ANALISIS KINERJA *RANDOM FOREST* DENGAN *FEATURE SELECTION* BERBASIS KARAKTERISTIK *MALWARE* DALAM MENDETEKSI SERANGAN *MALWARE*

SKRIPSI

Oleh :
ADE HASBULAH
NIM. 220605110079

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2026

ANALISIS KINERJA *RANDOM FOREST* DENGAN *FEATURE SELECTION* BERBASIS KARAKTERISTIK *MALWARE* DALAM MENDETEKSI SERANGAN *MALWARE*

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
ADE HASBULAH
NIM. 220605110079

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

HALAMAN PERSETUJUAN

ANALISIS KINERJA *RANDOM FOREST* DENGAN *FEATURE SELECTION* BERBASIS KARAKTERISTIK *MALWARE* DALAM MENDETEKSI SERANGAN *MALWARE*

SKRIPSI

Oleh:
ADE HASBULAH
220605110079

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 12 Juni 2026

Pembimbing I,	Pembimbing II,
Dr. M. Amin Hariyadi, M.T	Nurizal Dwi Priandani, M.Kom
NIP. 19670118 2005011 001	NIP. 19920830 2022031 001

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M.Kom
NIP. 19841010 201903 1 012

HALAMAN PENGESAHAN

ANALISIS KINERJA *RANDOM FOREST* DENGAN *FEATURE SELECTION* BERBASIS KARAKTERISTIK *MALWARE* DALAM MENDETEKSI SERANGAN *MALWARE*

SKRIPSI

Oleh:
ADE HASBULAH
NIM. 220605110079

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal: 23 Juni 2026

Susunan Dewan Penguji

Ketua Penguji	: Johan Ericka Wahyu Prakasa, M.Kom NIP. 19831213 201903 1 004	( )
Anggota Penguji I	: Ajib Hanani, M.T NIP. 19840731 2023211 013	( )
Anggota Penguji II	: Dr. M. Amin Hariyadi, M.T NIP. 19670118 2005011 001	( )
Anggota Penguji III	: Nurizal Dwi Priandani, M.Kom NIP. 19920830 2022031 001	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M.Kom
NIP. 19841010 201903 1 012

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

1. Nama : Ade Hasbulah
   NIM : 220605110079
   Fakultas / Program Studi : Sains dan Teknologi / Teknik Informatika
   Judul Skripsi : Analisis Kinerja Random Forest Dengan Feature Selection Berbasis Karakteristik Malware Dalam Mendeteksi Serangan Malware

Menyatakan dengan sebenarnya bahwa Skirpsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsian ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Juni 2026
Yang membuat pernyatan,

Ade Hasbulah
NIM.220605110079

MOTTO

"Failure is only the opportunity to begin again, only this time more wisely"

"In the darkest times, hope is something you give yourself, that is the meaning of inner strength"
(uncle iroh)

HALAMAN PERSEMBAHAN

Dengan penuh rasa syukur kepada Allah SWT atas limpahan rahmat dan kemudahan-Nya, akhirnya skripsi ini dapat terselesaikan dengan baik. Karya ini penulis persembahkan kepada:

Ayah, Ibu, dan kakak

atas dukungan, doa, dan kasih sayang tanpa henti yang selalu menjadi sumber kekuatan dan motivasi.

Dosen pembimbing dan para pengajar

atas bimbingan, arahan, serta ilmu yang sangat berarti bagi penulis.

Teman-teman yang ikut serta membantu dari awal pembuatan skripsi

yang telah menjadi bagian dari perjalanan ini melalui kebersamaan dan semangat yang selalu membara dalam mengerjakan skripsi.

Dan yang terakhir, untuk diri sendiri

Terima kasih telah terus berusaha dan tidak menyerah sehingga bisa menyelesaikan skripsi ini dengan baik hingga akhir.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillahirabbil'alamin, segala puji syukur penulis panjatkan ke hadirat Allah SWT. atas limpahan rahmat, nikmat, dan hidayah-Nya, sehingga penulis dapat menyelesaikan tugas skripsi yang berjudul "ANALISIS KINERJA *RANDOM FOREST* DENGAN *FEATURE SELECTION* BERBASIS KARAKTERISTIK *MALWARE* DALAM MENDETEKSI SERANGAN *MALWARE*" dengan baik dan lancar. Shalawat serta salam tak lupa penulis sampaikan kepada Nabi Muhammad ShallAllahu 'Alaihi Wasallam, sosok teladan mulia yang telah menuntun kita dari zaman kegelapan menuju zaman kebenaran, Islam, dan ilmu pengetahuan yang kita rasakan manfaatnya hingga hari ini.

Tugas skripsi ini disusun dalam rangka memenuhi salah satu syarat kelulusan untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang. Dalam proses penyusunan tugas ini, penulis mendapatkan bantuan, dukungan, serta doa dari banyak pihak, baik secara langsung maupun tidak langsung. Untuk itu, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang.
2. Dr. H. Agus Mulyono, S.Pd., M.Kes, selaku Dekan Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang.
3. Supriyono, M.Kom., selaku Ketua Program Studi S1 Teknik Informatika UIN Maulana Malik Ibrahim Malang.

4. Dr. M. Amin Hariyadi, M.T, selaku Dosen Pembimbing I, yang dengan kesabaran dan dedikasinya membimbing penulis selama penyusunan skripsi.
5. Nurizal Dwi Priandani, M.Kom, selaku Dosen Pembimbing II, yang senantiasa memberikan arahan dan masukan yang sangat berharga dalam pengerjaan skripsi ini.
6. Johan Ericka Wahyu Prakasa, M.Kom, selaku Dosen Penguji yang telah memberikan kritik, saran, dan bimbingan untuk menyempurnakan skripsi.
7. Ajib Hanani, M.T, selaku Dosen Penguji yang telah memberikan kritik, saran, dan bimbingan untuk menyempurnakan skripsi ini.
8. Nia Faricha, S.Si., admin Program Studi Teknik Informatika, yang membantu dalam urusan administrasi dan selalu mengingatkan tentang kelengkapan berkas.
9. Bapak dan Ibu tercinta, serta kakak saya, yang senantiasa menjadi sumber kekuatan melalui doa, kasih sayang, dan dukungan tanpa henti.
10. Teman dekat kamar ma'had, yang selalu ada ketika penulis sedang dalam kesulitan dalam hal apapun.
11. Teman-teman kost, yang senantiasa memberikan waktu luang kepada penulis untuk sejenak menghibur diri.
12. Seluruh keluarga, teman, sahabat, dan kerabat penulis yang tidak dapat penulis sebutkan satu per satu, yang turut memberikan bantuan, semangat, dukungan, dan doa untuk penulis.

Dan terakhir tak lupa terimakasih kepada diri sendiri, yang dengan tekad kuat mampu melewati setiap tantangan, rintangan, dan kesulitan selama perjalanan

ini. Penulis menyadari bahwa penelitian dalam tugas skripsi ini masih jauh dari kata sempurna dan memiliki berbagai keterbatasan. Oleh karena itu, dengan penuh kerendahan hati, penulis membuka diri untuk menerima kritik dan saran yang membangun dari para pembaca guna menjadi bahan evaluasi dan pengembangan di masa mendatang. Penelitian ini juga memiliki potensi untuk dilanjutkan dan dikembangkan lebih jauh dalam penelitian berikutnya, sehingga dapat melengkapi kekurangan yang ada. Penulis berharap, karya ini tidak hanya memberikan manfaat bagi pembaca, tetapi juga dapat berkontribusi positif bagi masyarakat secara luas.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Malang, 12 Juni 2026

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Hasbulah, Ade. 2026. **Analisa Kinerja *Random Forest* Dengan *Feature selection* Berbasis Karakteristik *Malware* Dalam Mendeteksi Serangan *Malware***. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. M. Amin Hariyadi, M.T (II) Nurizal Dwi Priandani, M.Kom.

Kata Kunci: *Feature selection, Malware*, *MTA-KDD'19, Random Forest*.

Serangan *malware* pada lingkungan *cloud server* terus meningkat seiring berkembangnya teknologi *cloud computing*, sehingga diperlukan metode deteksi yang akurat dan efisien. Penelitian ini bertujuan untuk menganalisis kinerja algoritma *Random Forest* dengan *feature selection* dalam mendeteksi *malware* menggunakan dataset MTA-KDD'19. Tahapan penelitian meliputi praproses data, normalisasi, pembagian data dengan skenario 70:30, 80:20, dan 90:10, pelatihan model, serta evaluasi menggunakan *confusion matrix, accuracy, precision, recall, F1-score*, dan 10-*Fold Cross Validation*. Hasil penelitian menunjukkan bahwa model *Random Forest* dengan *feature selection* mampu menghasilkan performa yang baik pada seluruh skenario pengujian. Skenario 80:20 dan 90:10 memberikan hasil terbaik dengan *accuracy* 93,8%, *precision 93,8%, recall 93,8% , dan F1-score* 93,8%. Hasil uji ablasi menunjukkan bahwa beberapa fitur yang tidak dipilih masih memiliki kontribusi terhadap performa klasifikasi, sehingga penggunaan seluruh fitur menghasilkan performa yang lebih tinggi. Berdasarkan hasil tersebut, algoritma *Random Forest* dengan *feature selection* mampu mendeteksi *malware* pada lingkungan *cloud server* dengan akurasi yang baik. Meskipun demikian, hasil uji ablasi menunjukkan bahwa pengurangan fitur belum mampu meningkatkan performa model dibandingkan penggunaan seluruh fitur.

ABSTRACT

Hasbulah, Ade. 2026. **Performance Analysis of Random Forest with Feature Selection Based on Malware Characteristics in Detecting Malware Attacks**. Undergraduate thesis. Department of Computer Science, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University, Malang. Supervisors: (I) Dr. M. Amin Hariyadi, M.T (II) Nurizal Dwi Priandani, M.Kom.

The increasing prevalence of malware attacks in cloud server environments, driven by the advancement of cloud computing technology, necessitates accurate and efficient detection methods. This research aims to analyze the performance of the Random Forest algorithm with feature selection in detecting malware using the MTA-KDD'19 dataset. The research stages include data preprocessing, normalization, data splitting with scenarios of 70:30, 80:20, and 90:10, model training, and evaluation using a confusion matrix, accuracy, precision, recall, F1-score, and 10-Fold Cross Validation. The research results indicate that the Random Forest model with feature selection is capable of achieving good performance across all testing scenarios. The 80:20 and 90:10 scenarios yielded the best results with an accuracy of 93.8%, precision of 93.8%, recall of 93.8%, and an F1-score of 93.8%. Ablation test results show that some unselected features still contribute to classification performance, thus using all features results in higher performance. Based on these findings, the Random Forest algorithm with feature selection can effectively detect malware in cloud server environments with good accuracy. Nevertheless, the ablation test results suggest that feature reduction has not yet improved model performance compared to the use of all features.

Keywords: Feature selection, Malware, MTA-KDD'19, Random Forest.

مستخلص البحث

حسب الله، آدي. 2026. تحليل أداء الغابة العشوائية مع اختيار الميزات بناءً على خصائص البرامج الضارة في اكتشاف هجمات البرامج الضارة. رسالة ماجستير. قسم هندسة المعلومات، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية .نوريزال دوي برياندا، م.كوم (II) د. م. أمين هريادي، م.ت (I) :الحكومية، مالانج. المشرفون

.الغابة العشوائية ،MTA-KDD'19 ،الكلمات المفتاحية: اختيار الميزات، البرامج الضارة

تستمر هجمات البرامج الضارة في بيئة خوادم السحابة في الازدياد مع تطور تقنية الحوسبة السحابية، مما يتطلب طرق اكتشاف دقيقة وفعالة. تهدف هذه الدراسة إلى تحليل أداء خوارزمية الغابة العشوائية مع اختيار الميزات في اكتشاف البرامج الضارة باستخدام مجموعة ،تشمل مراحل البحث معالجة البيانات المسبقة، والتطبيع، وتقسيم البيانات بسيناريوهات 70:30 .MTA-KDD'19 بيانات والتحقق المتقاطع ،F1 و 90:10، وتدريب النموذج، والتقييم باستخدام مصفوفة الارتباك، والدقة، والاستدعاء، ومقياس ،80:20 أضعاف. أظهرت نتائج البحث أن نموذج الغابة العشوائية مع اختيار الميزات قادر على تحقيق أداء جيد في جميع سيناريوهات 10 الاختبار. قدم سيناريو 80:20 و 90:10 أفضل النتائج بدقة 93.8%، ودقة استدعاء 93.8%، واستدعاء 93.8%، ومقياس أظهرت نتائج اختبار الاستئصال أن بعض الميزات التي لم يتم اختيارها لا تزال تساهم في أداء التصنيف، وبالتالي فإن .%93.8 F1 استخدام جميع الميزات ينتج أداءً أعلى. بناءً على هذه النتائج، فإن خوارزمية الغابة العشوائية مع اختيار الميزات قادرة على اكتشاف البرامج الضارة في بيئة خوادم السحابة بدقة جيدة. ومع ذلك، تشير نتائج اختبار الاستئصال إلى أن تقليل الميزات لم يتمكن من .تحسين أداء النموذج مقارنة باستخدام جميع الميزات

BAB I

PENDAHULUAN

1.1 Latar Belakang

Evolusi teknologi *cloud computing* menjadikan *cloud server* sebagai infrastruktur utama dalam penyediaan layanan digital modern. Fleksibilitas dan skalabilitas yang ditawarkan *cloud server* memungkinkan berbagai organisasi menyimpan, mengelola, dan memproses data secara efisien. Namun, keterbukaan akses jaringan pada lingkungan *cloud* juga meningkatkan risiko terhadap berbagai ancaman keamanan siber, khususnya serangan *malware* yang memanfaatkan lalu lintas jaringan untuk menyusup, menyebar, dan mengeksploitasi sistem.

Malware merupakan salah satu ancaman keamanan yang paling dominan pada lingkungan *cloud server* karena mampu menyamar sebagai trafik normal, menyebar secara cepat, serta menyebabkan kebocoran data, kerusakan sistem, dan gangguan layanan. Seiring meningkatnya kompleksitas serangan siber, metode deteksi konvensional menjadi semakin sulit untuk mengidentifikasi pola serangan yang beragam. Maka dari itu, dibutuhkan sistem deteksi yang sanggup mengenali karakteristik *malware* dengan efektif dan akurat.

Cloud server menawarkan kemudahan dalam pengelolaan sumber daya komputasi dengan tingkat fleksibilitas dan skalabilitas yang tinggi. Menurut Mubarok dan Romli (2025).

Jaringan publik juga menjadikannya rentan terhadap berbagai ancaman keamanan, termasuk serangan *malware* yang dapat mengganggu ketersediaan, integritas, dan kerahasiaan data.

