JNATIA Volume 1, Nomor 3, Mei 2023 p-ISSN: 2986-3929
Jurnal Nasional Teknologi Informasi dan Aplikasinya

Analisis Performa Algoritma *K-Nearest Neighbor* dalam Klasifikasi Tingkat Kerontokan Rambut

Gede Dikka Widya Prana[a1], Luh Gede Astuti[a2]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Jalan Raya Kampus Udayana, Bukit Jimbaran, Kuta Selatan, Badung, Bali Indonesia
[1]dikkawidyapranagede@gmail.com
[2]lg.astuti@unud.ac.id

Abstract

Hair loss can lead to baldness and affect one's self-confidence. Normally, hair falls out in 80-120 strands per day, and the average number of hair follicles on the head is around 100,000. If the amount is reduced by 50%, it can be considered a disorder. Therefore, a classification of hair loss levels is necessary to determine appropriate actions. Previous study has shown that the K-Nearest Neighbor algorithm is capable of classifying various diseases. In this study, the Luke Hair Loss Dataset from the website kaggle.com, consisting of 400 data points, was used. To evaluate the method's feasibility, a confusion matrix was employed. The objective of this research is to analyze the performance of the K-Nearest Neighbor algorithm. Several scenarios were utilized, including testing the model before and after SMOTE oversampling, testing before and after data normalization, testing based on different K values, and testing with varying ratios of training and testing data. The results of this study indicate that the K-Nearest Neighbor algorithm achieved the highest accuracy value of 0,9853, precision of 0,9886, recall of 0,9833, and f1-score of 0,9856.

Keywords: *Hair Loss, Classification, K-Nearest Neighbor, Performance Test, Confusion Matrix*

1. Pendahuluan

Rambut rontok adalah permasalahan utama yang terjadi pada rambut. Rambut rontok menjadi salah satu permasalahan rambut yang sering ditemui [1]. Rambut rontok merupakan fase alami yang dialami setiap orang [2]. Normalnya, rambut rontok 80-120 helai per hari dan jumlah normal folikel rambut kepala adalah sekitar 100.000. Jika jumlahnya berkurang hingga 50%, maka dapat dikatakan sebagai kelainan. Kerontokan rambut dapat memengaruhi fungsi biologis rambut terhadap tubuh, apabila melebihi batas normal [3], [4]. Selain itu, kerontokan rambut yang dapat mengakibatkan kebotakan dan memengaruhi kepercayaan diri seseorang sehingga menjadi masalah yang sangat mengkhawatirkan [5]. Oleh karena itu, klasifikasi tingkat kerontokan rambut diperlukan untuk menentukan tindakan yang tepat dalam mengatasi masalah tersebut.

Klasifikasi dapat dilakukan dengan teknologi kecerdasan buatan dan melibatkan *data mining* untuk mengumpulkan serta menganalisis data. Salah satu algoritma yang umum digunakan untuk klasifikasi adalah *K-Nearest Neighbor.* Algoritma *K-Nearest Neighbor* (K-NN) merupakan salah satu algoritma yang mengklasifikasikan objek berdasarkan jarak terdekat antara data latih dengan objek yang akan diklasifikasikan [6]. Pada penelitian sebelumnya algoritma K-NN digunakan untuk memprediksi penyakit jantung. Berdasarkan hasil pengujian, didapatkan tingkat akurasi sebesar 81,31% dan *classification error* sebesar 18,68% [7]. Algoritma K-NN juga telah digunakan untuk klasifikasi penyakit parkinson. Melalui penelitian ini, diperoleh nilai akurasi sebesar 80% [8]. Selain itu, untuk mengklasifikasikan data tidak seimbang, algoritma K-NN pernah digunakan untuk mengklasifikasikan penyakit diabetes dengan penambahan *Synthetic Minority Oversampling* (SMOTE). K-NN dengan SMOTE, akurasinya lebih baik daripada akurasi yang dihasilkan tanpa menggunakan SMOTE dengan peningkatan akurasi tertinggi sebesar 8,25% [9].

Penelitian kali ini akan menggunakan K-NN untuk mengklasifikasikan tingkat kerontokan rambut. Adapun tujuan dari penelitian ini adalah untuk mendapatkan model K-NN dengan performa terbaik dalam mengklasifikasikan tingkat kerontokan rambut. Akan diujikan pengaruh penambahan SMOTE, normalisasi data, pemilihan nilai K, dan pembagian data uji dan data latih. Penelitian dilakukan mulai dari pengumpulan data, *preprocessing* data, tahap klasifikasi, dan evaluasi. Belum ada penelitian sebelumnya yang menganalisis performa algoritma *K-Nearest Neighbor* dalam klasifikasi tingkat kerontokan rambut. Diharapkan penelitian ini dapat bermanfaat bagi masyarakat dan dapat menjadi referensi serta pengembangan ilmu pengetahuan untuk penelitian selanjutnya.

