KOMPARASI ALGORITMA DECISION TREE DAN RANDOM FOREST DALAM MENDETEKSI INTRUSI JARINGAN INTERNET

SKRIPSI

Diajukan untuk memenuhi salah satu syarat kelulusan Program Sarjana

AQSHAL FARHAN MAULANA

NIM : 15200333

Program Studi Informatika

Fakultas Teknik dan Informatika

Universitas Bina Sarana Informatika Bekasi

2024

PERSEMBAHAN

Pendidikan itu mengobarkan api, bukan mengisi bejana.
(Socrates)

Dengan mengucap puji syukur kepada Allah S.W.T, skripsi ini kupersembahkan untuk:

1. Bapak Fadillah Maulana dan Ibu Siti Muniroh yang telah membesarkan dan selalu membimbing, mendukung, memotivasi, memberi apa yang terbaik, serta selalu mendoakan untuk meraih kesuksesan saya.
2. Diri saya sendiri yang telah berjuang dan bekerja keras selama ini. Terima kasih atas usaha dan ketekunan yang telah ditunjukkan. Mari kita terus berdoa, berusaha, dan pantang menyerah untuk masa depan yang lebih baik.
3. Teman-teman satu dosen pembimbing. Farhan, Renaldi dan Rizka yang telah menjadi sumber inspirasi dan dukungan selama proses pengerjaan skripsi ini. Dukungan, semangat, dan kerjasama kalian sangat berarti bagi saya.

Tanpa mereka,

karya ini tak akan pernah ada

SURAT PERNYATAAN KEASLIAN SKRIPSI

Yang bertanda tangan di bawah ini:

Nama	: Aqshal Farhan Maulana
NIM	: 15200333
Jenjang	: Sarjana (S1)
Program Studi	: Ilmu Komputer
Fakultas	: Teknik dan Informatika
Perguruan Tinggi	: Universitas Bina Sarana Informatika

Dengan ini menyatakan bahwa Skripsi yang telah saya buat dengan judul: **"Komparasi Algoritma Decision Tree dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet"** adalah asli (orsinil) atau tidak plagiat (menjiplak) dan belum pernah diterbitkan/dipublikasikan dimanapun dan dalam bentuk apapun.

Demikianlah surat pernyataan ini saya buat dengan sebenar-benarnya tanpa ada paksaan dari pihak manapun juga. Apabila di kemudian hari ternyata saya memberikan keterangan palsu dan atau ada pihak lain yang mengklaim bahwa (Skripsi) yang telah saya buat adalah hasil karya milik seseorang atau badan tertentu, saya bersedia diproses baik secara pidana maupun perdata dan kelulusan saya dari **Universitas Bina Sarana Informatika** dicabut/dibatalkan.

Dibuat di : Bekasi
Pada tanggal : 29 Juni 2024
Yang menyatakan,

Aqshal Farhan Maulana

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini, saya:

Nama	: Aqshal Farhan Maulana
NIM	: 15200333
Jenjang	: Sarjana (S1)
Program Studi	: Ilmu Komputer
Fakultas	: Teknik dan Informatika
Perguruan Tinggi	: Universitas Bina Sarana Informatika

Dengan ini menyatakan bahwa seluruh data, informasi, interpretasi serta pernyataan yang terdapat dalam karya ilmiah Penulis dengan judul "**Komparasi Algoritma Decision Tree dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet**" ini, kecuali yang disebutkan sumbernya adalah hasil pengamatan, penelitian, pengelolaan, serta pemikiran saya.

Penulis menyetujui untuk memberikan ijin kepada pihak **Universitas Bina Sarana Informatika** untuk mendokumentasikan karya ilmiah saya tersebut secara internal dan terbatas, serta tidak untuk mengunggah karya ilmiah Penulis pada repository Universitas Bina Sarana Informatika

Penulis bersedia untuk bertanggung jawab secara pribadi, tanpa melibatkan pihak Universitas Bina Sarana Informatika, atas materi/isi karya ilmiah tersebut, termasuk bertanggung jawab atas dampak atau kerugian yang timbul dalam bentuk akibat tindakan yang berkaitan dengan data, informasi, interpretasi serta pernyataan yang terdapat pada karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Bekasi
Pada Tanggal : 28 Juni 2024
Yang menyatakan,

Aqshal Farhan Maulana

PERSETUJUAN DAN PENGESAHAN SKRIPSI

Skripsi ini diajukan oleh:

Nama	: Aqshal Farhan Maulana
NIM	: 15200333
Jenjang	: Sarjana (S1)
Program Studi	: Informatika
Fakultas	: Teknik dan Informatika
Perguruan Tinggi	: Universitas Bina Sarana Informatika
Judul Skripsi	: Komparasi Algoritma Decision Tree Dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet

Telah dipertahankan pada periode 2024-1 dihadapan penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh Sarjana Komputer (S.Kom) pada Program Sarjana (S1) Program Studi Informatika di Universitas Bina Sarana Informatika.

Jakarta, 29 Juli 2024

PEMBIMBING SKRIPSI

Pembimbing I : Rian Septian Anwar, M.Kom. .....

DEWAN PENGUJI

Penguji I : Waeisul Bismi, M.Kom. .....

Penguji II : Rachmat Suryadithia, M.Kom. .....

PEDOMAN PENGGUNAAN HAK CIPTA

Skripsi sarjana yang berjudul **"Komparasi Algoritma Decision Tree dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet"** adalah hasil karya tulis asli Dimas Miftakhul Fakri dan bukan hasil terbitan sehingga peredaran karya tulis hanya berlaku di lingkungan akademik saja, serta memiliki hak cipta. Oleh karena itu, dilarang keras untuk menggandakan baik sebagian maupun seluruhnya karya tulis ini, tanpa seizin penulis.

Referensi kepustakaan diperkenankan untuk dicatat tetapi pengutipan atau peringkasan isi tulisan hanya dapat dilakukan dengan seizin penulis dan disertai ketentuan pengutipan secara ilmiah dengan menyebutkan sumbernya.

Untuk keperluan perizinan pada pemilik dapat menghubungi informasi yang tertera di bawah ini:

Nama : Aqshal Farhan Maulana

Alamat : Jl,Mawar No.7, RT 003/01, Aren Jaya, Bekasi Timur

No. Telp : 08978213218

E-mail : maulanaacouy@gmail.com

Kata Pengantar

Puji syukur ke hadirat Tuhan Yang Maha Esa, karena berkat rahmat dan karunia-Nya atas terselesaikannya Tugas Akhir dengan

judul : "**KOMPARASI ALGORITMA DECISION TREE DAN RANDOM FOREST DALAM MENDETEKSI INTRUSI JARINGAN INTERNET", yang merupakan salah satu syarat kelulusan** Program Sarjana (S1) program studi Ilmu Komputer Fakultas Teknik dan Informatika Universitas Bina Sarana Informatika.