Salah satu cara banyak dipakai dalam mendeteksi *malware* ialah penggunaan machine learning dengan *Intrusion Detection System* (IDS). *Machine learning* memudahkan sistem dalam mempelajari pola lalu lintas jaringan berdasarkan data historis sehingga mampu melakukan klasifikasi antara trafik normal dan trafik *malware* secara otomatis. Pendekatan ini dinilai lebih adaptif dibandingkan metode deteksi tradisional karena mampu mengenali karakteristik serangan dari data yang telah dipelajari.

Dari semua algoritma *machine learning*, *Random Forest* ialah salah satu metode yang digunakan di bidang keamanan jaringan karena memiliki kemampuan klasifikasi yang baik, tahan terhadap *overfitting*, juga bisa menangani data berdimensi tinggi. *Random Forest* bekerja dengan membangun beberapa pohon keputusan yang selanjutnya menghasilkan prediksi berdasar mekanisme *majority voting*. Beberapa peneliti menunjukkan kalau *Random Forest* sanggup memberikan akurasi tinggi dalam mendeteksi aktivitas serangan jaringan maupun *malware*.

Meskipun demikian, performa *machine learning* dipengaruhi dari kualitas fitur yang digunakan. Dataset lalu lintas jaringan umumnya memiliki banyak atribut yang tidak semuanya berkontribusi secara signifikan terhadap proses klasifikasi. Penggunaan seluruh fitur yang tersedia dapat meningkatkan kompleksitas komputasi, memperpanjang waktu pelatihan model, dan berpotensi menurunkan performa klasifikasi akibat adanya fitur tidak relevan. Maka dari itu, diperlukan

proses *feature selection* agar dipilih fitur yang representatif terhadap karakteristik *malware*.

Feature selection merupakan proses pemilihan fitur yang tepat dari sekumpulan fitur yang ada pada dataset. *feature selection* dilakukan berdasarkan karakteristik *malware* yang telah diidentifikasi pada penelitian-penelitian terdahulu. Fitur-fitur yang dipilih meliputi karakteristik TCP *Flag*, protokol komunikasi, ukuran paket, *interval* waktu antar paket, serta pola koneksi jaringan yang diketahui memiliki keterkaitan dengan aktivitas *malware*. Dengan menggunakan fitur yang relevan, model dapat mempelajari *malware* secara efektif lalu menghasilkan deteksi yang lebih dari sebelumnya.

Indonesia sebagai negara dengan tingkat pertumbuhan pengguna internet yang tinggi juga menghadapi risiko serangan *siber* yang semakin meningkat. Berbagai laporan menunjukkan bahwa aktivitas *malware* masih menjadi salah satu ancaman utama terhadap *server* dan layanan digital. Kondisi ini menunjukkan pentingnya pengembangan sistem deteksi *malware* yang mampu bekerja secara akurat dan efisien untuk mendukung keamanan infrastruktur teknologi informasi.

Dalam perspektif Islam, Allah SWT telah berfirman dalam surat *Al Baqarah* ayat 188 yang menjelaskan bahwasanya

وَلَا تَأْكُلُوْٓا اَمْوَالَكُمْ بَيْنَكُمْ بِالْبَاطِلِ وَتُدْلُوْا بِهَآ اِلَى الْحُكَّامِ لِتَأْكُلُوْا فَرِيْقًا مِّنْ اَمْوَالِ النَّاسِ بِالْاِثْمِ وَاَنْتُمْ تَعْلَمُوْنَ ۝

"Dan janganlah sebahagian kamu memakan harta sebahagian yang lain di antara kamu dengan jalan yang bathil dan (janganlah) kamu membawa (urusan) harta itu kepada hakim, supaya kamu dapat memakan sebahagian daripada harta benda orang lain itu dengan (jalan berbuat) dosa, Padahal kamu mengetahui".(QS. Al Baqarah : 188)

Menurut tafsir Jalalain pada ayat diatas dijelaskan bahwa larangan untuk orang-orang yang memakan harta orang-orang yang lain melalui jalan yang batil, yang adalah jalan haram menurut syariat islam, seperti mencuri milik orang lain, melakukan intimidasi dan sebagainya.

Berdasarkan kondisi ini, penelitian difokuskan dengan analisis performa *Random Forest* dengan *feature selection* berbasis karakteristik *malware* dalam mendeteksi serangan *malware* pada *cloud server* menggunakan dataset MTA-KDD'19. Dataset ini dipilih karena merepresentasikan lalu lintas jaringan pada lingkungan *cloud* secara realistis dan menyediakan data trafik normal maupun trafik *malware*. Evaluasi dilakukan menggunakan metrik *accuracy, precision, recall,* dan *F1-score* untuk mengetahui tingkat efektivitas model dalam mendeteksi serangan *malware*. Hasil penelitian diharapkan dapat memberikan kontribusi dalam pengembangan sistem deteksi *malware* yang akurat, efisien, dan sesuai untuk diterapkan pada lingkungan *cloud server*.

1.2 Identifikasi Masalah

Bagaimana performa metode *random forest* dengan *feature selection* dalam mendeteksi serangan *malware*?

1.3 Batasan Masalah

1. Analisis dibatasi pada data trafik jaringan yang terdapat dalam dataset MTA-KDD'19.
2. Parameter yang dianalisis difokuskan pada fitur-fitur trafik jaringan yang relevan dengan pola serangan *malware*.
3. Penelitian tidak membahas proses pembuatan karakteristik *malware* secara dinamis, melainkan menggunakan pola serangan yang telah tersedia pada dataset.

1.4 Tujuan Penelitian

Mengetahui performa metode *random forest* dengan *feature selection* dalam mendeteksi serangan *malware.*

1.5 Manfaat Penelitian

Penelitian ini memberikan kontribusi berupa analisis kinerja metode *random forest* dengan *feature selection* dalam mendeteksi serangan *malware* pada lingkungan *cloud server*. Hasil penelitian ini diharapkan dapat memberikan gambaran objektif mengenai tingkat akurasi, efektivitas, dan efisiensi metode *random forest* ketika diterapkan pada dataset yang merepresentasikan lalu lintas jaringan *cloud.*

Selain itu, penelitian ini dapat menjadi dasar evaluasi terhadap relevansi metode *random forest* di tengah berkembangnya pendekatan deteksi berbasis *anomaly* dan pembelajaran mesin yang lebih kompleks. Dengan adanya hasil evaluasi yang terukur, penelitian ini diharapkan mampu menjadi referensi dalam pengembangan sistem deteksi *malware* yang seimbang antara akurasi deteksi dan penggunaan sumber daya sistem.

Penelitian ini juga berkontribusi sebagai referensi akademik bagi penelitian selanjutnya yang ingin melakukan pengembangan, perbandingan, atau integrasi metode *random forest* dengan pendekatan deteksi lainnya pada lingkungan *cloud server*.

BAB II

STUDI PUSTAKA

2.1 Dasar Teori

2.1.1 *Malware*

Malware (*malicious software*) merupakan perangkat lunak berbahaya yang dirancang untuk merusak, mengganggu, atau memperoleh akses tidak sah ke sistem komputer. Jenis *malware* meliputi *virus, worm, trojan, ransomware, spyware*, dan *botnet*. Serangan *malware* pada *cloud server* dapat berdampak serius, seperti kebocoran data, gangguan layanan, hingga kerugian finansial(Fadhlurrohman et al., n.d.).

2.1.2 *Cloud server*

Cloud server merupakan infrastruktur komputasi yang menyediakan sumber daya seperti penyimpanan, pemrosesan, dan jaringan secara virtual melalui internet. Model layanan *cloud* terdiri dari *Infrastructure as a Service* (*IaaS*), *Platform as a Service* (*PaaS*), dan *Software as a Service* (*SaaS*). Meskipun *cloud server* menawarkan fleksibilitas dan skalabilitas, lingkungan ini juga rentan terhadap berbagai serangan *siber*, termasuk *malware* dan intrusi jaringan(Riza, 2023).

2.1.3 *Intrusion Detection System* (IDS)

Intrusion Detection System (*IDS*) adalah sistem keamanan yang berfungsi untuk memantau aktivitas jaringan atau *host* dan mendeteksi adanya aktivitas mencurigakan atau serangan. *IDS* diklasifikasikan menjadi dua jenis utama, yaitu

Host-based IDS (HIDS) dan *Network-based IDS* (*NIDS*)**.** Pada lingkungan *cloud server, IDS* berperan penting dalam mendeteksi serangan secara dini untuk mencegah dampak yang lebih besar(Aqarni, 2025).

2.1.4 Pengertian *Feature selection*

Feature selection merupakan proses pemilihan fitur-fitur yang paling relevan dari sekumpulan fitur yang tersedia dalam suatu dataset. Tujuan utama *feature selection* adalah mengurangi jumlah fitur yang digunakan tanpa menghilangkan informasi penting yang dibutuhkan dalam proses klasifikasi atau prediksi.

Pada dataset dengan jumlah fitur yang banyak, tidak semua fitur memiliki kontribusi yang signifikan terhadap proses pembelajaran model. Beberapa fitur dapat bersifat redundan, tidak relevan, atau memiliki korelasi yang rendah terhadap kelas target. Penggunaan fitur yang tidak relevan dapat meningkatkan kompleksitas komputasi, memperpanjang waktu pelatihan model, serta berpotensi menurunkan performa klasifikasi.

Oleh karena itu, *feature selection* digunakan untuk memilih fitur-fitur yang paling representatif sehingga model dapat bekerja lebih efisien dan menghasilkan performa yang lebih baik.

2.1.5 *Random Forest*

Random Forest merupakan salah satu algoritma *machine learning* yang termasuk dalam metode *ensemble learning* dan digunakan untuk proses klasifikasi maupun regresi. Algoritma ini bekerja dengan membangun banyak pohon keputusan (*decision tree*) secara acak, kemudian menggabungkan hasil prediksi

dari seluruh pohon untuk menentukan hasil akhir menggunakan metode *majority voting* pada klasifikasi atau rata-rata pada regresi.

Konsep utama *Random Forest* adalah mengombinasikan banyak *decision tree* agar menghasilkan model yang lebih akurat, dan mampu mengurangi risiko *overfitting* yang sering terjadi pada satu pohon keputusan tunggal. Setiap pohon pada *Random Forest* dibangun menggunakan subset data yang dipilih secara acak (*bootstrap sampling*) serta subset fitur yang juga dipilih secara acak pada setiap proses pemisahan node.

Pada proses klasifikasi, setiap pohon keputusan akan memberikan prediksi terhadap suatu data. Hasil akhir klasifikasi ditentukan berdasarkan jumlah suara terbanyak (*majority voting*) dari seluruh pohon keputusan yang terbentuk. Pendekatan ini membuat *Random Forest* memiliki kemampuan generalisasi yang baik terhadap data baru.

Dalam penelitian ini, *Random Forest* digunakan untuk mendeteksi *malware* pada lalu lintas jaringan *cloud server* berdasarkan fitur-fitur statistik trafik jaringan yang merepresentasikan pola perilaku *malware* dan trafik normal. Algoritma ini dipilih karena memiliki performa klasifikasi yang baik, serta mampu mengenali pola data jaringan secara efektif.

2.1.6 Metode Evaluasi Kinerja Sistem Deteksi

Evaluasi kinerja sistem deteksi *malware* bertujuan untuk mengukur kemampuan metode yang digunakan dalam mengklasifikasikan lalu lintas jaringan secara tepat. Pengukuran kinerja dilakukan dengan membandingkan hasil prediksi sistem terhadap label aktual pada data uji.

Metode evaluasi yang umum digunakan pada sistem deteksi *malware* meliputi *accuracy, precision, recall, dan F1-score. Accuracy* menunjukkan tingkat ketepatan sistem dalam mengklasifikasikan seluruh data. *Precision* mengukur kemampuan sistem dalam meminimalkan kesalahan deteksi terhadap lalu lintas normal, sedangkan *recall* menunjukkan kemampuan sistem dalam mendeteksi seluruh serangan *malware* yang ada. *F1-score* digunakan sebagai ukuran keseimbangan antara *precision* dan *recall.*

Selain metrik klasifikasi, evaluasi juga dapat mencakup aspek efisiensi sistem, seperti penggunaan sumber daya dan waktu pemrosesan, terutama ketika metode diterapkan pada lingkungan *cloud server* yang menuntut kinerja yang efisien.

Penggunaan metrik evaluasi tersebut memberikan gambaran yang komprehensif mengenai efektivitas metode *Random Forest* dalam mendeteksi serangan *malware* pada lingkungan *cloud server*.

2.2 Penelitian Terkait

Penelitian terkait diperlukan untuk memahami perkembangan riset dalam deteksi serangan *malware*, khususnya pada lingkungan *cloud server*, serta untuk mengidentifikasi kelebihan dan keterbatasan pendekatan yang telah digunakan sebelumnya. Melalui kajian penelitian terdahulu, dapat ditentukan posisi penelitian yang dilakukan serta kontribusi yang dihasilkan.

Tabel 2.1 Penelitian Terkait

No	Peneliti & Tahun	Metode	Dataset / Lingkungan	Hasil Penelitian
1	Breiman (2001)	*Random Forest*	Data Klasifikasi	*Random Forest* memiliki akurasi tinggi dan tahan terhadap *overfitting* dibandingkan *single decision tree.*
2	Almseidin et al. (2017)	Perbandingan Algoritma ML (RF, DT, NB, SVM)	KDD Cup 99	*Random Forest* dan *Decision Tree* memberikan performa terbaik dengan akurasi tinggi pada deteksi intrusi.
3	Sultana et al. (2019)	*Machine learning*-based IDS	NSL-KDD	*Random Forest* menghasilkan nilai precision dan recall yang lebih baik dibanding beberapa metode tradisional.
4	Kim et al. (2020)	*Random Forest* untuk IDS	CICIDS 2017	*Random Forest* menunjukkan performa stabil dengan akurasi tinggi dalam mendeteksi serangan jaringan.
5	Liotta et al. (2021)	*Feature selection* + *Machine learning*	Dataset IDS	Pemilihan fitur yang relevan mampu meningkatkan akurasi serta mengurangi kompleksitas komputasi.
6	Moustafa & Slay (2015)	*Feature selection* pada IDS	UNSW-NB15	Fitur yang dipilih secara tepat mampu meningkatkan kemampuan deteksi serangan dan mengurangi noise data.
7	Sharafaldin et al. (2019)	*Machine learning*-based IDS	CICIDS 2017	Dataset modern dan fitur yang representatif meningkatkan kualitas evaluasi model deteksi serangan.
8	Muthurajkumar et al. (2013)	*Intelligent Feature selection & Classification*	*Network Intrusion Dataset*	Pemilihan fitur yang efektif mampu meningkatkan performa klasifikasi dan efisiensi sistem IDS.
9	El-Soudani et al. (2025)	*Feature-based Intrusion Detection*	*IoT Traffic Dataset*	Fitur statistik paket dan header jaringan efektif membedakan trafik normal dan serangan.
10	Tavallaee et al. (2009)	Evaluasi Dataset IDS	NSL-KDD	Penggunaan dataset yang lebih representatif meningkatkan validitas hasil pengujian model deteksi intrusi.