2. Metode Penelitian

2.1. Pengumpulan Data

Data yang digunakan pada penelitian ini adalah data sekunder, yakni *Luke Hair Loss Dataset* yang diperoleh dari website kaggle.com [10]. Data ini berupa file ".csv" dengan 400 data dan 14 atribut seperti Tabel 1.

Tabel 1. Atribut *Dataset*

No	Atribut	Tipe	Deskripsi
1	date	Nominal	Tanggal observasi dilakukan
2	hair_loss	Nominal	Menunjukkan 4 tingkatan status kerontokan rambut (*few*, *medium*, *many*, *a lot*)
3	stay_up_late	Numerik	Menunjukkan durasi begadang dari durasi normal tidur (8 jam). Diberi label 0-8. Label 2 menunjukkan begadang 2 jam (tidur hanya 6 jam dari durasi normal tidur)
4	pressure_level	Nominal	Menyatakan empat tingkat tekanan yang dirasakan (*low*, *medium*, *high*, *very high*)
5	coffee_consumed	Numerik	Menyatakan jumlah, berapa cangkir kopi yang dikonsumsi dalam sehari
6	brain_working_duration	Numerik	Menunjukkan durasi pekerjaan yang membutuhkan kekuatan otak telah dilakukan, diukur dalam jam
7	school_assesssment	Nominal	Menunjukkan jenis penilaian yang telah dilakukan (*final exam*, *final exam revision*, *individual assessment*, *team assessment*, *none*)
8	stress_level	Nominal	Menunjukkan empat tingkat stress (*low*, *medium*, *high*, *very high*)
9	shampoo_brand	Nominal	Menunjukkan merek sampo
10	swimming	Nominal	Menunjukkan apakah berenang di hari itu (*yes*, *no*)
11	hair_washing	Nominal	Menunjukkan apakah keramas di hari itu (Y: *yes*, N: *no*)
12	hair_grease	Numerik	Menunjukkan volume pemakaian pelumas rambut, diukur dengan skala 1-5
13	dandruff	Nominal	Menunjukkan 3 tingkat kondisi kulit kepala/ketombe (*none*, *few*, *many*)
14	libido	Numerik	Menunjukkan tingkat hormon, diukur dengan skala 1-5 (secara subjektif)

2.2. *Preprocessing* Data

Preprocessing dilakukan untuk menyiapkan *dataset* yang akan dianalisis lebih lanjut, dengan tahapan sebagai berikut:

a. Analisis Distribusi Data

Analisis distribusi data dilakukan untuk mengetahui keseimbangan jumlah data setiap kelas. Apabila terdapat perbedaan yang signifikan, maka perlu dilakukan penyeimbangan *dataset*. *Synthetic Minority Oversampling Technique* (SMOTE) merupakan salah satu teknik *oversampling* yang menyintesis *dataset* minoritas hingga seimbang dengan kelas mayoritas [11].

b. Pemilihan Atribut yang akan Digunakan

Atribut yang dipilih harus relevan dengan masalah yang ingin dipecahkan atau tujuan analisis. Atribut memiliki hubungan langsung dengan variabel target atau variabel yang ingin diprediksi.

c.* Analisis Atribut *Missing Value

Analisis *missing value* dilakukan untuk mengetahui atribut yang hilang atau kosong. *Missing value* dapat diatasi dengan metode *replace* dan *imputation*.

d. Transformasi Data

Transformasi data dilakukan untuk mengubah variabel kategori menjadi representasi numerik agar dapat dimanfaatkan oleh algoritma *machine learning*. Transformasi data dapat dilakukan dengan *label encoding* sehingga mengurangi penggunaan memori dan meningkatkan kecepatan komputasi. Transformasi data juga dapat dilakukan dengan memberi label secara manual pada *value* suatu kolom [12].

e. Normalisasi Data

Normalisasi dilakukan untuk menyamakan rentang nilai (skala) pada data. *Min-Max* merupakan salah satu metode normalisasi yang akan menghasilkan data hasil normalisasi dengan rentang 0 hingga 1 [13].

2.3. Membangun Model

Algoritma *K-Nearest Neighbor* (K-NN) digunakan untuk mengklasifikasikan objek berdasarkan data pembelajaran yang memiliki jarak terdekat dengan objek yang akan diklasifikasikan [14]. Algoritma K-NN memiliki beberapa kelebihan, yaitu kemampuan untuk menangani data latih yang memiliki banyak *noise*, serta efektif ketika digunakan pada data latih yang besar [15].

