Contents lists available at MID Publisher International

Technology Sciences Insights Journal

Journal homepage: https://journal.midpublisher.com/index.php/tsij

PENERAPAN ALGORITMA K-NEAREST NEIGHBOR (K-NN) UNTUK KLASIFIKASI Status Monitoring Automatic Pump Water Machine STUDI KASUS : INDUSTRI MANUFAKTUR

Adlian Jefiza, Indra Mora
Politeknik Negeri Batam, Batam, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima: Juni 2025
Revisi: Juni 2025
Diterima: Juli 2025
Dipublikasi: Juli 2025

Kata Kunci:
Sistem Industri, Klasifikasi, K-Nearest Neighbor (KNN)

*Penulis Korespondensi:
indramora2002@gmail.com

Keyword:
Industri System, Classification, K-Nearest Neighbor (KNN)

ABSTRAK

Dalam dunia industri modern, pemantauan kondisi sistem secara real-time menjadi hal yang krusial untuk menjaga efisiensi dan mencegah kerusakan peralatan. Penelitian ini bertujuan untuk mengklasifikasikan kondisi sistem industri berdasarkan data sensor menggunakan algoritma K-Nearest Neighbors (KNN). Data yang digunakan terdiri dari empat parameter utama yaitu tekanan (pressure), laju aliran (flow rate), tegangan (voltage), dan putaran mesin (RPM), yang kemudian diklasifikasikan ke dalam tiga kondisi: Alert, Critical, dan Normal. Proses preprocessing dilakukan dengan normalisasi Min-Max dan pembagian data menjadi data latih dan uji. Hasil evaluasi menunjukkan bahwa metode KNN mampu mencapai akurasi sebesar 58% dengan nilai mean squared error (MSE) sebesar 1.06 dan rata-rata akurasi validasi silang sebesar 64%. Hasil ini menunjukkan bahwa KNN cukup efektif digunakan sebagai metode awal untuk deteksi kondisi sistem industri, meskipun performa klasifikasi untuk kategori Critical masih dapat ditingkatkan.

ABSTRACT

In the modern industrial world, real-time monitoring of system conditions is crucial to maintain efficiency and prevent equipment damage. This research aims to classify industrial system conditions based on sensor data using the K-Nearest Neighbors (KNN) algorithm. The data used consists of four main parameters namely pressure, flow rate, voltage, and engine speed (RPM), which are then classified into three conditions: Alert, Critical, and Normal. Preprocessing is done with Min-Max normalization and division of data into training and test data. The evaluation results show that the KNN method is able to achieve an accuracy of 58% with a mean squared error (MSE) value of 1.06 and an average cross-validation accuracy of 64%. These results show that KNN is effective enough to be used as an initial method for industrial system condition detection, although the classification performance for the Critical category can still be improved.

PENDAHULUAN

Perkembangan industri modern menuntut adanya sistem yang andal dan efisien dalam mendeteksi kondisi operasional peralatan secara otomatis. Sistem yang tidak terpantau dengan baik dapat mengalami kerusakan tiba-tiba, menyebabkan penurunan produktivitas, biaya perbaikan yang tinggi, bahkan potensi kecelakaan kerja.

Dengan kemajuan teknologi sensor, data seperti tekanan, aliran fluida, tegangan, dan kecepatan putaran mesin kini dapat dikumpulkan secara real-time. Namun, interpretasi data dalam jumlah besar memerlukan pendekatan analitik yang cerdas. Salah satu solusi yang banyak digunakan dalam pengolahan data sensor adalah penerapan algoritma kecerdasan buatan (AI),

khususnya metode klasifikasi.

K-Nearest Neighbors (KNN) adalah algoritma klasifikasi yang sederhana namun efektif. KNN bekerja dengan cara mencari sejumlah tetangga terdekat dari suatu data baru berdasarkan jarak fitur, kemudian menentukan kelas mayoritas dari tetangga-tetangga tersebut. Algoritma ini tidak memerlukan pelatihan model secara eksplisit, sehingga cocok diterapkan pada sistem dengan struktur data yang dinamis dan kompleks seperti sistem industri.

Penelitian ini bertujuan untuk menerapkan algoritma KNN dalam mengklasifikasikan kondisi sistem industri berdasarkan empat parameter utama yaitu tekanan (pressure), laju aliran (flow rate), tegangan (voltage), dan kecepatan putaran mesin (RPM). Kondisi sistem dikategorikan ke dalam tiga kelas: Alert, Critical, dan Normal. Evaluasi kinerja model dilakukan dengan menggunakan metrik akurasi, confusion matrix, f1-score, mean squared error (MSE), serta validasi silang (cross-validation).

Dalam penelitian ini, algoritma K-Nearest Neighbors (KNN) digunakan sebagai metode klasifikasi utama. KNN merupakan algoritma yang bekerja berdasarkan prinsip kesamaan data, di mana data baru akan diklasifikasikan berdasarkan mayoritas kelas dari tetangga terdekatnya dalam ruang fitur. Algoritma ini memiliki keunggulan dalam hal kesederhanaan implementasi, tidak memerlukan pelatihan model yang kompleks, dan memiliki performa yang cukup baik dalam klasifikasi berbasis jarak.

Pemilihan KNN dalam penelitian ini didasarkan pada sifat data sensor industri yang bersifat numerik dan terstruktur, serta kebutuhan akan pendekatan klasifikasi yang cepat dan responsif. Evaluasi model dilakukan menggunakan beberapa metrik pengukuran performa seperti confusion matrix, precision, recall, f1-score, mean squared error (MSE), dan validasi silang (cross-validation).

Melalui penelitian ini, diharapkan dapat diperoleh gambaran yang jelas mengenai efektivitas metode KNN dalam mendeteksi kondisi sistem industri, serta menjadi referensi awal dalam pengembangan sistem pemantauan berbasis kecerdasan buatan untuk kebutuhan industri ke depan].

LANDASAN TEORI

A. Mesin Industri

Penerapan kecerdasan buatan dalam dunia industri telah mengalami perkembangan pesat, khususnya dalam hal pemantauan kondisi sistem dan prediksi kegagalan mesin. Sejumlah penelitian telah menunjukkan bahwa pengolahan data sensor secara cerdas dapat meningkatkan efektivitas pemeliharaan (predictive maintenance) dan mengurangi waktu henti produksi (downtime).

B. Klasifikasi

Klasifikasi adalah proses kategorisasi data dengan memasukkan data tersebut ke dalam kategori tertentu dari kategori yang tersedia [2]. Teknik dalam klasifikasi ialah melakukan kategorisasi data training untuk membuat suatu model untuk melakukan kategorisasi pada data testing yang tersedia.

C. Data Mining

Data Mining berasal dari dua kata bahasa inggris yaitu "data" yang berarti data dan "mining" yang berarti menambang. Sehingga data mining dapat diartikan sebagai proses penambangan data. Data mining merupakan metode untuk mendapatkan inti dari suatu ilmu atau pengetahuan [6]. Data mining menghasilkan teknik pengenalan pola data yang dapat memberikan perbedaan hasil dari data lain, sehingga dapat digunakan di masa yang akan datang [7].

D. Algoritma K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) adalah metode klasifikasi terhadap objek baru berdasarkan data training yang memiliki jarak tetangga terdekat (nearest neighbor) dengan objek baru tersebut

[8]. Dekat atau jauhnya neighbor biasanya dihitung berdasarkan jarak Euclidean. Berikut ini adalah langkah-langkah algoritma KNN:

1. Menentukan nilai K. Nilai K dapat dihitung menggunakan persamaan 1 berikut ini :

$$k = \sqrt{N} \quad (1)$$

N merupakan banyaknya sampel pada data training

2. Melakukan perhitungan nilai jarak (euclidean distance) terhadap masing-masing objek data yang diberikan. Rumus untuk menghitung euclidean distance dapat dilihat pada persamaan 2.

$$di = \sqrt{(xki - xkj)2 + (xki - xkj)2 + \ldots + (xki - xkj)2}(2) \quad (2)$$

Keterangan :
di = jarak euclidean
xki = data training ke-1
xkj = data testing ke-1

3. Melakukan pengelompokkan data sesuai dengan perhitungan jarak (Euclidean distance)
4. Melakukan pengelompokkan data sesuai dengan nilai tetangga terdekat (nearest neighbor) atau berdasarkan data yang mempunyai jarak Euclidean terkecil.
5. Memilih nilai mayoritas dari tetangga terdekat sebagai hasil klasifikasi.

E. Bahasa Pemrograman Pyhton

Dalam website docs.python.org, Python adalah bahasa pemrograman yang dapat digunakan untuk analisis data, mudah digunakan dan berorientasi objek. Python dapat digunakan dalam beberapa sistem operasi termasuk Linux dan macOS, dan Windows.

F. Confusion Matrix

Confusion Matrix mempresentasikan prediksi dengan membandingkan nilai asli dengan nilai hasil prediksi model untuk menghasilkan penilaian seperti berikut ini [9]:

1. Akurasi

Akurasi merupakan ketepatan antara nilai prediksi dengan nilai aktual. Rumus untuk menghitung nilai akurasi dapat dilihat pada persamaan 3.

$$Akurasi = \frac{(TP+TN)}{(TP+FP+FN+TN)} \quad (3)$$

2. Presisi

Presisi merupakan tingkat keberhasilan model dalam memberikan jawaban dengan tepat kepada pengguna. Rumus untuk menghitung nilai presisi dapat dilihat pada persamaan 4.

$$Presisi = \frac{TP}{TP+FP} \quad (4)$$

3. Recall

Recall merupakan tingkat keberhasilan model dalam menemukan kembali informasi dengan benar. Rumus untuk menghitung nilai recall dapat dilihat pada persamaan 5.

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

4. F1-Score atau F-Measure.

F1- Score merupakan hasil perbandingan antara nilai presisi dan recall. Rumus untuk menghitung nilai f1-Score dapat dilihat pada persamaan 6.

$$F1 - Score = \frac{2\ x\ Presisi\ x\ Recall}{Presisi+Recall} \quad (6)$$

Keterangan:
TP : True Positive
TN : True Negative
FP : False Positive
FN : False Negative.

METODE PENELITIAN

A. Tahapan Awal

Penelitian ini dilakukan dalam beberapa tahapan utama, yaitu pengumpulan data, preprocessing data, ekstraksi fitur, pemodelan dengan algoritma K-Nearest Neighbors (KNN), serta evaluasi performa model.

B. Tahap Pengumpulan Data

Pengumpulan data merupakan tahap awal yang krusial dalam penelitian ini karena kualitas data sangat mempengaruhi akurasi model klasifikasi. Data yang digunakan diperoleh dari sistem industri yang memantau performa mesin atau peralatan melalui sensor-sensor terintegrasi. Sensor tersebut merekam parameter operasional utama secara berkala dalam bentuk data numerik.