Selama melaksanakan Praktik Kerja Lapangan dan dalam menyelesaikan Tugas akhir ini, penulis telah banyak menerima bimbingan, pengarahan, petunjuk yang membantu hingga akhir dari penulisan laporan ini. Untuk itu penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada yang terhormat :

1. Rektor Universitas Bina Sarana Informatika.
2. Dekan Fakultas Teknik dan Informatika Universitas Bina Sarana Informatika.
3. Ketua Program Studi Ilmu Komputer Fakultas Teknik dan Informatika Universitas Bina Sarana Informatika.
4. Bapak Rian Septian Anwar, M.Kom. selaku Dosen Pebimbing yang telah memberikan petunjuk dan pengarahan dalam penyelesaian tugas akhir ini.
5. Bapak/ibu dosen program studi informatika Universitas Bina Sarana Informatika yang telah memberikan pengetahuan kepada penulis.
6. Staf, karyawan dan dosen di lingkungan Fakultas Teknik dan Informatika Universitas Bina Sarana Informatika.
7. Kedua Orang Tua, yang selalu memberikan do'a, dukungan dan dorongan tiada henti.
8. Rekan-rekan mahasiswa Informatika 15.8A.05

Akhir kata, semoga laporan yang sederhana ini dapat bermanfaat.

Serta pihak lain yang terlalu banyak untuk disebut satu persatu sehingga terwujudnya penulisan ini. Penulis menyadari bahwa penulisan Skripshit ini belum dikatakan sempurna. Untuk itu penulis dengan sangat terbuka menerima kritik dan saran yang pembaca berikan. Demikian penulisan Skripsi ini penulis susun , semoga dapat bermanfaat bagi semua pihak termasuk penulis sendiri. Akhir kata penulis ucapkan terimakasih. Wassalamu'alaikum Wr.Wb.

Bekasi, Agustus 2024

Aqshal Farhan Maulana

Abstraksi

Keamanan jaringan merupakan aspek krusial dalam dunia digital saat ini, terutama dengan meningkatnya frekuensi dan kompleksitas serangan siber. Intrusion Detection System (IDS) menjadi salah satu alat penting untuk mengidentifikasi dan merespons serangan terhadap jaringan komputer. Penelitian ini bertujuan untuk membandingkan kinerja dua algoritma machine learning, Decision Tree dan Random Forest, dalam mendeteksi intrusi jaringan internet dengan menggunakan dataset NSL-KDD, versi yang lebih baik dari KDD 99.

Dalam penelitian ini, kedua algoritma diterapkan pada dataset NSL-KDD yang berisi 125.973 sampel trafik jaringan, menggunakan pustaka scikit-learn dan bahasa pemrograman Python. Kinerja masing-masing algoritma dievaluasi berdasarkan akurasi deteksi intrusi. Hasil penelitian menunjukkan bahwa algoritma Decision Tree mencapai akurasi sebesar 93%, sementara algoritma Random Forest mencapai akurasi yang sangat baik sebesar 99%.

Penelitian ini menyimpulkan bahwa meskipun kedua algoritma menunjukkan kinerja yang baik, Random Forest lebih unggul dalam hal akurasi deteksi intrusi pada dataset NSL-KDD. Temuan ini diharapkan dapat memberikan kontribusi signifikan dalam pengembangan sistem keamanan jaringan yang lebih efektif dan efisien, serta memberikan wawasan bagi penelitian lebih lanjut dalam bidang deteksi intrusi menggunakan teknologi machine learning.

Kata Kunci: Keamanan Jaringan, Intrusion Detection System (IDS), Random Forest, Decision Tree, Deteksi Intrusi, Machine Learning, NSL-KDD, Scikit-learn, Python.

Abstract

Network security is a crucial aspect in today's digital world, especially with the increasing frequency and complexity of cyberattacks. Intrusion Detection System (IDS) is one of the important tools to identify and respond to attacks on computer networks. This research aims to compare the performance of two machine learning algorithms, Decision Tree and Random Forest, in detecting internet network intrusions using the NSL-KDD dataset, an improved version of KDD 99.

In this study, both algorithms were applied to the NSL-KDD dataset containing 125,973 network traffic samples, using the scikit-learn library and the Python programming language. The performance of each algorithm was evaluated based on intrusion detection accuracy. The results showed that the Decision Tree algorithm achieved an accuracy of 93%, while the Random Forest algorithm achieved an excellent accuracy of 99%.

This study concludes that although both algorithms show good performance, Random Forest is superior in terms of intrusion detection accuracy on the NSL-KDD dataset. The findings are expected to significantly contribute to the development of a more effective and efficient network security system, as well as provide insights for further research in the field of intrusion detection using machine learning technology.

Keywords: Network Security, Intrusion Detection System (IDS), Random Forest, Decision Tree, Intrusion Detection, Machine Learning, NSL-KDD, Scikit-learn, Python.

Daftar Isi

Daftar Gambar

Daftar Tabel

Daftar Lampiran

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Saat ini, penggunaan internet semakin meluas, sehingga hampir setiap aspek kehidupan seperti pendidikan, industri, bisnis, kesehatan, dan lainnya sangat bergantung pada koneksi internet untuk berkomunikasi dan berbagi informasi dengan siapa saja, di mana saja, dan kapan saja. Namun, hal ini juga membawa risiko, karena ada pihak-pihak yang melakukan serangan ke jaringan bisnis untuk merusak atau mencuri informasi pribadi. Pada April 2023, Indonesia mencatat jumlah anomali trafik tertinggi di dunia, dengan total 27.476.788 anomali. Puncaknya terjadi pada 18 April 2023, dengan 1.600.334 anomali dalam satu hari. Jenis serangan yang terdeteksi meliputi Malware, Trojan, Information Leak, exploit, DoS, APT, dan lainnya, dengan Malware sebagai jenis serangan yang paling dominan. Sektor administrasi pemerintahan menjadi target utama dengan 101 kasus peretasan, sementara sektor lainnya mencatat 22 aduan terkait serangan siber (Mindara, 2023).

Oleh karena itu, diperlukan sistem keamanan seperti Intrusion Detection System (IDS) yang dapat memantau lalu lintas jaringan untuk mendeteksi aktivitas berbahaya dan memberikan peringatan secara real-time. IDS yang efektif dan efisien sangat dibutuhkan untuk mengamankan setiap lalu lintas jaringan. Penggunaan Machine Learning dapat membantu IDS dalam mengidentifikasi setiap lalu lintas jaringan yang masuk. Machine Learning adalah disiplin ilmu yang mengembangkan algoritma dan model statistik yang memungkinkan sistem komputer menjalankan tugas tanpa instruksi eksplisit, melainkan berdasarkan pola dan inferensi. Penerapan Machine Learning pada deteksi intrusi sangat bermanfaat meskipun sulit, karena memerlukan

keterampilan dan teknik tingkat tinggi. Pola intrusi dapat bervariasi secara signifikan, dan tanda-tanda intrusi dapat tersembunyi di antara sejumlah besar trafik jaringan, menambah kesulitan bagi administrator jaringan.