Pemodelan dilakukan menggunakan data yang sebelumnya telah di-*preprocessing*. Pemodelan pada penelitian ini adalah klasifikasi data dengan algoritma *K-Nearest Neighbor* menggunakan bahasa pemrograman Python dan memanfaatkan *library scikit-learn* atau *sklearn*.

2.4. Analisis Hasil dan Kesimpulan

Pada penelitian ini, evaluasi dilakukan dengan *confusion matrix*, untuk menghitung nilai akurasi, presisi, *recall*, dan *f1-score*.

3. Hasil dan Diskusi

3.1. *Preprocessing* Data

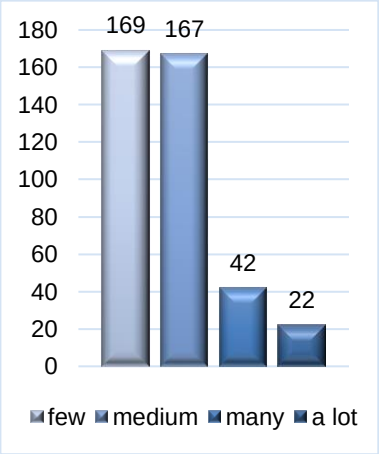
a. Analisis Distribusi Data

Distribusi kelas dari *Luke Hair Loss Dataset* memiliki selisih yang sangat signifikan seperti yang ditunjukkan oleh grafik pada Gambar 2. Dalam hal ini, dilakukan penyeimbangan

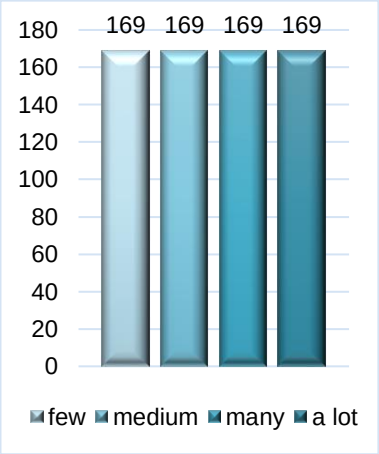
kelas dengan metode SMOTE *oversampling* (*synthetic minority oversampling technique*), memanfaatkan *library imblearn* seperti pada Gambar 1. Sehingga diperoleh keseimbangan kelas seperti yang ditunjukkan oleh grafik pada Gambar 3.

```
from imblearn.over_sampling import SMOTE
smote = SMOTE(k_neighbors=3,random_state=1)
x_resampled, y_resampled = smote.fit_resample(x, y)
```

Gambar 1. SMOTE O*versampling*



Gambar 2. Distribusi kelas sebelum SMOTE *Oversampling*



Gambar 3. Distribusi kelas setelah SMOTE *Oversampling*

b. Pemilihan Atribut yang akan Digunakan

Atribut yang akan diteliti sebagai parameter menentukan klasifikasi tingkat kerontokan rambut adalah "stay_up_late", "pressure_level", "coffee_consumed", "brain_working_duration", "school_assesssment", "stress_level", "shampoo_brand", "swimming", "hair_washing", "hair_grease", "dandruff", dan "libido". Dalam hal ini, atribut "date" dihapus karena tidak relevan terhadap tujuan klasifikasi. Hal ini dapat dilihat pada Gambar 4.

```
# menghapus kolom "date " karena tidak diperlukan
dataset.drop(['date '], axis=1, inplace=True)
```

Gambar 4. Penghapusan Kolom "date"

c.* Analisis Atribut *Missing Value

Pada *dataset* ditemukan *missing value* pada atribut "school_assesssment", "dandruff", dan "hair_grease". *Missing value* pada atribut "school_assesssment" dan "dandruff" di-*replace* menjadi "No", karena "None" di sini bukan merupakan *missing value*, melainkan menyatakan *tidak ada*. Kemudian, *missing value* pada "hair_grease" diatasi dengan *imputation* yakni mengisi dengan modusnya. Hal ini dapat dilihat pada Gambar 5.

```
# mengganti nan dengan no, karna sebenarnya bukan missing value
dataset['school_assesssment'] = dataset['school_assesssment'].fillna('No')
dataset['dandruff'] = dataset['dandruff'].fillna('No')
```

```
# mengganti missing values pada hair_grease dengan modusnya
modus = dataset['hair_grease'].mode()[0]
dataset['hair_grease'].fillna(modus, inplace=True)
```

