JOURNAL OF COMPUTER SCIENCE AND INFORMATICS ENGINEERING
VOL. 5, NO. 4 (2026)
DOI: https://doi.org/10.55537/cosie.v5i4.1832

e-ISSN 2827-8356
p-ISSN 2827-9999

Implementasi Metode Algoritma *Random Forest* Dan *Naive Bayes* Dalam Memprediksi Gangguan Jaringan Kominfo Tangsel

Implementation of the Random Forest and Naive Bayes Algorithms in Predicting Network Outages at the Kominfo South Tangerang

Raafa Syahidul Haq Irsi[1], Indra Kristianto[2]
Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Pamulang
[1]raffasyahidul@gmail.com, [2]dosen02597@unpam.ac.id

Received: July 17, 2026 | Revised: July 22, 2026 | Accepted: August 31, 2026 | Published: September 13, 2026

Abstrak

Ketergantungan operasional Diskominfo Kota Tangerang Selatan terhadap infrastruktur jaringan yang andal menjadikan gangguan jaringan sebagai salah satu faktor yang dapat memengaruhi kualitas layanan operasional serta produktivitas pegawai. Sistem *monitoring* jaringan yang digunakan saat ini masih bersifat reaktif dan belum didukung kemampuan analisis prediktif yang memanfaatkan data historis hasil *capturing* jaringan. Penelitian ini bertujuan mengembangkan model prediksi gangguan jaringan berbasis web dengan pendekatan *Cross Industry Standard Process for Data Mining (CRISP-DM)* melalui integrasi *severity score* dan pelabelan berbasis K-Means sebelum proses klasifikasi menggunakan algoritma *Random Forest* dan *Naive Bayes*. Dataset penelitian berasal dari hasil *capturing* trafik jaringan aktual di lingkungan Diskominfo Kota Tangerang Selatan yang terdiri atas 5.819 rekaman dengan empat parameter utama, yaitu *bandwidth*, *latency*, *packet loss*, dan *uptime*. Setelah melalui tahap *data preparation*, hasil pengujian menunjukkan bahwa algoritma *Random Forest* memperoleh *accuracy* sebesar *95,78%*, *precision 94,74%*, *recall 97,63%*, dan *F1-score 96,16%*, sedangkan *Naive Bayes* menghasilkan *accuracy 94,50%*, *precision 92,88%*, *recall 97,29%*, dan *F1-score 95,03%*. Analisis *Feature Importance* menunjukkan bahwa *packet loss* merupakan parameter yang paling berpengaruh terhadap keputusan klasifikasi dengan kontribusi sebesar *82,32%*. Berdasarkan keseluruhan hasil evaluasi, *Random Forest* dipilih sebagai model dengan performa terbaik dan diimplementasikan pada aplikasi prediksi berbasis Flask untuk mendukung proses pemantauan serta analisis kondisi jaringan. Penelitian ini bertujuan mengembangkan model prediksi gangguan jaringan berbasis web dengan pendekatan *Cross Industry Standard Process for Data Mining* (CRISP-DM) melalui integrasi *severity score* dan pelabelan berbasis K-Means untuk membentuk label berbasis karakteristik data sebelum proses klasifikasi menggunakan algoritma Random Forest dan Naive Bayes.

Kata kunci: *Machine Learning*, *Random Forest*, *Naive Bayes*, Prediksi gangguan jaringan, CRISP-DM.

Abstract

The operational activities of the South Tangerang City Communication and Information Office (Diskominfo) rely heavily on a stable network infrastructure, making network disruptions a critical issue that directly affects service quality and employee productivity. The existing network monitoring system remains reactive and does not yet incorporate predictive analytics based on historical network capturing data. This study proposes a web-based network disruption prediction model using the Cross Industry Standard Process for Data Mining (CRISP-DM) framework by integrating a severity score and K-Means-based labeling to generate data-driven class labels before applying the Random Forest and Naïve Bayes classification algorithms. The dataset consists of 5,819 records collected from actual network traffic capturing within the operational environment of the South Tangerang City Communication and Information Office, using four primary network parameters: bandwidth, latency, packet loss, and uptime. After the data preparation stage, the Random Forest model achieved an accuracy of 95.78%, precision of 94.74%, recall of 97.63%, and an F1-score of 96.16%. In comparison, the Naïve Bayes model achieved an accuracy of 94.50%, precision of 92.88%, recall of 97.29%, and an F1-score of 95.03%. Feature Importance analysis identified packet loss as the most influential predictor, contributing 82.32% to the classification decision. Based on the overall evaluation, Random Forest demonstrated superior predictive

DOI: https://doi.org/10.55537/cosie.v5i4.1832

performance and was deployed as the primary model in a Flask-based web application to support proactive network monitoring and network condition analysis.

Keywords: Machine Learning, Random Forest, Naive Bayes, Network Disruption Prediction,CRISP-DM