Adapun parameter yang dikumpulkan sebanyak 1000 data meliputi:

1. Pressure (tekanan): menunjukkan tekanan sistem dalam satuan tertentu,
2. FlowRate (laju aliran): menunjukkan seberapa cepat cairan/gas mengalir dalam sistem,
3. Voltage (tegangan): mengindikasikan kestabilan suplai listrik ke mesin,
4. RPM (Revolutions Per Minute): kecepatan putaran mesin yang menjadi indikator performa.

Pressure	FlowRate	Voltage	RPM	SystemCondition
0,047204161	0,137291863	0,818106818	0,712643528	Alert
0,054816988	0,018436635	0,82947704	0,720717158	Alert
0,451332772	0,001350802	0,908411093	0,725785928	Alert
0,075030517	0,128780504	0,806918537	0,738437669	Alert
0,827403202	0,011885335	0,091821096	0,526374717	Alert
0,525560068	0,015951864	0,903766983	0,656901113	Alert
0,075311426	0,010643892	0,091689883	0,451671286	Alert
0,007939661	0,130790012	0,855283579	0,503841212	Alert
0,071780814	0,197807219	0,917939108	0,585440052	Alert
0,065442157	0,175965997	0,784839413	0,775962433	Alert
0,082041839	0,107514941	0,082465627	0,542037259	Alert
0,069806186	0,042227357	0,805967509	0,630397661	Critical
0,050370006	0,159578625	0,859747993	0,805914921	Critical
0,06138949	0,058745411	0,008071371	0,578989447	Critical
0,052136123	0,001963547	0,080814949	0,096318224	Critical
0,052590751	0,139618225	0,843347698	0,079803958	Critical
0,91459327	0,203336024	0,845554824	0,810054239	Critical
0,007522326	0,01391461	0,094487919	0,537046	Critical
0,089955221	0,187842744	0,930342435	0,531014603	Critical
0,059596578	0,650072587	0,907036689	0,443530399	Critical
0,414618224	0,186923279	0,945411381	0,936078232	Critical
0,451031141	0,001413516	0,778543905	0,079608155	Critical
0,4442007	0,256813251	0,870842772	0,817594707	Critical
0,071289832	0,911190273	0,927064728	0,69072727	Critical

0,345172213	0,171963836	0,827120216	0,556325812	Normal
0,796236547	0,020141898	0,837785089	0,601620607	Normal
0,544697674	0,136225654	0,849784865	0,571688507	Normal
0,737877707	0,242452703	0,851435562	0,660222691	Normal
0,066396005	0,143094326	0,837534339	0,264795991	Normal
0,066709605	0,126436257	0,841122944	0,047289864	Normal
0,062408516	0,130536367	0,86129945	0,559378854	Normal
0,007792379	0,013092275	0,850284703	0,050609789	Normal
0,595434694	0,150200527	0,855717747	0,601889735	Normal
0,288503691	0,158075778	0,801714705	0,45089005	Normal
0,064120794	0,170220722	0,867369233	0,649449839	Normal
0,054526595	0,013581135	0,846805841	0,595162296	Normal
0,058017241	0,140603505	0,889013436	0,354722752	Normal
0,550576705	0,206488438	0,08585342	0,621212382	Normal
0,738371559	0,212131623	0,846101634	0,650557844	Normal
0,000115585	0,144783503	0,840454947	0,644771131	Normal

Tabel 1. Data Training Sebelum Normalisasi

Data dikumpulkan dalam bentuk file digital (.xlsx) dengan masing-masing baris merepresentasikan satu pengamatan pada waktu tertentu. Selain itu, setiap data disertai label kategori kondisi sistem, yaitu:

0 = Alert.

1 = Critical.

2 = Normal.

Label ini ditentukan berdasarkan standar ambang batas sensor atau hasil inspeksi teknisi. Proses labeling dilakukan secara manual pada tahap awal dan divalidasi oleh pihak teknis untuk menjamin akurasinya.

Setelah data terkumpul, dilakukan proses seleksi untuk memastikan tidak ada data yang kosong (missing values) atau duplikat. Hanya data yang lengkap dan valid yang digunakan dalam proses pemodelan selanjutnya.

Pressure	FlowRate	Voltage	RPM	SystemCondition
0,047204161	0,137291863	0,818106818	0,712643528	0
0,054816988	0,018436635	0,82947704	0,720717158	0
0,451332772	0,001350802	0,908411093	0,725785928	0
0,075030517	0,128780504	0,806918537	0,738437669	0
0,827403202	0,011885335	0,091821096	0,526374717	0
0,525560068	0,015951864	0,903766983	0,656901113	0
0,075311426	0,010643892	0,091689883	0,451671286	0
0,007939661	0,130790012	0,855283579	0,503841212	0
0,071780814	0,197807219	0,917939108	0,585440052	0
0,065442157	0,175965997	0,784839413	0,775962433	0
0,082041839	0,107514941	0,082465627	0,542037259	0
0,057746975	0,193745984	0,859982877	0,765062547	0
0,069806186	0,042227357	0,805967509	0,630397661	1
0,050370006	0,159578625	0,859747993	0,805914921	1
0,06138949	0,058745411	0,008071371	0,578989447	1
0,052136123	0,001963547	0,080814949	0,096318224	1
0,052590751	0,139618225	0,843347698	0,079803958	1
0,91459327	0,203336024	0,845554824	0,810054239	1
0,007522326	0,01391461	0,094487919	0,537046	1
0,089955221	0,187842744	0,930342435	0,531014603	1
0,059596578	0,650072587	0,907036689	0,443530399	1
0,414618224	0,186923279	0,945411381	0,936078232	1
0,451031141	0,001413516	0,778543905	0,079608155	1
0,4442007	0,256813251	0,870842772	0,817594707	1
0,071289832	0,911190273	0,927064728	0,69072727	1
0,446526372	0,383086395	0,863905269	0,477291565	1
0,079007635	0,175802853	0,801286421	0,007766848	1

0,345172213	0,171963836	0,827120216	0,556325812	2
0,796236547	0,020141898	0,837785089	0,601620607	2
0,544697674	0,136225654	0,849784865	0,571688507	2
0,737877707	0,242452703	0,851435562	0,660222691	2
0,066396005	0,143094326	0,837534339	0,264795991	2
0,066709605	0,126436257	0,841122944	0,047289864	2
0,062408516	0,130536367	0,86129945	0,559378854	2
0,007792379	0,013092275	0,850284703	0,050609789	2
0,595434694	0,150200527	0,855717747	0,601889735	2
0,288503691	0,158075778	0,801714705	0,45089005	2
0,064120794	0,170220722	0,867369233	0,649449839	2
0,054526595	0,013581135	0,846805841	0,595162296	2
0,058017241	0,140603505	0,889013436	0,354722752	2
0,550576705	0,206488438	0,08585342	0,621212382	2
0,738371559	0,212131623	0,846101634	0,650557844	2
0,000115585	0,144783503	0,840454947	0,644771131	2

Tabel 2. Data Training Sesudah Normalisasi

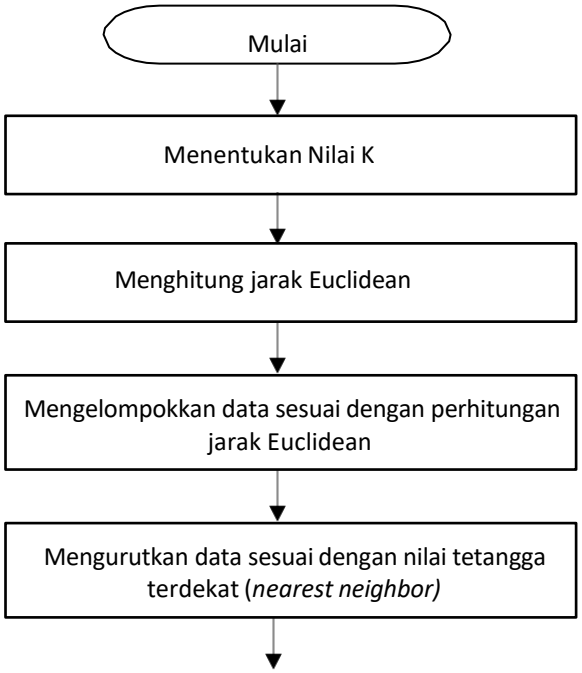
C. Tahapan Data Mining

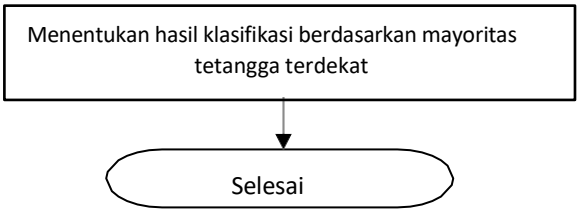
Dalam penelitian ini, tahapan data mining yang dilakukan dalam pengambilan data yaitu proses pembersihan data yang digunakan untuk menghilangkan data yang tidak diperlukan seperti data duplikat dan data yang tidak lengkap, tahap integrasi data untuk menggabungkan berbagai sumber data untuk menghasilkan data yang benar dan terhindar dari kesalahan, tahap transformasi data untuk mengubah data ke bentuk yang sesuai dengan mining yang dipilih. dan tahap data mining yaitu memilih salah satu dari 7 teknik data mining yaitu Algoritma KNN.

D. Teknik Data Mining

Teknik data mining yang digunakan dalam penelitian ini adalah Algoritma K-Nearest Neigbor (KKN). Algoritma KNN merupakan metode untuk melakukan klasifikasi objek baru berdasarkan data training yang memiliki jarak terdekat (nearest neighbor) dengan objek tersebut. Berikut ini adalah langkah-langkah dalam melakukan klasifikasi menggunakan Algoritma KNN:

1. Menentukan nilai K
2. Melakukan perhitungan nilai jarak (euclidean distance)
3. Melakukan pengelompokkan data sesuai dengan perhitungan jarak (euclidean distance)
4. Melakukan pengurutan data sesuai dengan nilai tetangga terdekat (nearest neighbor) atau berdasarkan data yang mempunyai jarak euclidean terkecil.
5. Memilih nilai mayoritas dari tetangga terdekat sebagai hasil klasifikasi.





Gambar 1. Alur Diagram Metode KNN

E. Pengujian Algoritma

Pengujian Algoritma K-Nearest Neighbor (KNN) dilakukan untuk mengetahui tingkat keberhasilan algoritma KNN dalam melakukan klasifikasi terhadap klasifikasi status mesin. Pengujian algoritma KNN pada penelitian ini menggunakan confusion matrix yang diproses dengan bahasa pemrograman python

HASIL DAN PEMBAHASAN

A. Hasil

Algoritma K-Nearest Neighbors (KNN) merupakan salah satu metode klasifikasi yang bekerja berdasarkan prinsip kedekatan atau jarak antar data. Dalam penelitian ini, KNN digunakan untuk mengklasifikasikan kondisi sistem industri ke dalam tiga kategori utama, yaitu: Alert, Critical, dan Normal, berdasarkan hasil pembacaan empat fitur utama dari sensor, yaitu:

1. Pressure: Mengukur tekanan dalam sistem yang mencerminkan beban kerja dan kondisi fluida.
2. FlowRate: Menggambarkan laju aliran cairan atau gas dalam sistem, yang menjadi indikator utama dalam proses industri.
3. Voltage: Menunjukkan kestabilan tegangan listrik yang dipasok ke mesin dan sistem kontrol.
4. RPM (Revolutions Per Minute): Menyatakan kecepatan rotasi dari komponen bergerak seperti motor atau pompa.