Algoritma Decision Tree dan Random Forest digunakan dalam analisis dataset jaringan internet karena kemampuan mereka untuk menangani data berdimensi tinggi, pola non-linear, interpretasi yang mudah dipahami, ketahanan terhadap overfitting, dan fleksibilitas dalam menangani data yang tidak seimbang. Perbandingan antara kedua algoritma ini dilakukan untuk mengevaluasi keunggulan relatif masing-masing dalam hal akurasi, kecepatan komputasi, kemampuan adaptasi terhadap pola intrusi yang kompleks, dan kemudahan interpretasi hasil.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah dari penelitian ini yaitu:

1. Algoritma manakah yang lebih efektif dalam mendeteksi intrusi jaringan internet antara Decision Tree dan Random Forest?
2. Bagaimana pengaruh ukuran dataset terhadap kinerja deteksi intrusi menggunakan algoritma Decision Tree dan Random Forest pada jaringan internet?
3. Apakah ada perbedaan hasil *accuracy, precision, recall, dan f1-score* antara algoritma Decision Tree dan Random Forest?
4. Apakah terdapat perbedaan dalam waktu komputasi antara algoritma Decision Tree dan Random Forest dalam mendeteksi intrusi pada jaringan internet?

1.3 Tujuan dan Manfaat

Berdasarkan latar belakang dan rumusan masalah di atas maka tujuan dari penelitian ini, antara lain:

1. Membandingkan kinerja algoritma Decision Tree dan Random Forest dalam mendeteksi intrusi pada jaringan internet.
2. Menganalisis faktor-faktor yang memengaruhi keunggulan salah satu algoritma dalam mendeteksi intrusi jaringan internet.
3. Menentukan apakah terdapat perbedaan signifikan dalam akurasi, waktu komputasi, dan penanganan data tidak seimbang antara algoritma Decision Tree dan Random Forest dalam konteks deteksi intrusi jaringan internet.

Manfaat dari penelitian ini yaitu:

1. Penelitian ini dapat membantu meningkatkan keamanan jaringan internet dengan mengidentifikasi metode yang lebih efektif dalam mendeteksi intrusi, sehingga dapat mencegah ancaman keamanan.
2. Penulis dapat memperdalam pemahaman dan keahlian dalam bidang keamanan jaringan dan machine learning, khususnya dalam penggunaan dan evaluasi algoritma Decision Tree dan Random Forest.
3. Pembaca, terutama yang memiliki minat atau bekerja di bidang keamanan jaringan, dapat memperoleh pemahaman yang lebih baik tentang bagaimana algoritma Decision Tree dan Random Forest dapat diterapkan dalam deteksi intrusi.

1.4 Hipotesis

Berdasarkan asumsi yang telah ditentukan berdasarkan pengamatan atau literatur sebelumnya. Berikut beberapa hipotesis yang relevan untuk penelitian ini:

H0 : Tidak terdapat perbedaan efektivitas yang signifikan antara Decision Tree dan Random Forest dalam mendeteksi intrusi jaringan internet.

H1 : Terdapat perbedaan efektivitas yang signifikan antara Decision Tree dan Random Forest dalam mendeteksi intrusi jaringan internet.

1.5 Batasan Masalah

Penelitian ini berfokus pada analisis dan komparasi antara algoritma Decision Tree dan Random Forest dalam mendeteksi intrusi jaringan internet. Batasan masalah yang diidentifikasi dalam penelitian ini meliputi:

1. Penelitian ini hanya akan fokus pada perbandingan metode Decision Tree dan Random Forest. Algoritma lain tidak akan ditinjau dalam studi ini.
2. Bahasa pemograman yang digunakan adalah Python dengan library Scikit-learn.
3. Penelitian ini akan membatasi fokus pada jenis jaringan yang mencurigakan (anomaly) dan jaringan yang sah (normal)
4. Penelitian ini hanya akan fokus pada deteksi intrusi, bukan pada respons terhadap intrusi yang terdeteksi. Pengembangan strategi respons terhadap serangan yang terdeteksi di luar cakupan penelitian ini.

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Internet

Internet adalah jaringan global komputer yang saling terhubung, memungkinkan komunikasi dan pertukaran informasi di antara komputer-komputer yang terhubung di berbagai lokasi, baik di dalam maupun di luar negeri. Internet sering dianggap sebagai jaringan komputer global yang menyimpan dan memfasilitasi akses terhadap berbagai jenis informasi, serta sebagai platform untuk komunikasi data berupa suara, gambar, video, dan teks. Informasi ini dapat diakses melalui layanan yang disediakan oleh operator atau pemilik jaringan komputer, serta oleh pemilik informasi yang mempercayakan data mereka kepada penyedia layanan Internet. Secara konseptual, Internet dapat dianggap sebagai perpustakaan besar yang berisikan jutaan bahkan miliaran informasi atau data, termasuk teks, grafik, audio, dan animasi, serta berbagai bentuk media elektronik lainnya (Luthfansa & Rosiani, 2021). Perkembangan teknologi internet telah membawa evolusi yang tidak hanya menciptakan media baru, tetapi juga mengubah berbagai aspek kehidupan manusia. Hal ini mencakup perubahan dalam cara kita berkomunikasi dan berinteraksi, yang sebelumnya sulit untuk dibayangkan (Mulawarman dkk, 2020).

2.1.2 Trafik Jaringan

Trafik jaringan mengacu pada jumlah data yang bergerak melalui jaringan komputer dalam suatu periode waktu tertentu. Trafik ini, juga dikenal sebagai trafik data, dipecah menjadi paket-paket yang dikirim melalui jaringan dan kemudian

disusun kembali oleh perangkat atau komputer penerima. Lalu lintas jaringan dapat mengalir dalam dua arah, yaitu utara-selatan dan timur-barat. Volume lalu lintas mempengaruhi kualitas jaringan; lalu lintas yang tinggi dapat menyebabkan penurunan kecepatan unduh atau masalah pada koneksi Voice over Internet Protocol (VoIP). Selain itu, tingginya volume trafik juga dapat menjadi indikasi adanya serangan, yang mempengaruhi aspek keamanan jaringan.

Trafik yang bergerak melalui jaringan biasanya bersifat heterogen dan terdiri dari aliran dari berbagai aplikasi dan utilitas. Menghubungkan arus lalu lintas dengan aplikasi yang menghasilkannya disebut sebagai klasifikasi lalu lintas (atau identifikasi lalu lintas), yang merupakan langkah penting untuk memprioritaskan, melindungi, atau memblokir lalu lintas tertentu. Klasifikasi trafik yang akurat dan menyeluruh memungkinkan pelaksanaan berbagai tindakan atau layanan jaringan, seperti akuntansi, pemantauan, kontrol, dan pengoptimalan, dengan tujuan akhir meningkatkan kinerja dan keamanan jaringan (Wang et al., 2020). Jaringan internet harus mempunyai IP address yang dikenal jaringan global (IP Publik). IP publik didapatkan dari penyedia layanan internet dengan cara berlangganan dengan biaya tertentu (Muh. Sahal, 2021).