Beberapa penelitian dalam beberapa tahun terakhir menunjukkan bahwa penerapan *machine learning* pada sistem deteksi *malware* dan intrusi terus berkembang seiring meningkatnya kompleksitas serangan siber. Sharafaldin et al. (2019) memperkenalkan dataset CICIDS2017 yang dirancang untuk merepresentasikan lalu lintas jaringan modern, termasuk berbagai jenis serangan dan aktivitas *malware*. Penelitian tersebut menunjukkan bahwa penggunaan dataset yang representatif sangat penting untuk menghasilkan evaluasi sistem deteksi yang lebih akurat dan realistis.

Penelitian yang dilakukan oleh Kim et al. (2020) menerapkan algoritma *Random Forest* untuk mendeteksi serangan jaringan menggunakan dataset CICIDS2017. Hasil penelitian menunjukkan bahwa *Random Forest* mampu memberikan performa klasifikasi yang baik dengan tingkat akurasi yang tinggi. Kemampuan *Random Forest* dalam membangun banyak pohon keputusan dan menggabungkan hasil prediksi melalui mekanisme *majority voting* membuat algoritma ini mampu menangani data berdimensi besar dan mengurangi risiko *overfitting*.

Sementara itu, Almseidin et al. (2017) melakukan perbandingan beberapa algoritma *machine learning* pada sistem *Intrusion Detection System* (IDS) menggunakan dataset KDD Cup 99. Hasil penelitian menunjukkan bahwa *Random Forest* dan *Decision Tree* memberikan performa yang lebih baik dibandingkan algoritma lainnya dalam mendeteksi aktivitas serangan jaringan. Penelitian tersebut menegaskan bahwa *Random Forest* merupakan salah satu algoritma yang efektif untuk permasalahan klasifikasi pada bidang keamanan jaringan.

Penelitian oleh Sultana et al. (2019) mengkaji penerapan *machine learning* pada sistem deteksi intrusi menggunakan dataset NSL-KDD. Hasil penelitian menunjukkan bahwa penggunaan algoritma *machine learning* mampu meningkatkan nilai *precision, recall*, dan akurasi dibandingkan pendekatan deteksi tradisional. Penelitian ini juga menunjukkan bahwa pemilihan fitur yang tepat memiliki pengaruh besar terhadap kualitas hasil klasifikasi.

Di sisi lain, Liotta et al. (2021) meneliti penerapan *feature selection* pada sistem deteksi intrusi jaringan. Hasil penelitian menunjukkan bahwa pemilihan fitur

yang relevan mampu mengurangi kompleksitas data, mempercepat proses komputasi, serta meningkatkan performa klasifikasi. Dengan menghilangkan fitur yang kurang relevan, model dapat lebih fokus mempelajari karakteristik utama yang membedakan lalu lintas normal dan lalu lintas serangan.

Penelitian yang dilakukan oleh Muthurajkumar et al. (2013) juga menunjukkan bahwa *feature selection* berperan penting dalam meningkatkan efisiensi dan akurasi sistem deteksi intrusi. Penelitian tersebut menyimpulkan bahwa penggunaan fitur-fitur yang memiliki keterkaitan kuat dengan karakteristik serangan mampu meningkatkan kemampuan model dalam mengidentifikasi aktivitas berbahaya pada jaringan.

Selain itu, penelitian El-Soudani et al. (2025) menunjukkan bahwa fitur statistik paket dan karakteristik header jaringan dapat digunakan untuk membedakan trafik normal dan trafik serangan secara efektif. Fitur-fitur seperti distribusi *TCP Flag*, ukuran paket, *interval* waktu antar paket, serta pola koneksi jaringan terbukti memiliki kontribusi penting dalam proses klasifikasi lalu lintas jaringan.

Berdasarkan kajian penelitian terdahulu tersebut, dapat disimpulkan bahwa *Random Forest* memiliki kemampuan yang baik dalam melakukan klasifikasi pada sistem deteksi *malware* dan intrusi. Selain itu, penerapan *feature selection* mampu meningkatkan efisiensi komputasi serta membantu model dalam mempelajari karakteristik serangan secara lebih efektif. Namun demikian, penelitian yang secara khusus mengkaji kinerja *Random Forest* dengan *feature selection* berbasis karakteristik *malware* pada dataset MTA-KDD'19 masih relatif terbatas.

Oleh karena itu, penelitian ini berfokus pada analisis kinerja algoritma *Random Forest* dengan *feature selection* berbasis karakteristik *malware* menggunakan dataset MTA-KDD'19 untuk mengevaluasi efektivitas, akurasi, *precision, recall,* dan *F1-score* dalam mendeteksi serangan *malware* pada lingkungan *cloud server*.

Berdasarkan penelitian-penelitian tersebut, dapat disimpulkan bahwa kombinasi *Random Forest* dan *feature selection* berbasis karakteristik *malware* memiliki potensi yang besar untuk meningkatkan kemampuan deteksi *malware*, sekaligus mempertahankan efisiensi komputasi pada lingkungan *cloud server*.

BAB III

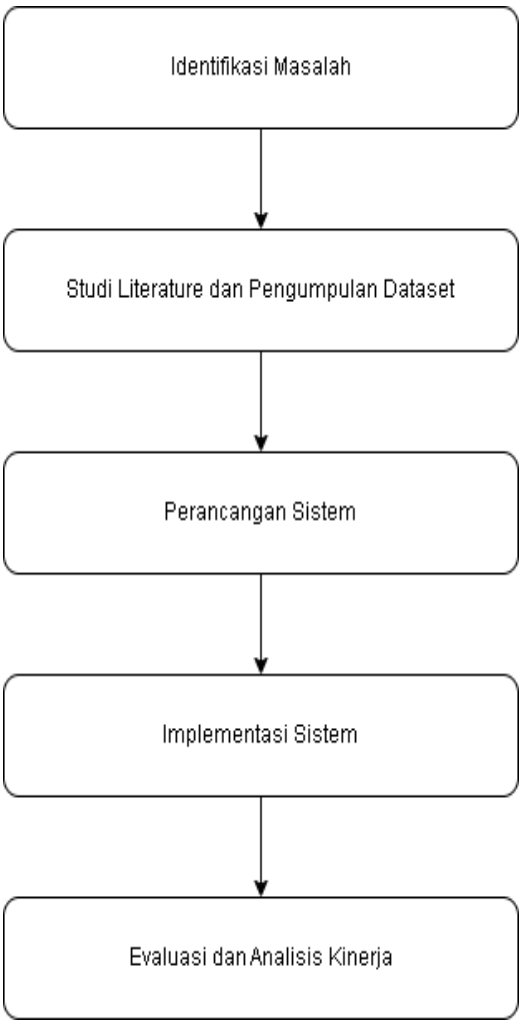
METODE PENELITIAN

Bab ini menjelaskan tahapan-tahapan metodologi penelitian yang digunakandalam membangun dan mengevaluasi sistem deteksi *malware* berbasis *Random Forest*. Metode yang diusulkan memanfaatkan karakteristik lalu lintas jaringan yang direpresentasikan dalam bentuk fitur statistik, kemudian dipelajari menggunakan algoritma *Random Forest* sebagai pembentuk *model deteksi.*

Alur penelitian disusun secara sistematis mulai dari pemuatan dataset MTA-KDD'19, proses integrasi dan praproses data, pembagian data latih dan data uji, pemodelan *Random Forest*, hingga evaluasi kinerja model dan analisis hasil. Seluruh tahapan tersebut digambarkan dalam bentuk diagram alir untuk memudahkan pemahaman proses penelitian secara menyeluruh.

3.1 Prosedur Penelitian

Dalam pelaksanaan penelitian deteksi serangan *malware* menggunakan metode *Random Forest dengan feature selection*, peneliti menyusun suatu prosedur penelitian sebagai acuan agar setiap tahapan penelitian dapat berjalan secara sistematis dan terarah. Acuan tersebut disajikan dalam bentuk diagram alur penelitian yang ditunjukkan pada Gambar 3.1

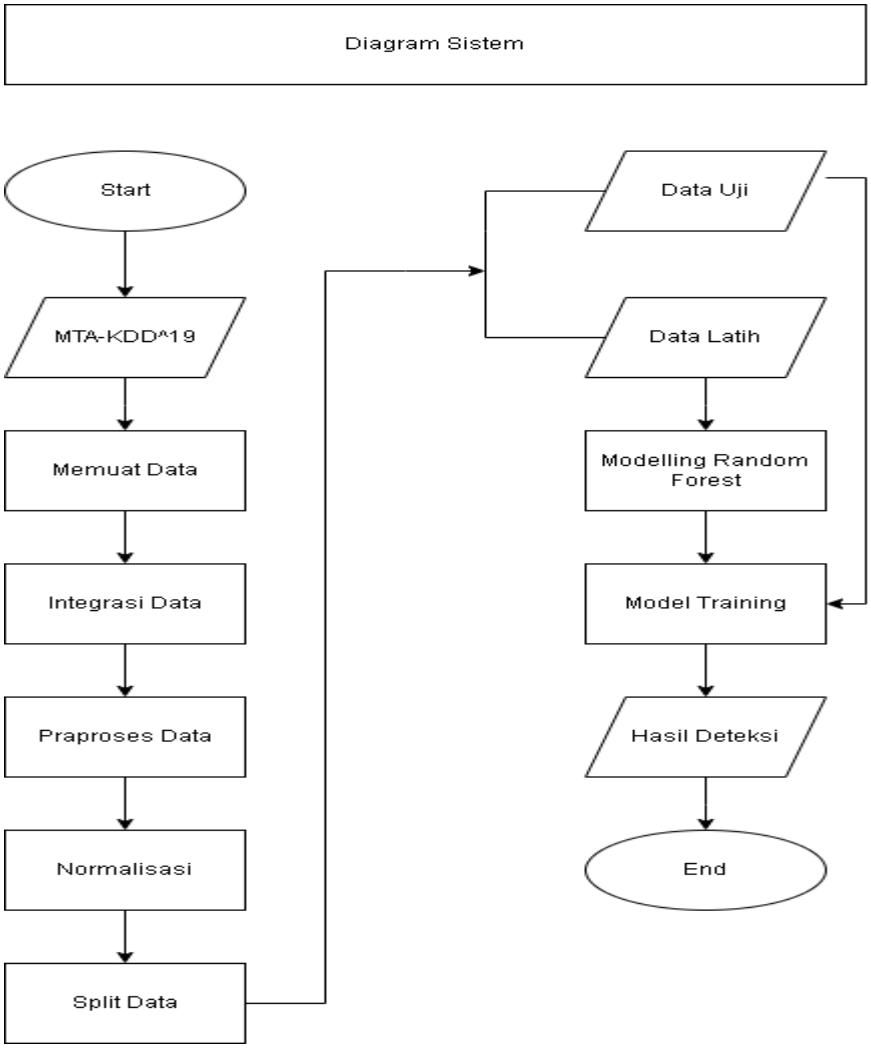


Gambar 3. 1 Alur Penelitian

Berdasarkan diagram alur penelitian di atas, proses penelitian diawali dengan identifikasi masalah yang berkaitan dengan analisis kinerja metode *Random Forest* dalam mendeteksi serangan *malware* pada *cloud server*. Selanjutnya dilakukan studi literatur untuk memperkuat landasan teori serta menentukan pendekatan yang tepat. Setelah metodologi penelitian dirancang, peneliti melakukan pengumpulan dan pemilihan dataset yang relevan. Tahap berikutnya

meliputi perancangan model deteksi, implementasi serta eksperimen, kemudian dilakukan evaluasi dan analisis kinerja model. Tahapan terakhir adalah penarikan kesimpulan dan penyusunan saran berdasarkan hasil penelitian yang diperoleh.

Untuk menjelaskan alur kerja sistem yang dikembangkan dalam penelitian ini, disusun diagram sistem yang menggambarkan proses pengolahan data hingga menghasilkan output deteksi.



Gambar 3. 2 Diagram Sistem

Berdasarkan Gambar 3.2, terdapat beberapa tahapan proses penelitian yang dilakukan untuk menyelesaikan penelitian ini. Penelitian diawali dengan pemuatan dataset MTA-KDD'19 yang terdiri dari data serangan *malware* dan data lalu lintas normal (*legitimate*). Setelah dataset berhasil dimuat, tahap selanjutnya adalah integrasi data, yaitu menggabungkan seluruh data agar dapat diproses secara menyeluruh dalam satu kesatuan dataset. Berikutnya, dataset yang telah terintegrasi akan melalui tahap praproses data dengan tujuan untuk meningkatkan kualitas data yang akan digunakan dalam proses deteksi. Tahap praproses ini meliputi pembersihan data, penyesuaian format, serta penanganan data yang tidak sesuai. Setelah tahap praproses selesai, dataset kemudian dibagi menjadi data latih dan data uji, di mana data latih digunakan untuk membangun model dan data uji digunakan untuk pengujian kinerja model.

Tahap selanjutnya adalah pemodelan menggunakan algoritma *Random Forest* dengan memanfaatkan data latih. Proses ini bertujuan untuk membangun model deteksi berbasis *Random Forest dengan feature selection* yang mampu mengenali pola serangan *malware* pada lingkungan *cloud server*.

Model yang telah dibangun kemudian digunakan pada tahap evaluasi model untuk mengukur tingkat keberhasilan deteksi terhadap data uji. Setelah proses evaluasi selesai, dilakukan analisa hasil untuk mengetahui kinerja metode *Random Forest* berdasarkan metrik evaluasi yang digunakan. Tahap terakhir dalam penelitian ini adalah penarikan kesimpulan, yang berisi rangkuman hasil penelitian serta kesimpulan mengenai efektivitas metode *Random Forest* dalam mendeteksi serangan *malware* pada *cloud server.*

3.1.1 Akuisisi Data

Tahap akuisisi data merupakan proses pengumpulan data yang digunakan sebagai dasar dalam penelitian. Data yang digunakan dalam penelitian ini diperoleh dari dataset MTA-KDD'19 (*Malware Traffic Analysis – KDD* 2019), yang merupakan dataset lalu lintas jaringan untuk penelitian deteksi *malware* pada lingkungan *cloud server.*

Dataset MTA-KDD'19 dikembangkan melalui simulasi dan pengamatan aktivitas jaringan pada lingkungan *cloud* nyata, dengan tujuan merepresentasikan lalu lintas normal dan lalu lintas yang mengandung serangan *malware.* Data dikumpulkan dalam bentuk *flow-based network traffic*, sehingga tidak memuat isi paket (payload) jaringan. Pendekatan ini dipilih untuk menjaga privasi serta menyesuaikan dengan kondisi sistem *cloud* modern yang umumnya membatasi inspeksi payload.