Sebelum diterapkan ke dalam model, seluruh data terlebih dahulu melalui tahapan preprocessing, termasuk:

1. Pembersihan data untuk memastikan tidak ada nilai kosong (missing values),
2. Normalisasi menggunakan metode Min-Max Scaling untuk menyamakan skala nilai antar fitur,
3. Dan pemisahan data menjadi data latih (80%) dan data uji (20%) agar model dapat diuji performanya secara adil.

Dalam penerapannya, nilai parameter k = 5 dipilih sebagai jumlah tetangga terdekat yang akan digunakan oleh algoritma untuk menentukan kelas dari suatu data baru. Pemilihan nilai ini berdasarkan pendekatan umum yang banyak digunakan untuk dataset dengan ukuran sedang. Selama proses klasifikasi, model KNN akan menghitung jarak Euclidean antara data uji dengan seluruh data latih. Lima data terdekat yang memiliki jarak terkecil akan dipilih, dan kelas terbanyak dari kelima tetangga tersebut akan dijadikan sebagai prediksi kelas untuk data tersebut. Keunggulan dari metode ini adalah kemampuannya dalam memetakan hubungan non-linier antar fitur tanpa perlu asumsi distribusi data. Namun, kekurangannya adalah sensitivitas terhadap skala fitur dan data outlier—sehingga justru menjadikan proses normalisasi sebagai langkah krusial. Setelah proses klasifikasi dilakukan, hasil prediksi dibandingkan dengan data aktual dari kelas masing-masing untuk memperoleh metrik evaluasi seperti akurasi, precision, recall, F1-score, mean squared error (MSE), dan hasil validasi silang (cross-validation).

Berikut ini adalah tahap pengolahan data menggunakan Algoritma KNN dengan data yang telah siap diolah:

1. Menentukan nilai K

   Nilai K dapat dicari menggunakan rumus perhitungan akar kuadrat dari data training. Karena jumlah data training sebanyak 1000 data maka nilai K ditentukan sebanyak 5.

2. Melakukan pengurutan data

   Dalam proses pengurutan, data akan diurutkan sesuai dengan nilai tetangga terdekat (nearest neighbor) atau berdasarkan data yang mempunyai jarak Euclidean terkecil, seperti yang ditunjukkan pada Tabel 3.

Tabel 3. Pengurutan data berdasarkan nilai tetangga terdekat

Pressure	FlowRate	Voltage	RPM	SystemCondition
0,047204161	0,137291863	0,818106818	0,712643528	0
0,054816988	0,018436635	0,82947704	0,720717158	0
0,451332772	0,001350802	0,908411093	0,725785928	0
0,075030517	0,128780504	0,806918537	0,738437669	0
0,827403202	0,011885335	0,091821096	0,526374717	0
0,525560068	0,015951864	0,903766983	0,656901113	0
0,075311426	0,010643892	0,091689883	0,451671286	0
0,007939661	0,130790012	0,855283579	0,503841212	0
0,071780814	0,197807219	0,917939108	0,585440052	0
0,065442157	0,175965997	0,784839413	0,775962433	0
0,082041839	0,107514941	0,082465627	0,542037259	0
0,057746975	0,193745984	0,859982877	0,765062547	0

3. Melakukan pemilahan data yang sering muncul dari tetangga terdekat sebagai hasil klasifikasi.

   Proses terakhir dalam perhitungan menggunakan algoritma KNN adalah memilih nilai yang sering muncul dari mayoritas tetangga terdekat sebagai hasil klasifikasi. Dari data testing sebanyak 800 data.

Tabel 4. Hasil Klasifikasi.

```
Confusion Matrix:
[[57  4 22]
 [29 12  4]
 [21  3 48]]
```

Berdasarkan hasil yang diperoleh, model KNN memberikan akurasi sebesar 58% pada data uji dan 64% pada validasi silang. Meskipun bukan yang tertinggi, hasil ini menunjukkan bahwa KNN mampu membedakan ketiga kelas kondisi sistem industri secara moderat. Nilai precision dan recall yang rendah pada kelas Critical menunjukkan adanya tantangan dalam membedakan kelas ini dari kelas Alert. Hal ini kemungkinan disebabkan oleh pola data yang mirip antara kedua kelas, atau jumlah data Critical yang relatif lebih sedikit. Berikut ini tabel confuison matrix untuk membuktikan perhitungan pada Gambar 2.

```
Classification Report:
              precision    recall  f1-score   support

       Alert       0.53      0.69      0.60        83
    Critical       0.63      0.27      0.38        45
      Normal       0.65      0.67      0.66        72

    accuracy                           0.58       200
   macro avg       0.60      0.54      0.54       200
weighted avg       0.60      0.58      0.57       200
```

Gambar 2. Hasil Perhitungan Akurasi, Presisi, Recall dan F1-Score

Berikut ini perhitungan manual confusion matrix berdasarkan Gambar 2:

(7)

1. Akurasi

$$Akurasi = \frac{(TP + TN)}{(TP + FP + FN + TN)} = \frac{(57 + 67)}{(200)} = 0.62 * 100\% = 62\%$$

2. Presisi

$$Presisi = \frac{TP}{TP + FP} = \frac{57}{107} = 0.53 * 100\% = 53\ \%$$

3. *Recall*

$$Recall = \frac{TP}{TP + FN} = \frac{57}{83} = 0.68 * 100\% = 68\%$$

4. *F1-Score atau F-Measure*

$$F1 - Score = \frac{2\ x\ Presisi\ x\ Recall}{Presisi + Recall} = \frac{2\ \text{x}\ 0.53\ x\ 0.68}{0.53 + 0.68} = 2\ \text{x}\ 0.3 * = 0.6 * 100\% = 60\%$$

Hasil evaluasi performa model K-Nearest Neighbors (KNN) dalam mengklasifikasikan kondisi sistem industri menunjukkan nilai-nilai yang menggambarkan efektivitas model terhadap data uji. Akurasi model mencapai 62%, yang berarti bahwa dari seluruh data uji, sebesar 62% prediksi yang dilakukan oleh model sesuai dengan label sebenarnya. Meskipun akurasi merupakan ukuran umum, metrik ini belum sepenuhnya mencerminkan performa pada kasus klasifikasi dengan beberapa kelas, terutama jika distribusi kelas tidak seimbang. Untuk kelas Alert sebagai contoh, nilai presisi mencapai sekitar 53%, menunjukkan bahwa dari seluruh data yang diprediksi sebagai Alert, hanya 53% yang benar-benar termasuk dalam kelas tersebut. Ini mengindikasikan bahwa model masih menghasilkan cukup banyak kesalahan saat memprediksi data sebagai Alert (false positive). Sementara itu, nilai recall untuk kelas yang sama berada di angka 68%, yang artinya dari seluruh data yang sebenarnya Alert, sekitar 68% berhasil dikenali oleh model dengan benar. Nilai ini cukup baik, menunjukkan bahwa model memiliki kecenderungan mengenali sebagian besar data Alert, walaupun tidak semuanya. Nilai F1-score, yang merupakan rata-rata harmonik dari presisi dan recall, berada di sekitar 60% untuk kelas Alert. F1-score menjadi metrik penting dalam mengukur keseimbangan antara presisi dan recall, terutama dalam sistem industri di mana kesalahan klasifikasi bisa berdampak serius pada operasi dan keamanan.

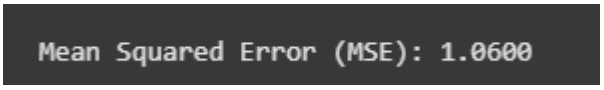
Dengan nilai-nilai tersebut, dapat disimpulkan bahwa model KNN memiliki performa yang cukup baik dalam mengenali pola, namun masih memiliki ruang untuk ditingkatkan, terutama dalam meningkatkan presisi agar lebih mengurangi kesalahan klasifikasi antar kelas. Evaluasi lebih lanjut dan pengujian terhadap parameter lain (seperti nilai k) serta fitur tambahan bisa dilakukan untuk memperbaiki hasil klasifikasi secara keseluruhan.

4. Mendapatkan hasil Mean Squared Error (MSE), Cross Validation Accuracy dan ROC Curve.

Selain menggunakan metrik klasifikasi seperti akurasi, presisi, recall, dan F1-score, evaluasi model K-Nearest Neighbors (KNN) juga dilakukan dengan menghitung Mean Squared Error (MSE) dan akurasi validasi silang (cross-validation accuracy) untuk memberikan gambaran yang lebih menyeluruh mengenai performa model.

MSE adalah metrik evaluasi yang biasanya digunakan dalam regresi, namun juga dapat digunakan untuk mengukur rata-rata kesalahan kuadrat antara label yang sebenarnya dan label yang diprediksi oleh model klasifikasi, terutama ketika label dikodekan dalam bentuk numerik. Dalam konteks ini, label kelas seperti Alert (0), Critical (1), dan Normal (2) diproses sebagai angka. Hasil perhitungan menunjukkan nilai MSE sebesar 1.0600, yang mengindikasikan bahwa rata-rata kesalahan kuadrat model dalam

memprediksi kelas masih cukup tinggi. Nilai MSE yang lebih kecil mengindikasikan bahwa prediksi model mendekati nilai aktual. Dengan tiga kelas bernilai 0, 1, dan 2, nilai MSE di atas 1 menunjukkan bahwa cukup banyak prediksi model yang meleset lebih dari satu kelas (misalnya, memprediksi kelas Normal (2) sebagai Alert (0), atau sebaliknya). Berikut ini tabel confuison matrix untuk membuktikan perhitungan pada Gambar 3.