2.1.3 Intrusi Jaringan

Intrusi jaringan merupakan upaya yang tidak sah untuk mengakses, mengendalikan, atau merusak sumber daya dan data pada suatu jaringan komputer. Intrusi dapat terjadi melalui berbagai metode, termasuk eksploitasi kerentanan perangkat lunak, serangan brute force, dan teknik rekayasa sosial. Tujuan utama dari intrusi ini adalah untuk mencuri informasi, merusak data, atau mengganggu layanan jaringan.

Serangan jaringan dapat diklasifikasikan berdasarkan metode dan tujuan serangan, di antaranya:

1. Serangan DoS dan DDoS: Serangan Denial of Service (DoS) dan Distributed Denial of Service (DDoS) bertujuan untuk membuat layanan jaringan tidak dapat diakses oleh pengguna yang berwenang dengan cara membanjiri sistem dengan lalu lintas yang berlebihan (Brij B. Gupta, 2021).
2. Injeksi SQL: Serangan ini mengeksploitasi kerentanan dalam aplikasi web untuk menjalankan perintah SQL yang tidak sah, memungkinkan penyerang untuk mengakses, mengubah, atau menghapus data dalam basis data (Ettore Galluccio, 2020).
3. Man in the Middle (MitM): Penyerang menyusup di antara dua pihak yang berkomunikasi untuk mencuri atau memanipulasi informasi yang ditransmisikan tanpa sepengetahuan kedua belah pihak (Alex Khang, 2023).
4. Phishing: Metode serangan rekayasa sosial di mana penyerang menyamar sebagai entitas tepercaya untuk memperoleh informasi sensitif seperti kata sandi dan nomor kartu kredit (Dedik Kurniawan, 2023).

2.1.4 Intrusion Detection System

Intrusion Detection System (IDS) adalah komponen kunci dalam menjaga keamanan jaringan komputer. IDS berfungsi untuk mendeteksi dan merespons aktivitas mencurigakan atau berbahaya yang dapat mengancam integritas, kerahasiaan, dan ketersediaan sistem serta data dalam jaringan (Cybellium Ltd, 2023). Penelitian ini berfokus pada penggunaan algoritma machine learning, khususnya Decision Tree dan Random Forest, dalam mendeteksi intrusi jaringan internet. Bagian ini akan

membahas konsep dasar IDS, klasifikasi IDS, peran machine learning dalam IDS, dan studi-studi terdahulu yang relevan.

Intrusion Detection System (IDS) adalah mekanisme keamanan yang mendeteksi dan merespons aktivitas mencurigakan dalam jaringan komputer dengan memantau lalu lintas jaringan atau aktivitas sistem untuk mengidentifikasi pola yang menunjukkan serangan. IDS dapat dibagi menjadi tiga kategori utama berdasarkan metode deteksi: Signature-Based IDS, Anomaly-Based IDS dan Hybrid-Based IDS (Saranya et al., 2020).

Selain metode deteksi, IDS juga dapat diklasifikasikan berdasarkan lokasi implementasinya, yaitu IDS berbasis host (HIDS) atau IDS berbasis jaringan (NIDS). HIDS digunakan pada sebuah host tunggal untuk memantau semua aktivitas dan mendeteksi aktivitas mencurigakan serta pelanggaran kebijakan keamanan. Sebaliknya, NIDS diterapkan pada level jaringan untuk melindungi seluruh perangkat dan infrastruktur jaringan dari serangan. NIDS secara terus-menerus memonitor lalu lintas jaringan untuk mendeteksi potensi serangan dan pelanggaran keamanan (Ahmad et al., 2021).

2.1.5 Machine Learning

Sejarah *Machine learning* dimulai pada tahun 1943, di mana Warren McCulloch memperkenalkan konsep dan cara kerja jaringan saraf (Purba Daru Kusuma, 2020). *Machine learning* adalah cabang dari *Artificial Intelligence* yang berfokus pada penciptaan algoritma dan teknik yang memungkinkan komputer belajar dari data serta membuat keputusan atau prediksi tanpa perlu diprogram secara eksplisit. *Machine Learning* menggambarkan kemampuan suatu sistem untuk belajar dari data pelatihan

yang diberikan, sehingga dapat mengotomatiskan proses pembuatan model analitis dan menyelesaikan tugas-tugas terkait (Janiesch, 2021).

Beberapa definisi dan konsep dasar dari machine learning meliputi:

1. Pembelajaran dari Data

   Machine learning memungkinkan sistem komputer untuk belajar dari data yang diberikan. Dengan menganalisis data pelatihan, sistem dapat mengidentifikasi pola dan mengembangkan model yang dapat digunakan untuk memprediksi hasil atau melakukan klasifikasi pada data baru.

2. Algoritma Machine Learning

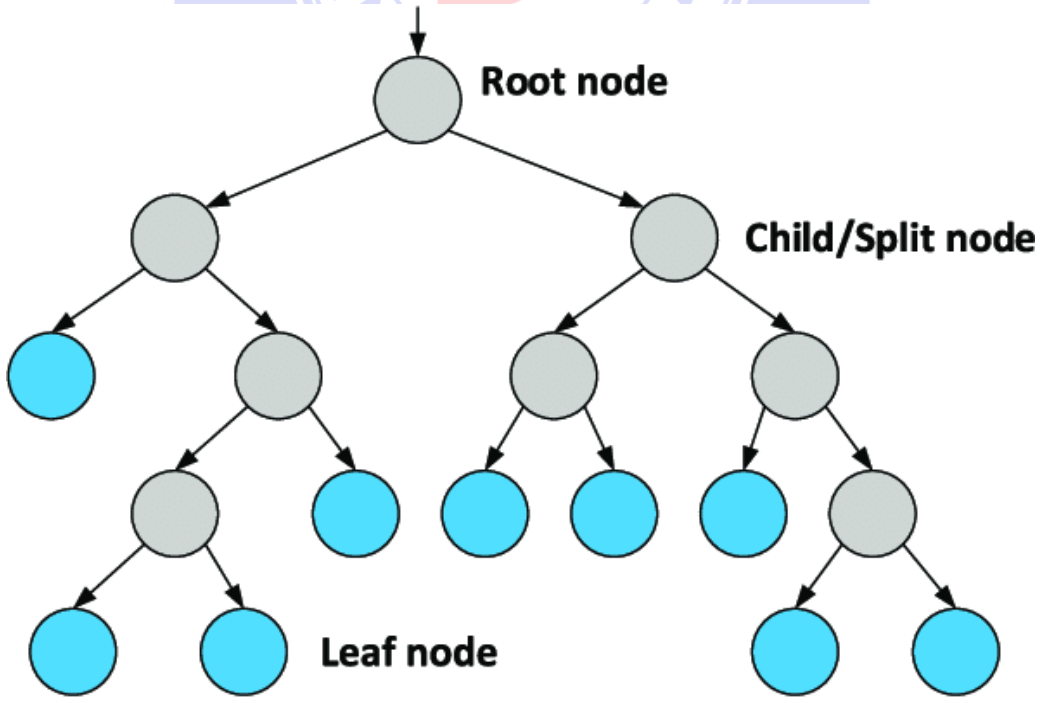
   Terdapat berbagai jenis algoritma machine learning, termasuk algoritma pembelajaran terawasi (supervised learning), pembelajaran tak terawasi (unsupervised learning), dan pembelajaran penguatan (reinforcement learning). Algoritma pembelajaran terawasi menggunakan data berlabel untuk melatih model, sedangkan pembelajaran tak terawasi menggunakan data tanpa label untuk menemukan struktur yang tersembunyi dalam data. Pembelajaran penguatan melibatkan interaksi dengan lingkungan untuk belajar melalui umpan balik dan imbalan.