1. PENDAHULUAN

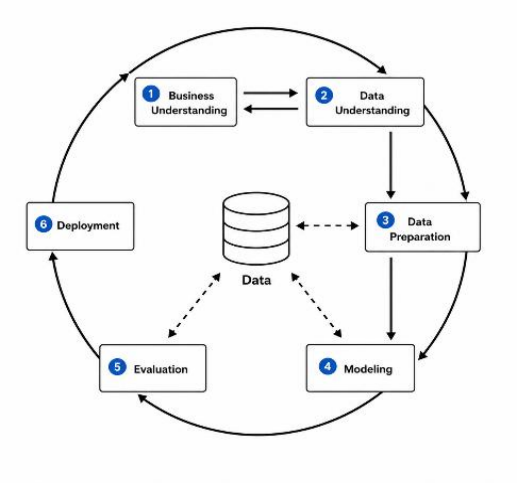
Gangguan jaringan dapat menurunkan kualitas layanan komunikasi, meningkatkan nilai *packet loss*, serta mengurangi ketersediaan layanan operasional yang bergantung pada konektivitas jaringan. Oleh karena itu, pengembangan metode yang mampu memprediksi potensi gangguan sebelum terjadi menjadi semakin penting untuk meningkatkan kualitas jaringan dan mendukung kegiatan pemeliharaan secara proaktif [1]. Pada Diskominfo Kota Tangerang Selatan, aktivitas *monitoring* jaringan menghasilkan data trafik aktual melalui proses *capturing* menggunakan Wireshark. Data tersebut memuat parameter *bandwidth*, *latency*, *packet loss*, dan *uptime* yang berpotensi dimanfaatkan sebagai data historis untuk membangun model berbasis *Machine Learning*. Namun, hingga saat ini data hasil *capturing* masih lebih banyak digunakan untuk keperluan pemantauan dan analisis jaringan dibandingkan sebagai dasar pengembangan model prediksi gangguan [2]. Perkembangan *Machine Learning* telah mendorong penerapan berbagai algoritma klasifikasi untuk mendeteksi dan memprediksi kondisi jaringan. Algoritma *Random Forest* dikenal memiliki performa yang baik dalam proses klasifikasi data jaringan [3], sementara pendekatan *Network Fault Prediction* berbasis *Machine Learning* dinilai mampu meningkatkan keandalan jaringan melalui identifikasi potensi gangguan sebelum terjadinya kegagalan layanan [4].

Berbagai penelitian menunjukkan bahwa algoritma *Machine Learning* mampu memberikan hasil klasifikasi yang baik pada berbagai kasus prediksi. *Random Forest* dikenal efektif dalam mengenali pola pada data jaringan, sedangkan *Naive Bayes* tetap banyak digunakan karena proses pelatihannya sederhana serta mampu memberikan performa yang kompetitif pada dataset berukuran besar [5]. Meskipun demikian, sebagian besar penelitian masih menggunakan satu algoritma klasifikasi atau mengevaluasi setiap model secara terpisah [6]. Sementara itu, penerapan K-Means untuk membentuk pola data sebelum proses klasifikasi serta pemanfaatan data trafik jaringan aktual hasil *capturing* menggunakan Wireshark masih belum banyak dilakukan. Kondisi tersebut membuka peluang untuk mengembangkan pendekatan *Hybrid Machine Learning* yang mampu memanfaatkan karakteristik data jaringan secara lebih optimal [7]. Berdasarkan kondisi tersebut, kasus ini mengembangkan model prediksi gangguan jaringan menggunakan pendekatan *Hybrid Machine Learning* dengan menggabungkan K-Means, *Random Forest*, dan *Naive Bayes* menggunakan 5.819 data trafik jaringan aktual hasil *capturing* Wireshark di lingkungan Diskominfo Kota Tangerang Selatan. Model yang dihasilkan kemudian diimplementasikan ke dalam aplikasi berbasis Flask sehingga dapat dimanfaatkan untuk mendukung proses *monitoring* jaringan dan pengambilan keputusan secara lebih cepat melalui antarmuka berbasis web [8]. Kontribusi utama penelitian ini tidak hanya terletak pada perbandingan performa algoritma *Random Forest* dan *Naive Bayes*, tetapi juga pada penerapan proses pembentukan label menggunakan pendekatan K-Means berbasis *severity score* sebelum tahap klasifikasi dilakukan. Penelitian ini juga memanfaatkan data trafik jaringan aktual yang diperoleh dari hasil *capturing* menggunakan Wireshark serta mengimplementasikan model yang dihasilkan ke dalam aplikasi berbasis Flask. Dengan pendekatan tersebut, model dapat dimanfaatkan sebagai pendukung analisa kegiatan *monitoring* jaringan secara operasional.

2. METODE PENELITIAN

Studi ini menggunakan metodologi *Cross Industry Standard Process for Data Mining* (CRISP-DM) sebagai kerangka kerja dalam pengembangan model. Metodologi tersebut dipilih karena menyediakan tahapan yang terstruktur, mulai dari pemahaman permasalahan, pengolahan data, pembangunan model, evaluasi, hingga implementasi [9].

DOI: https://doi.org/10.55537/cosie.v5i4.1832



Gambar 1. Metode CRISP-DM

2.1 Bussiness Understanding

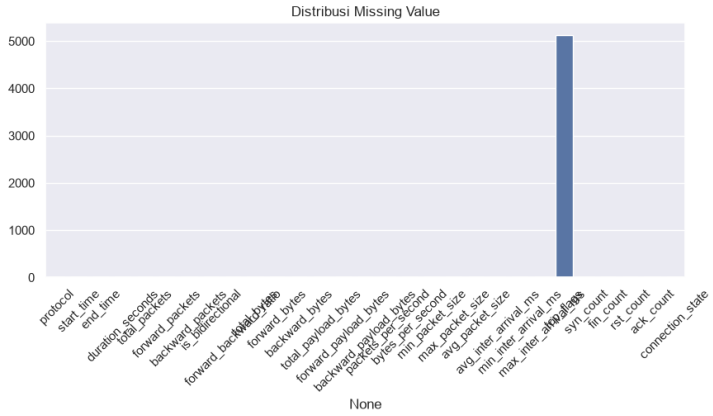
Tahap ini dilakukan untuk mengidentifikasi permasalahan utama dalam proses *monitoring* jaringan di kominfo Kota Tangerang Selatan. Hasil observasi menunjukkan bahwa tim teknisi jaringan ingin menaganalisa data hasil *capturing* dari setiap rekamaan penggunaan jaringan pada perangkat yang digunakan oleh para peagawai, Kondisi tersebut menunjukkan perlunya model prediksi yang mampu mengidentifikasi potensi gangguan berdasarkan pola historis parameter jaringan sebelum memengaruhi kinerja operasional.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan membangun model prediksi gangguan jaringan menggunakan pendekatan *Hybrid Machine Learning* dengan menggabungkan algoritma K-Means, *Random Forest*, dan *Naive Bayes*. Model yang dikembangkan diharapkan dapat membantu administrator jaringan dalam mendeteksi potensi gangguan lebih awal melalui analisis parameter.