Gambar 5. *Replace* dan *Imputing Missing Value*

d. Transformasi Data

Seperti pada Gambar 6, transformasi data dilakukan dengan *label encoding* pada kolom "pressure_level", "school_assessment", "stress_level", "shampoo_brand", "swimming", "hair_washing", dan "dandruff", karena masih nominal. Kemudian, *labeling* juga dilakukan pada kolom "hair_loss": "few" diberi label "0", "medium" diberi label "1", "many" diberi label "2", dan "a lot" diberi label "3".

```
# membuat mapping antara nilai awal dan label yang diinginkan
label_mapping = {
    'Few': 0,
    'Medium': 1,
    'Many': 2,
    'A lot': 3
}

# mengubah nilai pada kolom 'hair_loss' menggunakan mapping
dataset['hair_loss'] = dataset['hair_loss'].map(label_mapping)

# encoding data nominal
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
columns_to_transform = [2, 5, 6, 7, 8, 9, 11]
for col in columns_to_transform:
    dataset.iloc[:, col] = le.fit_transform(dataset.iloc[:, col].values)
dataset
```

Gambar 6. *Labeling* kolom "hair_loss" dan *Label Encoding* Kolom Nominal

e. Normalisasi Data

Normalisasi dilakukan untuk menyamakan rentang nilai (skala) pada data. Dalam penelitian ini, digunakan *min-max normalization* untuk normalisasi atribut, seperti pada Gambar 7.

```
from sklearn.preprocessing import MinMaxScaler
# membuat objek MinMaxScaler
scaler = MinMaxScaler()
# melakukan normalisasi pada kolom yang ditentukan
x22 = scaler.fit_transform(x22)
```

Gambar 7. Normalisasi dengan *Min-Max Normalization*

3.2. Model dan Pengujian Model

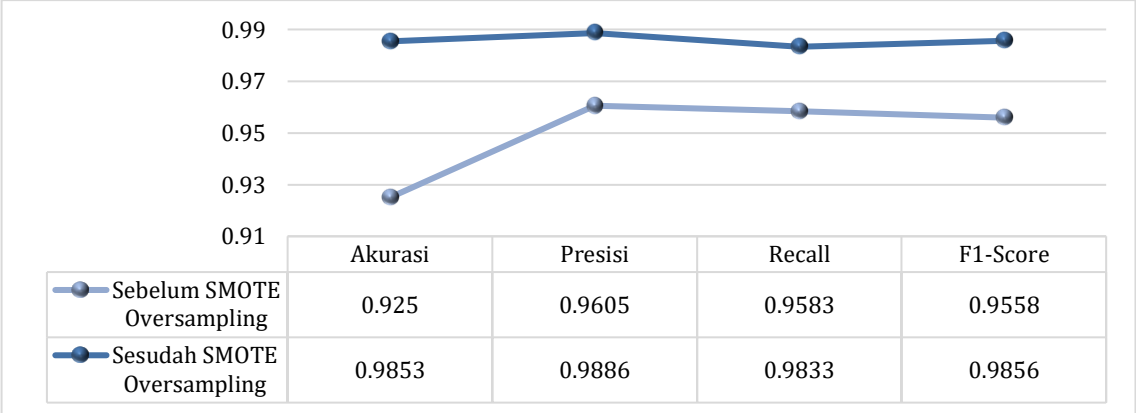
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x1,y1,test_size=0.10, random_state=1)

from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
model.fit(x_train, y_train)

y_pred=model.predict(x_test)