3. Proses Pengembangan Model

   Proses pengembangan model dalam machine learning melibatkan beberapa langkah, termasuk pengumpulan dan persiapan data, pemilihan dan pelatihan model, evaluasi kinerja model, dan penerapan model pada data baru. Selama proses ini, data dibagi menjadi set pelatihan dan set pengujian untuk memvalidasi kinerja model dan memastikan bahwa model dapat menggeneralisasi dengan baik pada data yang belum pernah dilihat sebelumnya.

2.1.6 Decision Tree

Decision Tree adalah algoritma induksi yang telah digunakan untuk klasifikasi dalam banyak masalah (Partha Majumdar, 2023). Algoritma ini didasarkan pada pemisahan fitur dan pengujian data masing-masing fitur. Proses pemisahan terus berlanjut hingga setiap cabang dapat dilabeli hanya dengan satu klasifikasi. Pohon keputusan adalah representasi yang lebih dari sekadar representasi yang setara dengan set pelatihan. Oleh karena itu, pohon keputusan dapat digunakan untuk memprediksi data dari contoh lain yang tidak ada dalam set pelatihan. Pohon keputusan digunakan secara luas sebagai sarana untuk menghasilkan aturan klasifikasi karena adanya algoritma sederhana namun sangat kuat yang disebut *Top-Down Induction of Decision Trees* (TDIDT). Algoritma ini dijamin untuk memberikan pohon keputusan yang sesuai dengan data yang disediakan oleh dua algoritma yang paling terkenal, yaitu ID3 dan C4.5 (Guezzaz et al., 2021).



Sumber: (Camana Acosta et al., 2020)

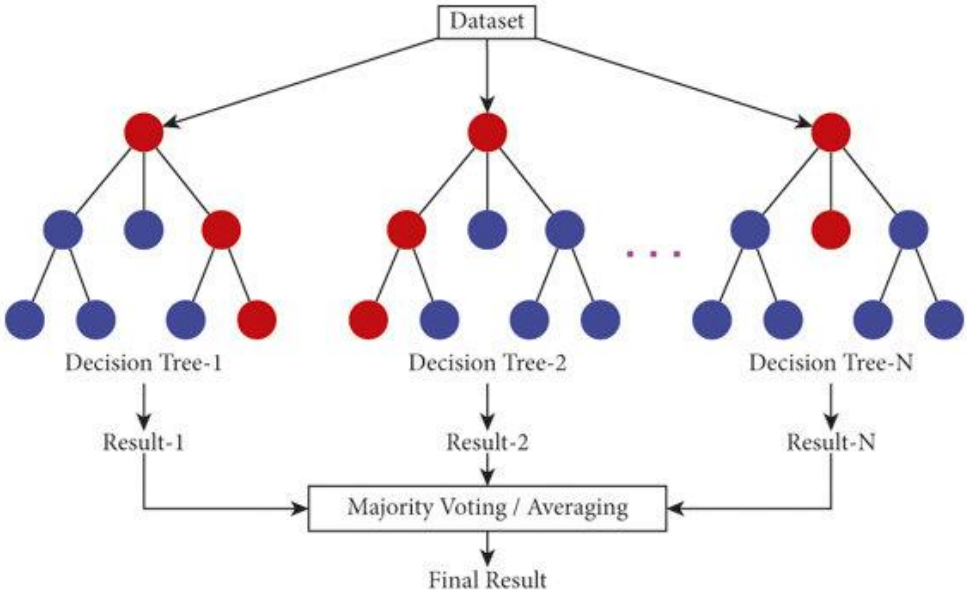
Gambar II.1 Konsep Decision Tree

Decision Tree sangat populer dan sering digunakan dalam klasifikasi dan regresi. Pohon keputusan membagi dataset menjadi subset berdasarkan data fitur, membentuk struktur pohon dengan node internal yang mewakili fitur, cabang yang mewakili aturan keputusan, dan daun yang mewakili hasil atau kelas.

Pohon klasifikasi digunakan untuk memprediksi hasil kategorikal atau diskrit, seperti jawaban "ya" atau "tidak" atau label kelas seperti "lulus" atau "tidak lulus". Di sisi lain, pohon regresi digunakan untuk memprediksi hasil yang bersifat kontinu atau numerik, seperti memprediksi pendapatan seseorang berdasarkan tingkat pendidikan, pengalaman kerja, dan faktor lainnya.

2.1.7 Random Forest

Random forest dianggap sebagai salah satu metode pembelajaran ensemble dalam subkelompok pembelajaran ensemble homogen. Dalam random forest, setiap pohon keputusan, atau dengan kata lain, setiap pembelajar dasar, memiliki akses ke subset acak dari vektor fitur (Savargiv et al., 2021).



Sumber: (Khan et al., 2021)

Gambar II.2 Konsep Random Forest

Prinsip dasar Random Forest melibatkan teknik ensemble learning yang menggabungkan beberapa pohon keputusan untuk menghasilkan model yang lebih kuat, berdasarkan prinsip bahwa kombinasi beberapa model lebih baik daripada satu model saja (T.N. Nguyen, 2022). Setiap pohon dalam Random Forest dilatih pada subset data yang dipilih secara acak melalui bootstrap sampling, yang meningkatkan keragaman model. Selain itu, ketika membangun setiap pohon, Random Forest hanya mempertimbangkan subset acak dari fitur untuk memilih pemisah pada setiap node, yang membantu mengurangi overfitting dan membuat pohon lebih spesifik terhadap fitur tertentu.