2.2 Data Understanding

Difokuskan untuk memahami karakteristik dataset sebelum memasuki proses pemodelan. Data penelitian diperoleh dari hasil *capturing* menggunakan Wireshark, sehingga mencerminkan kondisi operasional jaringan yang sebenarnya. Pada tahap ini dilakukan identifikasi struktur dan tipe data, analisis distribusi setiap atribut, serta pemeriksaan *missing value* dan data duplikat sebagai dasar dalam menentukan tahapan *preprocessing*.

Proses *preprocessing* diawali dengan pemeriksaan struktur dataset menggunakan Pandas untuk memastikan kesesuaian tipe data pada setiap atribut serta mengidentifikasi keberadaan data duplikat dan *missing value*. Tahap ini bertujuan menjamin kualitas dan konsistensi dataset sehingga data yang digunakan pada proses pembentukan model tidak memengaruhi hasil pelatihan maupun performa klasifikasi [10].



Gambar 2. *Missing Value*

DOI: https://doi.org/10.55537/cosie.v5i4.1832

Eksplorasi data memanfaatkan Pandas untuk membaca, memfilter, dan mengelola dataset, sedangkan NumPy digunakan untuk mendukung komputasi numerik selama pengolahan data. Penggunaan kedua *library* tersebut membuat proses analisis data menjadi lebih efisien sebelum dilanjutkan ke tahap pembangunan model klasifikasi.

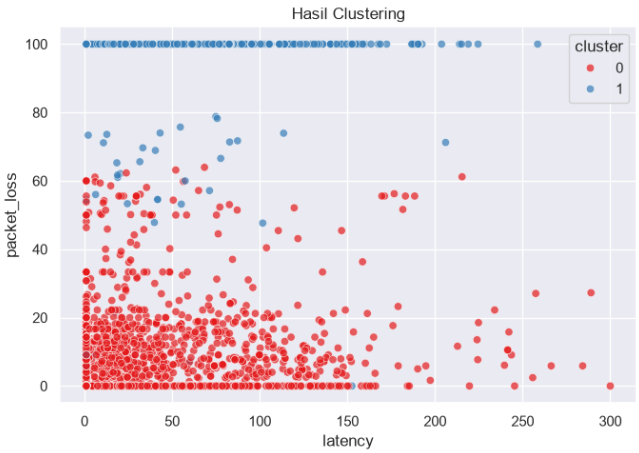
2.3 Data Preparation

Bagian ini bertujuan menyiapkan dataset agar dapat digunakan pada proses pembentukan model *Machine Learning*. Tahapan yang dilakukan meliputi pembersihan data, penghapusan data duplikat, transformasi tipe data, *feature scaling*, pembentukan *severity score*, proses *clustering* menggunakan K-Means, serta pembentukan label berdasarkan hasil *clustering*. Seluruh tahapan tersebut dilakukan untuk menghasilkan dataset yang lebih berkualitas dan siap digunakan pada proses klasifikasi.

Tabel 1. Dataset Berdasarkan *severity_score*

bandwidth	latency	packet_loss	uptime	Cluster	severity_score
80.488	74.932	78.792	80.192	1	49.076461
266.387	18.890	28.838	80.152	0	20.699193
329.760	206.031	71.227	80.965	1	62.168298
48.904	12.716	73.598	80.057	1	38.881773
508.174	2.1142	73.364	80.035	1	35.081346

Pada penelitian ini, proses pembentukan label diawali dengan menghitung *severity score* sebagai indikator tingkat keparahan kondisi jaringan. Nilai tersebut diperoleh dari kombinasi parameter *bandwidth*, *latency*, *packet loss*, dan *uptime*, sehingga mampu merepresentasikan kualitas jaringan secara menyeluruh. Semakin tinggi nilai *severity score*, semakin besar indikasi bahwa kualitas jaringan mengalami penurunan, sedangkan nilai yang lebih rendah menunjukkan kondisi jaringan yang relatif stabil. Nilai *severity score* kemudian digunakan sebagai acuan untuk mengenali pola kondisi jaringan sebelum memasuki tahap *clustering*. Dengan pendekatan ini, proses pelabelan didasarkan pada karakteristik data hasil pengelompokan (*data-driven labeling*), sehingga tidak bergantung pada penentuan label secara manual.



Gambar 3. Hasil *Clustering*

Sebelum proses *clustering*, seluruh atribut numerik dinormalisasi menggunakan *MinMaxScaler* yang tersedia pada *Scikit-learn*. Normalisasi dilakukan agar setiap atribut berada pada rentang nilai yang sama sehingga tidak terjadi dominasi nilai tertentu pada proses pengelompokan data. Seluruh proses komputasi numerik selama tahap *preprocessing* dilakukan menggunakan *NumPy*, sedangkan implementasi normalisasi dan algoritma K-Means memanfaatkan *Scikit-learn* [11].

Jumlah *cluster* ditentukan sebanyak dua karena penelitian ini bertujuan mengelompokkan kondisi jaringan ke dalam dua kategori operasional, yaitu Normal dan Gangguan. Pemilihan konfigurasi tersebut juga disesuaikan dengan kebutuhan pada tahap klasifikasi, sehingga hasil *clustering* dapat dimanfaatkan sebagai dasar pembentukan label yang konsisten. Hasil evaluasi menggunakan *Silhouette Score* sebesar 0,3647 menunjukkan bahwa pemisahan antar-*cluster* sudah cukup baik, sehingga layak digunakan sebagai dasar proses pelabelan dalam penelitian ini.