Proses akuisisi data pada dataset MTA-KDD'19 dilakukan dengan merekam aliran lalu lintas jaringan (network flow) selama aktivitas normal dan aktivitas *malware* berlangsung. Setiap aliran lalu lintas kemudian diekstraksi menjadi sejumlah fitur statistik yang merepresentasikan karakteristik komunikasi jaringan, seperti distribusi flag TCP, rata-rata panjang paket, interval waktu antar paket, dan jumlah koneksi.

Dataset yang diperoleh tersedia dalam format *Comma Separated Values* (CSV) dan dikategorikan ke dalam dua kelas, yaitu lalu lintas normal (*legitimate*) dan lalu lintas *malware*. Dataset ini bersifat publik dan telah digunakan secara luas

dalam penelitian sebelumnya, sehingga memiliki validitas dan kredibilitas yang memadai untuk digunakan sebagai sumber data penelitian.

Dengan menggunakan dataset MTA-KDD'19, penelitian ini tidak melakukan pengambilan data secara langsung dari jaringan nyata, melainkan memanfaatkan data yang telah terstandarisasi untuk memastikan proses evaluasi metode dapat dilakukan secara terukur dan dapat direproduksi.

3.1.2 Memuat Data

Tahap memuat data merupakan langkah awal untuk memasukkan dataset ke dalam lingkungan pengolahan data. Dataset yang digunakan dalam penelitian ini adalah MTA-KDD'19, yang diperkenalkan oleh Letteri *et al.* sebagai dataset lalu lintas jaringan untuk mendukung penelitian deteksi *malware* pada infrastruktur *cloud*.

Berbeda dengan dataset berbasis payload, MTA-KDD'19 tidak menyertakan isi paket jaringan, sehingga lebih aman dari sisi privasi dan lebih realistis untuk diterapkan pada sistem *cloud modern*.

Berikut karakteristik dataset MTA-KDD'19 tertera pada tabel 3.1

Tabel 3. 1 Karakteristik MTA-KDD'19

Aspek	Keterangan
Jenis data	Lalu lintas jaringan (*flow-based*)
Jumlah fitur	33 fitur *numerik*
Kelas	*Legitimate* (normal) dan *Malware*
Label	*Biner* (0 = normal, 1 = *malware*)
Lingkungan	*Cloud server*

Berdasar tabel 3.1, Dataset terbagi antara *malware* dan *legitimate* yang dimuat dari berkas CSV terpisah, kemudian diperiksa struktur dan konsistensinya.

Data dimuat menggunakan *library* Pandas agar dapat diproses lebih lanjut dalam bentuk *DataFrame*. Adapun fitur-fitur yang ada pada dataset MTA-KDD'19 yaitu sebanyak 33 fitur seperti pada tabel 3.2

Tabel 3. 2 Fitur-fitur MTA-KDD'19

No.	Nama Fitur	No.	Nama Fitur
1	FinFlagDist	18	1stPktLen
2	SynFlagDist	19	MaxLenrx
3	RstFlagDist	20	MinLenrx
4	PshFlagDist	21	StdDevLenrx
5	AckFlagDist	22	AvgLenrx
6	DNServerIP	23	MinIATrx
7	TCPoverIP	24	AvgIATrx
8	UDPoverIP	25	NumPorts
9	MaxLen	26	FlowLEN
10	MinLen	27	FlowLENrx
11	StdDevLen	28	repeated_pkts_ratio
12	AvgLen	29	NumCon
13	MaxIAT	30	NumIPdst
14	MinIAT	31	Start_flow
15	AvgIAT	32	DeltaTimeFlow
16	AvgWinFlow	33	HTTPpkts
17	PktsIOratio		

Fitur-fitur dataset yang digunakan untuk *metode Random Forest berbasis feature selection* antara lain *'SynFlagDist', 'RstFlagDist', 'AckFlagDist', 'DNSoverIP', 'TCPoverIP', 'UDPoverIP', 'AvgLen', 'StdDevLen', 'AvgIAT', 'DeltaTimeFlow', 'NumCon', 'NumIPdst', 'HTTPpkts'.*

Pemilihan fitur tersebut dijelaskan berdasarkan tabel berikut ini

Tabel 3. 3 Referensi Pemilihan Fitur

Kategori Fitur	Fitur	Alasan Pemilihan	Dukungan Penelitian
TCP Flag	SynFlagDist, RstFlagDist, AckFlagDist	Mendeteksi pola serangan jaringan (SYN flood, scan)	A. Liotta, *Supervised feature selection techniques in network intrusion detection Volume* 101
Protokol	TCPoverIP, UDPoverIP, DNSoverIP	Mengidentifikasi jenis komunikasi jaringan	Muthurajkumar, S. *et al. Intelligent feature selection* and *classification techniques for intrusion detection in networks*: *J Wireless Com Network* 2013
Statistik Paket	AvgLen, StdDevLen	Membedakan pola trafik normal vs *malware*	. & El-Soudani, M. *Intrusion detection using TCP/IP single packet header binary image for IoT. Cybersecurity* **8**, 104 (2025)
Waktu	AvgIAT, DeltaTimeFlow	Deteksi pola berbasis waktu (*burst traffic*)	A. Liotta, *Supervised feature selection techniques in network intrusion detection Volume* 101
Flow/Koneksi	NumCon, NumIPdst, HTTPpkts	Representasi perilaku komunikasi jaringan	A. Liotta, *Supervised feature selection techniques in network intrusion detection Volume* 101

3.1.3 Integrasi Data

Setelah kedua dataset berhasil dimuat, dilakukan proses integrasi data dengan cara menggabungkan dataset *malware* dan dataset *legitimate* menjadi satu dataset lengkap.

Pada tahap integrasi data, dataset *malware* dan *legitimate* digabungkan menjadi satu dataset utuh. Dataset kemudian diacak (*shuffle*) untuk menghindari bias urutan data sebelum masuk ke tahap praproses dan pelatihan model.

Skema label kelas yang digunakan adalah sebagai berikut:

Tabel 3. 4 Label kelas

Label	Keterangan
0	Lalu lintas normal
1	Lalu lintas *malware*

Berdasarkan hasil integrasi data, diperoleh distribusi kelas sebagai berikut:

Tabel 3. 5 Distribusi Kelas

Kelas	Jumlah Data
Malware	34.349
Normal	30.205

Distribusi kelas yang relatif seimbang mendukung proses pelatihan model secara optimal.

3.1.4 Praproses Data

Tahap praproses data bertujuan untuk memastikan kualitas dan konsistensi data sebelum digunakan dalam proses pelatihan model *Random Forest*.

Langkah praproses yang dilakukan dalam penelitian ini meliputi:

1. Pemisahan fitur dan label kelas
2. Pemeriksaan nilai kosong (*missing values*)
3. Pemeriksaan kesesuaian tipe data numerik

Karena seluruh fitur pada dataset bersifat numerik, proses *encoding* kategori tidak diperlukan.

3.1.5 Normalisasi Data

Pada penelitian ini digunakan metode *Min-Max Normalization*, yaitu teknik yang mengubah nilai data ke dalam rentang tertentu, biasanya antara 0 hingga 1. Berikut rumus *Min-Max Normalization*:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (3.1)$$

Keterangan:

1. X= nilai asli data
2. X_{min}= nilai minimum pada fitur
3. X_{max}= nilai maksimum pada fitur
4. X'= nilai hasil normalisasi

3.1.6 Split Data

Dataset yang telah dipraproses kemudian dibagi menjadi data latih dan data uji menggunakan metode *train-test split*.

Proporsi pembagian data ada 3 bagian yaitu:

1. 70% data latih 30% data uji
2. 80% data latih 20% data uji
3. 90% data latih 10% data uji

Pembagian dilakukan secara *stratified* untuk menjaga proporsi kelas *malware* dan *normal* tetap seimbang pada kedua subset data.

Tujuan pembagian data ini adalah untuk mengevaluasi kemampuan model dalam mendeteksi *malware* pada data yang belum pernah dilihat sebelumnya.

3.1.7 Pembentukan Model Menggunakan *Random Forest*

Tahap ini merupakan inti dari penelitian. *Algoritma Random Forest* digunakan untuk membentuk model deteksi berbasis perilaku lalu lintas jaringan.

Random Forest bekerja dengan membangun sejumlah pohon keputusan (*decision tree*) dan menggabungkan hasil prediksi dari setiap pohon melalui mekanisme *majority voting*.

Secara matematis, prediksi akhir *Random Forest* dinyatakan sebagai:

$$\hat{\gamma} = "mode"\{h\_1\ (x), h\_2\ (x), \dots, h\_n\ (x)\} \quad (3.2)$$

Parameter utama yang digunakan dalam penelitian ini adalah:

1. *n_estimators* = 100
2. *class_weight = balanced*
3. *random_state* = 42

Model yang terbentuk berfungsi sebagai model deteksi, di mana pola lalu lintas *malware* disimpan secara implisit dalam struktur model.

3.1.8 **Skenario Pengujian**

Skenario pengujian dalam penelitian ini bertujuan untuk mengevaluasi kinerja model *Random Forest* dengan *feature selection* dalam mendeteksi serangan *malware* pada *cloud server*. Pengujian dilakukan dengan membagi dataset ke dalam data latih *(training data)* dan data uji *(testing data)* menggunakan beberapa variasi proporsi pembagian data (*train-test split*).

Pembagian data dilakukan untuk mengetahui pengaruh jumlah data latih terhadap performa model dalam mengklasifikasikan data uji. Semakin besar data latih, diharapkan model dapat mengenali pola jaringan dengan lebih baik, namun tetap perlu diuji terhadap data yang belum pernah dilihat sebelumnya.

Dalam penelitian ini digunakan tiga skenario pembagian data sebagai berikut:

1. **70% Data Latih – 30% Data Uji**

   Pada skenario pertama, dataset dibagi menjadi 70% sebagai data latih 30% sebagai data uji. Pada tahap ini, model *Random Forest* dilatih menggunakan 70% data untuk membentuk pola berdasarkan karakteristik serangan dan lalu lintas normal. Selanjutnya, model diuji menggunakan 30% data yang tidak digunakan saat pelatihan untuk mengukur kemampuan generalisasi model. Skenario ini digunakan sebagai *baseline* awal untuk melihat performa model dengan proporsi pembagian data yang umum digunakan dalam penelitian *machine learning.*

2. **80% Data Latih – 20% Data Uji**

   Pada skenario kedua, dataset dibagi menjadi 80% sebagai data latih 20% sebagai data uji. Dengan jumlah data latih yang lebih besar dibandingkan skenario pertama, model memiliki lebih banyak informasi untuk mempelajari pola *malware*.

   Pengujian dilakukan terhadap 20% data uji untuk melihat apakah peningkatan jumlah data latih berdampak pada peningkatan nilai *accuracy,*

precision, recall, dan *F1-score.* Skenario ini bertujuan untuk mengevaluasi performa model ketika jumlah data pelatihan ditambah.

3. 90% Data Latih – 10% Data Uji

Pada skenario ketiga, dataset dibagi menjadi 90% sebagai data latih 10% sebagai data uji. Pada skenario ini, model dilatih menggunakan sebagian besar data yang tersedia.

Tujuannya adalah untuk mengetahui performa maksimal yang dapat dicapai model ketika hampir seluruh dataset digunakan untuk pelatihan. Meskipun data latih sangat besar, pengujian tetap dilakukan terhadap 10% data yang tidak terlihat sebelumnya untuk memastikan model tidak mengalami *overfitting* dan tetap mampu menggeneralisasi pola dengan baik.

Pada setiap skenario, performa model dievaluasi menggunakan metrik berikut:

1. *Accuracy*
2. *Precision*
3. *Recall*
4. *F1-Score*

Hasil dari ketiga skenario kemudian dibandingkan untuk menentukan konfigurasi pembagian data yang memberikan performa terbaik dalam mendeteksi serangan *malware* pada *cloud server.*

3.2 Uji Coba Awal Implementasi Sistem

Untuk mengevaluasi kinerja sistem deteksi *malware* pada tahap uji coba awal, dilakukan pengukuran menggunakan metrik evaluasi klasifikasi, yaitu *precision, recall, F1-score,* dan *accuracy*.

Evaluasi ini bertujuan untuk melihat kemampuan sistem dalam mengklasifikasikan lalu lintas jaringan ke dalam kelas *normal* dan *malware* secara seimbang, serta untuk mengetahui tingkat keandalan metode *Random Forest* sebelum dilakukan pengujian akhir.

Hasil uji coba awal dengan data 40% dataset disajikan dalam bentuk *classification report*, sebagaimana ditunjukkan pada gambar berikut.

	precision	recall	f1-score	support
Normal	0.894164	0.929931	0.911697	3625.000000
Malware	0.936133	0.903202	0.919373	4122.000000
accuracy	0.915709	0.915709	0.915709	0.915709
macro avg	0.915149	0.916567	0.915535	7747.000000
weighted avg	0.916495	0.915709	0.915781	7747.000000

Gambar 3. 3 Hasil Uji Coba Awal

Berdasarkan *classification report* pada Gambar 3.3 , sistem deteksi *malware* pada tahap uji coba awal menunjukkan kinerja yang cukup baik dengan nilai *accuracy* sebesar 91,57%. Untuk kelas normal, sistem memperoleh nilai *precision* sebesar 89,42% dan *recall* sebesar 92,99%, yang menunjukkan bahwa sebagian besar data normal berhasil diidentifikasi dengan benar, meskipun masih terdapat sejumlah kecil kesalahan klasifikasi sebagai *malware.*

Pada kelas *malware*, sistem memperoleh nilai *precision* sebesar 93,61% dan *recall* sebesar 90,32%, yang menandakan bahwa sistem mampu mengenali sebagian besar serangan *malware* dengan tingkat ketepatan yang tinggi. Nilai F1-

score pada kedua kelas berada di atas 91%, menunjukkan keseimbangan yang baik antara *precision* dan *recall.*

Selain itu, nilai *macro average* dan *weighted average* yang relatif seimbang mengindikasikan bahwa sistem tidak bias terhadap salah satu kelas, meskipun jumlah data pada masing-masing kelas berbeda. Secara keseluruhan, hasil ini menunjukkan bahwa metode *Random Forest* memiliki performa awal yang layak untuk dilanjutkan pada tahap pengujian akhir.

