa performa model relatif konsisten pada setiap pembagian data, meskipun terjadi penurunan akurasi seiring berkurangnya jumlah data latih. Kondisi ini mengindikasikan bahwa jumlah data pelatihan berpengaruh terhadap kemampuan model dalam mempelajari pola data meteorologi. Pada algoritma *K-Nearest Neighbor* (K-NN), proses klasifikasi dilakukan berdasarkan kedekatan antar data, sehingga kualitas dan jumlah data latih menjadi faktor penting dalam menentukan hasil prediksi. Penggunaan jarak *Euclidean* berfungsi untuk menghitung kedekatan antar data dalam ruang fitur, sedangkan pembobotan *Distance* memberikan pengaruh yang lebih besar kepada tetangga terdekat dalam proses penentuan kelas. Secara keseluruhan, model baseline telah mampu melakukan klasifikasi dengan cukup baik, namun hasil yang diperoleh masih dapat ditingkatkan melalui optimasi parameter untuk memperoleh performa yang lebih optimal dalam mengklasifikasikan intensitas curah hujan.

TABEL 2
(Metrik Evaluasi *Baseline)*

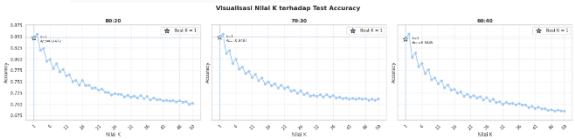
Split Data	*Precision*	*Recall*	*F1-Score*	*Support*
80:20	0.24	0.29	0.25	1617
70:30	0.23	0.28	0.24	2426
60:40	0.25	0.32	0.25	3234

Hasil menunjukan pada pembagian data 80:20, model menghasilkan nilai *precision* sebesar 0.24, *recall* sebesar 0.29, dan *F1-score* sebesar 0.25 dengan total data uji sebanyak 1617. Hasil tersebut menunjukkan bahwa kemampuan model dalam mengklasifikasikan seluruh kelas masih tergolong rendah dan belum merata. Nilai recall yang lebih tinggi dibandingkan precision menunjukkan bahwa model cukup mampu mengenali sebagian data pada setiap kelas, namun masih sering melakukan kesalahan dalam menentukan kelas yang tepat.

Pada pembagian data 70:30, model memperoleh nilai *precision* sebesar 0.23, *recall* sebesar 0.28, dan *F1-score* sebesar 0.24 dengan jumlah data uji sebanyak 2426. Hasil ini menunjukkan bahwa perubahan proporsi data belum memberikan peningkatan yang signifikan terhadap performa model. Model masih cenderung mengalami kesalahan klasifikasi, terutama pada kelas minoritas.

Sementara itu, pada pembagian data 60:40, model menghasilkan *precision* sebesar 0.25, *recall* sebesar 0.32, dan *F1-score* sebesar 0.25 dengan total data uji sebanyak 3234. Nilai recall yang meningkat menunjukkan bahwa model lebih mampu mengenali data pada berbagai kelas. Namun, nilai *precision* dan *F1-score* yang relatif tetap menunjukkan bahwa model masih belum optimal dalam menghasilkan klasifikasi yang akurat secara keseluruhan.

E. Model K-NN *GridSearchCV*



GAMBAR 5
(Visualisasi Nilai K terhadap Test Accuracy)

Berdasarkan hasil pengujian pada Gambar 4.10, nilai k = 1 menghasilkan akurasi tertinggi pada seluruh pembagian data, yaitu 80:20, 70:30, dan 60:40, dengan nilai akurasi berada pada kisaran 0,84. Hasil tersebut menunjukkan bahwa penggunaan nilai k yang kecil, khususnya k = 1, mampu meningkatkan akurasi model K-NN karena proses klasifikasi dilakukan berdasarkan tetangga terdekat yang memiliki kemiripan terhadap data uji.

Namun, penggunaan nilai k = 1 menunjukkan bahwa proses klasifikasi mempertimbangkan satu tetangga terdekat sehingga model menjadi sangat bergantung pada data latih. Akibatnya, model cenderung mempelajari pola data pelatihan secara berlebihan sehingga kemampuan generalisasi terhadap data baru menjadi kurang optimal.