DOI: https://doi.org/10.55537/cosie.v5i4.1832

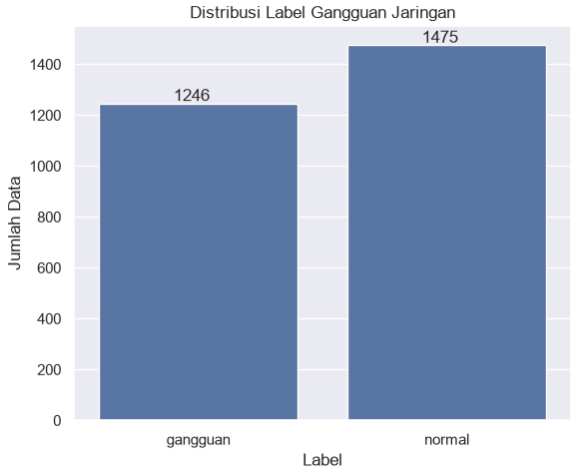
Tabel 2. Dataset Berdasarkan Cluster & Label

bandwidth	latency	packet_loss	uptime	Cluster	label
80.49	74.93	78.79	80.19	1	gangguan
266.39	18.89	28.84	80.15	0	normal
329.76	206.03	71.23	80.97	1	gangguan
48.90	12.71	73.60	80.05	1	gangguan
508.17	2.11	73.36	80.03	1	gangguan

Tabel 2 menampilkan contoh hasil pelabelan berdasarkan *cluster* yang terbentuk. Data yang berada pada *cluster* dengan karakteristik tingkat keparahan lebih tinggi diberi label Gangguan, sedangkan data pada *cluster* dengan tingkat keparahan lebih rendah diberi label Normal. Proses ini menghasilkan dataset berlabel yang selanjutnya digunakan sebagai target pada tahap klasifikasi.

Cluster dengan rata-rata *severity score* yang lebih rendah menunjukkan kondisi jaringan yang relatif stabil sehingga ditetapkan sebagai label Normal. Sebaliknya, *cluster* yang memiliki rata-rata *severity score* lebih tinggi menggambarkan kondisi jaringan dengan tingkat keparahan yang lebih besar. Kondisi tersebut dipengaruhi oleh meningkatnya nilai *latency* dan *packet loss*, sehingga *cluster* tersebut dipetakan sebagai label Gangguan.

Hasil ini mengindikasikan bahwa penerapan K-Means dengan 2 cluster mampu mengelompokkan data berdasarkan karakteristik *bandwidth*, *latency*, *packet loss*, *uptime*, dan severity score secara cukup representatif. Meskipun belum termasuk kategori *cluster* yang sangat kuat, hasil pengelompokan ini telah memadai untuk digunakan sebagai dasar pembentukan label dan analisis karakteristik data [12].



Gambar 4. Distribusi Label

Hasil *clustering* kemudian digunakan sebagai dasar pembentukan label kelas yang terdiri atas Normal dan Gangguan. Dataset yang telah melalui proses *preprocessing*, normalisasi, pembentukan *severity score*, *clustering*, serta pelabelan selanjutnya digunakan sebagai masukan pada tahap *Modeling* untuk membangun model klasifikasi menggunakan algoritma *Random Forest* dan *Naive Bayes*.

Tabel 3. Dataset Berdasarkan Label

Label	Jumlah Data	Persentase Dominan
Gangguan	1.475	54.21%
Nornal	1.246	45.79 %
Total	2.721	100 %

2.4 Modeling

Sebelum proses pelatihan pada model dimulai, dataset hasil *preprocessing* dan *clustering* dibagi menjadi data pelatihan (*training set*) dan data pengujian (*testing set*) menggunakan metode *train-test split*. Pembagian data bertujuan memastikan model dilatih menggunakan sebagian data, sedangkan sisa data digunakan untuk mengevaluasi kemampuan model dalam melakukan prediksi terhadap data yang belum

DOI: https://doi.org/10.55537/cosie.v5i4.1832

pernah dipelajari sebelumnya [13].

Tabel 4. Dataset Berdasarkan Label

Dataset	Jumlah Data	Persentase Dominan
Data Training	2.176	80%
Data Testing	545	20 %
Total	2.721	100 %

Proses pelatihan model diimplementasikan menggunakan *Scikit-learn* yang menyediakan berbagai algoritma klasifikasi, utilitas *preprocessing*, serta mekanisme validasi model dalam satu lingkungan pengembangan. Pemanfaatan *Scikit-learn* mempermudah integrasi proses *train-test split*, pelatihan model, prediksi, hingga evaluasi performa sehingga seluruh tahapan eksperimen dapat dilakukan secara konsisten.

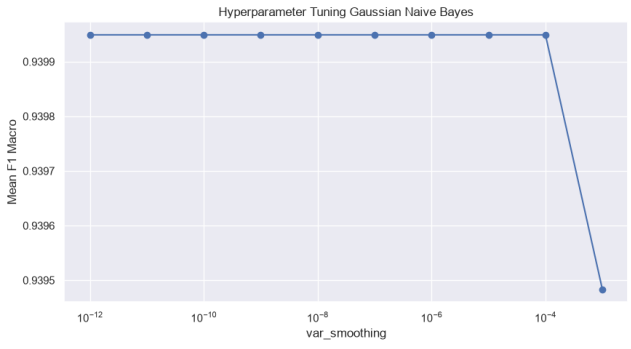
Model pertama yang dibangun menggunakan algoritma *Random Forest*. Implementasi dilakukan dengan membentuk sejumlah *decision tree* berdasarkan data pelatihan, kemudian menghasilkan prediksi akhir melalui mekanisme *majority voting*. Pada penelitian ini dilakukan proses *hyperparameter tuning* menggunakan *GridSearchCV* untuk memperoleh kombinasi parameter yang menghasilkan performa klasifikasi terbaik sebelum model digunakan pada proses pengujian [14].