BAB IV

HASIL DAN PEMBAHASAN

4,1 Hasil

Subbab ini menyajikan hasil implementasi serta analisis kinerja sistem deteksi *malware* berbasis *Random Forest* dengan *feature selection*. Sistem yang dibangun bertujuan untuk mengidentifikasi lalu lintas jaringan. Pengujian dilakukan menggunakan dataset MTA-KDD'19 yang telah melalui beberapa tahapan pengolahan data, meliputi proses pemuatan data, integrasi dataset *malware* dan *legitimate*, praproses data, serta pembagian data menjadi data latih (*training data*) dan data uji (*testing data*).

Model *Random Forest* kemudian dilatih menggunakan data latih untuk membentuk pola berbasis perilaku lalu lintas jaringan. Hasil pengujian model dianalisis menggunakan *confusion matrix* untuk mengetahui distribusi prediksi terhadap kelas sebenarnya, serta metrik evaluasi berupa *accuracy, precision, recall*, dan *F1-score.*

Dengan demikian, hasil pada subbab ini diharapkan dapat memberikan gambaran yang komprehensif mengenai efektivitas metode *Random Forest* dan *feature selection* dalam mendeteksi serangan *malware* pada lingkungan *cloud server*.

4.1.1 Hasil Pengolahan Data Penelitian

Pada tahap awal penelitian, dilakukan proses pengolahan data menggunakan dataset MTA-KDD'19 yang terdiri dari dua jenis data, yaitu data lalu lintas normal (*legitimate*) dan data serangan *malware*.

Kedua dataset tersebut dimuat dari file CSV terpisah, kemudian dilakukan proses integrasi menjadi satu dataset utuh. Setelah proses penggabungan data dilakukan, dataset diacak (*shuffle*) untuk menghindari bias urutan data. Selanjutnya, dilakukan pemberian label kelas dengan skema sebagai berikut:

Tabel 4. 1 Label Kelas

Label	Keterangan
0	Lalu lintas normal
1	Lalu lintas *malware*

Berdasarkan hasil pengolahan data, diperoleh distribusi data sebagai berikut:

Tabel 4. 2 Distribusi Kelas

Kelas	Jumlah Data
Malware	34.349
Normal	30.205

Distribusi kelas yang relatif seimbang ini mendukung proses pelatihan model agar tidak bias terhadap salah satu kelas.

Selanjutnya, dilakukan pemilihan fitur (*feature selection*) dari total 33 fitur yang tersedia pada dataset. Dalam penelitian ini digunakan 13 fitur utama, yaitu:

Tabel 4. 3 Feature

No	*Feature*
1	SynFlagDist
2	RstFlagDist
3	AckFlagDist
4	DNSoverIP
5	TCPoverIP
6	UDPoverIP
7	AvgLen
8	StdDevLen
9	AvgIAT
10	DeltaTimeFlow
11	NumCon
12	NumIPdst
13	HTTPpkts

Pemilihan fitur ini bertujuan untuk mengurangi kompleksitas model sekaligus mempertahankan informasi penting yang merepresentasikan pola serangan *malware.*

4.1.2 Hasil Praproses Data

Tahap praproses data dilakukan untuk memastikan kualitas data sebelum digunakan dalam proses pelatihan model. Langkah-langkah praproses yang dilakukan meliputi:

1. Data cleaning

   Tahap *data cleaning* dilakukan untuk memastikan kualitas data sebelum digunakan pada proses pelatihan dan pengujian model klasifikasi. Data yang bersih dan konsisten sangat penting karena dapat memengaruhi performa model *machine learning*, khususnya pada algoritma klasifikasi seperti *Random Forest*.

Pada penelitian ini, proses *data cleaning* difokuskan pada pemeriksaan *missing values* atau nilai kosong pada setiap fitur dalam dataset. Pemeriksaan *missing values* dilakukan menggunakan pustaka *Pandas* pada Python dengan menghitung jumlah nilai kosong pada masing-masing atribut.

Tujuan dari proses ini adalah untuk memastikan bahwa seluruh data dapat digunakan secara optimal tanpa menyebabkan gangguan pada proses pelatihan model. Nilai kosong pada dataset umumnya dapat menyebabkan penurunan performa model, menghasilkan bias pada prediksi, maupun menimbulkan error saat proses komputasi berlangsung.

Hasil pemeriksaan *missing values* menunjukkan bahwa dataset berada dalam kondisi baik dan lengkap, sehingga kualitas data dinilai cukup memadai untuk digunakan dalam proses klasifikasi trafik jaringan.

Kondisi dataset yang bersih ini juga membantu meningkatkan reliabilitas hasil pengujian model karena proses pelatihan dilakukan menggunakan data yang utuh tanpa kehilangan informasi penting.

Adapun hasil pemeriksaan *missing values* pada masing-masing fitur ditunjukkan pada tabel berikut.

Tabel 4. 4 Hasil *Data Cleansing*

No	Fitur	Missing Count
1	FinFlagDist	0
2	SynFlagDist	0
3	RstFlagDist	0
4	PshFlagDist	0
5	AckFlagDist	0
6	DNSoverIP	0
7	TCPoverIP	0
8	UDPoverIP	0
9	MaxLen	0
10	MinLen	0
11	StdDevLen	0
12	AvgLen	0
13	MaxIAT	0
14	MinIAT	0
15	AvgIAT	0
16	AvgWinFlow	0
17	PktsIOratio	0
18	1stPktLen	0
19	MaxLenrx	0
20	MinLenrx	0
21	StdDevLenrx	0
22	AvgLenrx	0
23	MinIATrx	0
24	AvgIATrx	0
25	NumPorts	0
26	FlowLEN	0
27	FlowLENrx	0
28	repeated_pkts_ratio	0
29	NumCon	0
30	NumIPdst	0
31	Start_flow	0
32	DeltaTimeFlow	0
33	HTTPpkts	0

Berdasarkan tabel tersebut, seluruh fitur memiliki nilai *missing count* sebesar 0. Hal ini menunjukkan bahwa dataset tidak mengalami

permasalahan data kosong dan siap digunakan pada tahapan *preprocessing* berikutnya.

2. Pemisahan fitur dan label

Pemisahan fitur dan label dilakukan agar model *Random Forest* dapat mempelajari pola hubungan antar fitur terhadap kelas target tanpa mengetahui label secara langsung selama proses pelatihan. Dengan demikian, model akan membangun aturan klasifikasi berdasarkan pola statistik dan karakteristik data yang terdapat pada fitur-fitur input.

Berikut merupakan contoh hasil pemisahan fitur dan label pada dataset.

Tabel 4. 5 Contoh Pemisahan Fitur

No	SynFlagDist	RstFlagDist	AckFlagDist	DNSoverIP	TCPoverIP	UDPoverIP	AvgLen	StdDevLen	AvgIAT	DeltaTimeFlow	NumCon	NumIPdst	HTTPpkts
0	-0.216194	-0.716937	0.607692	-0.154516	0.154711	-0.156991	1.055396	1.095308	0.946340	0.929731	-0.190341	-0.194497	-0.772901
1	0.316746	-0.716937	0.366734	-0.154516	0.154711	-0.156991	1.049255	1.120492	0.871846	0.696459	-0.190341	-0.194497	1.512202
2	-0.216194	-0.716937	0.745368	-0.154516	0.154711	-0.156991	1.086119	1.081383	1.138956	1.161626	-0.190341	-0.194497	-0.772901
3	-0.216194	-0.716937	0.325073	-0.154516	0.154711	-0.156991	1.223243	0.839563	1.147592	0.843500	-0.190341	-0.194497	-0.772901
4	-0.216194	-0.716937	0.622892	-0.154516	0.154711	-0.156991	1.126810	1.091091	0.505560	0.688897	-0.190341	-0.194497	-0.772901

Tabel 4. 6 Contoh Pemisahan Label

No	Label
0	0
1	0
2	0
3	1
4	0

Berdasarkan tabel tersebut, dapat dilihat bahwa setiap baris data fitur memiliki pasangan label yang merepresentasikan kelas target. Dataset hasil pemisahan ini kemudian digunakan pada tahap pembagian data pelatihan dan pengujian untuk proses klasifikasi.

3. Pemeriksaan tipe data

Tahap berikutnya dalam proses *preprocessing* adalah pemeriksaan tipe data pada setiap atribut dalam dataset. Pemeriksaan ini dilakukan untuk memastikan bahwa seluruh fitur memiliki format data yang sesuai dan dapat diproses dengan baik oleh algoritma *machine learning*, khususnya algoritma *Random Forest* yang digunakan pada penelitian ini.

Dalam implementasi *machine learning*, tipe data sangat memengaruhi proses pelatihan model. Algoritma klasifikasi umumnya bekerja lebih optimal ketika data input berbentuk numerik. Sebaliknya, apabila dataset masih mengandung data kategorikal atau teks (*string*), maka diperlukan proses transformasi seperti *encoding* agar data dapat dikenali dan diolah oleh model.

Pemeriksaan tipe data dilakukan menggunakan fungsi pemeriksaan atribut pada pustaka *Pandas* di Python untuk melihat jenis data dari masing-masing fitur. Hasil pemeriksaan menunjukkan bahwa seluruh atribut pada dataset telah memiliki tipe data numerik, baik dalam bentuk *integer* maupun *floating point*. Kondisi ini menunjukkan bahwa data sudah berada dalam format yang sesuai untuk proses komputasi dan pelatihan model klasifikasi.

Karena seluruh fitur telah berbentuk numerik, maka tidak diperlukan proses *encoding* seperti *Label Encoding* maupun *One-Hot Encoding*. Dengan demikian, dataset dapat langsung digunakan pada tahap selanjutnya tanpa perlu transformasi tambahan terhadap tipe data.

4. Normalisasi data

Tahap selanjutnya pada proses *preprocessing* adalah normalisasi data. Normalisasi dilakukan untuk menyamakan rentang nilai antar fitur sehingga seluruh atribut memiliki skala yang sebanding sebelum digunakan pada proses pelatihan model klasifikasi.

Perbedaan skala antar fitur dapat memengaruhi proses pembelajaran model karena fitur dengan nilai yang lebih besar berpotensi mendominasi proses pembentukan pola dibandingkan fitur lainnya.

Pada penelitian ini, metode normalisasi yang digunakan adalah *Min-Max Normalization*. Metode ini bekerja dengan mentransformasikan nilai setiap fitur ke dalam rentang tertentu, yaitu antara 0 hingga 1. Proses transformasi dilakukan dengan menghitung nilai minimum dan maksimum dari setiap atribut, kemudian mengubah seluruh nilai data berdasarkan rentang tersebut.

Berikut merupakan contoh perubahan nilai data sebelum dan sesudah dilakukan proses normalisasi menggunakan metode *Min-Max Normalization.*

Tabel 4. 7 Normalisasi Data

No	Fitur	Sebelum	Sesudah
1	SynFlagDist	-0.216194	0.286645
2	RstFlagDist	-0.716937	0.000000
3	AckFlagDist	0.607692	0.527178
4	DNSoverIP	-0.154516	0.000000
5	TCPoverIP	0.154711	1.000000
6	UDPoverIP	-0.156991	0.000000
7	AvgLen	1.055396	0.590760
8	StdDevLen	1.095308	0.509254
9	AvgIAT	0.946340	0.639039
10	DeltaTimeFlow	0.929731	0.479740
11	NumCon	-0.190341	0.000000
12	NumIPdst	-0.194497	0.000000
13	HTTPpkts	-0.772901	0.000000

Berdasarkan tabel tersebut, dapat dilihat bahwa nilai fitur yang sebelumnya memiliki rentang berbeda telah berhasil ditransformasikan ke dalam rentang 0 hingga 1. Hasil normalisasi ini menunjukkan bahwa data telah berada dalam kondisi yang lebih seragam dan siap digunakan pada tahap pembagian data (*data splitting*), pelatihan model *Random Forest*, serta evaluasi performa model menggunakan berbagai skenario pengujian dan *cross validation*.

4.1.3 Hasil Pembagian Data (*Split Data*)

Setelah seluruh tahapan *preprocessing* selesai dilakukan, dataset kemudian dibagi menjadi dua bagian utama, yaitu data latih (*training data*) dan data uji (*testing data*). Proses pembagian data ini bertujuan untuk mempersiapkan data yang akan digunakan pada tahap pelatihan model serta pengujian performa model klasifikasi.

Data latih digunakan untuk melatih model *Random Forest* agar mampu mempelajari pola dan karakteristik trafik jaringan berdasarkan fitur-fitur yang tersedia. Sementara itu, data uji digunakan untuk mengevaluasi kemampuan model dalam melakukan prediksi terhadap data baru yang sebelumnya tidak pernah dilihat selama proses pelatihan.

Pada penelitian ini, proses pembagian data dilakukan menggunakan metode *Stratified Train-Test Split*. Metode ini dipilih karena mampu mempertahankan proporsi distribusi kelas pada data latih dan data uji sehingga keseimbangan jumlah data antara kelas *malware* dan kelas normal tetap terjaga. Penggunaan metode stratifikasi sangat penting pada permasalahan klasifikasi karena distribusi kelas yang tidak seimbang dapat menyebabkan model menjadi bias terhadap kelas tertentu.

Untuk memperoleh evaluasi performa model yang lebih komprehensif, penelitian ini menggunakan tiga skenario pembagian data, yaitu:

1. 70% data latih dan 30% data uji
2. 80% data latih dan 20% data uji
3. 90% data latih dan 10% data uji

Penggunaan beberapa skenario pembagian data bertujuan untuk mengetahui pengaruh jumlah data latih terhadap performa model klasifikasi.

Selain itu, penggunaan beberapa skenario juga dilakukan untuk menguji kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya. Dengan membandingkan hasil pada masing-masing skenario, dapat diketahui konfigurasi pembagian data yang memberikan performa terbaik,

Hasil pembagian data pada setiap skenario ditunjukkan pada tabel berikut.

Tabel 4. 8 Pembagian Data

No	Skenario Split	Data Latih	Data Uji	Total
1	70 : 30	45.197	19.367	64.564
2	80 : 20	51.651	12.913	64.564
3	90 : 10	58.107	6.457	64.564

Berdasarkan tabel tersebut, dapat dilihat bahwa jumlah total data pada setiap skenario tetap sama, yaitu sebanyak 64.564 data, namun proporsi pembagian antara data latih dan data uji berbeda sesuai dengan skenario yang digunakan. Pada skenario 70:30, jumlah data uji lebih besar sehingga evaluasi model dilakukan pada data yang lebih banyak. Sebaliknya, pada skenario 90:10, jumlah data latih lebih dominan sehingga model memiliki lebih banyak data untuk dipelajari.