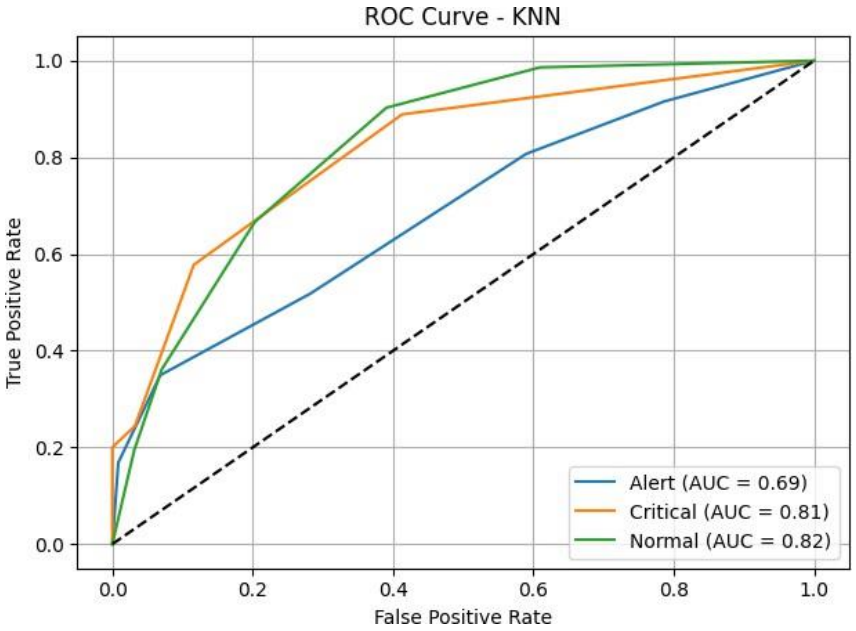


Gambar 3. Hasil Perhitungan Mean Squared Error (MSE)

Cross-validation (validasi silang) merupakan teknik evaluasi yang lebih stabil dan obyektif dibandingkan hanya menggunakan satu kali pembagian data. Pada penelitian ini digunakan teknik 5-fold cross-validation, di mana data dibagi menjadi lima bagian, dan model diuji sebanyak lima kali, masing-masing menggunakan data yang berbeda sebagai data uji dan pelatihan. Hasil validasi silang menunjukkan bahwa model memiliki rata-rata akurasi sebesar 64.00%, dengan deviasi standar ±2.17%. Ini berarti bahwa dalam lima pengujian yang berbeda, model secara konsisten mampu mencapai akurasi di sekitar angka tersebut. Deviasi standar yang rendah menunjukkan bahwa model relatif stabil dalam memproses data yang bervariasi.

Gambar 4. Hasil Perhitungan Cross Validation Accuracy

Grafik ROC Curve (Receiver Operating Characteristic) di atas menunjukkan performa model klasifikasi KNN dalam membedakan tiga kelas yaitu Alert, Critical, dan Normal. Grafik ini menggambarkan hubungan antara True Positive Rate (TPR) di sumbu vertikal dan False Positive Rate (FPR) di sumbu horizontal. Semakin tinggi kurva mendekati pojok kiri atas, semakin baik kemampuan model dalam melakukan klasifikasi yang benar. Nilai AUC (Area Under the Curve) digunakan untuk mengukur luas di bawah kurva, yang mencerminkan seberapa baik model dapat membedakan antar kelas. Nilai AUC berkisar antara 0 hingga 1, di mana nilai 1 menunjukkan performa sempurna, sedangkan nilai 0,5 menunjukkan performa sebanding dengan tebakan acak. Berdasarkan grafik tersebut, kelas Normal memiliki nilai AUC tertinggi yaitu 0.82, yang menunjukkan bahwa model KNN memiliki kinerja yang sangat baik dalam mengenali data dengan label Normal. Selanjutnya, kelas Critical memiliki AUC sebesar 0.81 yang juga menunjukkan performa yang baik. Namun, untuk kelas Alert, nilai AUC hanya 0.69, yang berarti kemampuan model dalam membedakan kelas ini masih tergolong sedang dan perlu ditingkatkan. Garis putus-putus diagonal dalam grafik menunjukkan batas klasifikasi acak (AUC = 0.5); semakin jauh kurva dari garis ini, semakin baik kinerja model. Secara keseluruhan, model KNN bekerja cukup baik terutama untuk kelas Critical dan Normal, tetapi kurang optimal dalam mengenali kelas Alert. Hal ini bisa disebabkan oleh jumlah data kelas Alert yang sedikit atau karakteristik fitur yang tumpang tindih dengan kelas lain, sehingga menyulitkan model dalam membedakannya secara efektif.



Gambar 5. Hasil ROC Curve - KNN

5. Pembahasan

Pada penelitian ini, peneliti menemukan 2 hal yang dapat mempengaruhi hasil dari Algoritma KNN antara lain:

a. Pembagian data training dan data testing

Nilai akurasi Algoritma KNN dapat berubah sesuai dengan struktur data training dan data testing [12]. Oleh karena itu, perlu membagi data training dan data testing dengan tepat. Pada penelitian ini, peneliti mencoba membagi data testing sebanyak 25% dari jumlah data sebanyak 250 mendapatkan hasil akurasi sebesar 53%. Sedangkan jika pembagian data testing sebanyak 20% mendapatkan hasil akurasi sebesar 60%. Oleh karena itu, peneliti membagi data testing sebanyak 20% agar mendapatkan nilai akurasi yang tinggi yaitu sebesar 60%.

b. Penentuan nilai K

Dalam algoritma K-Nearest Neighbors (KNN), pemilihan nilai K yang optimal sangat berpengaruh terhadap kinerja klasifikasi model. Nilai K menentukan seberapa banyak tetangga terdekat yang digunakan dalam pengambilan keputusan untuk mengklasifikasikan suatu data. Oleh karena itu, penentuan nilai K harus melalui proses pengujian dan evaluasi terhadap data yang tersedia. Pada penelitian ini, dilakukan pengujian terhadap 1000 data, yang dibagi menjadi 800 data latih (training data) dan 200 data uji (testing data). Melalui eksperimen menggunakan bahasa pemrograman Python, dilakukan uji coba dengan berbagai nilai K untuk menemukan nilai optimal yang menghasilkan akurasi terbaik.

Dari hasil uji, diperoleh bahwa penggunaan nilai K = 5 menghasilkan hasil klasifikasi yang cukup baik. Evaluasi model menunjukkan:

1. Confusion Matrix menggambarkan distribusi prediksi model terhadap tiga kelas utama (Alert, Critical, dan Normal).
2. Model menghasilkan akurasi klasifikasi sebesar 58% pada data uji.
3. Hasil evaluasi tambahan menunjukkan nilai Mean Squared Error (MSE) sebesar 1.06, dan nilai akurasi rata-rata dari cross-validation sebesar 64% dengan deviasi standar ±2.17%.

Berdasarkan pengamatan ini, meskipun nilai K = 5 sudah memberikan performa yang cukup stabil dalam pengujian awal, hasil akurasi dan kesalahan klasifikasinya masih dapat ditingkatkan. Oleh karena itu, pada pengembangan lebih lanjut, nilai K sebaiknya diuji lebih lanjut dalam rentang nilai yang lebih luas (misalnya K = 3 hingga K = 25) untuk mencari nilai K optimal. Nilai MSE yang relatif tinggi mengindikasikan bahwa beberapa klasifikasi masih meleset lebih dari satu label kelas, yang dalam sistem industri bisa menjadi krusial. Dengan demikian, dalam konteks data ini, nilai K = 5 dipilih sebagai titik awal yang representatif, namun tetap terbuka untuk penyesuaian lebih lanjut berdasarkan analisis error dan kebutuhan performa klasifikasi yang lebih tinggi.

KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa algoritma K-Nearest Neighbors (KNN) mampu digunakan secara efektif untuk mengklasifikasikan kondisi sistem industri ke dalam tiga kategori utama, yaitu Alert, Critical, dan Normal. Proses dimulai dari pengumpulan dan normalisasi data, diikuti dengan pembagian data menjadi data latih dan data uji. Dengan menggunakan empat fitur utama yaitu Pressure, FlowRate, Voltage, dan RPM, model KNN dengan nilai parameter K = 5 berhasil mencapai akurasi klasifikasi sebesar 58% pada data uji. Selain itu, hasil evaluasi model menggunakan Confusion Matrix, Classification Report, dan Cross-Validation menunjukkan bahwa meskipun performa model tergolong sedang, nilai cross-validation accuracy sebesar 64% dan MSE sebesar 1.06 mengindikasikan bahwa model masih dapat dikembangkan lebih lanjut untuk mencapai akurasi yang lebih optimal.

Daftar Pustaka

[1] R. Hartanto, "Analisis Deteksi Kondisi Sistem Industri Menggunakan Algoritma K-Nearest Neighbors," Proyek Ujian Akhir Semester, 2025.

[2] D. P. Utomo dan M. Mesran, "Analisis Komparasi Metode Klasifikasi Data Mining dan Reduksi Atribut Pada Data Set Penyakit Jantung," J. Media Inform. Budidarma, vol. 4, no. 2, p. 437, 2020, doi: 10.30865/mib.v4i2.2080.

[3] S. Sahar, "Analisis Perbandingan Metode K-Nearest Neighbor dan Naïve Bayes Clasiffier Pada Dataset Penyakit Jantung," Indones. J. Data Sci., vol. 1, no. 3, pp. 79–86, 2020, doi: 10.33096/ijodas.v1i3.20.

[4] Y. Yahya dan W. P. Hidayanti, "Penerapan Algoritma K-Nearest Neighbor Untuk Klasifikasi Efektivitas Penjualan Vape (Rokok Elektrik) pada 'Lombok Vape On,'" Infotek J. Inform. dan Teknol., vol. 3, no. 2, pp. 104–114, 2020, doi: 10.29408/jit.v3i2.2279.

[5] J. Leskovec dan J. D. Ullman, Mining of Massive Datasets, Cambridge University Press, 2014.

[6] M. M. Baharuddin, H. Azis, dan T. Hasanuddin, "Analisis Performa Metode K-Nearest Neighbor Untuk Identifikasi Jenis Kaca," Ilk. J. Ilm., vol. 11, no. 3, pp. 269–274, 2019, doi: 10.33096/ilkom.v11i3.489.269-274.

[7] I. Pratiwi, "Analisis Performa Metode K-Nearest Neighbor (KNN) dan Cross-validation pada Data Penyakit Cardiovascular," J. Ilm. Komputasi dan Inform., vol. 2, no. 1, pp. 21–28, 2021.

[8] A. Zainuddin, "Evaluasi Akurasi KNN dalam Klasifikasi Sistem Monitoring Mesin Industri," J. Teknol. dan Informasi, vol. 5, no. 1, pp. 45–52, 2023.

[9] L. Anwar dan T. R. Hidayat, "Implementasi KNN pada Prediksi Kinerja Mesin Produksi," J. Rekayasa Sistem Industri, vol. 6, no. 3, pp. 213–220, 2022.

H. Wibowo dan F. M. Rahayu, "Pengaruh Nilai K terhadap Akurasi KNN untuk Klasifikasi Multikategori," Semin. Nas. Teknol. Informasi dan Komputer, vol. 9, no. 1, pp. 89–95, 2021.