Tabel 5. Paramgrid *Random Forest*

Parameter	Kandidat Nilai	Kegunaan
n_estimators	50, 100	Menentukan jumlah pohon
max_depth	3, 5, 7	Menentukan kedalaman
min_samples_split	10, 20, 30	Minimum Jumlah data split
min_samples_leaf	5, 10, 15	Minumum jumlah data node
max_features	sqrt	Jumlah fitur keputusan tiap split

Proses *hyperparameter tuning* dilakukan menggunakan **GridSearchCV** yang dipadukan dengan skema *cross-validation* untuk memperoleh konfigurasi parameter terbaik pada algoritma **Random Forest**. Setiap kombinasi parameter dievaluasi menggunakan data pelatihan, kemudian dipilih berdasarkan hasil evaluasi dengan performa tertinggi. Dengan cara ini, model yang digunakan pada tahap pengujian merupakan hasil konfigurasi yang telah dioptimalkan melalui proses pencarian parameter. Pendekatan tersebut diterapkan untuk meningkatkan kemampuan generalisasi model sekaligus mengurangi risiko terjadinya *overfitting*.

Model kedua dibangun menggunakan algoritma *Naive Bayes* sebagai pembanding terhadap *Random Forest*. Algoritma ini dilatih menggunakan dataset yang sama sehingga kedua model memperoleh karakteristik data pelatihan yang identik. Pendekatan tersebut diterapkan agar proses perbandingan dilakukan pada kondisi eksperimen yang setara, sehingga perbedaan performa yang dihasilkan berasal dari karakteristik masing-masing algoritma, bukan dari perbedaan data pelatihan [15].



Gambar 5. *var_smoothing Naive Bayes*

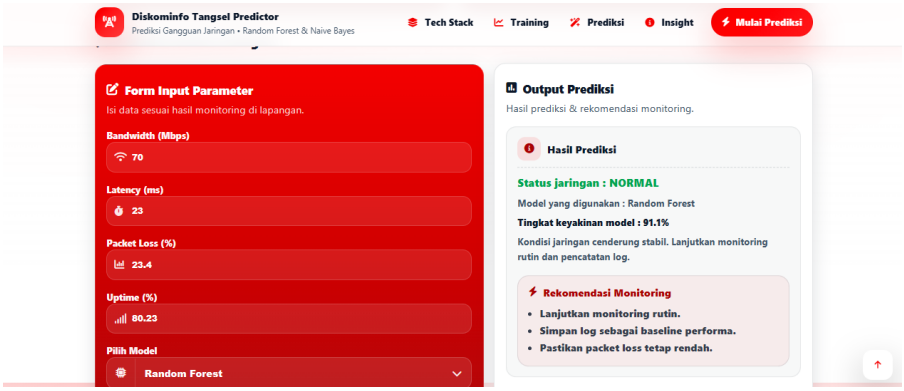
Berdasarkan Gambar 5. grafik Hyperparameter Tuning Gaussian *Naive Bayes*, perubahan nilai parameter var_smoothing dari 1e-12 hingga 1e-4 tidak memberikan pengaruh yang berarti terhadap performa model. Hal tersebut terlihat dari nilai Mean F1 Macro yang cenderung stabil pada seluruh rentang

DOI: https://doi.org/10.55537/cosie.v5i4.1832

parameter yang diuji. Nilai Mean F1 Macro tertinggi diperoleh pada var_smoothing = 1e-12 dengan nilai sekitar 0.9399. Meskipun demikian, selisih performa antar setiap nilai var_smoothing relatif kecil sehingga kualitas klasifikasi yang dihasilkan hampir sama. Setelah proses pelatihan selesai, kedua model digunakan untuk melakukan prediksi terhadap data pengujian. Hasil prediksi selanjutnya menjadi dasar dalam proses evaluasi menggunakan *confusion matrix*, *accuracy*, *precision*, *recall*, *F1-score*, serta analisis *feature importance* yang dibahas pada bagian hasil dan pembahasan. Tahapan ini memastikan setiap model dievaluasi menggunakan prosedur dan dataset yang sama sehingga hasil perbandingan dapat dianalisis secara objektif [16].

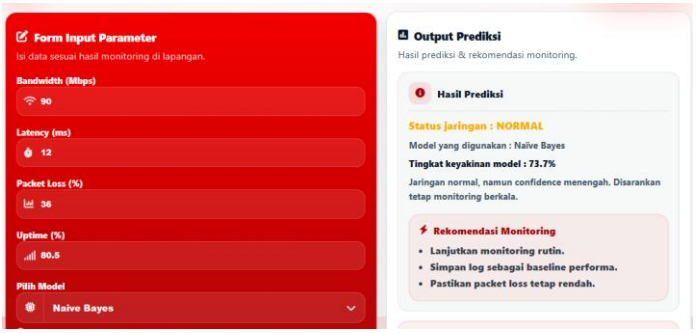
2.5 Deployment

Sebagai tahap akhir pada proses *Modeling*, model dengan konfigurasi terbaik disimpan menggunakan *Joblib* agar dapat dimuat kembali tanpa proses pelatihan ulang. Penyimpanan model dilakukan untuk mendukung tahap *deployment* pada aplikasi berbasis *Flask*, sehingga proses prediksi gangguan jaringan dapat dilakukan secara langsung melalui antarmuka web tanpa perlu membangun ulang model setiap kali sistem dijalankan [17].



Gambar 6. Form input paramater

Berdasarkan Gambar 6. Flask dashboard menyediakan *form* input yang terdiri atas empat parameter jaringan, yaitu *bandwidth* (Mbps), *latency* (ms), *packet loss* (%), dan *uptime* (%). Pengguna juga dapat memilih algoritma *Machine Learning* yang akan digunakan untuk melakukan prediksi, yaitu *Random Forest* atau *Naive Bayes*. Model menghasilkan prediksi Normal dengan tingkat keyakinan (*confidence score*) sebesar 91,1%. Hasil tersebut menunjukkan bahwa kondisi jaringan masih berada dalam keadaan stabil dan beroperasi pada kisaran yang normal.



