Jurnal Nasional Teknomedik | Vol. 1, No. 2, 27 Des 2024
Submitted: 10-09-2024, Accepted 10-12-2024, Published: 27-12-2024
ISSN 3062-8474 (media online)
Online at https://ejournal.umw.ac.id

Studi Komparatif Nilai K Pada Algoritma K-NN Untuk Klasifikasi Status Gizi Menggunakan Manhattan Distance

La Ode Hasnuddin S. Sagala[1], Desak Ketut Sutiari[2]
[1]Universitas Sembilanbelas November Kolaka
[2]Universitas Mandala Waluya

Corresponding author: (e-mail:hasnuddin.sagala@)gmail.com)

Intisari— Status gizi merupakan salah satu indikator yang dapat digunakan untuk mengetahui kondisi kesehatan dan perkembangan individu, terutama balita dan anak-anak. Salah satu algoritma yang sering digunakan dalam penentuan status gizi adalah K-Nearest Neighbor. Namun, dalam penerapan algoritma ini, pemilihan nilai K yang tepat sangat mempengaruhi akurasi proses klasifikasi. Pada penelitian ini diperoleh bahwa nilai K=3 memberikan hasil akurat lebih baik dibandingkan nilai K yang lain. Semakin tinggi nilai K maka keakuratan klasifikasi akan semakin menurun. Untuk penelitian berikutnya, perlu adanya penggunaan data yang banyak dan perbedaan komposisi antara training data dengan testing data.

Kata Kunci : k-NN, Manhattan Distance, Klasifikasi, Status Gizi

I. PENDAHULUAN

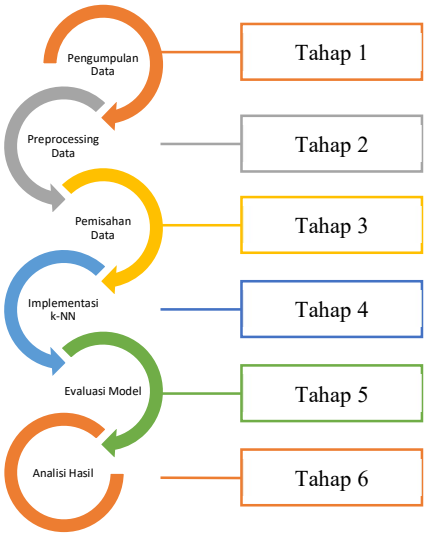
Status gizi adalah indikator penting dalam menentukan kesehatan dan perkembangan individu, terutama pada anak-anak dan remaja[1][2]. Klasifikasi status gizi yang akurat sangat penting untuk mengidentifikasi individu yang berisiko mengalami malnutrisi, baik gizi kurang maupun gizi lebih[3]. Salah satu metode yang sering digunakan untuk klasifikasi status gizi pada balita adalah algoritma K-Nearest Neighbor (K-NN)[4][5][6][7]. Algoritma ini dikenal karena kesederhanaannya namun efektif dalam melakukan klasifikasi berdasarkan kedekatan jarak antara data uji dan data latih.

Dalam penerapan algoritma K-NN, pemilihan nilai K yang tepat sangat krusial. Nilai K yang terlalu kecil atau terlalu besar dapat mempengaruhi akurasi klasifikasi secara signifikan. Nilai K yang kecil cenderung membuat model terlalu sensitif terhadap noise dalam data, sehingga bisa menghasilkan klasifikasi yang tidak akurat. Sebaliknya, nilai K yang besar dapat membuat model terlalu umum, sehingga kehilangan kemampuan untuk menangkap pola-pola penting dalam data.

Selain pemilihan nilai K, metrik jarak yang digunakan dalam algoritma K-NN juga memiliki dampak besar terhadap hasil klasifikasi. *Euclidean distance* adalah metrik jarak yang paling umum digunakan, namun metrik ini sangat sensitif terhadap outlier[8][9]. Oleh karena itu, Manhattan distance sering digunakan sebagai alternatif. Manhattan distance menghitung jarak absolut antara titik-titik data, sehingga lebih robust terhadap outlier dan dapat memberikan hasil yang lebih baik dalam beberapa kasus. Oleh karena itu, rumusan masalahnya adalah bagaimana pengaruh variasi nilai K pada algoritma K-NN terhadap akurasi klasifikasi status gizi menggunakan *Manhattan Distance*.