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='macro')
recall = recall_score(y_test, y_pred, average='macro')
f1_score = f1_score(y_test, y_pred, average='macro')
```

Gambar 8. Pemodelan *K-Nearest Neighbor* dan Perhitungan Evaluasi

Gambar 8 menunjukkan proses dari *split* data, pemodelan K-NN dengan *library sklearn* hingga *fitting* data latih, hasil prediksi, sampai perhitungan akurasi, presisi, *recall*, dan *f1-score*. Pengujian model yang dilakukan dalam penelitian ini menggunakan beberapa skenario, mulai dari pengujian model sebelum dan sesudah menggunakan SMOTE *oversampling*, pengujian model sebelum dan sesudah dilakukan normalisasi, pengujian model berdasarkan nilai K, serta pengujian model dengan beberapa perbandingan data latih dan data uji.

a. Pengujian Model sebelum dan sesudah SMOTE *Oversampling*

Pengujian dilakukan menggunakan data yang tidak dinormalisasi dengan perbandingan data latih dan data uji, yaitu 90:10 dan nilai K=3. Dari pengujian tersebut dihasilkan nilai akurasi, presisi, *recall*, dan *f1-score* yang diterangkan pada Gambar 9.

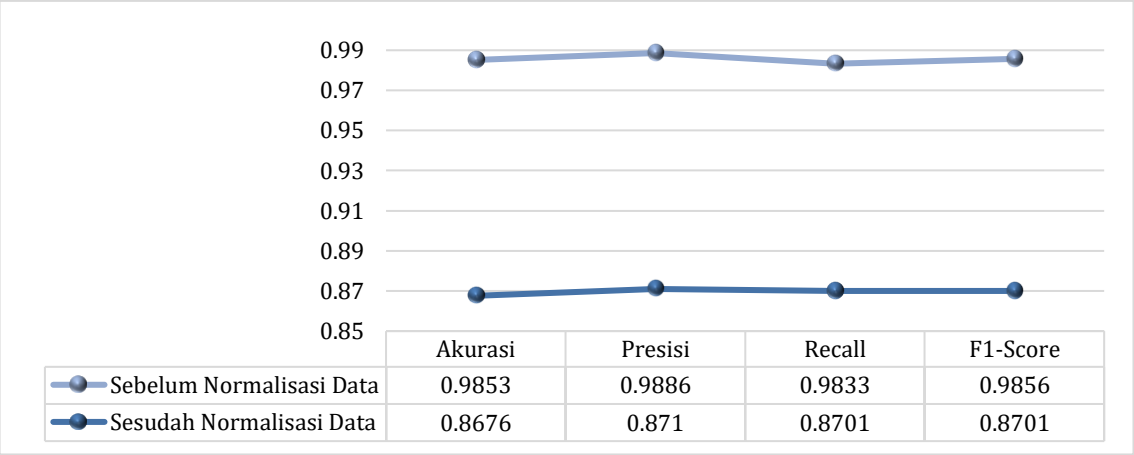


	Akurasi	Presisi	Recall	F1-Score
Sebelum SMOTE Oversampling	0.925	0.9605	0.9583	0.9558
Sesudah SMOTE Oversampling	0.9853	0.9886	0.9833	0.9856

Gambar 9. Hasil Uji sebelum dan sesudah SMOTE *Oversampling*

b. Pengujian Model sebelum dan sesudah Normalisasi Data

Pengujian dilakukan menggunakan data yang sudah dilakukan SMOTE *oversampling* dengan perbandingan data latih dan data uji, yaitu 90:10 dan nilai K=3. Dari pengujian tersebut dihasilkan nilai akurasi, presisi, *recall*, dan *f1-score* yang diterangkan pada Gambar 10.

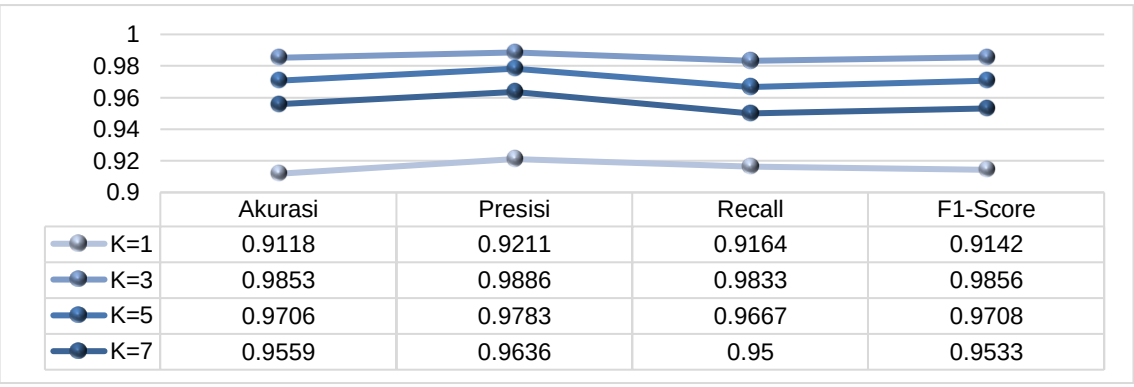


	Akurasi	Presisi	Recall	F1-Score
Sebelum Normalisasi Data	0.9853	0.9886	0.9833	0.9856
Sesudah Normalisasi Data	0.8676	0.871	0.8701	0.8701

Gambar 10. Hasil Uji sebelum dan sesudah Normalisasi Data

c. Pengujian Model Berdasarkan Nilai K

Pengujian dilakukan menggunakan data yang sudah dilakukan SMOTE *oversampling* dan tidak dinormalisasi dengan perbandingan data latih dan data uji, yaitu 90:19. Nilai K yang diuji adalah 1, 3, 5, dan 7. Dari pengujian tersebut dihasilkan nilai akurasi, presisi, *recall*, dan *f1-score* yang diterangkan pada Gambar 11.

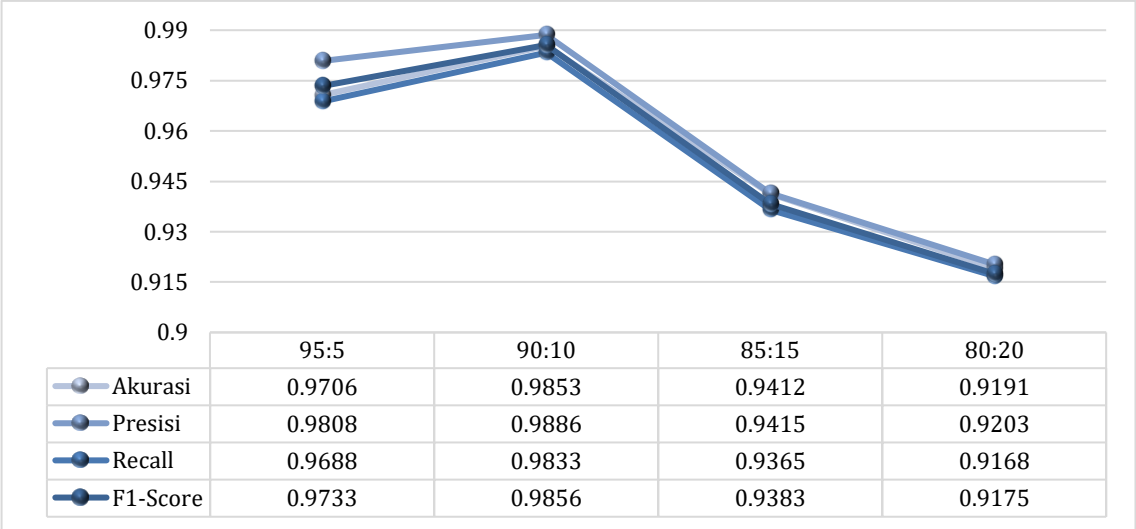


	Akurasi	Presisi	Recall	F1-Score
K=1	0.9118	0.9211	0.9164	0.9142
K=3	0.9853	0.9886	0.9833	0.9856
K=5	0.9706	0.9783	0.9667	0.9708
K=7	0.9559	0.9636	0.95	0.9533

Gambar 11. Hasil Uji Berdasarkan Nilai K

d. Pengujian dengan Perbandingan Data Latih Dan Data Uji

Pengujian dilakukan menggunakan data yang sudah dilakukan SMOTE *oversampling* dan tidak dinormalisasi dengan K=3, serta perbandingan data latih dan data uji, yaitu 95:5, 90:10, 85:15, dan 80:20. Dari pengujian tersebut dihasilkan nilai akurasi, presisi, *recall*, dan *f1-score* yang diterangkan pada Gambar 12.



	95:5	90:10	85:15	80:20
Akurasi	0.9706	0.9853	0.9412	0.9191
Presisi	0.9808	0.9886	0.9415	0.9203