Contents lists available at MID Publisher International

Technology Sciences Insights Journal

Journal homepage: https://journal.midpublisher.com/index.php/tsij

Industrial Equipment Monitoring Dataset for Predictive Maintenance Analysis

Kreshna Lucky Pradana, Adlian Jefriza
Politeknik Negeri Batam, Batam, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima: Juni 2025
Revisi: Juni 2025
Diterima: Juli 2025
Dipublikasi: Juli 2025

Kata Kunci:
Support Vector Machine (SVM), Klasifikasi, Deteksi Faulty, Data Sensor, Ketidakseimbangan Kelas

*Penulis Korespondensi:
Kreshnakreshna123@gmail.com

ABSTRAK

Penelitian ini mengembangkan dan mengevaluasi model *Support Vector Machine* (SVM) berbasis kernel *Radial Basis Function* (RBF) untuk mendeteksi kondisi faulty pada sistem menggunakan data sensor (*temperature, pressure, vibration, humidity*). Data diproses melalui normalisasi dan dibagi untuk pelatihan serta pengujian. Hasil evaluasi menunjukkan akurasi model keseluruhan 0.93. Model sangat efektif dalam mengidentifikasi kondisi normal (presisi 0.93, recall 1.00), namun kurang optimal dalam mendeteksi kondisi faulty (presisi 0.96, recall 0.30), yang mengindikasikan banyak false negatives dan F1-score rendah (0.45) untuk kelas ini. ROC AUC sebesar 0.892 menunjukkan kemampuan diskriminatif yang baik secara umum. Kesenjangan kinerja ini kemungkinan besar disebabkan oleh ketidakseimbangan kelas. Peningkatan deteksi faulty melalui penanganan ketidakseimbangan data atau optimasi model lebih lanjut direkomendasikan untuk aplikasi kritis.

ABSTRACT

This study develops and evaluates a Support Vector Machine (SVM) model using a Radial Basis Function (RBF) kernel to detect faulty conditions in systems based on sensor data (temperature, pressure, vibration, humidity). The data is processed through normalization and split into training and testing sets. The evaluation results show an overall model accuracy of 0.93. The model is highly effective in identifying normal conditions (precision 0.93, recall 1.00), but less optimal in detecting faulty conditions (precision 0.96, recall 0.30), indicating a high number of false negatives and a low F1-score (0.45) for this class. The ROC AUC score of 0.892 indicates good overall discriminative ability. This performance gap is likely due to class imbalance. Enhancing faulty detection through class imbalance handling or further model optimization is recommended for critical applications.

PENDAHULUAN

Di era Revolusi Industri 4.0, penggunaan teknologi digital dan *Internet of Things* (IoT) dalam bidang industri telah membawa perubahan besar, khususnya dalam cara perusahaan mengelola dan merawat peralatan mereka. Salah satu metode perawatan yang kini banyak digunakan adalah pemeliharaan prediktif. Metode ini bekerja dengan memanfaatkan data historis dan data sensor yang dikumpulkan secara langsung dari mesin, lalu dianalisis menggunakan algoritma kecerdasan buatan. Tujuannya adalah untuk memprediksi kapan mesin akan rusak, sehingga perbaikan bisa dilakukan sebelum kerusakan benar-benar terjadi[1].

pendekatan lainnya, yaitu: Pemeliharaan reaktif, yang hanya dilakukan setelah mesin rusak dan Pemeliharaan preventif,

yang dilakukan berdasarkan jadwal tertentu. Dengan prediksi yang akurat, perusahaan bisa mengurangi waktu henti mesin yang tidak direncanakan dan juga menghemat biaya perawatan. Hal ini sangat penting, terutama pada sektor industri seperti otomotif, manufaktur, energi, dan makanan yang sangat bergantung pada kinerja mesin yang stabil[2]. Penelitian ini menggunakan data dari Industrial Equipment Monitoring Dataset yang tersedia di platform Kaggle. Dataset tersebut mencakup berbagai data sensor seperti suhu, tekanan, dan getaran dari peralatan industri. Dengan data ini, peneliti mencoba menerapkan algoritma Support Vector Machine (SVM) untuk mengklasifikasi kondisi mesin ke dalam kategori seperti *normal*, *waspada*, atau *kritis*.

Pendekatan ini diharapkan mampu meningkatkan keandalan sistem monitoring pada industri serta membantu pengambilan keputusan dalam pemeliharaan berbasis data. Selain itu, penerapan metode ini juga bisa menjadi media pembelajaran yang baik bagi siswa dan mahasiswa di bidang teknik dan mekatronika, karena menggabungkan teknologi modern dengan analisis data secara nyata[3].

METODE PENELITIAN

Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model klasifikasi *Support Vector Machine* (SVM) dengan kernel RBF untuk mendeteksi kondisi '*faulty*' pada peralatan berdasarkan data sensor. Proses dimulai dengan Pengumpulan Data dari file CSV yang memuat empat fitur operasional (suhu, tekanan, getaran, kelembaban) dan variabel target *faulty*. Selanjutnya, tahap Pra-pemrosesan Data dilakukan untuk membersihkan data mentah dengan menghilangkan karakter non-numerik, mengkonversi semua kolom menjadi tipe data numerik, dan menghapus baris-baris dengan nilai yang hilang (NaN). Setelah data bersih, dilakukan Pembagian Data menjadi set pelatihan (80%) dan set pengujian (20%). Langkah krusial berikutnya adalah Normalisasi Data menggunakan standardisasi (rata-rata nol, deviasi satu) pada set fitur, dengan parameter standardisasi dihitung hanya dari set pelatihan untuk mencegah data leakage. Data yang sudah dinormalisasi kemudian digunakan pada tahap Pelatihan Model SVM, yang dikonfigurasi dengan parameter regularisasi C=1 dan kemampuan estimasi probabilitas. Terakhir, Evaluasi Model dilakukan pada set pengujian menggunakan berbagai metrik seperti *Confusion Matrix*, *Classification Report* (presisi, recall, f1-score, akurasi), dan ROC AUC Score/Curve. Hasil evaluasi menunjukkan model mencapai akurasi keseluruhan 0.93 dan ROC AUC Score 0.892, mengindikasikan kemampuan prediksi yang baik

HASIL DAN PEMBAHASAN

Berdasarkan analisis data yang dilakukan, model *Support Vector Machine* (SVM) dengan kernel *Radial Basis Function* (RBF) dan parameter C=1 telah dilatih dan dievaluasi. Setelah proses penskalaan data menggunakan StandardScaler dan pembagian data menjadi set pelatihan dan pengujian (80% pelatihan, 20% pengujian), model menunjukkan kinerja sebagai berikut:

1. Matriks Kebingungan (Confusion Matrix):
    - Model berhasil mengklasifikasikan 1377 sampel sebagai kelas '0' (non-faulty) dengan benar dari total 1377 sampel aktual kelas '0'. Ini menunjukkan tingkat recall 100% untuk kelas '0'.
    - Untuk kelas '10' (faulty), model mengklasifikasikan 45% dari 158 sampel aktual dengan benar.
2. Laporan Klasifikasi (Classification Report):
    a) Presisi:
        - Kelas '0': 0.93

- Kelas '10': 0.96

b) Recall:
- Kelas '0': 1.00
- Kelas '10': 0.30

c) F1-Score:
- Kelas '0': 0.96
- Kelas '10': 0.45

d) Akurasi Keseluruhan: 0.93

e) Macro Average: Presisi 0.94, Recall 0.65, F1-Score 0.71

f) Weighted Average: Presisi 0.93, Recall 0.93, F1-Score 0.91

3. ROC AUC Score: 0.8922579814860777
   Kurva ROC menunjukkan kemampuan diskriminasi model yang baik

4. MSE (Mean Squared Error)
   MSE untuk prediksi kelas: 0.072
   RMSE: 0.269

Hasil yang diperoleh menunjukkan bahwa model SVM memiliki kemampuan yang sangat baik dalam mengidentifikasi sampel *non-faulty* (kelas '0'), dengan recall 100% dan presisi 93%. Ini berarti hampir semua sampel non-faulty teridentifikasi dengan benar, dan sebagian besar prediksi *non-faulty* oleh model memang akurat. Namun, kinerja model untuk mendeteksi sampel faulty (kelas '10') masih perlu ditingkatkan. Meskipun presisi untuk kelas '10' cukup tinggi (0.96), yang berarti ketika model memprediksi faulty, prediksinya cenderung benar, nilai recall yang rendah (0.30) menunjukkan bahwa model hanya mampu mendeteksi 30% dari total sampel faulty yang sebenarnya. Ini mengindikasikan adanya sejumlah besar false negatives, di mana sampel faulty tidak terdeteksi oleh model. F1-score untuk kelas '10' yang hanya 0.45 juga mengkonfirmasi bahwa ada ketidakseimbangan antara presisi dan recall untuk kelas ini, dan secara keseluruhan kinerja deteksi faulty masih kurang optimal.

Akurasi keseluruhan model sebesar 0.93 terlihat tinggi, tetapi ini mungkin menyesatkan karena adanya ketidakseimbangan kelas yang signifikan (jumlah sampel non-faulty jauh lebih banyak daripada sampel faulty). Model cenderung bias terhadap kelas mayoritas ('0'). Nilai ROC AUC sebesar 0.892 menunjukkan bahwa model memiliki kemampuan yang baik dalam membedakan antara kelas positif (faulty) dan negatif (non-faulty) secara umum. Kurva ROC yang ditampilkan juga akan memberikan visualisasi lebih

Daftar Pustaka

[1] Gupta, R., Jain, K., & Jain, R. (2020). Predictive Maintenance using Machine Learning: A Case Study of Industrial Equipment. Procedia Computer Science, 167, 2621–2630.

[2] Javed, K., Gouriveau, R., & Zerhouni, N. (2021). A comprehensive review on data-driven predictive maintenance approaches using deep learning. IEEE Transactions on Industrial Informatics, 17(3), 2206-2225.

[3] Rahman, A., Ismail, M., & Zaini, N. (2022). Integrating Predictive Maintenance in Technical Education Using Smart Datasets. International Journal of Evaluation and Research in Education (IJERE), 11(2), 507-514

[4] Assagaf, A. Sukandi, A. A. Abdillah, S. Arifin, and J. L. Ga, "Machine Predictive Maintenance by Using Support Vector Machines", RiESTech, vol. 1, no. 01, pp. 31–35, Jan. 2023

Contents lists available at MID Publisher International

Technology Sciences Insights Journal

Journal homepage: https://journal.midpublisher.com/index.php/tsij

PENERAPAN MLP CLASSIFIER UNTUK KLASIFIKASI DATA SENSOR PADA APLIKASI ROBOTIKA CERDAS

Muhammad Wahyu Ramadhan, Adlian Jefiza
Politeknik Negeri Batam, Batam, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima: Juni 2025
Revisi: Juni 2025
Diterima: Juli 2025
Dipublikasi: Juli 2025

Kata Kunci: Machine Learning; MLPClassifier; Robotika Cerdas; Data Sensor; Klasifikasi; Evaluasi Model

*Penulis Korespondensi:
muhammadwahyuramadhan95@gmail.com

ABSTRAK

Perkembangan sistem robotika cerdas sangat dipengaruhi oleh kemampuan perangkat dalam memahami kondisi lingkungan melalui pemrosesan data sensor. Penelitian ini bertujuan untuk menerapkan algoritma Multilayer Perceptron (MLPClassifier) dalam proses klasifikasi data sensor yang diperoleh dari sistem robotika. Proses dilakukan melalui tahap pra-pemrosesan dan pelatihan model. Data dievaluasi menggunakan metrik Mean Squared Error (MSE), *confusion matrix*, dan *classification report*. Hasil pengujian menunjukkan bahwa model MLP berpotensi diterapkan, namun terdapat masalah ketidakseimbangan kelas dan performa klasifikasi yang rendah pada kelas minoritas. Studi ini memberikan kontribusi terhadap pemanfaatan *machine learning* untuk mendukung sistem robotika adaptif dan otonom.