II. METODE PENELITIAN

Penentuan nilai K pada saat proses klasifikasi memberikan pengaruh yang besar terhadap akurasi k-NN dalam mengidentifikasi nilai status gizi balita. Gambar 1 menunjukkan tahap-tahap penelitian ini.



Gambar 1. Tahapan Penelitian

2.1 Pengumpulan Data

Pada penelitian ini diawali dengan proses pengumpulan data. Peneliti menggunakan data dari website Kaggle.com, yang dimana di dalam data tersebut terdapat beberapa kriteria yang digunakan dalam menentukan status gizi pada balita. Indikator keberhasilan dalam proses ini adalah diperoleh data yang lengkap dan valid dari sumber terpercaya

2.2 Preprocessing Data

Setelah data terkumpul, dilanjutkan dengan proses Preprocessing Data. Proses ini bertujuan untuk membersihkan data dari missing values, outliers, dan melakukan proses normalisasi. Indikator keberhasilan dalam proses ini adalah dataset yang bersih dan siap digunakan untuk pemodelan.

2.3 Pemisahan Data

Data yang siap digunakan untuk pemodelan, akan dibagi menjadi dua bagian. Bagian yang pertama akan dijadikan sebagai *training data* atau data latih dan bagian yang kedua dijadikan sebagai data uji atau *testing data*. Komposisi antara *training data* dengan *testing data* adalah 80%:20%. Indikator keberhasilan dalam proses ini adalah pemisahan data yang seimbang dan representative.

2.4 Implementasi k-NN

Pada proses ini, algoritma K-NN akan diimplementasikan ke dataset (*training data* & *testing data*). Pada umumnya proses untuk mencari jarak antara *testing data* dengan *training data* adalah menggunakan *Euclidean Distance*. Namun penelitian ini akan menggunakan *Manhattan Distance* sebagai teknik mengukur kedekatan data. Indikator keberhasilan dalam proses ini adalah algoritma K-NN berjalan dengan baik dan dapat menguji berbagai nilai K.

2.5 Evaluasi Model

Tahap ini bertujuan untuk mengevaluasi model yang dihasilkan karena perbedaan nilai K yang digunakan. Metode evaluasi yang dipakai adalah Confusion Matrix dan Nilai K yang digunakan adalah 3,5,7,9,11,13,15,17,19, dan 21. Indikator keberhasilan dalam proses ini adalah peforma model yang terukur dengan jelas untuk setiap nilai K.

2.6 Analisis Hasil

Tahap ini merupakan tahap terakhir dalam penelitian ini. Tahap ini bertujuan untuk menganalisis hasil evaluasi dalam menentukan nilak K terbaik. Indikator keberhasilan dalam tahap ini adalah identifikasi nilai K yang optimal dalam memberikan peforma yang lebih baik.

III. HASIL DAN PEMBAHASAN

Pada penelitian ini, bahasa pemrograman yang digunakan adalah Bahasa Python. Langkah pertama yang dilakukan adalah mengimport beberapa library python yang akan digunakan dalam proses analisis. Gambar 2 menunjukkan beberapa *library* yang dipakai.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
import matplotlib.pyplot as plt
```

Gambar 2. *Library* python yang dipakai

Setelah mengimport beberapa library yang dibutuhkan, langkah berikutnya adalah pengumpulan data status gizi yang akan digunakan. Penelitian ini menggunakan data yang diambil dari website Kaggle.com. Data yang diperoleh, kemudian dibaca dan dicek menggunakan bahasa Python. Gambar 3A menunjukkan *source code* pembacaan dataset yang akan digunakan sedangkan Gambar 3B merupakan hasil *source code*.