Recall	0.9688	0.9833	0.9365	0.9168
F1-Score	0.9733	0.9856	0.9383	0.9175

Gambar 12. Hasil Uji Perbandingan Data

3.3. Analisis Hasil Pengujian Model

Hasil pengujian model berdasarkan skenario yang dilakukan dalam penelitian ini, sebagai berikut:

a. Skenario Pengujian Pertama

Membandingkan data sebelum dan sesudah SMOTE *oversampling* dari data yang tidak dinormalisasi dengan perbandingan data latih dan data uji, yaitu 90:10 dan nilai K=3. Dalam skenario ini, terjadi peningkatan signifikan pada semua metrik evaluasi yang digunakan setelah menggunakan SMOTE *oversampling*. Akurasi meningkat dari 0,925 menjadi 0,9853, presisi meningkat dari 0,9605 menjadi 0,9886, *recall* meningkat dari 0,9583 menjadi 0,9833, dan *f1-score* meningkat dari 0,9558 menjadi 0,9856. Secara keseluruhan, dengan menggunakan SMOTE *oversampling*, model dapat mengatasi ketidakseimbangan kelas pada dataset dan meningkatkan kemampuan klasifikasinya terhadap kelas minoritas.

b. Skenario Pengujian Kedua

Membandingkan data sebelum dan sesudah normalisasi dari data yang sudah dilakukan SMOTE *oversampling* dengan perbandingan data latih dan data uji, yaitu 90:10 dan nilai K=3. Dalam skenario ini, terjadi penurunan signifikan pada semua metrik evaluasi yang digunakan setelah normalisasi. Akurasi menurun dari 0,9853 menjadi 0,8676, presisi menurun dari 0,9886 menjadi 0,871, *recall* menurun dari 0,9833 menjadi 0,8701, dan *f1-score* menurun dari 0,9856 menjadi 0,8701. Secara keseluruhan, normalisasi mengakibatkan model menjadi kurang efektif dalam mengklasifikasikan dan mengenali kelas minoritas.

c. Skenario Pengujian Ketiga

Membandingkan data yang sudah dilakukan SMOTE *oversampling* dan tidak dinormalisasi dengan perbandingan data latih dan data uji, yaitu 90:10. Nilai K yang digunakan yaitu 1, 3, 5, dan 7. Hasil pengujian menunjukkan variasi performa model berdasarkan nilai K. Dengan K=1, diperoleh akurasi 0,9118, presisi 0,9211, *recall* 0,9164, dan *f1-score* 0,9142. Model dengan K=1 memiliki performa baik dengan presisi dan *recall* yang seimbang. Dengan K=3, terjadi peningkatan performa dengan akurasi 0,9853, presisi 0,9886, *recall* 0,9833, dan *f1-score* 0,9856. Model K=3 memberikan performa sangat baik dengan tingkat akurasi, presisi, *recall*, dan *f1-score* yang tinggi. Dengan K=5, terjadi sedikit penurunan

performa dengan akurasi 0,9706, presisi 0,9783, *recall* 0,9667, dan *f1-score* 0,9708. Namun, performa masih baik dan mendekati K=3. Dengan K=7, terjadi penurunan lebih lanjut pada performa dengan akurasi 0,9559, presisi 0,9636, *recall* 0,95, dan *f1-score* 0,9533. Pada K=7, model memiliki performa lebih rendah dibandingkan dengan K yang lebih kecil. Secara keseluruhan, K=3 memberikan performa terbaik dengan akurasi, presisi, *recall*, dan *f1-score* yang tinggi. Nilai K yang terlalu rendah atau tinggi dapat mengakibatkan penurunan performa.

d. Skenario Pengujian Keempat