Gambar 7. Output *Naive Bayes*

Pada Gambar 7. model *Naive Bayes* mengklasifikasikan kondisi jaringan ke dalam kategori Normal dengan tingkat keyakinan (*confidence score*) sebesar 73,7%. Hasil tersebut menunjukkan bahwa karakteristik parameter jaringan yang dimasukkan masih lebih mendekati kondisi Normal dibandingkan Gangguan.

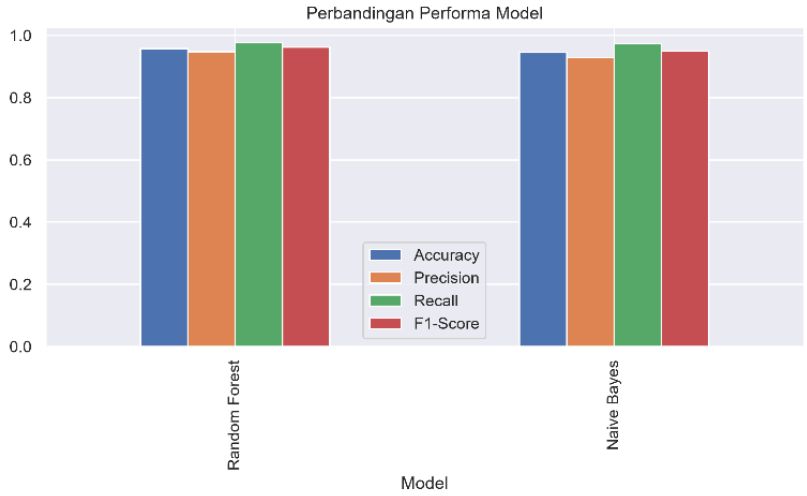
3. HASIL DAN PEMBAHASAN

3.1 Hasil Pembahasan Data Setelah *Preprocessing*

Setelah melalui tahap *preprocessing*, diperoleh 2.721 data yang siap digunakan dengan empat parameter utama, yaitu *bandwidth*, *latency*, *packet loss*, dan *uptime*. Pelabelan data dilakukan menggunakan

DOI: https://doi.org/10.55537/cosie.v5i4.1832

pendekatan *severity scoring* sehingga menghasilkan dua kelas, yaitu Normal sebanyak 1.475 data (54,2%) dan Gangguan sebanyak 1.246 data (45,8%). Hasil validasi menggunakan K-Means Clustering memperoleh nilai Silhouette Score sebesar 0,3648, yang menunjukkan bahwa kedua kelompok data memiliki tingkat pemisahan yang cukup baik. Model yang telah dibangun sudah diimplementasikan ke dalam Flask App sehingga dapat dimanfaatkan secara langsung oleh tim teknisi untuk memprediksi kondisi jaringan.



Gambar 8. Perbandingan Algoritma

Tabel 6. Perbandingan Algoritma *Random Forest* dan *Naive Bayes*

Model Algoritma	Precision	Akurasi	Recall	F1-Score
Random Forest	94.74	95.78 %	97.63	96.16
Naive Bayes	92.88	94.50 %	97.29	95.03

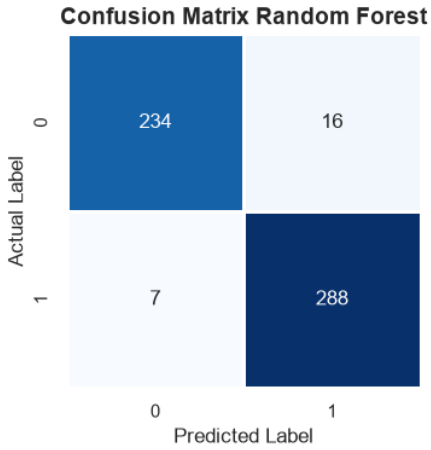
3.2 Perbandingan Confuson Matrix

Berdasarkan hasil Confusion Matrix, algoritma *Random Forest* mampu membedakan kondisi jaringan Normal dan Gangguan dengan sangat baik. Model berhasil mengklasifikasikan 234 data Normal sebagai True Negative dan 288 data Gangguan sebagai *True Positive*, sehingga sebagian besar data pengujian berhasil diprediksi sesuai dengan kondisi sebenarnya.

Kesalahan klasifikasi terdiri atas 16 data Normal yang diprediksi sebagai Gangguan (*False Positive*) dan 7 data Gangguan yang diprediksi sebagai Normal (*False Negative*). Jumlah kesalahan tersebut relatif kecil, terutama pada *False Negative*, yang menunjukkan bahwa model mampu mendeteksi sebagian besar gangguan jaringan.

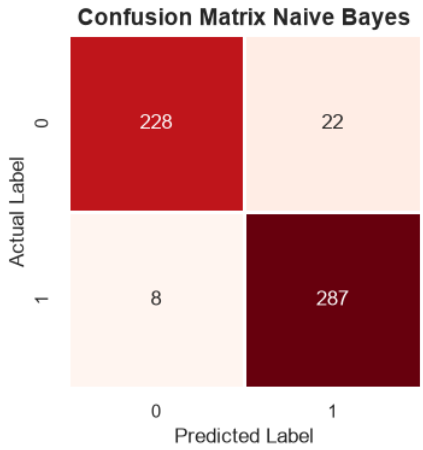
Confusion Matrix digunakan untuk mengidentifikasi jenis kesalahan prediksi yang dihasilkan oleh model. Dari seluruh jenis kesalahan, *False Negative*, yaitu kondisi gangguan yang diprediksi sebagai normal, menjadi kesalahan yang paling perlu diperhatikan.

DOI: https://doi.org/10.55537/cosie.v5i4.1832



Gambar 9. Confusion Matrix - *Random Forest*

Dari total 545 data pengujian, 522 data berhasil diklasifikasikan dengan benar. Hasil tersebut menunjukkan bahwa algoritma *Random Forest* memiliki tingkat akurasi yang tinggi dengan jumlah kesalahan prediksi yang relatif rendah pada kedua kelas.