```
# Misalkan dataset diambil dari file CSV
data = pd.read_csv('dataku.csv')

# Memeriksa beberapa baris pertama dari dataset
data.head()
```

Gambar 3A. Source code pembacaan dataset

	NAMA BALITA	JENIS KELAMIN	UMUR (BULAN)	BERAT(KG)	TINGGI (CM)	STATUS GIZI
0	Adhe Fitri	P	24	5.8	65.0	GIZI_KURANG
1	Andi Hariati	P	24	5.5	59.0	GIZI_NORMAL
2	Anwar Amir	L	28	6.7	71.5	GIZI_KURANG
3	Asmar	L	30	8.1	72.5	GIZI_NORMAL
4	Eka Andriyani	P	28	6.9	73.0	GIZI_KURANG

Gambar 3B. Hasil source code

Setelah proses pembacaan dataset, dilanjutkan proses *Preprocessing data*. Terdapat 3 (tiga) bentuk yang dilakukan dalam proses ini yaitu menghapus *missing values*, memilih fitur dan label, serta melakukan Normalisasi data. Tujuan *missing values* adalah menghapus data data yang masuk kategori data tidak lengkap. Tujuan pemilihan fitur dan label adalah memberi tanda ke data yang masuk dalam kategori fitur dan yang masuk dalam kategori label. Sedangkan yang terakhir yaitu proses normalisasi, bertujuan untuk menyeragamkan nilai dari data yang dikumpulkan agar mudah diolah.

```python
# Menghapus missing values
data = data.dropna()

# Memilih fitur dan label
features = data[['UMUR (BULAN)', 'BERAT(KG)', 'TINGGI (CM)']]
labels = data['STATUS GIZI']

# Normalisasi data
features = (features - features.min()) / (features.max() - features.min())
print (features)
```

Gambar 4A. Source code preprocessing data

	UMUR (BULAN)	BERAT(KG)	TINGGI (CM)
0	0.40	0.369231	0.326531
1	0.40	0.346154	0.204082
2	0.48	0.438462	0.459184
3	0.52	0.546154	0.479592
4	0.48	0.453846	0.489796
..	...	...	...
195	0.64	0.430769	0.306122
196	0.68	0.615385	0.551020
197	0.76	0.846154	0.530612
198	0.80	0.769231	0.510204
199	0.86	0.692308	0.489796

[200 rows x 3 columns]

Gambar 4B. Hasil *source code*

Langkah berikutnya adalah proses pemisahan data. Proses ini bertujuan untuk memecah dataset menjadi dua bagian yaitu *training* data dan *testing* data. Pada penelitian ini, komposisi yang digunakan adalah 80% untuk training data dan 20% untuk testing data. Gambar 5 menunjukkan proses pemecahan dataset.

```python
# Memisahkan data menjadi data pelatihan dan data pengujian
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.2, random_state=42)
```

Gambar 5. Proses pemisahan dataset

Testing data yang telah dibuat pada proses sebelumnya, kemudian akan diimplementasikan kedalam proses algoritma K-Nearest Neoghbor (K-NN). Pada proses ini, nilai K akan divariasi yaitu 3,5,7,9,11,13,15,17,19, dan 21. Sedangkan untuk menentukan jarak terdekat antara data yaitu menggunakan metode *Manhattan Distance*. Gambar 6 menunjukkan proses implementasi data ke dalam algoritma K-NN serta menggunakan *Manhattan Distance*.

```python
k_values = range(3, 21)
accuracies = []
precisions = []
recalls = []
f1_scores = []

for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k, metric='manhattan')
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)

    accuracies.append(accuracy_score(y_test, y_pred))
    precisions.append(precision_score(y_test, y_pred, average='macro'))
    recalls.append(recall_score(y_test, y_pred, average='macro'))
    f1_scores.append(f1_score(y_test, y_pred, average='macro'))
```

Gambar 6. Implementasi k-NN

Evaluasi model merupakan langkah terakhir pada penelitian ini. Tahap ini akan menggunakan metode *Confussion Matrix* untuk mengetahui *Accuracy*, *Precision*, *Recall,* dan *F1-score*. Gambar 7A menunjukkan *source code* yang digunakan dan Gambar 7B merupakan hasil *source code* dengan memvariasikan nilai K.

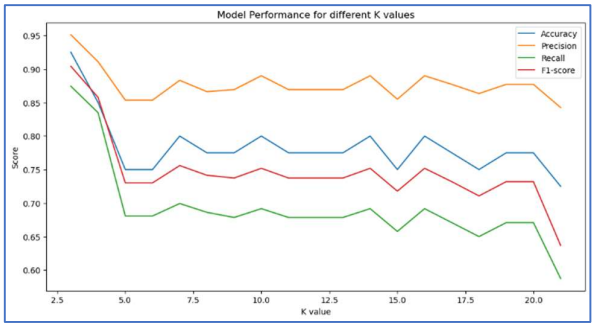
```python
# Menampilkan hasil evaluasi untuk berbagai nilai K
for i, k in enumerate(k_values):
    print(f'K={k}: Accuracy={accuracies[i]}, Precision={precisions[i]}, Recall={recalls[i]}, F1-score={f1_scores[i]}')