Hasil pembagian data menunjukkan bahwa proses *stratified split* berhasil mempertahankan distribusi kelas secara seimbang pada setiap skenario. Dengan demikian, data hasil pembagian ini dinilai layak digunakan pada tahap pelatihan model *Random Forest* dan evaluasi performa menggunakan *confusion matrix*, *accuracy, precision, recall, F1-score*, serta 10-*Fold Cross Validation*.

4.1.4 Hasil Pelatihan Model *Random Forest*

Setelah proses pembagian data selesai dilakukan, tahap berikutnya adalah pelatihan model klasifikasi menggunakan algoritma *Random Forest*. Pada penelitian ini, model dibangun menggunakan pendekatan *machine learning*, di mana fitur-fitur yang digunakan merupakan representasi pola dari karakteristik lalu lintas jaringan *malware* dan trafik normal. Fitur-fitur seperti *SynFlagDist*, *AckFlagDist*, *AvgLen*, *AvgIAT*, *DeltaTimeFlow*, *NumCon*, dan atribut jaringan lainnya dipilih karena mampu merepresentasikan pola perilaku komunikasi jaringan yang berkaitan dengan aktivitas *malware*. Dengan demikian, proses deteksi tidak dilakukan menggunakan *pola statis* seperti pada IDS tradisional, melainkan menggunakan pola statistik trafik jaringan yang dipelajari secara otomatis oleh model *machine learning*.

Algoritma *Random Forest* digunakan untuk membangun model klasifikasi yang mampu mengenali pola-pola tersebut. *Random Forest* merupakan metode *ensemble learning* yang bekerja dengan membangun sejumlah pohon keputusan (*decision tree*) secara acak dan menghasilkan prediksi akhir menggunakan mekanisme *majority voting*. Pendekatan ini memungkinkan model memiliki kemampuan generalisasi yang baik serta lebih tahan terhadap *overfitting*. Model dilatih menggunakan data latih dari masing-masing skenario pembagian data, yaitu 70:30, 80:20, dan 90:10. Pada proses pelatihan, *Random Forest* membangun banyak pohon keputusan menggunakan subset data dan subset fitur yang dipilih secara acak (*bootstrap sampling* dan *feature randomness*). Setiap pohon kemudian mempelajari pola hubungan antar fitur terhadap kelas target.

Ketika data baru dimasukkan ke dalam model, setiap pohon keputusan akan menghasilkan prediksi kelas berdasarkan pola yang telah dipelajari sebelumnya. Hasil akhir klasifikasi ditentukan berdasarkan jumlah suara terbanyak (*majority voting*) dari seluruh pohon keputusan.

Dengan demikian, penelitian ini mengombinasikan keunggulan kemampuan *machine learning Random Forest* dalam melakukan klasifikasi secara otomatis dan adaptif terhadap data trafik jaringan.

4.1.5 Hasil Evaluasi Model

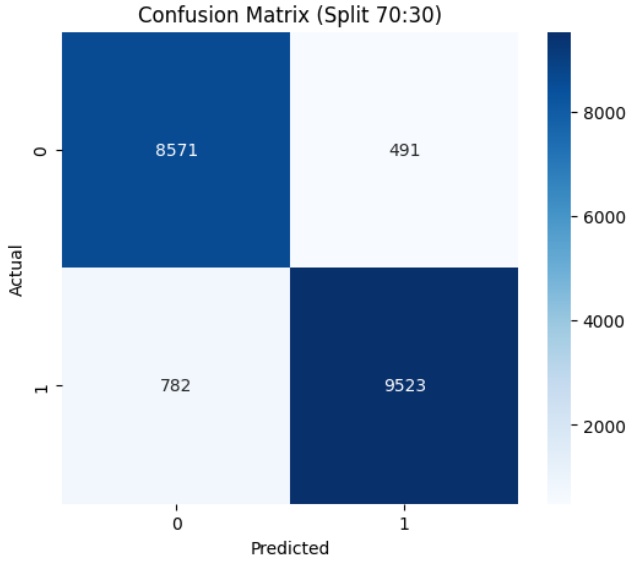
Setelah proses pelatihan model selesai dilakukan, tahap berikutnya adalah evaluasi performa model *Random Forest* pada masing-masing skenario pembagian data. Evaluasi dilakukan untuk mengukur kemampuan model dalam mengklasifikasikan trafik jaringan ke dalam kelas *malware* dan normal berdasarkan data yang belum pernah dilihat sebelumnya.

Evaluasi dilakukan pada tiga skenario pembagian data, yaitu 70:30, 80:20, dan 90:10. Tujuan penggunaan beberapa skenario adalah untuk mengetahui pengaruh proporsi data latih dan data uji terhadap performa model serta mengukur kemampuan generalisasi model *Random Forest* dalam mendeteksi pola serangan *malware.*

1. **Skenario 1 (70% data latih – 30% data uji)**

   Pada skenario pertama, dataset dibagi menjadi 70% data latih dan 30% data uji. Skenario ini memberikan proporsi data uji yang lebih besar dibandingkan skenario lainnya sehingga evaluasi dilakukan pada jumlah data yang lebih banyak. Pengujian ini bertujuan untuk mengetahui

kemampuan model dalam melakukan generalisasi terhadap data baru yang belum pernah dipelajari sebelumnya.



Gambar 4. 1 *Confusion Matrix* (70:30)

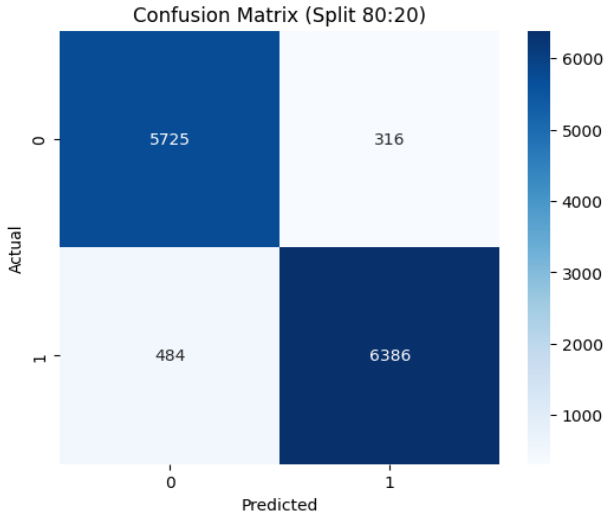
Berdasarkan confusion matrix pada skenario 70:30, model berhasil mengklasifikasikan sebagian besar data dengan benar. Sebanyak 8.571 data normal berhasil diprediksi dengan tepat sebagai kelas normal (*True Negative*), sedangkan 9.523 data *malware* berhasil dikenali dengan benar sebagai *malware* (*True Positive*).

Namun demikian, masih terdapat beberapa kesalahan klasifikasi. Sebanyak 491 data normal diprediksi sebagai *malware* (*False Positive*), sedangkan 782 data *malware* diprediksi sebagai normal (*False Negative*). Nilai kesalahan ini relatif kecil dibandingkan jumlah total data sehingga performa model tetap tergolong sangat baik.

Hasil evaluasi menunjukkan bahwa model *Random Forest* pada skenario 70:30 memperoleh nilai *accuracy* sebesar 0,934 dengan *precision* sebesar 0,935, *recall* sebesar 0,934, dan *F1-score* sebesar 0,934. Nilai tersebut menunjukkan bahwa model memiliki kemampuan yang baik dalam mendeteksi pola *malware* maupun trafik normal secara seimbang.

2. **Skenario 2 (80% data latih – 20% data uji)**

Pada skenario kedua, dataset dibagi menjadi 80% data latih dan 20% data uji. Penambahan jumlah data latih bertujuan agar model memiliki lebih banyak informasi untuk mempelajari pola karakteristik trafik jaringan sehingga diharapkan dapat meningkatkan performa klasifikasi.



Gambar 4. 2 *Confusion Matrix* (80:20)

Berdasarkan *confusion matrix* pada skenario 80:20, model menunjukkan peningkatan performa dibandingkan skenario sebelumnya. Sebanyak 5.725 data normal berhasil diprediksi dengan benar sebagai kelas

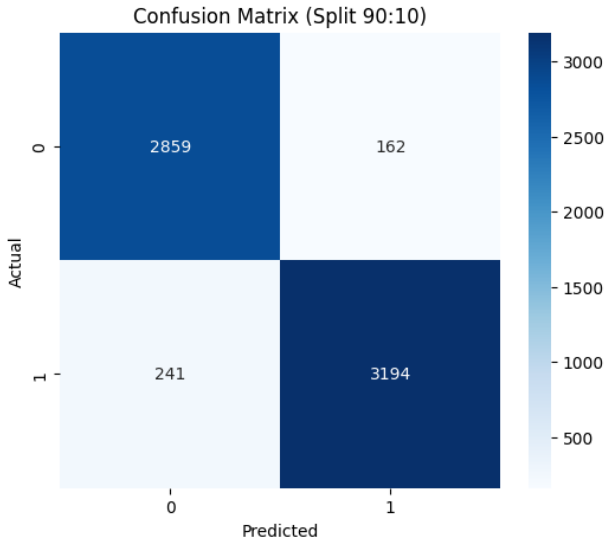
normal, sedangkan 6.386 data *malware* berhasil dikenali dengan tepat sebagai *malware.*

Kesalahan klasifikasi pada skenario ini juga mengalami penurunan. Sebanyak 316 data normal diprediksi sebagai *malware*, sedangkan 484 data *malware* diprediksi sebagai normal. Jumlah kesalahan yang lebih sedikit menunjukkan bahwa model mampu mempelajari pola data dengan lebih baik ketika jumlah data latih ditingkatkan.

Hasil evaluasi menunjukkan bahwa model memperoleh accuracy sebesar 0,938 dengan *precision, recall*, dan *F1-score* masing-masing sebesar 0,938. Nilai tersebut menunjukkan bahwa model memiliki performa klasifikasi yang baik pada skenario pembagian data 80:20.

3. **Skenario 3 (90% data latih – 10% data uji)**

Pada skenario ketiga, dataset dibagi menjadi 90% data latih dan 10% data uji. Skenario ini memberikan jumlah data latih terbesar sehingga model memiliki lebih banyak data untuk mempelajari pola karakteristik trafik jaringan *malware* dan normal.



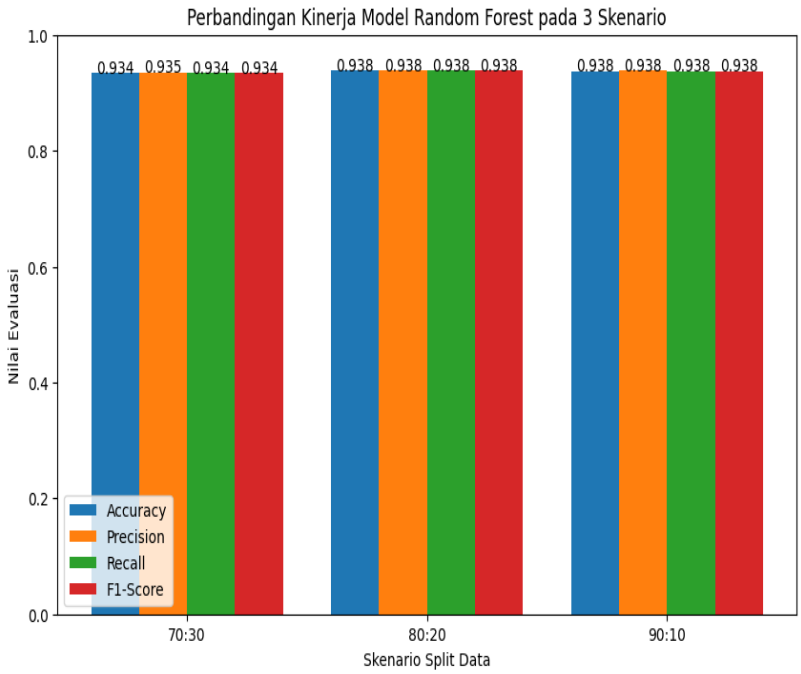
Gambar 4. 3 *Confusion Matrix* (90:10)

Berdasarkan *confusion matrix* pada skenario 90:10, model berhasil mengklasifikasikan sebagian besar data dengan benar. Sebanyak 2.859 data normal berhasil diprediksi dengan tepat sebagai kelas normal, sedangkan 3.194 data *malware* berhasil dikenali dengan benar sebagai *malware*.

Kesalahan klasifikasi pada skenario ini tergolong kecil, yaitu sebanyak 162 data normal diprediksi sebagai *malware* dan 241 data *malware* diprediksi sebagai normal. Hasil ini menunjukkan bahwa model tetap mampu mempertahankan performa klasifikasi yang baik meskipun jumlah data uji lebih sedikit dibandingkan skenario lainnya.

Nilai evaluasi pada skenario 90:10 menunjukkan accuracy sebesar 0,938 dengan *precision, recall, dan F1-score* sebesar 0,938. Hasil tersebut menunjukkan bahwa penambahan jumlah data latih memberikan kontribusi positif terhadap kemampuan model dalam mengenali pola trafik *malware*.

Untuk melihat perbandingan performa model secara keseluruhan, dilakukan visualisasi menggunakan diagram batang berdasarkan nilai *accuracy, precision, recall, dan F1-score* pada ketiga skenario pembagian data.



Gambar 4. 4 Perbandingan Kinerja Model *Random Forest*

Berdasarkan grafik perbandingan, seluruh skenario menunjukkan performa yang sangat baik dengan nilai evaluasi di atas 93%. Skenario 80:20 dan 90:10 menghasilkan performa tertinggi dengan *accuracy* sebesar 0,938, sedangkan skenario 70:30 memperoleh *accuracy* sebesar 0,934.

Perbedaan nilai performa antar skenario relatif kecil, yang menunjukkan bahwa model *Random Forest* memiliki kemampuan generalisasi yang baik terhadap berbagai proporsi pembagian data. Selain itu, nilai *precision, recall, dan F1-score*

yang seimbang menunjukkan bahwa model mampu mendeteksi *malware* dengan baik tanpa menghasilkan terlalu banyak kesalahan klasifikasi.

Secara keseluruhan, hasil evaluasi membuktikan bahwa algoritma *Random Forest* efektif digunakan untuk mendeteksi *malware* pada lalu lintas jaringan *cloud server* dengan tingkat akurasi yang tinggi.

4.1.6 Hasil K-Fold Cross Validation

Penelitian ini juga melakukan evaluasi menggunakan metode 10-Fold Cross Validation. Metode ini digunakan untuk memperoleh estimasi performa model yang lebih baik dan mengurangi kemungkinan bias akibat satu kali pembagian data.