Gambar 10. Confusion Matrix – *Naive Bayes*

Berdasarkan Gambar 10. hasil Confusion Matrix, algoritma *Naive Bayes* mampu mengklasifikasikan kondisi jaringan Normal dan Gangguan dengan baik. Model berhasil mengidentifikasi 228 data Normal sebagai True Negative dan 287 data Gangguan sebagai *True Positive*, sehingga sebagian besar data pengujian berhasil diprediksi sesuai dengan kondisi sebenarnya.

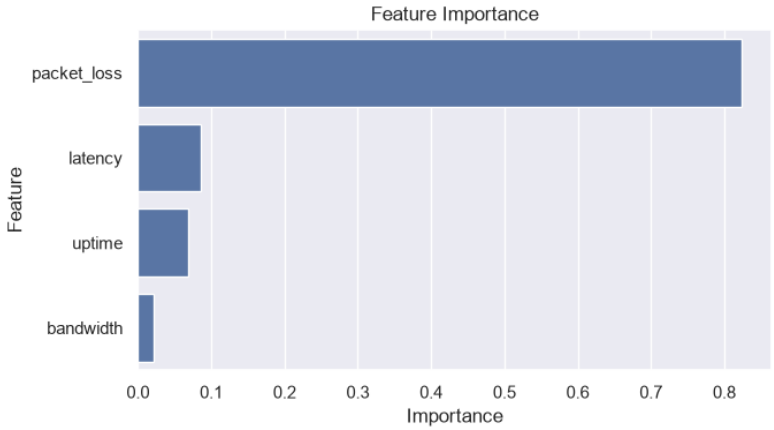
Kesalahan klasifikasi terdiri atas 22 data Normal yang diprediksi sebagai Gangguan (*False Positive*) dan 8 data Gangguan yang diprediksi sebagai Normal (*False Negative*). Meskipun jumlahnya relatif kecil, nilai *False Positive* yang lebih tinggi menunjukkan bahwa model cenderung lebih sering memberikan peringatan gangguan pada kondisi jaringan yang masih normal.

Dari total 545 data pengujian, 515 data berhasil diklasifikasikan dengan benar. Hasil tersebut menunjukkan bahwa algoritma *Naive Bayes* memiliki tingkat akurasi yang tinggi, meskipun performanya masih sedikit berada di bawah *Random Forest* dalam membedakan kondisi jaringan Normal dan Gangguan

3.3 Analisa *Feature Importance*

Analisis Feature Importance digunakan untuk mengetahui besarnya kontribusi setiap parameter jaringan dalam proses klasifikasi. Melalui analisis ini dapat diidentifikasi fitur yang paling berpengaruh dalam membedakan kondisi jaringan Normal dan Gangguan.

DOI: https://doi.org/10.55537/cosie.v5i4.1832



Gambar 10. *Feature Importance*

Hasil analisis *Feature Importance* menunjukkan bahwa *packet loss* merupakan parameter yang paling berpengaruh dalam membedakan kondisi jaringan Normal dan Gangguan. Dominasi nilai *importance* pada parameter ini mengindikasikan bahwa peningkatan *packet loss* lebih sering berkaitan dengan penurunan kualitas jaringan dibandingkan parameter lainnya. Di sisi lain, *latency* dan *uptime* juga berkontribusi dalam proses klasifikasi, meskipun pengaruhnya tidak sebesar *packet loss*. Temuan tersebut menunjukkan bahwa model tidak hanya mampu mencapai tingkat akurasi yang tinggi, tetapi juga dapat mengidentifikasi parameter yang memiliki peran paling signifikan dalam proses prediksi gangguan jaringan.

4. KESIMPULAN

Kedua, hasil pengujian menunjukkan bahwa algoritma *Random Forest* memberikan performa yang lebih baik dibandingkan *Naive Bayes*. Pada 545 data pengujian, *Random Forest* memperoleh accuracy sebesar 95,78%, precision 94,74%, recall 97,63%, dan F1-Score 96,16%. Sementara itu, *Naive Bayes* menghasilkan accuracy 94,50%, precision 92,88%, recall 97,29%, dan F1-Score 95,03%. Keunggulan tersebut juga didukung oleh hasil *5-fold cross-validation*, di mana *Random Forest* memperoleh rata-rata F1-Score sebesar 0,9551, lebih tinggi dibandingkan *Naive Bayes* yang mencapai 0,9467. Analisis *Feature Importance* menunjukkan bahwa *packet loss* merupakan faktor yang paling berpengaruh dalam menentukan kondisi jaringan dengan kontribusi sebesar 82,32%, diikuti oleh *latency* (8,58%), *uptime* (6,90%), dan *bandwidth* (2,20%). Hasil tersebut menunjukkan bahwa *packet loss* menjadi indikator utama dalam mendeteksi gangguan jaringan. Berdasarkan keseluruhan hasil evaluasi, algoritma *Random Forest* dipilih sebagai model terbaik pada penelitian ini. Penelitian ini menggunakan data hasil *capturing* jaringan yang berasal dari satu lingkungan operasional, yaitu Diskominfo Kota Tangerang Selatan. Oleh sebab itu, performa model yang diperoleh masih mencerminkan karakteristik jaringan pada lingkungan tersebut dan belum dapat menggambarkan kinerjanya pada jaringan dengan topologi, perangkat, maupun pola trafik yang berbeda. Proses pembentukan label juga disusun berdasarkan pendekatan *severity score* dan *clustering* yang disesuaikan dengan karakteristik dataset penelitian. Untuk mengetahui tingkat generalisasi model secara lebih menyeluruh, diperlukan pengujian menggunakan dataset dari organisasi lain atau pada lingkungan jaringan dengan skenario operasional yang berbeda.