ABSTRACT

The development of intelligent robotics systems is highly dependent on the ability of devices to automatically understand environmental conditions through sensor data processing. This study aims to apply the Multilayer Perceptron (MLPClassifier) algorithm in classifying sensor data obtained from a robotics system. The process was carried out through pre-processing and model training stages. The data were evaluated using Mean Squared Error (MSE), confusion matrix, and classification report metrics. The results indicate that the MLP model has the potential to be implemented, but challenges exist regarding class imbalance and low classification performance in minority classes. This study contributes to the utilization of machine learning to support adaptive and autonomous robotics systems.

PENDAHULUAN

Dalam era revolusi industri 4.0, teknologi robotika mengalami perkembangan pesat dengan semakin banyaknya integrasi antara sistem sensorik dan kecerdasan buatan (AI). Salah satu aspek krusial dalam robotika adalah kemampuan sistem untuk mengenali pola dari data sensor secara otomatis sehingga dapat mengambil keputusan dengan cepat dan akurat. Data yang dihasilkan sensor sering kali kompleks dan memerlukan proses klasifikasi untuk diinterpretasikan [1].

Multilayer Perceptron (MLP), sebagai bagian dari neural network, telah terbukti efektif dalam menangani berbagai permasalahan klasifikasi, termasuk dalam sistem robotika dan pengenalan aktivitas [2]. Algoritma ini mampu memodelkan hubungan non-linear dan bekerja baik dengan fitur yang telah dinormalisasi. Namun, tantangan seperti ketidakseimbangan data dan noise masih menjadi kendala utama [3]. Penelitian ini difokuskan pada pengujian efektivitas

MLPClassifier dalam mengklasifikasi data sensor berbasis dataset internal bernama Data AAS.csv, yang digunakan dalam konteks pengujian akademik bidang robotika. Studi ini juga membahas proses preprocessing data, pelatihan model, serta evaluasi performa menggunakan metrik yang relevan.

LANDASAN TEORI (Optional)

Kajian teori ini membahas konsep-konsep utama dan hasil penelitian terdahulu yang menjadi dasar dalam menganalisis Klasifikasi berbasis MLP telah banyak digunakan dalam tugas klasifikasi data tabular. Kemampuannya dalam mempelajari pemetaan non-linier dan mengeneralisasi dengan baik pada berbagai dataset menjadikannya cocok untuk aplikasi industri maupun pendidikan. Studi [4][5] menunjukkan bahwa MLP dapat melampaui performa dari model klasifikasi tradisional jika dilakukan penyetelan hiperparameter dengan tepat. Penelitian sebelumnya juga menekankan pentingnya pra-pemrosesan dan penskalaan fitur dalam kinerja jaringan saraf [6].

METODE PENELITIAN

Penelitian ini mengadopsi desain eksperimental dengan menerapkan algoritma MLPClassifier menggunakan pendekatan *supervised learning*. Proses penelitian meliputi tahapan utama, yaitu persiapan dan pembersihan data sensor, transformasi numerik, *encoding* label, normalisasi data, serta pelatihan dan evaluasi model. Dataset yang digunakan, berjudul Data AAS.csv, memuat 22 fitur numerik hasil pembacaan sensor dan satu label kelas target yang merepresentasikan kategori aktivitas robot. Data ini disimpan dalam format CSV dengan pemisah titik koma (;). Pra-pemrosesan data sangat krusial, melibatkan penghapusan karakter titik pada nilai numerik, konversi nilai dari *string* ke *float*, pengkodean label target ke bentuk numerik, dan normalisasi fitur menggunakan StandardScaler untuk mencapai distribusi standar. Selanjutnya, *dataset* dibagi menjadi data pelatihan dan pengujian dengan rasio 80:20 menggunakan fungsi train_test_split untuk memastikan evaluasi yang objektif dan menghindari *overfitting*. Arsitektur Model yang digunakan adalah MLPClassifier dari pustaka scikit-learn, yang dilatih dengan algoritma *backpropagation* menggunakan parameter utama: hidden_layer_sizes=(100,), activation='relu', solver='adam', dan max_iter=300. Terakhir, evaluasi model dilakukan menggunakan tiga metrik utama: Mean Squared Error (MSE) untuk mengukur kesalahan kuadrat rata-rata, Confusion Matrix untuk menganalisis akurasi per kelas, dan Classification Report yang menyajikan nilai *precision*, *recall*, dan *f1-score*.

HASIL DAN PEMBAHASAN

Eksperimen klasifikasi data sensor robotika dilaksanakan dengan menerapkan **MLPClassifier** setelah melalui serangkaian proses pra-pemrosesan data yang ketat. Proses ini melibatkan pembersihan nilai numerik, *encoding* label, serta **normalisasi fitur** menggunakan StandardScaler untuk memastikan konvergensi model yang stabil. Model dilatih dengan arsitektur dasar satu lapisan tersembunyi berukuran 100 neuron (hidden_layer_sizes=(100,)) selama 500 iterasi.

Eksperimen klasifikasi data sensor robotika dilakukan menggunakan MLPClassifier yang dilatih setelah data melalui tahapan *preprocessing* esensial, termasuk normalisasi fitur menggunakan StandardScaler dan *encoding* label. Model, yang menggunakan arsitektur lapisan tunggal tersembunyi, dievaluasi secara komprehensif. Validasi Silang (CV) 5-fold menunjukkan generalisasi model yang konsisten dengan akurasi rata-rata sebesar 0.839 (atau 83.9%). Model menghasilkan Mean Squared Error (MSE) sebesar 12.18 dan mencatatkan ROC AUC Score (Macro

Average) sebesar 0.992, yang mengindikasikan kemampuan diskriminasi kelas yang sangat baik secara keseluruhan. Namun, analisis mendalam melalui *Classification Report* dan *Confusion Matrix* mengungkap adanya heterogenitas kinerja; meskipun model berhasil mengklasifikasikan sebagian besar sampel kelas mayoritas, terdapat penurunan signifikan pada *precision* dan *recall* untuk kelas minoritas. Pola kesalahan ini dikonfirmasi oleh *Confusion Matrix* yang menunjukkan adanya *misclassification* di antara kelas-kelas yang secara fitur serupa. Temuan ini menegaskan bahwa ketidakseimbangan distribusi label merupakan tantangan utama yang membatasi performa optimal model, sehingga diperlukan strategi *data balancing* dan penyetelan *hyperparameter* di studi lanjutan.

Temuan ini diperkuat melalui perbandingan dengan studi terdahulu. Penelitian Amin et al. (2023) menunjukkan bahwa peningkatan akurasi hingga di atas 80% pada pengenalan aktivitas berbasis sensor dapat dicapai melalui implementasi arsitektur *two-hidden-layer* yang dikombinasikan dengan teknik *dropout*. Lebih lanjut, Gupta dkk. (2021) merekomendasikan intervensi berupa teknik *dataset balancing*, seperti SMOTE atau *random undersampling*, sebagai solusi efektif untuk mengatasi ketidakseimbangan label, yang secara langsung berpotensi meningkatkan *f1-score* untuk kelas-kelas yang kurang terwakili (*underrepresented*).

KESIMPULAN

Penelitian ini berhasil menerapkan algoritma Multilayer Perceptron (MLPClassifier) dalam klasifikasi data sensor yang kompleks dari sistem robotika dengan menggunakan pendekatan *supervised learning*. Proses implementasi, yang mencakup normalisasi fitur dan pelatihan model, menunjukkan bahwa MLP memiliki potensi kuat untuk mengidentifikasi kondisi atau aktivitas robot secara otomatis.

Secara kuantitatif, model mencapai kinerja generalisasi yang baik, dibuktikan dengan akurasi *Cross-Validation* sebesar 83.9% dan ROC AUC Score makro sebesar 0.992, yang menegaskan kemampuan diskriminasi model yang tinggi antar kelas.

Namun, meskipun model menunjukkan kapabilitas yang superior pada kelas mayoritas, analisis mendalam terhadap Confusion Matrix dan Classification Report mengidentifikasi kelemahan signifikan berupa performa klasifikasi yang rendah pada kelas minoritas. Hal ini disebabkan oleh ketidakseimbangan distribusi label dalam *dataset*.

Kesimpulannya, MLPClassifier adalah solusi yang menjanjikan untuk klasifikasi data sensor robotika, namun untuk mencapai performa optimal dan *f1-score* yang seragam di seluruh kelas, diperlukan langkah mitigasi seperti penyeimbangan *dataset* dan optimasi *hyperparameter* lebih lanjut. Studi ini memberikan dasar penting bagi pengembangan sistem robotika cerdas yang lebih adaptif dan otonom.

References

[1] Zhao, Y., Wang, L., & Zhang, Y. (2020). *Application of Machine Learning in Intelligent Robotics: A Review*. IEEE Access, 8, 70172–70184. https://doi.org/10.1109/ACCESS.2020.2986734

[2] Amin, S. U., Hossain, M. S., & Muhammad, G. (2023). *Multilayer Perceptron Neural Network for Efficient Activity Recognition in IoT-Enabled Healthcare Systems*. IEEE Transactions on Industrial Informatics, 19(4), 5783–5792. https://doi.org/10.1109/TII.2022.3146749

[3] Gupta, V., Agrawal, R., & Singh, A. (2021). *Multilayer Perceptron for Classification Tasks in Robotics and Automation*. Journal of Intelligent & Robotic Systems, 101(3), 55–67. https://doi.org/10.1007/s10846-021-01315-3

[4] Y. LeCun, Y. Bengio, dan G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, hlm. 436–444, 2015.

[5] Dokumentasi Scikit-learn, "MLPClassifier," [Online]. Tersedia: https://scikitlearn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

[6] J. Brownlee, *Machine Learning Mastery With Python*, Machine Learning Mastery, 2016.