Selain itu, grafik menunjukkan bahwa semakin besar nilai k yang digunakan, maka akurasi model cenderung mengalami penurunan secara bertahap. Pada saat nilai k meningkat hingga mencapai k = 50, nilai akurasi menurun ke kisaran 0,69–0,71. Kondisi ini terjadi karena semakin banyak tetangga yang dipertimbangkan dalam proses klasifikasi dapat menyebabkan model kehilangan detail pola antar data sehingga hasil klasifikasi menjadi kurang optimal.

Pola penurunan akurasi tersebut terlihat konsisten pada seluruh pembagian data. Oleh karena itu, dapat disimpulkan bahwa penggunaan nilai k kecil memberikan performa terbaik pada penelitian ini, khususnya pada nilai k = 1.

TABEL 2
(Akurasi *GridSearchCV)*

Split Data	*K*	*Accuracy*	*Matrix*	*Weight*
80:20	1	0.8472	*Manhattan*	*Uniform*
70:30	1	0.8487	*Manhattan*	*Uniform*
60:40	1	0.8445	*Manhattan*	*Uniform*

Berdasarkan hasil optimasi menggunakan *GridSearchCV* dengan rentang nilai K dari 1 hingga 50, diperoleh bahwa pada seluruh pembagian data, kombinasi parameter terbaik adalah K = 1, jarak *Manhattan*, dan pembobotan *Uniform*. Pada pembagian data 80:20, model menghasilkan akurasi sebesar 0.8472. Pada pembagian data 70:30, akurasi meningkat menjadi 0.8487, yang merupakan nilai tertinggi dari seluruh pengujian. Sementara itu, pada pembagian data 60:40, model memperoleh akurasi sebesar 0.8445. Hasil tersebut menunjukkan bahwa kombinasi parameter yang diperoleh melalui proses optimasi mampu memberikan kinerja klasifikasi yang konsisten pada berbagai pembagian data.

Terpilihnya K = 1 sebagai parameter terbaik menggambarkan bahwa data yang digunakan memiliki pola lokal yang cukup kuat, sehingga kelas suatu data lebih tepat ditentukan berdasarkan tetangga terdekatnya. Kondisi ini menunjukkan bahwa data dengan karakteristik meteorologi yang cenderung berada pada kelas yang sama. Selain itu, penggunaan metrik jarak *Manhattan* menunjukkan bahwa perhitungan kedekatan berdasarkan penjumlahan selisih *absolut* antar atribut lebih sesuai dalam merepresentasikan karakteristik data dibandingkan metrik jarak lainnya yang diuji. Dengan demikian, proses identifikasi tetangga terdekat dapat dilakukan secara lebih akurat.

Sementara itu, penggunaan pembobotan *Uniform* menunjukkan bahwa setiap tetangga yang terlibat dalam proses klasifikasi memberikan kontribusi yang sama terhadap penentuan kelas. Kombinasi ketiga parameter tersebut berhasil meningkatkan akurasi model dari 0.7767 sebelum optimasi menjadi 0.8487 setelah optimasi, sehingga dapat disimpulkan bahwa optimasi parameter menggunakan *GridSearchCV* memberikan pengaruh terhadap peningkatan kinerja algoritma *K-Nearest Neighbor* dalam klasifikasi curah hujan.

TABEL 2
(Matrix Evaluasi *GridSearchCV)*

Split Data	*Precision*	*Recall*	*F1-Score*	*Support*
80:10:10	0.25	0.28	0.25	1617
70:15:15	0.25	0.28	0.26	2426
60:20:20	0.27	0.32	0.28	3234