# Menampilkan confusion matrix untuk nilai K terbaik
best_k = k_values[np.argmax(accuracies)]
knn = KNeighborsClassifier(n_neighbors=best_k, metric='manhattan')
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)

conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix for K={best_k}:\n{conf_matrix}')
```

Gambar 7A. *Source code* evaluasi model

```
K=3: Accuracy=0.925, Precision=0.9511278195488722, Recall=0.8743421052631578, F1-score=0.9041192725403251
K=4: Accuracy=0.85, Precision=0.9108455882352942, Recall=0.8348684210526316, F1-score=0.8576902025177887
K=5: Accuracy=0.75, Precision=0.8536184210526316, Recall=0.680701754385965, F1-score=0.7301065162907268
K=6: Accuracy=0.75, Precision=0.8536184210526316, Recall=0.680701754385965, F1-score=0.7301065162907268
K=7: Accuracy=0.8, Precision=0.8831521739130435, Recall=0.6993421052631579, F1-score=0.7559523809523809
K=8: Accuracy=0.775, Precision=0.8662587412587412, Recall=0.6861842105263158, F1-score=0.7414837398373983
K=9: Accuracy=0.775, Precision=0.8693181818181819, Recall=0.6785087719298245, F1-score=0.7373820357263229
K=10: Accuracy=0.8, Precision=0.8899999999999999, Recall=0.6916666666666667, F1-score=0.7518393939393393
K=11: Accuracy=0.775, Precision=0.8693181818181819, Recall=0.6785087719298245, F1-score=0.7373820357263229
K=12: Accuracy=0.775, Precision=0.8693181818181819, Recall=0.6785087719298245, F1-score=0.7373820357263229
K=13: Accuracy=0.775, Precision=0.8693181818181819, Recall=0.6785087719298245, F1-score=0.7373820357263229
K=14: Accuracy=0.8, Precision=0.8899999999999999, Recall=0.6916666666666667, F1-score=0.7518393939393393
K=15: Accuracy=0.75, Precision=0.855, Recall=0.6576754385964912, F1-score=0.7178030303030303
K=16: Accuracy=0.8, Precision=0.8899999999999999, Recall=0.6916666666666667, F1-score=0.7518393939393393
K=17: Accuracy=0.775, Precision=0.8771367521367521, Recall=0.6708333333333334, F1-score=0.7319444444444443
K=18: Accuracy=0.75, Precision=0.8634259259259259, Recall=0.65, F1-score=0.7106884057971015
K=19: Accuracy=0.775, Precision=0.8771367521367521, Recall=0.6708333333333334, F1-score=0.7319444444444443
K=20: Accuracy=0.775, Precision=0.8771367521367521, Recall=0.6708333333333334, F1-score=0.7319444444444443
K=21: Accuracy=0.725, Precision=0.8425925925925926, Recall=0.5875, F1-score=0.6368788819875777
Confusion Matrix for K=3:
[[ 3  1  0  0]
 [ 0 12  0  0]
 [ 0  0  4  1]
 [ 0  1  0 18]]
```

Gambar 7B. Hasil *source code*

Setelah didapatkan nilai *accuracy*, *precision*, *recall*, dan *F1-Score* setiap nilai K. Hasil tersebut di visualisasikan dalam bentuk grafik perbandingan seperti Gambar 8.



Gambar 8. Grafik dengan memvariasikan nilai K

Pada grafik terlihat bahwa nilai K = 3 memberikan peforma klasifikasi yang lebih baik dibandingan nilai K yang lain. Semakin tinggi nilai K maka hasil peforma klasifikasi yang diberikan semakin menurun.