2.1.8 Python

Python adalah bahasa pemrograman komputer tingkat tinggi dan serba guna yang cocok untuk berbagai aplikasi (Kong, 2020). Bahasa ini dirancang untuk membuat kode sumber mudah dibaca dan dipahami. Python memiliki pustaka yang sangat lengkap, memudahkan programmer dalam mengembangkan aplikasi sesuai kebutuhan dengan kode yang sederhana. Selain itu, hampir semua pustaka Python dapat digunakan secara gratis (Irfan Ardiansah, 2023).

2.1.9 Scikit-Learn

Scikit-learn adalah library perangkat lunak popular di seluruh dunia yang dikembangkan untuk mendukung machine learning di Python. Library ini menyediakan berbagai alat sederhana dan efisien untuk analisis data dan pemodelan prediktif (Kevin Jolly, 2018). Scikit-learn memanfaatkan integrasi yang kuat dengan NumPy dan SciPy, sehingga memfasilitasi implementasi algoritma machine learning dengan mudah dan efisien. Fitur utama yang terdapat dalam scikit-learn adalah:

1. API Scikit-learn yang sederhana memudahkan para peneliti dan praktisi untuk mengimplementasikan model machine learning tanpa memerlukan pengetahuan mendalam tentang algoritma yang digunakan.
2. Scikit-learn memiliki modul preprocessing yang kuat untuk menangani berbagai tugas preprocessing data seperti normalisasi, standarisasi, imputer untuk nilai yang hilang, dan encoding fitur kategorikal.
3. Scikit-learn bekerja baik dengan library visualisasi seperti Matplotlib dan Seaborn, yang dapat memberikan visualisasi data dan hasil model dengan mudah.

2.1.10 Evaluation Measurement

Pengukuran evaluasi adalah langkah pengujian yang dilakukan untuk membandingkan metode yang digunakan dalam penelitian. Tahap evaluasi ini merupakan tahap akhir dari penelitian, di mana hasil atau kesimpulan dapat ditarik (Jason Brownlee, 2016). Metode evaluasi yang digunakan mencakup *recall, precision, f1-score, dan accuracy.*

1. Recall

   Recall atau sensitivitas adalah ukuran kemampuan model untuk mendeteksi semua sampel positif yang benar-benar ada dalam dataset. Rumus recall adalah:

$$\text{Recall} = \frac{TP}{TP + FN}$$

   TP (True Positive): Jumlah sampel positif yang berhasil diidentifikasi dengan benar oleh model. FN (False Negative): Jumlah sampel positif yang tidak berhasil diidentifikasi oleh model (dikenali sebagai negatif).

2. Precision

Precision adalah ukuran kemampuan model untuk mengidentifikasi sampel positif dengan benar dari semua sampel yang diprediksi sebagai positif. Rumus *precision* adalah:

$$\text{Precision} = \frac{TP}{TP + FP}$$

TP (True Positive): Jumlah sampel positif yang berhasil diidentifikasi dengan benar oleh model. FP (False Positive): Jumlah sampel negatif yang salah diidentifikasi sebagai positif oleh model.

3. F1-*Score*

F1-*Score* adalah rata-rata yang berkaitan dengan *precision* dan *recall*, yang memberikan gambaran umum tentang keseimbangan antara kedua metrik tersebut. Rumus F1-*Score* adalah:

$$\text{F1} - \text{Score} = 2 \,. \frac{Precision. Recall}{Precision + Recall}$$

F1-Score berguna ketika kita membutuhkan keseimbangan antara precision dan recall, terutama ketika kita memiliki distribusi kelas yang tidak seimbang.

4. Accuracy

Accuracy adalah ukuran kemampuan model untuk mengidentifikasi sampel positif dan negatif dengan benar dari keseluruhan sampel dalam dataset. Rumus dari accuracy adalah:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

TP (True Positive): Jumlah sampel positif yang diidentifikasi dengan benar oleh model.

TN (True Negative): Jumlah sampel negatif yang diidentifikasi dengan benar oleh model.

FP (False Positive): Jumlah sampel negatif yang salah diidentifikasi sebagai positif oleh model.

FN (False Negative): Jumlah sampel positif yang tidak diidentifikasi oleh model (teridentifikasi sebagai negatif).

2.2 Penelitian Terkait

Dalam penelitian tentang perbandingan algoritma decision tree dan random forest, merujuk pada referensi dari penelitian sebelumnya sangat penting untuk menghindari plagiarisme atau duplikasi. Tujuan utamanya adalah memastikan bahwa penelitian dengan tema serupa dapat berkembang lebih lanjut dengan kontribusi baru yang dibuat oleh penulis. Berikut adalah beberapa hasil dari penelitian sebelumnya yang berkaitan dengan penelitian ini.

Tabel II.1
Penelitian Terkait

No.	Judul	Nama Peneliti	Metode	Hasil
1	Study on Decision Tree and KNN Algorithm for Intrusion Detection System	(Ashwini Pathak & Sakshi Pathak, 2020)	Decision Tree	Model deteksi intrusi menggunakan dataset NSL-KDD yang dirancang menggunakan Decision Tree menghasilkan akurasi sebesar 99.15%, penelitian ini berhasil melakukan peningkatan

				keakurasian pelatihan dengan sangat baik.
2	A Hybrid Intrusion Detection Model Using EGA-PSO and Improved Random Forest Method	(Liu et al., 2021)	Random Forest	Pada model ini dataset NSL-KDD diproses dengan menggunakan beberapa metode, salah satunya Random Forest untuk eksperimen *Binary Class Clasification*. Tingkat akurasi yang didapatkan dari proses training penelitian ini yaitu sebesar 97.498%, tertinggi kedua setelah metode HNIDS dibandingkan dengan metode lainnya.

3	Intrusion detection system combined enhanced random forest with SMOTE algorithm	(Wu et al., 2022)	Random Forest	Deteksi intrusi jaringan berdasarkan Random Forest yang disempurnakan dan teknik Synthetic Minority Oversampling Technique (SMOTE) yang menggunakan dataset NSL-KDD dengan akurasi klasifikasi sebesar 99,72% pada dataset training set dan 78,47% pada dataset testing set.
4	Perbandingan Kinerja Algoritma K-Nearest Neighbors (K-NN) Dan Decision Tree dalam Deteksi	(Kasmara et al., 2024)	K-NN dan Decision Tree	Model deteksi paket malis menggunakan algoritma variasi nilai K pada K-NN dan Decisio Tree Hasilnya menunjukkan bahwa K-NN memiliki akurasi 91.54%, sedangkan Decision Tree memiliki