Berdasarkan hasil evaluasi model menggunakan *GridSearchCV* menunjukkan bahwa pada pembagian data 80:20 diperoleh nilai *precision* sebesar 0.25, *recall* 0.28, dan *F1-score* 0.25 dengan jumlah data uji sebanyak 1617. Nilai tersebut menunjukkan bahwa kemampuan model dalam mengklasifikasikan data pada seluruh kelas masih tergolong rendah dan belum optimal. Pada pembagian data 70:30, *precision* tetap berada pada nilai 0.25, *recall* sebesar 0.28, dan *F1-score* meningkat sedikit menjadi 0.26 dengan jumlah data uji sebanyak 2426. Sementara itu, pada pembagian data 60:40 diperoleh peningkatan nilai *precision* menjadi 0.27, *recall* sebesar 0.32, dan *F1-score* sebesar 0.28 dengan jumlah data uji sebanyak 3234, sehingga menunjukkan performa yang lebih baik dibandingkan pembagian data lainnya. Hasil tersebut menunjukkan bahwa penambahan proporsi data uji pada pembagian 60:40 memberikan peningkatan terhadap kemampuan model dalam melakukan klasifikasi, meskipun performanya secara keseluruhan masih belum terlalu tinggi.

F. Analisis Hasil

Berdasarkan hasil pengujian, model *baseline* menggunakan parameter (k = 5), *metric Euclidean*, dan *weight Distance* menghasilkan akurasi yang lebih rendah dibandingkan model hasil optimasi. Pada kombinasi parameter tersebut, model masih cenderung mengikuti pola kelas mayoritas karena proses klasifikasi mempertimbangkan lima tetangga terdekat. Akibatnya, beberapa data pada kelas minoritas masih sering salah diprediksi sehingga akurasi hanya mencapai 0.7767 pada split data 80:20, 0.7704 pada split data 70:30, dan 0.7696 pada split data 60:40.

Hasil optimasi menggunakan *GridSearchCV* menunjukkan bahwa kombinasi parameter terbaik adalah (k = 1), *metric Manhattan*, dan *weight Uniform*. Kombinasi parameter ini menghasilkan performa yang lebih baik karena model hanya menggunakan satu tetangga terdekat yang paling mirip dengan data uji. Dengan demikian, model menjadi lebih fokus dalam mengenali pola setiap kelas dan tidak terlalu dipengaruhi oleh kelas mayoritas di sekitarnya.

Penggunaan *metric Manhattan* menghasilkan akurasi yang lebih tinggi dibandingkan Euclidean karena perhitungan jaraknya lebih sesuai dengan pola data meteorologi pada penelitian ini. *Manhattan* mampu memberikan pemisahan jarak antar data yang lebih jelas sehingga proses penentuan tetangga terdekat menjadi lebih optimal. Hal tersebut membuat model lebih mudah membedakan antar kelas curah hujan.

Selain itu, penggunaan *weight Uniform* memberikan hasil terbaik karena seluruh tetangga memiliki bobot yang sama dalam proses klasifikasi. Pada saat nilai (k = 1), hanya terdapat satu tetangga yang digunakan sehingga pembobotan *Distance* tidak memberikan pengaruh tambahan terhadap hasil prediksi. Oleh sebab itu, kombinasi *Uniform* lebih stabil dan menghasilkan akurasi yang lebih tinggi.

Kombinasi parameter terbaik tersebut berhasil meningkatkan akurasi pada seluruh pembagian data. Pada split data 80:20, akurasi meningkat dari 0.7767 menjadi 0.8472. Pada split data 70:30, akurasi meningkat dari 0.7704 menjadi 0.8487 dan menjadi hasil terbaik dari seluruh pengujian. Sedangkan pada split data 60:40, akurasi

meningkat dari 0.7696 menjadi 0.8445. Hasil ini menunjukkan bahwa parameter hasil optimasi mampu meningkatkan kemampuan model dalam mengenali pola data curah hujan dibandingkan parameter *default* pada model *baseline*.

G. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa algoritma *K-Nearest Neighbor* (K-NN) dapat digunakan untuk klasifikasi curah hujan berdasarkan data meteorologi. Pada pengujian awal sebelum dilakukan optimasi parameter, model baseline memperoleh akurasi sebesar 0.7704 sehingga menunjukkan bahwa performa model masih belum optimal dalam mengenali pola data pada setiap kelas. Model baseline menggunakan parameter (k = 5*), Metric Euclidean*, dan *Weight Distance*.