IV. KESIMPULAN

Berdasarkan hasil analisis menunjukkan bahwa dengan menggunakan nilai K=3 dapat memberikan hasil peforma yang baik pada klasifikasi status gizi balita. Namun hal terbalik terjadi jika semakin tinggi nilai K-nya maka hasil peforma klasifikasi semakin menurun. Kedepannya penelitian ini dapat ditinjau dari pengaruh banyaknya data yang digunakan dan perbandingan komposisi *training data* dengan *testing data*.

REFERENSI

[1] D. Capriani and A. F. Jamir, "Hubungan Status Gizi Dengan Perkembangan Motoric Kasar Anak Usia Prasekolah Di TK Idhata Sangalla," *Media Publ. Penelit. Kebidanan*, vol. 6, no. 1, pp. 93–99, 2023, [Online]. Available: https://institutgrahaananda.ac.id/jurnal/index.php/mppk/article/view/83

[2] R. T. Haryanti, T. Susilowati, and I. M. Sari, "Hubungan Intensitas Penggunaan Gadget terhadap Status Gizi pada Siswa SMK Batik 2 Surakarta," *ASJN (Aisyiyah Surakarta J. Nursing)*, vol. 3, no. 1, pp. 27–33, 2022, doi: 10.30787/asjn.v3i1.897.

[3] A. Amirullah, A. T. Andreas Putra, and A. A. Daud Al Kahar, "Deskripsi Status Gizi Anak Usia 3 Sampai 5 Tahun Pada Masa Covid-19," *Murhum J. Pendidik. Anak Usia Dini*, vol. 1, no. 1, pp. 16–27, 2020, doi: 10.37985/murhum.v1i1.3.

[4] N. Mutiara, S. Maylita, H. Z. Zahro', and N. Vendyansyah, "Penerapan Metode K-Nearest Neighbor (Knn) Untuk Menentukan Status Gizi Balita (Studi Kasus : Posyandu Ananda Kelurahan Langkai, Kota Palangka Raya, Kalimantan Tengah)," *J. Mhs. Tek. Inform.*, vol. 6, no. 2, pp. 1–5, 2022.

[5] S. Lonang, A. Yudhana, and M. K. Biddinika, "Rancangan Sistem Klasifikasi Kekurangan Gizi Balita Dengan Metode K-Nearest Neighbor," *J. Inform. dan Rekayasa Perangkat Lunak*, vol. 5, no. 1, p. 73, 2023, doi: 10.36499/jinrpl.v5i1.7834.

[6] H. Saleh, M. Faisal, and R. I. Musa, "Klasifikasi Status Gizi Balita Menggunakan Metode K-Nearest Neighbor," *Simtek J. Sist. Inf. dan Tek. Komput.*, vol. 4, no. 2, pp. 120–126, 2019, doi: 10.51876/simtek.v4i2.60.

[7] M. Annabaa' Aulia, R. Goejantoro, and M. N. Hayati, "Penerapan Metode Klasifikasi K-Nearest Neighbor (Studi Kasus : Data Status Gizi Balita di Puskesmas Baqa Samarinda Seberang)," pp. 128–142, 2023.

[8] K. W. Pakuani and R. Kurniawan, "Kajian Penentuan Nilai Epsilon Optimal Pada Algoritma DMDBSCAN Dan Pemetaan Daerah Rawan Gempa Bumi Di Indonesia Tahun 2014-2020," *Semin. Nas. Off. Stat.*, vol. 2021, no. 1, pp. 991–1000, 2021, doi: 10.34123/semnasoffstat.v2021i1.847.

[9] Z. R. Fadilah and A. W. Wijayanto, "Perbandingan Metode Klasterisasi Data Bertipe Campuran: One-Hot-Encoding, Gower Distance, dan K-Prototype Berdasarkan Akurasi (Studi Kasus: Chronic Kidney Disease Dataset)," *J. Appl. Informatics Comput.*, vol. 7, no. 1, pp. 57–67, 2023, doi: 10.30871/jaic.v7i1.5857.