Pada metode 10-Fold Cross Validation, data latih pada masing-masing skenario dibagi menjadi sepuluh bagian (*fold*) dengan ukuran yang hampir sama. Proses validasi dilakukan secara berulang sebanyak sepuluh kali, di mana pada setiap iterasi sembilan *fold* digunakan sebagai data pelatihan dan satu *fold* digunakan sebagai data pengujian. Proses ini dilakukan secara bergantian hingga seluruh *fold* pernah menjadi data uji.

Melalui pendekatan tersebut, model akan menghasilkan sepuluh nilai akurasi pada setiap skenario pembagian data. Nilai-nilai tersebut kemudian dihitung rata-rata (*mean accuracy*) dan standar deviasinya (*standard deviation*) untuk mengukur tingkat performa model.

Adapun hasil 10-*Fold Cross Validation* pada masing-masing skenario ditunjukkan pada tabel berikut.

Tabel 4. 9 Hasil 10 *Fold Cross Validation*

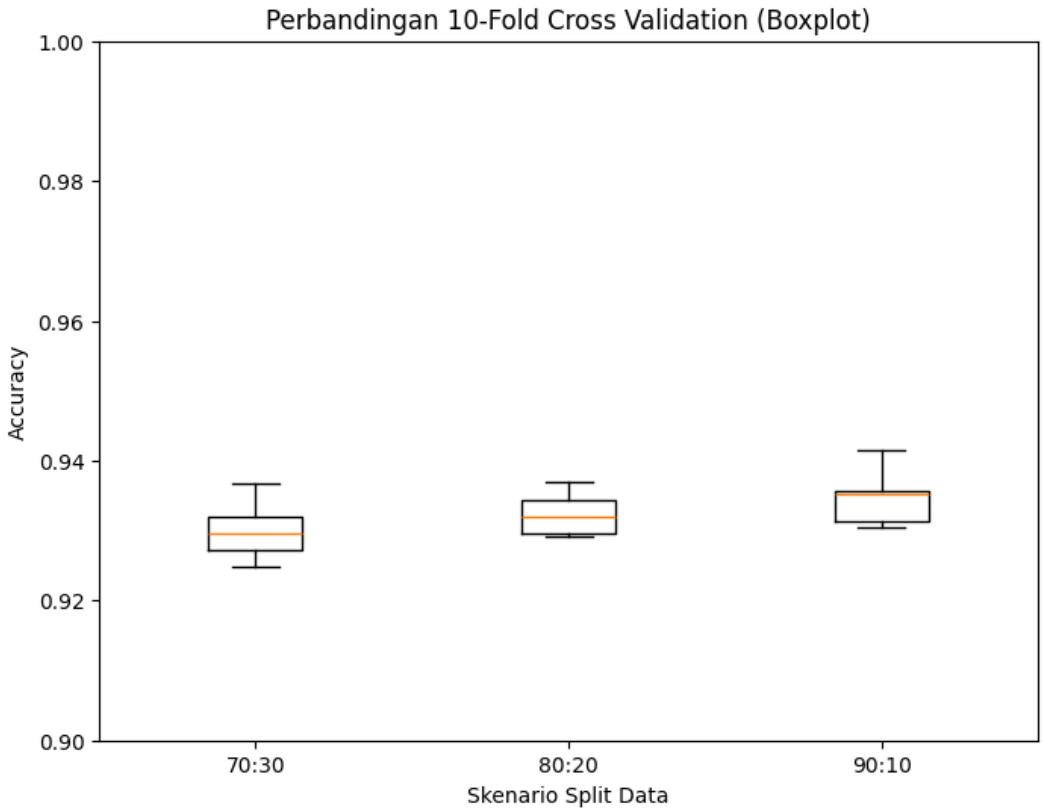
No	Split	k	Mean CV Accuracy	Std Dev
0	70:30	10	0,9298	0,0036
1	80:20	10	0,9322	0,0027
2	90:10	10	0,9344	0,0032

Berdasarkan tabel tersebut, seluruh skenario menunjukkan nilai rata-rata akurasi (*Mean CV Accuracy*) yang tinggi, yaitu di atas 92%. Hal ini menunjukkan bahwa model *Random Forest* mampu mempertahankan performa klasifikasi yang baik pada berbagai pembagian data pelatihan dan pengujian.

Skenario 90:10 menghasilkan nilai rata-rata akurasi tertinggi sebesar 0,9344, diikuti skenario 80:20 sebesar 0,9322, dan skenario 70:30 sebesar 0,9298. Hasil ini menunjukkan bahwa semakin besar jumlah data latih yang digunakan, maka model cenderung memiliki kemampuan pembelajaran pola yang lebih baik sehingga performa klasifikasi meningkat.

Selain nilai rata-rata akurasi, standar deviasi (*Std Dev*) pada seluruh skenario juga tergolong kecil, yaitu berada pada rentang 0,0027 hingga 0,0036. Nilai standar deviasi yang rendah menunjukkan bahwa performa model relatif baik pada setiap *fold* dan tidak mengalami fluktuasi yang signifikan selama proses validasi.

Untuk melihat distribusi performa model pada masing-masing fold, dilakukan visualisasi menggunakan *boxplot* sebagai berikut.



Gambar 4. 5 Perbandingan 10 *Fold Cross Validation (Boxplot)*

Berdasarkan boxplot tersebut, distribusi nilai akurasi pada ketiga skenario terlihat cukup rapat dengan rentang variasi yang kecil. Hal ini menunjukkan bahwa model *Random Forest* memiliki konsistensi performa yang baik pada seluruh proses validasi.

Skenario 90:10 menunjukkan median akurasi yang sedikit lebih tinggi dibandingkan skenario lainnya, yang menandakan bahwa penggunaan jumlah data latih yang lebih besar memberikan kontribusi positif terhadap kemampuan model dalam mengenali pola *malware*. Namun demikian, perbedaan performa antar skenario relatif kecil sehingga dapat disimpulkan bahwa model memiliki kemampuan generalisasi yang baik pada berbagai konfigurasi pembagian data.

Secara keseluruhan, hasil 10-*Fold Cross Validation* memperkuat hasil evaluasi sebelumnya bahwa algoritma *Random Forest* mampu menghasilkan performa deteksi *malware* yang akurat, dan konsisten.

4.2 Pembahasan

4.2.1 Analisis Pengaruh Split Data

Pada penelitian ini digunakan tiga skenario pembagian data (*split data*), untuk mengetahui pengaruh proporsi data latih dan data uji terhadap performa model *Random Forest* dalam mendeteksi *malware* pada lalu lintas jaringan. Perbedaan proporsi pembagian data memengaruhi jumlah data yang digunakan model selama proses pembelajaran. Semakin besar jumlah data latih, maka model memiliki lebih banyak informasi untuk mempelajari pola karakteristik trafik jaringan. Sebaliknya, semakin besar data uji, maka proses evaluasi menjadi lebih representatif karena model diuji menggunakan data yang lebih banyak dan lebih beragam.

Pada skenario 70:30, model memperoleh accuracy sebesar 0,934 dengan mean cross validation sebesar 0,9298. Skenario ini memiliki jumlah data uji paling besar dibandingkan skenario lainnya, sehingga proses evaluasi dilakukan terhadap data yang lebih banyak. Meskipun demikian, model tetap mampu menghasilkan performa yang tinggi. Hal ini menunjukkan bahwa pola *malware* yang direpresentasikan melalui fitur-fitur jaringan dapat dikenali dengan baik oleh model *Random Forest*.

Pada skenario 80:20, performa model mengalami peningkatan dengan accuracy sebesar 0,938 dan mean cross validation sebesar 0,9322. Penambahan

jumlah data latih memberikan lebih banyak informasi kepada model untuk mempelajari karakteristik trafik jaringan sehingga kesalahan klasifikasi menjadi lebih sedikit dibandingkan skenario sebelumnya.

Sementara itu, pada skenario 90:10, model memperoleh performa tertinggi dengan accuracy sebesar 0,938 dan mean cross validation sebesar 0,9344. Jumlah data latih yang lebih besar memungkinkan model mempelajari pola serangan *malware* secara lebih optimal. Namun, karena jumlah data uji lebih sedikit, evaluasi dilakukan pada data yang lebih terbatas dibandingkan skenario lainnya.

Selain nilai *accuracy*, standar deviasi hasil *cross validation* pada seluruh skenario juga tergolong kecil, yaitu berada pada rentang 0,0027 hingga 0,0036. Hal ini menunjukkan bahwa model memiliki tingkat performa yang baik dan tidak mengalami fluktuasi performa yang signifikan selama proses validasi.

Secara umum, hasil penelitian menunjukkan bahwa peningkatan proporsi data latih cenderung memberikan peningkatan performa model. Namun, perbedaan performa antar skenario relatif kecil sehingga dapat disimpulkan bahwa model *Random Forest* memiliki kemampuan generalisasi yang baik pada berbagai konfigurasi pembagian data.

Hasil ini juga menunjukkan bahwa algoritma *Random Forest* efektif digunakan dalam mendeteksi *malware* karena mampu menghasilkan performa klasifikasi yang akurat pada seluruh skenario pengujian.

Dengan demikian, skenario pembagian data 90:10 dapat dianggap sebagai konfigurasi terbaik pada penelitian ini karena menghasilkan nilai accuracy dan mean cross validation tertinggi. Namun demikian, skenario 80:20 juga

menunjukkan performa yang sangat baik dan lebih seimbang antara jumlah data latih dan data uji, sehingga dapat menjadi pilihan yang lebih optimal dalam implementasi nyata.

4.2.2 Analisis Feature Importance dan Implikasinya

Selain digunakan untuk proses klasifikasi, algoritma *Random Forest* juga memiliki kemampuan untuk mengukur tingkat kepentingan setiap fitur (*feature importance*) terhadap hasil prediksi model.

Analisis *feature importance* dilakukan untuk mengetahui fitur-fitur mana yang paling berpengaruh dalam membedakan trafik *malware* dan trafik normal pada *cloud server*.

Pada penelitian ini, nilai *feature importance* diperoleh dari proses pelatihan model *Random Forest* berdasarkan kontribusi masing-masing fitur dalam membentuk pohon keputusan (*decision tree*). Semakin besar nilai *importance* suatu fitur, maka semakin besar pula pengaruh fitur tersebut dalam membantu model melakukan klasifikasi.

Analisis ini penting dilakukan karena dapat memberikan pemahaman mengenai karakteristik trafik jaringan yang paling relevan dalam proses deteksi *malware*.

Selain itu, hasil *feature importance* juga dapat digunakan sebagai dasar untuk optimasi model, pengurangan fitur yang kurang relevan, serta pengembangan sistem deteksi *malware* berbasis perilaku jaringan (*behavior-based detection*).

Nilai *feature importance* pada algoritma *Random Forest* diperoleh berdasarkan kontribusi setiap fitur dalam mengurangi tingkat *impurity*

(ketidakmurnian) pada setiap node pohon keputusan yang terbentuk selama proses pelatihan. Pada penelitian ini, *Random Forest* menggunakan kriteria Gini *Impurity* untuk menentukan fitur terbaik yang digunakan sebagai pemisah (*split*) pada setiap *node*.

Nilai Gini *Impurity* dihitung menggunakan Persamaan berikut:

$$Gini(D) = 1 - \sum_{i=1}^{n} p_i^2 \quad (4.1)$$

Keterangan:

1. D= himpunan data pada suatu node
2. p_i= proporsi data pada kelas ke-i
3. n= jumlah kelas

Pada setiap proses *split*, *Random Forest* akan menghitung penurunan *impurity* yang dihasilkan oleh suatu fitur. Besarnya kontribusi fitur dihitung menggunakan Persamaan:

$$\Delta Gini = Gini(parent) - \left(\frac{N_{left}}{N_{parent}} Gini(left) + \frac{N_{right}}{N_{parent}} Gini(right)\right) \quad (4.2)$$

Keterangan:

1. $Gini(parent)$= *impurity node* sebelum pemisahan
2. $Gini(left)$= *impurity node* kiri
3. $Gini(right)$= *impurity node* kanan
4. N_{parent}= jumlah data pada *node* induk
5. N_{left}= jumlah data pada *node* kiri
6. N_{right}= jumlah data pada *node* kanan

Semakin besar nilai $\Delta Gini$, semakin besar kontribusi fitur tersebut dalam memisahkan data *malware* dan normal. Setelah seluruh pohon pada *Random Forest* terbentuk, kontribusi setiap fitur dijumlahkan dari seluruh *node* dan seluruh pohon keputusan. Nilai tersebut kemudian dinormalisasi sehingga total *importance* seluruh fitur bernilai 1.

Persamaan normalisasi *feature importance* adalah:

$$FI_j = \frac{\sum_{t=1}^{T} \Delta Gini_{j,t}}{\sum_{k=1}^{M} \sum_{t=1}^{T} \Delta Gini_{k,t}} \quad (4.3)$$

Keterangan:

1. FI_j= *feature importance* fitur ke-j
2. T= jumlah pohon pada *Random Forest*
3. M= jumlah fitur
4. $\Delta Gini_{j,t}$= penurunan *impurity* yang dihasilkan fitur ke-j pada pohon ke-t

Hasil normalisasi inilah yang menghasilkan nilai *feature importance* pada Tabel 4.10. Semakin besar nilai *feature importance* suatu fitur, maka semakin besar kontribusinya dalam proses klasifikasi *malware* dan normal.

Berikut merupakan hasil analisis *feature importance* pada model *Random Forest* yang digunakan dalam penelitian ini.

Tabel 4. 10 *Feature Importance*

No	*Feature*	*Importance*
1	AvgLen	0.192210
2	StdDevLen	0.171086
3	DeltaTimeFlow	0.159903
4	AvgIAT	0.150250
5	AckFlagDist	0.116248
6	SynFlagDist	0.075833
7	HTTPpkts	0.044819
8	RstFlagDist	0.034606
9	NumIPdst	0.023397
10	NumCon	0.019764
11	DNSoverIP	0.005509
12	TCPoverIP	0.004184
13	UDPoverIP	0.002194

Berdasarkan tabel tersebut, fitur AvgLen memiliki nilai *importance* tertinggi sebesar 0,192210. Hal ini menunjukkan bahwa rata-rata panjang paket (*Average Packet Length*) merupakan fitur yang paling berpengaruh dalam membedakan trafik *malware* dan trafik normal. *Malware* umumnya memiliki pola ukuran paket tertentu yang berbeda dari komunikasi jaringan normal sehingga fitur ini menjadi sangat penting dalam proses klasifikasi.