Optimasi parameter menggunakan *GridSearchCV* terbukti mampu meningkatkan performa algoritma *K-Nearest Neighbor* (K-NN). Hasil pengujian menunjukkan bahwa akurasi model meningkat menjadi 0.8487 setelah dilakukan optimasi parameter. Parameter terbaik yang diperoleh pada penelitian ini adalah (k = 1), *Metric Manhattan*, dan *Weight Uniform*. Kombinasi parameter tersebut menghasilkan performa klasifikasi paling optimal karena model lebih fokus pada satu tetangga terdekat yang memiliki kemiripan paling tinggi dengan data uji. Selain itu, penggunaan Metric Manhattan menghasilkan perhitungan jarak yang lebih sesuai terhadap pola data meteorologi, sedangkan *Weight Uniform* memberikan hasil klasifikasi yang lebih stabil pada penggunaan (k = 1).

Penggunaan balancing data dengan kombinasi SMOTE + *Tomek Link* juga memberikan pengaruh terhadap performa model *K-Nearest Neighbor* (K-NN). Proses *balancing* membantu memperbaiki distribusi data antar kelas sehingga model menjadi lebih baik dalam mengenali kelas minoritas dibandingkan sebelum dilakukan balancing data. Meskipun demikian, model masih mengalami kesulitan dalam membedakan beberapa kelas yang memiliki pola serupa, terutama pada kelas dengan jumlah data yang lebih sedikit.

REFERENSI

[1] R. Sakti and A. Daulay, “Analisis Kritis dan Pengembangan Algoritma K-Nearest Neighbor (KNN): Sebuah Tinjauan Literatur,” vol. 4, no. 2, pp. 131–141, 2024.

[2] V. Kumar, N. Kedam, O. Kisi, and S. Alsulamy, “A Comparative Study of Machine Learning Models for Daily and Weekly Rainfall Forecasting,” *Water Resour. Manag.*, vol. 39, no. 1, pp. 271–290, 2025, doi: 10.1007/s11269-024-03969-8.

[3] P. Akurasi, C. Algoritma, N. Untuk, and K. Curah, “Perbandingan akurasi algoritma c4.5 dan,” vol. 9, no. 2, pp. 628–638, 2024.

[4] E. A. Firdaus, J. Manurung, H. Saragih, and M. Azhar, “Optimization of XGBoost hyperparameters using grid search and random search for credit card default prediction,” vol. 14, no. 2, pp. 269–280, 2025.

[5] D. Ardiyansyah, N. Oktafiani, and U. T. Madura, “PERBANDINGAN METODE PENGUKURAN JARAK PADA K-NEAREST,” vol. 1, no. 2, pp. 116–122, 2024.

[6] Z. Cahya, D. Cahya, T. Nugroho, A. Zuhri, and W. Agusta, “CNN Model with Parameter Optimisation for Fine-Grained Banana Ripening Stage Classification,” *ACM Int. Conf. Proceeding Ser.*, vol. 13, pp. 90–94, 2022, doi: 10.1145/3575882.3575900.

[7] P. Probst, M. Wright, and A. Boulesteix, “Hyperparameters and Tuning Strategies for Random Forest,” no. 2010, pp. 1–18, 2018.

[8] F. Pedregosa, R. Weiss, and M. Brucher, “Scikit-learn : Machine Learning in Python,” vol. 12, pp. 2825–2830, 2011.

[9] N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE : Synthetic Minority Over-sampling Technique,” vol. 16, pp. 321–357, 2002.

[10] I. Tomek, “I’yj*,” pp. 769–772, 1976.

[11] V. A. Semenov, A. V Kravtsov, and N. Y. Gnedin, “Weather Prediction using Advanced Machine Learning Techniques Weather Prediction using Advanced Machine Learning Techniques,” 2021, doi: 10.1088/1742-6596/2089/1/012059.

[12] O. S. Riza *et al.*, “Bibliometric Study : Rainfall Classification - Prediction using Machine Learning Methods,” vol. x, no. x, pp. 206–218, 2023.

[13] Z. A. Dwiyanti and C. Prianto, “Prediksi Cuaca Kota Jakarta menggunakan Metode Random Forest : Studi Optimalitas,” vol. 17, no. 2, pp. 127–137, 2023.

[14] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.