Mengatur perbandingan antara data latih dan data uji yang menggunakan data yang sudah dilakukan SMOTE *oversampling* dan tidak dinormalisasi dengan K=3. Perbandingan data latih dan data uji yang digunakan, yaitu 95:5, 90:10, 85:15, dan 80:20. Hasil pengujian menunjukkan variasi performa model berdasarkan perbandingan data latih dan data uji. Dengan perbandingan 95:5, diperoleh akurasi 0,9706, presisi 0,9808, *recall* 0,9688, dan *f1-score* 0,9733. Model memiliki performa baik dalam mengklasifikasikan sampel dengan perbandingan yang tidak seimbang. Dengan perbandingan 90:10, terjadi peningkatan performa dengan akurasi 0,9853, presisi 0,9886, *recall* 0,9833, dan *f1-score* 0,9856. Model memiliki performa sangat baik dengan perbandingan yang lebih seimbang. Dengan perbandingan 85:15, terjadi penurunan performa dengan akurasi 0,9412, presisi 0,9415, *recall* 0,9365, dan *f1-score* 0,9383. Meskipun terjadi penurunan, performa model masih cukup baik dengan perbandingan yang tidak seimbang. Dengan perbandingan 80:20, terjadi penurunan lebih lanjut pada performa dengan akurasi 0,9191, presisi 0,9203, *recall* 0,9168, dan *f1-score* 0,9175. Pada perbandingan yang lebih tidak seimbang, model cenderung memiliki performa lebih rendah. Secara keseluruhan, perbandingan 90:10 memberikan performa terbaik dengan akurasi, presisi, *recall*, dan *f1-score* yang tinggi.

4. Kesimpulan

Penelitian ini menggunakan K-NN dalam mengklasifikasikan tingkat kerontokan rambut dengan 4 skenario pengujian untuk menentukan nilai akurasi, presisi, *recall*, dan *f1-score*. Performa terbaik diperoleh saat data di-SMOTE *oversampling* dan tidak dinormalisasi, dengan perbandingan data 90:10 dan K=3. Pada kondisi ini, diperoleh akurasi sebesar 0,9853, presisi 0,9886, *recall* 0,9833, dan *f1-score* 0,9856. Untuk menghasilkan hasil klasifikasi yang lebih baik dan lebih berkualitas perlu dilakukan penelitian lebih lanjut seperti eksplorasi teknik *oversampling*, pemilihan metode normalisasi yang lebih sesuai, penentuan nilai K dan perbandingan data yang lebih optimal, melakukan evaluasi tambahan seperti kurva ROC, melibatkan validasi silang untuk menghindari bias, serta eksplorasi metode dan algoritma yang berbeda untuk membandingkan sehingga diperoleh model dengan performa terbaik.

Daftar Pustaka

[1] B. Harris, “KERONTOKAN DAN KEBOTAKAN PADA RAMBUT HAIR LOSS AND ALOPECIA,” *Ibnu Sina: Jurnal Kedokteran dan Kesehatan-Fakultas KedokteranUniversitas Islam Sumatera Utara*, vol. 20, no. 2, hlm. 159–168, 2021.

[2] W. N. Suhery, M. Febrina, dan I. Permatasari, “Formulasi Mikroemulsi dari Kombinasi Minyak Kelapa Murni (Virgin Coconut Oil) dan Minyak Dedak Padi (Rice Bran Oil) Sebagai Penyubur Rambut,” *Traditional Medicine Journal*, vol. 23, no. 1, hlm. 40–46, Apr 2018.

[3] L. Sulastri, S. Asih, dan R. Amelia, “UJI AKTIVITAS PENYUBUR RAMBUT EMULGEL EKSTRAK ETANOL BUAH CABAI GENDOT (Capsicum annum Var.Abbreviata) PADA MENCIT PUTIH (Mus musculus) JANTAN,” *Medical Sains*, vol. 4, no. 2, hlm. 101–110, Mar 2020.