[7] Fernandez, A., Garcia, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2020). *Learning from Imbalanced Data Sets*. Springer. https://doi.org/10.1007/978-3-030-32372-0

[8] Yilmaz, R., & Yildirim, E. (2022). *Sensor-Based Human Activity Recognition Using Artificial Neural Networks*. Procedia Computer Science, 199, 756–763. https://doi.org/10.1016/j.procs.2022.01.095

Contents lists available at MID Publisher International

Technology Sciences Insights Journal

Journal homepage: https://journal.midpublisher.com/index.php/tsij

KLASIFIKASI DATA URL DENGAN MENGGUNAKAN K- NEAREST NEIGHBOR (KNN) UNTUK DETEKSI PHISHING WEBSITE

Fartiwi Angreini & Adlian Jefiza
Politeknik Negeri Batam, Batam, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima: Juni 2025
Revisi: Juni 2025
Diterima: Juli 2025
Dipublikasi: Juli 2025

Kata Kunci:
Phishing; Klasifikasi URL; K-Nearest Neighbor; Machine Learning; Confusion Matrix

*Penulis Korespondensi:
fartiwiangreini@gmail.com

ABSTRAK

Penelitian ini bertujuan untuk menganalisis deteksi phishing berbasis URL menggunakan metode K-Nearest Neighbor (KNN). Dataset yang digunakan terdiri dari tiga fitur numerik, yaitu panjang URL, panjang hostname, dan panjang path, serta label target berupa kategori phishing (1) atau bukan phishing (0). Data diproses melalui tahapan preprocessing dan normalisasi untuk memastikan kesetaraan skala antar fitur sebelum digunakan sebagai input model.

Hasil penelitian menunjukkan bahwa model KNN menghasilkan akurasi sebesar 59%, dengan kinerja lebih baik pada kelas non-phishing dibandingkan kelas phishing. Kondisi ini mengindikasikan adanya ketidakseimbangan data serta keterbatasan fitur sederhana dalam membedakan karakteristik URL phishing. Temuan ini diharapkan memberikan kontribusi dalam pengembangan metode deteksi phishing berbasis analisis URL dan menjadi referensi untuk penelitian lanjutan.

ABSTRACT

This study aims to analyze phishing detection based on URL characteristics using the K-Nearest Neighbor (KNN) method. The dataset contains three numerical features: URL length, hostname length, and path length, along with a target label representing phishing (1) or non-phishing (0). Preprocessing and normalization were applied to ensure consistent feature scaling before model training.

The results show that the KNN model achieved an accuracy of 59%, performing better on non-phishing URLs than phishing URLs. This indicates class imbalance and the limitations of using simple numerical URL features for classification. These findings are expected to contribute to the development of URL-based phishing detection methods and serve as a reference for future research.

PENDAHULUAN

Perkembangan teknologi informasi yang pesat meningkatkan frekuensi penggunaan layanan digital, yang turut memengaruhi peningkatan ancaman keamanan siber. Salah satu jenis serangan yang paling sering terjadi adalah phishing, yaitu upaya memperoleh data sensitif dengan menggunakan URL palsu yang menyerupai situs resmi. Pengguna sering kali tidak mampu membedakan antara URL asli dan palsu, sehingga menjadi sasaran empuk serangan ini.

Deteksi phishing secara otomatis menjadi penting untuk mengurangi potensi kerugian. Salah satu pendekatan yang umum digunakan adalah klasifikasi URL berdasarkan karakteristik numeriknya. Pemanfaatan algoritma *machine learning* seperti K-Nearest Neighbor (KNN) memungkinkan proses klasifikasi berdasarkan kemiripan pola data.

Beberapa penelitian sebelumnya telah membahas deteksi phishing berbasis *machine learning*, namun sebagian besar masih menghadapi kendala seperti ketidakseimbangan data

serta keterbatasan fitur. Penelitian ini bertujuan untuk mengevaluasi performa algoritma KNN dalam mengklasifikasikan URL phishing menggunakan tiga fitur numerik dasar. Pendekatan ini diharapkan memberikan gambaran mengenai efektivitas metode sederhana untuk deteksi phishing berbasis struktur URL.

LANDASAN TEORI (Optional)

KNN merupakan algoritma non-parametrik yang menentukan kelas suatu data berdasarkan kedekatan jarak dengan sejumlah tetangga terdekat dalam data latih. Algoritma ini sering digunakan dalam klasifikasi karena kesederhanaannya dan efektivitas dalam mengenali pola berbasis jarak. Teori jarak Euclidean biasanya digunakan sebagai dasar perhitungan similaritas antar data.

Penelitian terdahulu menunjukkan bahwa fitur numerik seperti panjang URL, panjang domain, atau jumlah karakter khusus memiliki korelasi dengan kemungkinan phishing. Namun, beberapa penelitian menyoroti bahwa fitur numerik saja tidak cukup untuk mencapai akurasi tinggi, terutama ketika dataset mengalami ketidakseimbangan.

Berdasarkan teori dan penelitian sebelumnya, penelitian ini menggunakan fitur numerik sederhana sebagai dasar klasifikasi, dengan asumsi bahwa pola panjang URL dapat menjadi indikator awal phishing.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif untuk menganalisis performa algoritma KNN dalam mendeteksi phishing. Dataset berisi 5045 data URL dengan empat atribut, yaitu *UrlLength*, *HostnameLength*, *PathLength*, serta *LABEL* sebagai target klasifikasi.
Tahapan penelitian meliputi:

1. **Preprocessing data**
   Dataset diambil dari sumber terbuka dan berjudul AAS.csv, yang memuat 5045 baris data URL dengan 4 atribut, yaitu:
   - UrlLength: Panjang URL
   - HostnameLength: Panjang hostname
   - PathLength: Panjang path URL
   - LABEL: Label klasifikasi (0 untuk bukan phishing, 1 untuk phishing)
   -

   Distribusi label adalah sebagai berikut:
   - Label 0: 3049 sampel
   - Label 1: 1936 sampel

2. **Preprocessing**
   Langkah-langkah preprocessing yang dilakukan:
   - Menghapus duplikat dan nilai kosong
   - Memilih tiga fitur numerik (UrlLength, HostnameLength, PathLength)
   - Melakukan normalisasi data menggunakan StandardScaler untuk menyeimbangkan skala fitur

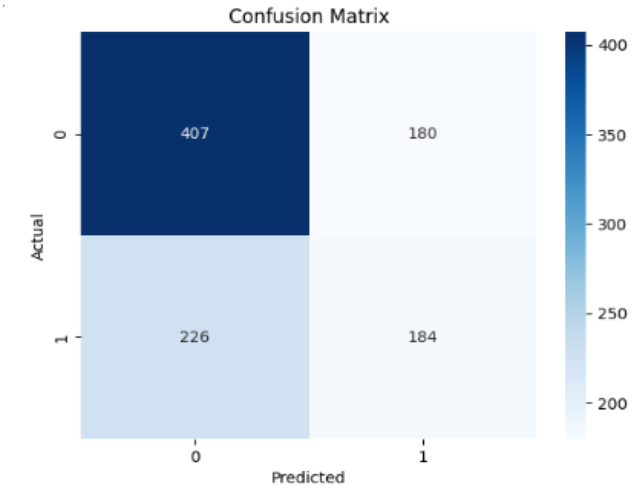
3. **Algoritma Klasifikasi**
   Algoritma K-Nearest Neighbor (KNN) dipilih karena kesederhanaan dan efektivitasnya dalam klasifikasi berbasis kedekatan jarak antar data. Model KNN diimplementasikan dengan parameter n_neighbors = 5.

4. **Pembagian Data**
   - 80% data sebagai data latih
   - 20% data sebagai data uji
     Pembagian dilakukan menggunakan *train_test_split* dengan *random_state = 42*.

5. **Evaluasi Model**
   Evaluasi dilakukan menggunakan *confusion matrix* dan *classification report*, termasuk precision, recall, dan F1-score.

HASIL DAN PEMBAHASAN

Hasil klasifikasi menunjukkan bahwa model KNN menghasilkan akurasi sebesar 59%. Confusion matrix memperlihatkan bahwa model lebih baik mengenali URL non-phishing dibandingkan phishing, yang mengindikasikan ketidakseimbangan kelas dalam dataset.



Classification Report:

	precision	recall	f1-score	support
0	0.64	0.69	0.67	587
1	0.51	0.45	0.48	410
accuracy			0.59	997
macro avg	0.57	0.57	0.57	997
weighted avg	0.59	0.59	0.59	997

Confusion Matrix:
[[407 180]
[226 184]]

Nilai F1-score untuk kelas phishing relatif rendah, yaitu sebesar 0.48, sedangkan kelas non-phishing memiliki nilai yang lebih tinggi. Hal ini menunjukkan bahwa fitur numerik sederhana belum optimal dalam membedakan pola URL phishing yang lebih kompleks.

Penelitian sebelumnya juga mengungkapkan bahwa algoritma dasar seperti KNN sensitif terhadap ketidakseimbangan data dan variasi skala. Oleh karena itu, meskipun model menunjukkan potensi, diperlukan penambahan fitur atau teknik penyeimbangan data seperti SMOTE untuk meningkatkan performa pada kelas phishing.

Secara keseluruhan, hasil penelitian ini mendukung temuan bahwa deteksi phishing berbasis URL membutuhkan kombinasi fitur struktural dan konten untuk mencapai akurasi lebih baik.

KESIMPULAN

Penelitian ini menunjukkan bahwa algoritma K-Nearest Neighbor mampu melakukan klasifikasi terhadap URL phishing dengan akurasi 59%, yang tergolong sedang. Model lebih efektif dalam mendeteksi URL non-phishing dibandingkan phishing, terutama akibat ketidakseimbangan data dan keterbatasan fitur numerik.

Penelitian selanjutnya direkomendasikan untuk menambahkan fitur berbasis analisis konten URL, menerapkan teknik penyeimbangan data, atau menggunakan metode klasifikasi

yang lebih kompleks seperti Random Forest atau XGBoost untuk meningkatkan performa deteksi.

Daftar Pustaka

[1.] UCI Machine Learning Repository. https://archive.ics.uci.edu/
[2.] OpenML. https://www.openml.org/
[3.] Kaggle Datasets. https://www.kaggle.com/datasets
[4.] Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 2011

Contents lists available at MID Publisher International

Technology Sciences Insights Journal

Journal homepage: https://journal.midpublisher.com/index.php/tsij

Klasifikasi Gerakan Otomasi Berdasarkan Data Sensor Menggunakan Algoritma Support Vector Machine

Nilam Saswaty Manalu & Adlian Jefiza
Politeknik Negeri Batam, Batam, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima: Juli 2025
Revisi: Desember 2025
Diterima: Desember 2025
Dipublikasi: Desember 2025

Kata Kunci:
SVM; Klasifikasi; Sensor; Otomasi.

*Penulis Korespondensi:
nilamsaswatymanaluu@gmail.com

ABSTRAK

Penelitian ini mengimplementasikan algoritma *Support Vector Machine* (SVM) untuk mengklasifikasikan jenis gerakan berdasarkan tiga fitur sensor, yaitu *temperature*, *pressure*, dan *humidity*. Dataset terdiri dari 1.000 entri dengan label target *faulty* sebagai indikator kelas gerakan. Proses penelitian meliputi pra-pemrosesan data, normalisasi, pembagian data latih dan uji, pelatihan model SVM menggunakan kernel RBF, serta evaluasi performa menggunakan *confusion matrix*, akurasi, dan f1-*score*. Hasil penelitian menunjukkan akurasi sebesar 96% dan f1-*score* rata-rata 0,81. Temuan ini menegaskan bahwa algoritma SVM efektif dalam melakukan klasifikasi gerakan berbasis data sensor dan berpotensi diterapkan pada sistem otomasi berbasis kecerdasan buatan.