DAFTAR PUSTAKA

[1] K. Murphy, A. Lavignotte, and C. Lepers, “Fault Prediction for Heterogeneous Telecommunication Networks Using Machine Learning: A Survey,” *IEEE Trans. Netw. Serv. Manag.*, vol. 21, no. 2, pp. 2515–2538, 2024, doi: 10.1109/TNSM.2023.3340351.

[2] M. M. Alamin, A. R. Firmansyah, A. Bittuqoh, C. B. Adzimi, M. I. Wahyudi, and M. Z. AT, “Pengukuran Performa Jaringan Internet Menggunakan Quality of Service dengan Wireshark,” *Nusant. Comput. Des. Rev.*, vol. 3, no. 1, pp. 9–14, 2025, doi: 10.55732/ncdr.v3i1.1633.

[3] K. Putra, F., & Andesa, “Prediksi Nilai Redaman Jaringan Fiber Optik Untuk Menilai Kinerja Jaringan Menggunakan *Random Forest* Regression,” vol. x, no. x, pp. 3347–3361, 2025, [Online]. Available: https://doi.org/10.33022/ijcs.v14i2.4796

[4] A. K. Sah and K. Venkatesh, “Anomaly-Based Intrusion Detection in Network Traffic using Machine Learning: A Comparative Study of Decision Trees and *Random Forest*s,” *Proc. 2nd IEEE Int. Conf. Netw. Commun. 2024, ICNWC 2024*, 2024, doi: 10.1109/ICNWC60771.2024.10537451.

[5] S. N. Ervansah, A. P. Kusuma, and Y. Primasari, "Penerapan *Naive Bayes* pada Sistem Pakar Pendeteksi Jaringan Internet di Rosi Cell.Net," *JASIEK (Jurnal Apl. Sains, Informasi, Elektron. dan Komputer)*, vol. 6, no. 2, pp. 167–176, 2024, doi: 10.26905/jasiek.v6i2.14049.

[6] Y. Yuliani, "Perbandingan Algoritma Klasifikasi untuk Deteksi Intrusi pada Jaringan Komputer (Literature Review)," *J. Multidiscip. Inq. Sci. Technol. Educ. Res.*, vol. 1, no. 3c, pp. 1687–1695, 2024.

[7] R. Ben Said, Z. Sabir, and I. Askerzade, "CNN-BiLSTM: A Hybrid Deep Learning Approach for Network Intrusion Detection System in Software-Defined Networking With Hybrid Feature Selection," *IEEE Access*, vol. 11, no. November, pp. 138732–138747, 2023, doi: 10.1109/ACCESS.2023.3340142.

[8] Elan Suherlan, Shindy Arti, Siti Nabilah, and Zahwah Hazimah, "Penerapan Flask Framework Untuk Deployment Model Machine Learning Dalam Mendukung Analisis Adaptasi Mahasiswa Pada Pembelajaran Daring," *PINTER J. Pendidik. Tek. Inform. dan Komput.*, vol. 9, no. 1, pp. 110–117, 2025, doi: 10.21009/pinter.9.1.15.

[9] A. Rianti *et al.*, "CRISP-DM: Metodologi Proyek Data Science," *Pros. Semin. Nas. Teknol. Inf. dan Bisnis*, pp. 107–114, 2023, [Online]. Available: https://ojs.udb.ac.id/index.php/Senatib/article/view/3015

[10] A. Pannadhitthana Candra, "Analisis Data Menggunakan Python: Memperkenalkan Pandas dan NumPy," *J. Inf. Syst. Educ. Dev.*, vol. 3, no. 1, pp. 11–16, 2025, doi: 10.62386/jised.v3i1.118.

[11] R. Syahri, T. Informatika, and S. Selatan, "Algoritma K-Means Clustering : Sebuah Studi Literatur K-Means Clustering Algorithm : a Literatur Study," vol. x, no. x, pp. 1–7, 2023, doi: 10.12345/juri.

[12] A. Fauzan, N. Suarna, I. Ali, and H. Susana, "Penerapan Algoritma K-Means Clustering Untuk Meningkatkan Model Pengelompokan Dan Kinerja Jaringan Wi-Fi Secara Optimal," *J. Inform. dan Tek. Elektro Terap.*, vol. 13, no. 2, 2025, doi: 10.23960/jitet.v13i2.6272.

[13] R. More and J. S. Bradbury, "Assessing Data Augmentation-Induced Bias in Training and Testing of Machine Learning Models," *Proc. - 2025 IEEE Int. Conf. Softw. Anal. Evol. Reengineering - Companion, SANER-C 2025*, pp. 57–60, 2025, doi: 10.1109/SANER-C66551.2025.00015.

[14] I. Saputri, P. Arsi, and K. N. Isnaini, "Effectiveness Hyperparameter Tuning on *Random Forest*, Linear Discriminant Analysis, Logistic Regression and *Naive Bayes* Algorithms for Detecting Dos Network Attacks," *J. Tek. Inform.*, vol. 6, no. 1, pp. 87–104, 2025, doi: 10.52436/1.jutif.2025.6.1.4175.

[15] W. Djatmiko, Kusrini, and Hanafi, "Perbandingan *Naive Bayes* dan *Random Forest* untuk Prediksi Perilaku Peserta Program Rujuk Balik," *J. Fasilkom*, vol. 13, no. 3, pp. 358–367, 2023, doi: 10.37859/jf.v13i3.6070.

[16] D. Lusiyanti, S. Musdalifah, A. Sahari, and I. Al Fajri, "Evaluasi Kinerja Algoritma Machine learning pada Dataset Skala Besar," *MathVision J. Mat.*, vol. 7, no. 1, pp. 84–92, 2025, doi: 10.55719/mv.v7i1.1661.

[17] T. Adekoya-Cole, S. Fernando, A. Maroju, C. K. Samal, S. Aggarwal, and B. Brahma, "Building a dynamic opinion dashboard to categorize tweets for real-time sentiment analysis," *Int. J. Inf. Technol.*, vol. 18, no. 1, pp. 5–11, 2026, doi: 10.1007/s41870-025-02495-z.