	Paket Malis pada Jaringan			92.41%. Penelitian ini menunjukkan bahwa algoritma Decision Tree memiliki kinerja yang sedikit lebih baik daripada K-Nearest Neighbor (K-NN) dalam mendeteksi paket malis.
5	Komparasi Performa Tree-Based Classifier Untuk Deteksi Anomali Pada Data Berdimensi Tinggi dan Tidak Seimbang	(Kurniabudi et al., 2022)	Random Forest	Komparasi Tree-based seperti REPTree, J48, Random Tree dan Random Forest pada deteksi trafik jaringan anomali. Penelitian ini menghasilkan sistem deteksi anomali yang memiliki performa yang tinggi. Untuk data eksperimen digunakan dataset CICIDS-2017, yang memiliki dimensionalitas data yang tinggi dan

				mengandung data tidak seimbang. Hasil pengujian memperlihatkan Random Tree memiliki akurasi 99,983% dan Random Forest 99,984%.
6	Analisa Perbandingan Klasifier Decision Tree, Random Forest, Dan Adaboost Dalam Mendeteksi Serangan Siber	(Fiqri, 2020)	Decision Tree, Random Forest dan Adaboost	Pendeteksi serangan menggunakan klasifier Decision Tree, Random Forest, dan AdaBoost. Dataset yang digunakan adalah dataset KDDcup99 dan dataset manual. Fitur yang digunakan berjumlah 14 dari total 41 fitur dalam KDDcup99 Hasil yang didapatkan adalah klasifier Decision Tree menjadi klasifier yang paling efisien dalam hal

				waktu dan performa. Dengan hasil pelatihan klasifier 9,35 detik,memprediksi serangan 1,42 detik, Precision 96,41%, Recall 100%, dan Accuracy 97,05%. Random Forest merupakan klasifier kedua yang efisien untuk mendeteksi serangan karena dibandingkan Decision Tree, Random Forest memiliki hasil yang fluktuatif pada performanya. AdaBoost kurang efisien untuk mendeteksi serangan dikarenakan waktu yang dibutuhkan untuk melatih klasifier (178.64 detik) dan

				memprediksi serangan (21.56 detik) terlalu lama.

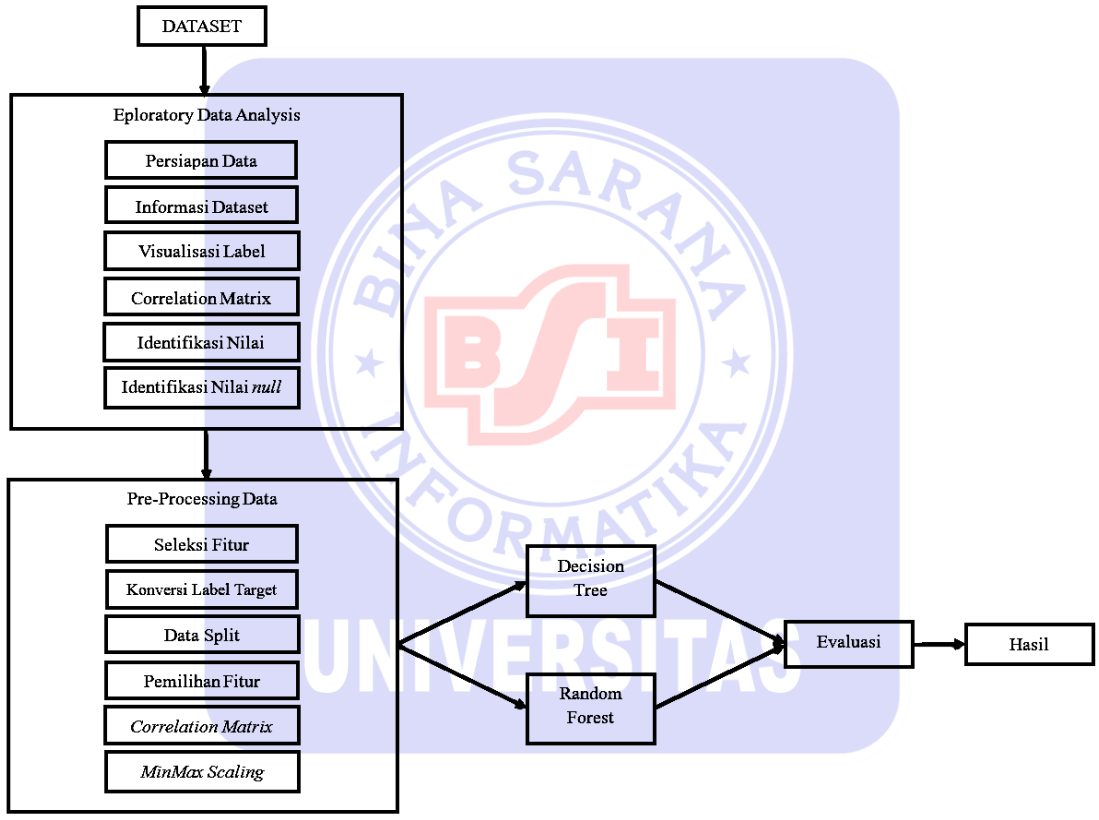
Sumber: Penelitian Terkait

BAB III

METODOLOGI PENELITIAN

3.1 Proses dan Langkah Penelitian

Proses dan Langkah Penelitian yang akan dilakukan dalam penelitian ini dapat dilihat dari gambar di bawah ini:



Sumber: Proses dan Langkahh Penelitian

Gambar III.1 Alur Penelitian

3.1.1 Dataset

Pada penelitian ini, dataset yang digunakan adalah NSL-KDD versi yang lebih baik dari dataset KDD Cup 99, yang digunakan untuk mengevaluasi sistem deteksi intrusi (IDS). NSL-KDD dikembangkan untuk mengatasi beberapa masalah yang ada pada dataset KDD Cup 99, yang sering digunakan dalam penelitian keamanan siber

untuk melatih dan menguji algoritma deteksi intrusi. KDD Cup 99 adalah dataset yang dihasilkan dari simulasi jaringan militer yang terinfeksi berbagai jenis serangan. Dataset NSL-KDD diunduh dari website Kaggle https://www.kaggle.com/datasets/hassan06/nslkdd/data. Dataset berukuran 19 MB. Dataset ini terdiri dari 125.973 baris data trafik jaringan dan memiliki 43 fitur untuk mendukung pembelajaran mesin lebih baik.

3.1.2 Exploratory Data Analysis

Pada tahap ini, dilakukan analisis eksplorasi data untuk mendapatkan informasi dari dataset yang akan digunakan, dengan langkah-langkah berikut:

1. Persiapan Data

   Pada tahap ini library pandas digunakan untuk membaca dan memasukkan data ke dalam model agar dapat diolah, dataset NSL-KDD diatur dan disesuaikan untuk langkah-langkah pemrosesan lebih lanjut.

2. Informasi Dataset

   Menampilkan informasi umum tentang dataset, seperti jumlah baris dan kolom, serta tipe data tiap kolom.

3. Visualisasi Label

   Dengan menggunakan matplotlib, dibuat pie chart untuk menggambarkan distribusi label. Terdapat label normal (53.5%), neptune (32.7%), satan (10.4%), dan label lainnya. Semua label selain "normal" menunjukkan adanya lalu lintas jaringan yang tidak wajar atau anomali, yang kemudian dikategorikan ke dalam label "anomaly".