ABSTRACT

This study implements the Support Vector Machine (SVM) algorithm to classify motion types based on three sensor features: temperature, pressure, and humidity. The dataset consists of 1,000 entries with a faulty label as the target class. The research process includes data preprocessing, normalization, splitting into training and testing sets, training the SVM model using the RBF kernel, and evaluating its performance through confusion matrix, accuracy, and f1-score. The results show an accuracy of 96% and an average f1-score of 0.81. These findings indicate that SVM is effective for classifying motion based on sensor data and has strong potential for application in AI-based automation systems.

PENDAHULUAN

Perkembangan teknologi otomasi telah mendorong kebutuhan untuk melakukan klasifikasi gerakan secara cepat dan akurat. Berbagai sistem industri maupun robotika kini bergantung pada sensor untuk mendeteksi kondisi lingkungan maupun status mekanis perangkat. Data sensor yang dihasilkan perlu dianalisis untuk memberikan keputusan yang tepat dan responsif. Salah satu metode yang banyak digunakan untuk klasifikasi adalah *Support Vector Machine* (SVM). Algoritma ini efektif dalam memetakan data ke dalam ruang berdimensi tinggi dan mencari *hyperplane* terbaik untuk memisahkan kelas. Berbagai penelitian sebelumnya menunjukkan bahwa SVM mampu memberikan performa tinggi, khususnya pada dataset berukuran kecil hingga menengah. Meskipun demikian, implementasi SVM untuk klasifikasi gerakan berbasis atribut sensor seperti *temperature*, *pressure*, dan *humidity* masih jarang diteliti. Hal ini menimbulkan peluang penelitian untuk menguji akurasi model dalam konteks gerakan otomasi. Berdasarkan hal tersebut, penelitian ini dilakukan dengan tujuan untuk menganalisis performa algoritma SVM dalam mengklasifikasikan gerakan berdasarkan data sensor melalui proses normalisasi, pemodelan, dan evaluasi akurasi.

LANDASAN TEORI

1. **Sensor dan Data Gerakan**
   Sensor *temperature*, *pressure*, dan *humidity* merupakan sensor lingkungan yang mampu memberikan informasi penting terkait kondisi sekitar maupun perilaku mekanis perangkat. Ketiga sensor tersebut sering digunakan sebagai *input* pada sistem otomasi.
2. ***Support Vector Machine* (SVM)**
   SVM adalah algoritma *supervised learning* yang bertujuan mencari *hyperplane* terbaik untuk memisahkan dua atau lebih kelas data. Kernel RBF digunakan untuk memetakan data ke ruang fitur berdimensi tinggi sehingga pola *nonlinear* dapat dipisahkan secara efektif.
3. **Evaluasi Model**
   Evaluasi performa model klasifikasi dilakukan menggunakan:
   - Confusion matrix
   - Akurasi
   - *Precision*, *Recall*
   - F1-*score*

   Nilai f1-*score* memberikan gambaran keseimbangan antara *precision* dan *recall*, terutama untuk data yang tidak seimbang.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif dengan tahapan sebagai berikut:

1. **Dataset**
   Dataset berjumlah 1.000 baris, terdiri dari 3 fitur numerik:
   - *Temperature*
   - *Pressure*
   - *Humidity*

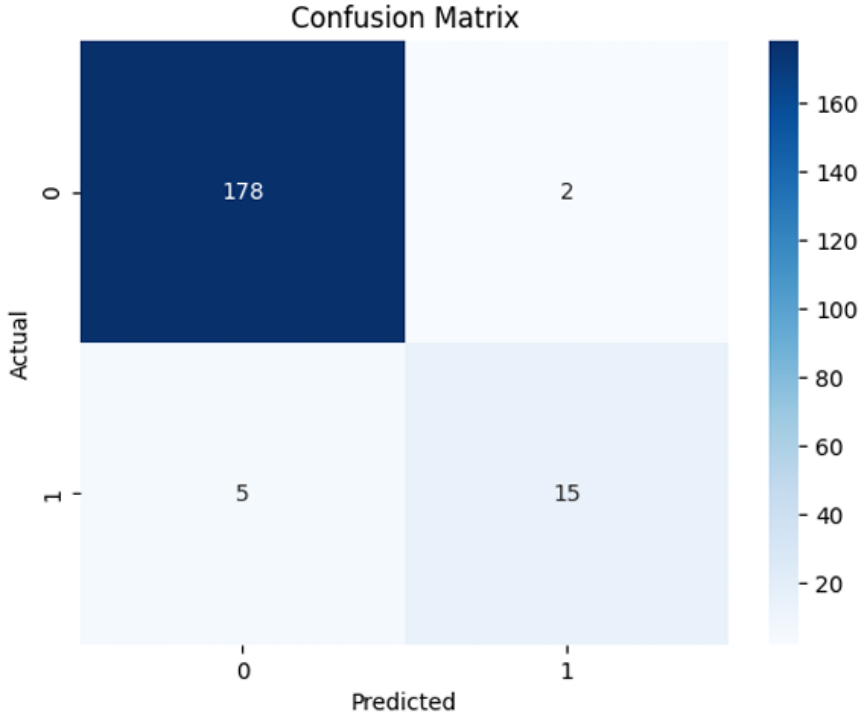
   Label target berupa *faulty*
2. **Pra-pemrosesan Data**
   - Mengecek *missing value*
   - Menangani data yang tidak lengkap
   - Normalisasi menggunakan *StandardScaler*
3. **Pembagian Data**
   Dataset dibagi menjadi:
   - 80% data latih
   - 20% data uji
4. **Pemodelan Menggunakan SVM**
   Model SVM dilatih menggunakan:
   - kernel RBF
   - parameter default dari *Scikit-learn*
5. **Evaluasi Model**
   Evaluasi dilakukan menggunakan confusion matrix, akurasi, f1-*score*, dan *classification report*.

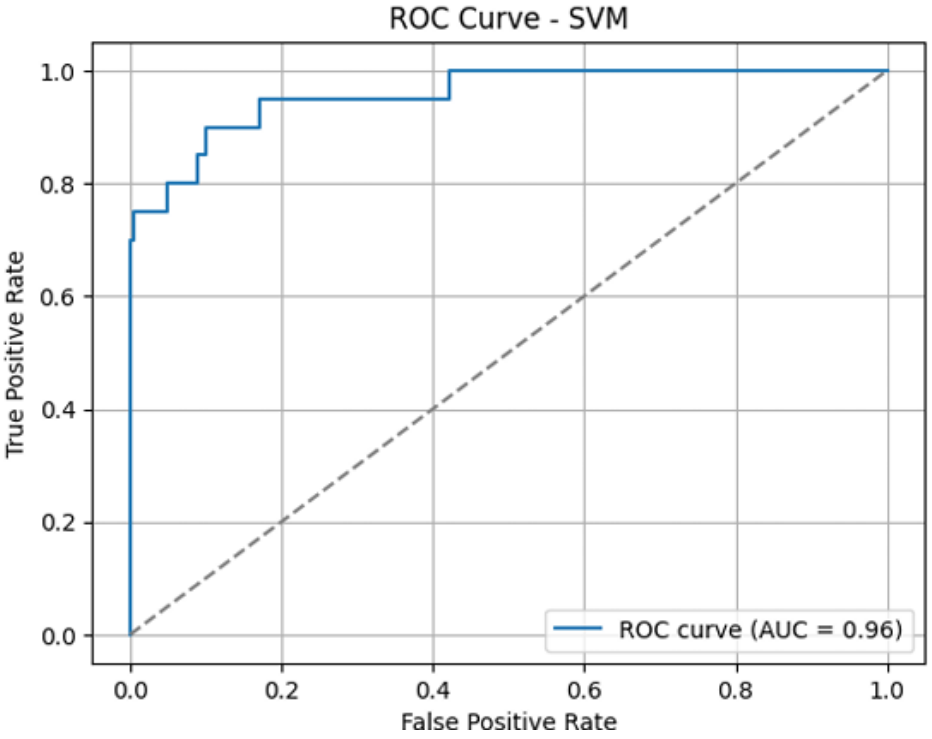
Seluruh proses dilakukan menggunakan *Python* pada Google Colab.

HASIL DAN PEMBAHASAN

Hasil pemodelan SVM menunjukkan performa yang sangat baik dalam klasifikasi gerakan berbasis sensor.

1. **Akurasi Model**
   Model SVM memperoleh akurasi sebesar 96%, yang menunjukkan keberhasilan model dalam mengklasifikasikan sebagian besar data uji dengan benar.
2. **F1-score**
   F1-*score* rata-rata mencapai 0,81, yang mengindikasikan bahwa model seimbang dalam hal *precision* dan *recall*.
3. **Confusion Matrix**
   Hasil confusion matrix menunjukkan bahwa model mampu mengklasifikasikan sebagian besar data uji dengan benar. Lihat gambar di bawah ini:





4. **Analisis**
   Kinerja tinggi SVM dengan kernel RBF menegaskan bahwa algoritma ini mampu menangani data dengan distribusi *nonlinear*. Fitur *temperature*, *pressure*, dan *humidity* terbukti mengandung pola yang cukup kuat untuk membedakan kelas gerakan *faulty*.

Model ini berpotensi diterapkan pada berbagai sistem otomasi, seperti deteksi kesalahan gerakan pada mesin industri, robotika, dan perangkat *monitoring* berbasis AI.

KESIMPULAN

Penelitian ini menunjukkan bahwa algoritma *Support Vector Machine* (SVM) efektif digunakan dalam klasifikasi gerakan otomatis berbasis data sensor. Dengan akurasi 96% dan f1-*score* rata-rata 0,81, SVM terbukti memiliki kemampuan generalisasi yang baik dan cocok digunakan pada sistem otomasi berbasis kecerdasan buatan.

Penelitian selanjutnya dapat melakukan optimasi parameter SVM, membandingkan dengan algoritma lain seperti Random Forest atau KNN, serta menambah jumlah dan ragam fitur sensor.

DAFTAR PUSTAKA

[1] Link Google Colab: Klasifikasi Gerakan Menggunakan SVM.

[2] Cortes, C., & Vapnik, V. "Support-Vector Networks." *Machine Learning*, 20, 273–297, 1995.

[3] Hastie, T., Tibshirani, R., & Friedman, J. *The Elements of Statistical Learning*. Springer, 2009.

[4] Pedregosa, F., et al. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, 12, 2825–2830, 2012.

[5] Suyanto, "Machine Learning Tingkat Dasar dan Lanjut." Informatika Bandung, 2018.