[4] E. Collins, Rollando, dan E. Monica, “Pembuatan Serum Penumbuh Rambut Kombinasi Minyak Kemiri (Aleurites moluccanus) dan Ekstrak Buah Apel (Pyrus malus L.),” *Jurnal Farmasi Ma Chung: Sains Teknologi dan Klinis Komunitas*, vol. 1, no. 1, hlm. 32–41, 2023.

[5] Magfirah, M. H. Angka, dan Rizka, “Pemanfaatan Kembang Sepatu sebagai Shampo untuk Perawatan Rambut Rontok,” *Jurnal Pengabdian Kepada Masyarakat (DiMas)*, vol. 4, no. 1, hlm. 25–28, 2022, doi: 10.53359/dimas.v4i1.37.

[6] L. Hakim, “PENGENALAN EMOSI PADA MANULA BERBASIS SINYAL SPO2 DAN PULSE RATE MENGGUNAKAN METODE SUPPORT VECTOR MACHINE,” Magister, Institut Teknologi Sepuluh Nopember, Surabaya, 2018.

[7] H. W. Dhany, “Performa Algoritma K-Nearest Neighbour dalam Memprediksi Penyakit Jantung,” dalam *Seminar Nasional Informatika (SENATIKA)*, 2021, hlm. 176–179. [Daring]. Tersedia pada: https://www.kaggle.com/

[8] R. D. Y. Prakoso, B. S. Wiriaatmadja, dan F. W. Wibowo, “Sistem Klasifikasi Pada Penyakit Parkinson Dengan Menggunakan Metode K-Nearest Neighbor,” dalam *Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)*, 2020, hlm. 63–68.

[9] A. G. Pertiwi, N. Bachtiar, R. Kusumaningrum, I. Waspada, dan A. Wibowo, “Comparison of performance of k-nearest neighbor algorithm using smote and k-nearest neighbor algorithm without smote in diagnosis of diabetes disease in balanced data,” dalam *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jun 2020, hlm. 1–8. doi: 10.1088/1742-6596/1524/1/012048.

[10] Luke X, “Luke Hair Loss Dataset,” *Kaggle.com*, 2022. https://www.kaggle.com/datasets/lukexun/luke-hair-loss-dataset (diakses 15 April 2023).

[11] G. Gumelar *dkk.*, “Kombinasi Algoritma Sampling dengan Algoritma Klasifikasi untuk Meningkatkan Performa Klasifikasi Dataset Imbalance,” dalam *Prosiding Seminar Nasional Sistem Informasi dan Teknologi (SISFOTEK)*, 2021, hlm. 250–255.

[12] Nurahman dan J. Susanto, “Klasterisasi Data Penerima Bantuan Langsung Tunai Menggunakan Algoritma K-Means,” *JURIKOM (Jurnal Riset Komputer)*, vol. 10, no. 2, hlm. 461–470, Apr 2023, doi: 10.30865/jurikom.v10i2.5807.

[13] M. Nishom, “Perbandingan Akurasi Euclidean Distance, Minkowski Distance, dan Manhattan Distance pada Algoritma K-Means Clustering berbasis Chi-Square,” *Jurnal Informatika: Jurnal Pengembangan IT (JPIT)*, vol. 4, no. 1, hlm. 20–24, Jan 2019, doi: 10.30591/jpit.v4i1.1253.

[14] N. Nafi’ Dzikrulloh, Indriati, dan B. D. Setiawan, “Penerapan Metode K-Nearest Neighbor (KNN) dan Metode Weighted Product (WP) Dalam Penerimaan Calon Guru Dan Karyawan Tata Usaha Baru Berwawasan Teknologi (Studi Kasus : Sekolah Menengah Kejuruan Muhammadiyah 2 Kediri),” *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 1, no. 5, hlm. 378–385, 2017, [Daring]. Tersedia pada: http://j-ptiik.ub.ac.id

[15] W. Yustanti, “Algoritma K-Nearest Neighbour untuk Memprediksi Harga Jual Tanah,” *Jurnal Matematika, Statistika, & Komputasi*, vol. 9, no. 1, hlm. 57–68, Jul 2012.