Fitur dengan kontribusi terbesar berikutnya adalah StdDevLen, DeltaTimeFlow, dan AvgIAT. Ketiga fitur tersebut berkaitan dengan karakteristik statistik lalu lintas jaringan, seperti variasi ukuran paket dan pola waktu komunikasi

antar paket. Tingginya nilai *importance* pada fitur-fitur ini menunjukkan bahwa perilaku *malware* dapat dikenali melalui pola trafik jaringan yang tidak normal.

Selain itu, fitur AckFlagDist dan SynFlagDist juga memiliki kontribusi yang cukup besar terhadap proses klasifikasi. Hal ini menunjukkan bahwa distribusi flag TCP pada komunikasi jaringan dapat menjadi indikator penting dalam mendeteksi aktivitas *malware*, terutama yang berkaitan dengan pola koneksi abnormal atau komunikasi mencurigakan. Sementara itu, fitur seperti DNSoverIP, TCPoverIP, dan UDPoverIP memiliki nilai *importance* yang relatif kecil. Hal ini menunjukkan bahwa fitur-fitur tersebut memiliki pengaruh yang lebih rendah dibandingkan fitur statistik trafik lainnya. Namun demikian, fitur-fitur tersebut tetap memberikan kontribusi terhadap performa model secara keseluruhan ketika dikombinasikan dengan fitur lain.

Hasil analisis *feature importance* menunjukkan bahwa fitur-fitur yang berkaitan dengan ukuran paket, variasi paket, interval waktu komunikasi, dan pola aliran trafik jaringan merupakan faktor utama dalam mendeteksi *malware* pada jaringan. Temuan ini memperkuat bahwa penelitian menggunakan *feature selection*, di mana pola perilaku jaringan direpresentasikan melalui fitur-fitur statistik trafik. Algoritma *Random Forest* kemudian memanfaatkan fitur tersebut untuk membangun aturan klasifikasi secara otomatis melalui struktur pohon keputusan.

Implikasi dari hasil analisis ini adalah bahwa sistem deteksi *malware* dapat difokuskan pada fitur-fitur dengan kontribusi terbesar untuk meningkatkan efisiensi dan efektivitas proses deteksi. Selain itu, hasil *feature importance* juga dapat

menjadi dasar pengembangan penelitian lanjutan dalam membangun sistem keamanan berbasis analisis perilaku jaringan yang lebih adaptif dan akurat.

4.2.3 Analisis Hasil Penelitian dan Interpretasi Model

Berdasarkan seluruh hasil pengujian yang telah dilakukan, algoritma *Random Forest* menunjukkan kemampuan yang baik dalam mendeteksi *malware* pada lalu lintas jaringan.

Hasil pengujian menunjukkan bahwa peningkatan jumlah data latih cenderung meningkatkan performa model. Skenario 90:10 menghasilkan nilai *accuracy* dan *mean cross validation* tertinggi dibandingkan skenario lainnya karena model memperoleh data pelatihan yang lebih banyak untuk mempelajari pola trafik jaringan. Namun demikian, perbedaan performa antar skenario relatif kecil sehingga menunjukkan bahwa model memiliki kemampuan generalisasi yang baik pada berbagai konfigurasi pembagian data.

Selain itu, hasil analisis *feature importance* menunjukkan bahwa fitur-fitur yang berkaitan dengan panjang paket, *interval* waktu komunikasi, dan karakteristik aliran trafik memiliki kontribusi terbesar dalam proses deteksi *malware*. Hal ini menunjukkan bahwa *malware* memiliki pola komunikasi jaringan tertentu yang dapat dikenali melalui karakteristik statistik trafik.

Penggunaan *Random Forest* pada penelitian ini juga memberikan beberapa keunggulan, seperti kemampuan klasifikasi yang tinggi, serta kemampuan dalam mengurangi risiko *overfitting*. Metode ini mampu membangun pola klasifikasi berdasarkan banyak pohon keputusan sehingga hasil prediksi menjadi lebih konsisten.

Meskipun demikian, penelitian ini masih memiliki keterbatasan, terutama pada penggunaan dataset yang berasal dari lingkungan tertentu sehingga pola trafik yang digunakan belum sepenuhnya merepresentasikan seluruh kondisi jaringan *cloud server* di dunia nyata. Selain itu, penelitian ini masih berfokus pada proses analisis model dan belum sampai pada implementasi sistem deteksi secara *real-time*.

Secara keseluruhan, hasil penelitian menunjukkan bahwa algoritma *Random Forest* efektif digunakan untuk mendeteksi *malware* pada jaringan *cloud server* dengan tingkat akurasi yang baik.

4.2.4 Analisis Uji Ablasi *Feature selection*

Uji ablasi dilakukan untuk mengevaluasi pengaruh penggunaan *feature selection* terhadap kinerja algoritma *Random Forest* dalam mendeteksi *malware* pada dataset MTA-KDD'19. Pengujian dilakukan dengan membandingkan dua skenario, yaitu model yang menggunakan seluruh 33 fitur (tanpa *feature selection*) dan model yang menggunakan 13 fitur hasil *feature selection*. Kedua model diuji menggunakan parameter dan skenario pengujian yang sama yaitu 80% data latih banding 20% data uji untuk memastikan perbandingan yang adil. Evaluasi dilakukan berdasarkan nilai *accuracy, precision, recall, F1-score*, hasil 10-*Fold Cross Validation*, serta waktu pelatihan model. Hasil uji ablasi digunakan untuk mengetahui apakah *feature selection* mampu meningkatkan performa klasifikasi dan efisiensi komputasi dibandingkan penggunaan seluruh fitur yang tersedia.

Tabel 4. 11 Uji Ablasi

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*Mean CV*	Waktu Training
Tanpa *Feature selection*	99,95%	100%	99,91%	99,96%	99,97%	10,15 detik
Dengan *Feature selection*	92,91%	94,39%	92,13%	93,25%	93,29%	9,90 detik

Berdasarkan hasil pengujian, model *Random Forest* yang menggunakan seluruh 33 fitur memperoleh *accuracy* sebesar 99,95%, *precision* sebesar 100%, recall sebesar 99,91%, dan F1-score sebesar 99,96%. Sementara itu, model yang menggunakan 13 fitur hasil *feature selection* memperoleh accuracy sebesar 92,91%, precision sebesar 94,39%, recall sebesar 92,13%, dan F1-score sebesar 93,25%.

Hasil tersebut menunjukkan bahwa penggunaan seluruh fitur memberikan performa klasifikasi yang lebih baik dibandingkan penggunaan fitur hasil seleksi. Hal ini mengindikasikan bahwa sebagian fitur yang dieliminasi masih mengandung informasi penting yang berkontribusi dalam proses identifikasi *malware* pada dataset MTA-KDD'19.

Dari sisi efisiensi komputasi, model dengan *feature selection* memiliki waktu pelatihan sebesar 9,90 detik, sedikit lebih cepat dibandingkan model tanpa *feature selection* sebesar 10,15 detik. Namun selisih waktu hanya sekitar 0,25 detik atau 2,5%, sehingga pengurangan jumlah fitur belum memberikan peningkatan efisiensi yang signifikan.

Berdasarkan hasil tersebut dapat disimpulkan bahwa *feature selection* pada penelitian ini berhasil mengurangi jumlah fitur dari 33 menjadi 13 fitur, namun menyebabkan penurunan performa klasifikasi. Dengan demikian, fitur-fitur yang

dieliminasi masih memiliki kontribusi terhadap proses deteksi *malware* sehingga penggunaan seluruh fitur menghasilkan kemampuan klasifikasi yang lebih baik.

Dalam penelitian ini, akurasi menjadi indikator penting untuk menilai sejauh mana model mampu menghasilkan deteksi yang tepat sesuai dengan kondisi sebenarnya. Ketepatan informasi ini sejalan dengan prinsip verifikasi yang diajarkan dalam Al-Qur'an, sebagaimana termaktub dalam QS. Al-Hujurat ayat 6. Allah berfirman:

يَا أَيُّهَا الَّذِينَ آمَنُوا إِنْ جَاءَكُمْ فَاسِقٌ بِنَبَإٍ فَتَبَيَّنُوا أَنْ تُصِيبُوا قَوْمًا بِجَهَالَةٍ

فَتُصْبِحُوا عَلَىٰ مَا فَعَلْتُمْ نَادِمِينَ

"Wahai orang-orang yang beriman, jika seorang fasik datang kepadamu membawa berita penting, maka telitilah kebenarannya agar kamu tidak mencelakakan suatu kaum karena ketidaktahuan(-mu) yang berakibat kamu menyesali perbuatanmu itu." (Q.S. Al-Hujurat: 6)

Menurut Tafsir Kementerian Agama Republik Indonesia, ayat ini menegaskan pentingnya kehati-hatian melakukan pemeriksaan, verifikasi, dan ketelitian terhadap setiap informasi sebelum diambil sebagai dasar keputusan. Menurut Tafsir Ibnu Katsir, ayat ini berfungsi sebagai peringatan agar tidak bersandar pada informasi yang meragukan. Dalam konteks penelitian ini, ayat tersebut memberikan landasan filosofis bahwa model klasifikasi harus menghasilkan prediksi yang akurat agar tidak menyesatkan dalam interpretasinya.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa metode *Random Forest* dengan *feature selection* mampu mendeteksi serangan *malware* pada dataset MTA-KDD'19 dengan performa yang baik. Skenario 80:20 dan 90:10 memberikan hasil terbaik dengan *accuracy* 93,8%, *precision 93,8%, recall 93,8%, dan F1-score* 93,8%. Namun, berdasarkan hasil uji ablasi, penggunaan seluruh fitur menghasilkan performa yang lebih tinggi dibandingkan model dengan *feature selection*, yang menunjukkan bahwa beberapa fitur yang tidak dipilih masih memiliki kontribusi dalam proses klasifikasi *malware*.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, beberapa saran yang dapat diberikan adalah sebagai berikut:

1. Bagi Peneliti Selanjutnya

   Penelitian selanjutnya disarankan untuk menggunakan dataset dengan jumlah data dan variasi jenis *malware* yang lebih banyak agar model mampu mengenali pola serangan yang lebih kompleks.

2. Bagi Pengembang Sistem Keamanan Jaringan

   Hasil penelitian ini dapat dimanfaatkan sebagai dasar dalam pengembangan sistem deteksi *malware* pada *cloud server*, khususnya dengan

memanfaatkan fitur-fitur trafik jaringan yang terbukti memiliki pengaruh besar terhadap proses deteksi *malware*.

DAFTAR PUSTAKA

Aqarni, A. (2025). A hybrid model for detecting and preventing attacks on *cloud* computing systems using blockchain technology. *International Journal of Advanced and Applied Sciences*, *12*(4), 79–87. https://doi.org/10.21833/ijaas.2025.04.010

Fadhlurrohman, M., Muliawati, A., & Hananto, B. (n.d.). *Analisis Kinerja Intrusion Detection Systempada Deteksi Anomali dengan Metode Decision Tree Terhadap Serangan Siber Analysis of Intrusion Detection SystemPerformance on Anomaly Detection with Decision Tree Method Against Cyber Attacks*. *8*(Pratomo 2016), 90–94.

Kamil, A., Tahir, M., Juliah, S., Rahmat, A. L., & Mahendra, Y. D. (2025). *SISTEM KEAMANAN BERBASIS HOST-BASED INTRUSION DETECTION SYSTEM(HIDS) MENGGUNAKAN WAZUH*. *9*(3), 5460–5466.

Lata, S., & Singh, D. (2022). *Intrusion Detection System*in *cloud* environment: Literature survey & future research directions. *International Journal of Information Management Data Insights*, *2*(2), 100134. https://doi.org/10.1016/j.jjimei.2022.100134

Mubarok, K., & Romli, M. A. (2025). *Implementation of Rule Based Method in Detecting Brute Force Attacks on Owncloud Implementasi Metode Rule Based dalam Mendeteksi Serangan Brute Force pada Owncloud*. *5*(January), 159–167.

Nugroho, E. P., Nugraha, E., & Zulfikar, M. N. (2019). *Sistem Reporting Keamanan pada Jaringan Cloud Computing Melalui bot Telegram dengan Menggunakan Teknik Intrussion Detection and Prevention System*. *5*(2), 49–57.

Pramudya, P. B. (2022). Implementation of *Intrusion Detection System*using SNORT to prevent threats in network servers. *Journal of Soft Computing Exploration*, *3*(2), 93–98. https://doi.org/10.52465/joscex.v3i2.80

Riza, F. (2023). *Jurnal Sistim Informasi dan Teknologi Sistem Deteksi Intrusi pada Server secara Realtime Menggunakan Seleksi Fitur dan Firebase Cloud Messaging*. *5*, 7–9. https://doi.org/10.37034/jsisfotek.v5i1.161

Saputra, F. A., Salman, M., Hasim, J. A. N., Nadhori, I. U., & Ramli, K. (2022). The Next-Generation NIDS Platform: *Cloud*-Based Snort NIDS Using Containers and Big Data. *Big Data and Cognitive Computing*, *6*(1). https://doi.org/10.3390/bdcc6010019

Shah, S. A. R., & Issac, B. (2018). Performance comparison of intrusion detection systems and application of *machine learning* to Snort system. *Future Generation Computer Systems*, *80*, 157–170. https://doi.org/10.1016/j.future.2017.10.016

Sommestad, T., Holm, H., & Steinvall, D. (2022). Variables influencing the effectiveness of intrusion detection systems. *Information Security Journal*, *31*(6), 711–728. https://doi.org/10.1080/19393555.2021.1975853

Sowmya, T., & Mary Anita, E. A. (2023). A comprehensive review of AI based intrusion detection system. *Measurement: Sensors*, *28*(October 2022), 100827. https://doi.org/10.1016/j.measen.2023.100827

Thakkar, A., & Lohiya, R. (2020). A Review of the Advancement in Intrusion Detection Datasets. *Procedia Computer Science*, *167*(2019), 636–645. https://doi.org/10.1016/j.procs.2020.03.330

Tiana, D., Supriyadi, O., Wahyudi, B., & Rimbawa, D. (2025). *KOMPUTA : Jurnal Ilmiah Komputer dan Informatika Studi Pustaka : Optimalisasi Deteksi Malware melalui Integrasi Pembelajaran Mesin Heuristik dan Big Data untuk Keamanan Siber Literature Study : Optimizing Malware Detection Through Integration of Heuristic Machine learning and Big Data for Cybersecurity KOMPUTA : Jurnal Ilmiah Komputer dan Informatika*. *14*(1). https://doi.org/10.34010/komputa.v14i1.

Al-Mahalli, J., & As-Suyuthi, J. (2010). *Tafsir Jalalain*. Jakarta: Pustaka Amani.

Letteri, I., Penna, G. Della, Vita, L. Di, & Grifa, M. T. (2020). *MTA-KDD ' 19 : A Dataset for Malware Traffic Detection ∗*. 1–13.