4. Correlation Matrix

Menggunakan heatmap untuk memvisualisasikan korelasi antar fitur dalam dataset.

5. Kemiringan (Skewness)

   Menghitung nilai skewness untuk setiap kolom numerik.

6. Identifikasi Nilai *null*

   Mengidentifikasi data yang hilang pada setiap baris kolom.

3.1.3 Pre-Processing Data

Tahap ini melibatkan proses pra-pemrosesan data sebelum digunakan dalam model, dengan langkah-langkah sebagai berikut:

1. Seleksi Fitur

   Menghapus fitur yang tidak berpengaruh signifikan terhadap proses klasifikasi model.

2. Konversi Fitur

   Mengonversi beberapa fitur yang memiliki data kategori menjadi data numerik. Data kategori tidak dapat digunakan langsung oleh algoritma model, sehingga perlu diubah menjadi data numerik agar dapat diproses oleh model.

3. Data Split

   Membagi dataset menjadi data training (70%) dan data testing (30%).

4. Pemilihan Fitur

   Mengidentifikasi pengaruh setiap fitur terhadap target digunakan untuk menentukan fitur mana yang akan digunakan sebagai node akar atau node daun dalam model pohon keputusan. Sebanyak 25 fitur dengan nilai pengaruh tertinggi terhadap target dipilih untuk digunakan dalam model.

5. Correlation Matrix

   Memvisualisasikan korelasi antar fitur setelah seleksi menggunakan heatmap dari library seaborn.

6. Normalisasi Data

   Mereskalakan semua fitur dataset ke dalam rentang yang ditentukan untuk meningkatkan kinerja model, yaitu dengan menerapkan MinMax Scaling.

3.1.4 Decision Tree

Pada tahap ini, dataset yang telah siap akan digunakan untuk proses training dan testing dengan metode Decision Tree. Selama proses training, Decision Tree akan dilatih untuk mencapai akurasi yang tinggi dalam melakukan klasifikasi.

Terdapat beberapa parameter model decision tree yang terdapat pada scikit-learn yang dapat diatur untuk kustomisasi model yaitu:

1. ccp_alpha

   Parameter pruning yang mengontrol pemangkasan pohon. Defaultnya 0.0 berarti tidak ada pemangkasan. Nilai lebih tinggi akan memangkas cabang yang tidak signifikan.

2. class_weight

   Menentukan bobot untuk setiap kelas dalam klasifikasi tidak seimbang. Bisa berupa None (tanpa bobot), 'balanced' (menyesuaikan bobot otomatis berdasarkan frekuensi kelas), atau dictionary yang menentukan bobot spesifik untuk setiap kelas.

3. Criterion

   Fungsi untuk mengukur kualitas split. Bisa 'gini' untuk impurity Gini atau 'entropy' untuk gain informasi.

4. max_depth

   Kedalaman maksimum pohon. Misalnya, nilai 3 berarti pohon hanya bisa memiliki hingga 3 level.

5. max_features

   Jumlah maksimum fitur yang dipertimbangkan untuk split. None berarti semua fitur dipertimbangkan.

6. max_leaf_nodes

   Jumlah maksimum node daun. None berarti tidak ada batasan.

7. min_impurity_decrease

   Ambang minimal untuk menurunkan impurity. Split dilakukan jika impurity decrease lebih besar dari nilai ini.

8. min_samples_leaf

   Jumlah minimum sampel yang diperlukan untuk menjadi node daun. Defaultnya 1.

9. min_samples_split

   Jumlah minimum sampel untuk melakukan split pada node internal. Defaultnya 2.

10. min_weight_fraction_leaf

    Fraksi minimal dari total weight yang diperlukan untuk menjadi node daun. Defaultnya 0.0.

11. random_state

    Seed acak untuk memastikan hasil dapat direproduksi. None berarti menggunakan generator acak global.

12. Splitter

    Strategi untuk memilih split pada setiap node. Bisa 'best' untuk split terbaik atau 'random' untuk split acak.

3.1.5 Random Forest

Pada tahap ini, dataset yang telah siap akan digunakan untuk proses training dan testing dengan metode Random Foret. Selama proses training, Random Forest akan dilatih untuk mencapai akurasi yang tinggi dalam melakukan klasifikasi.

Terdapat beberapa parameter model random forest yang terdapat pada scikit-learn yang dapat diatur untuk kustomisasi model yaitu:

1. Bootstrap

   Parameter jika menggunakan sampel bootstrap saat membangun pohon.

2. ccp_alpha

   Parameter pruning untuk mengontrol pemangkasan pohon.

3. class_weight

   Bobot untuk setiap kelas dalam klasifikasi tidak seimbang.

4. Criterion

   Fungsi untuk mengukur kualitas split.

5. max_depth

   Kedalaman maksimum pohon.

6. max_features

   Jumlah maksimum fitur yang dipertimbangkan untuk split.

7. max_leaf_nodes

   Jumlah maksimum node daun.

8. max_samples

   Jumlah atau proporsi sampel untuk bootstrap. Hanya relevan jika bootstrapb bernilai ‘True’.

9. min_impurity_decrease

Ambang minimal untuk menurunkan impurity. Split dilakukan jika impurity decrease lebih besar dari nilai ini.

10. min_samples_leaf

    Jumlah minimum sampel yang diperlukan untuk menjadi node daun.

11. min_samples_split

    Jumlah minimum sampel yang diperlukan untuk split pada node internal.

12. min_weight_fraction_leaf

    Fraksi minimal dari total weight untuk menjadi node daun.

13. n_estimators

    Jumlah pohon dalam hutan.

14. n_jobs

    Jumlah threads untuk mencocokkan model.

15. oob_score

    Parameter untuk menggunakan sampel di luar tas untuk memperkirakan akurasi.

16. random_state

    Seed acak untuk memastikan hasil dapat direproduksi.

17. Verbose

    Tingkat detail log yang dihasilkan.

18. warm_start

    Jika True, fit ulang akan menambahkan pohon baru ke hutan yang ada. Jika False, hutan akan dibangun kembali.

3.1.6 Evaluasi

Evaluasi dilakukan dengan mengkomparasikan hasil kinerja model Decision Tree dan Random Forest menggunakan metrik evaluasi Precision, Recall, F1-Score,

dan Accuracy. Hasil evaluasi kemudian divisualisasikan menggunakan plot figure dengan matplotlib.

3.1.7 Hasil

Hasil akhir dari penelitian ini akan membahas komparasi kinerja algoritma Decision Tree dan Random Forest dalam mendeteksi intrusi jaringan. Analisis ini mencakup kelebihan dan kekurangan masing-masing algoritma berdasarkan metrik evaluasi yang telah dihitung.

3.2 Metode Pengolahan dan Analisis Data

Penelitian ini menggunakan Python sebagai perangkat lunak utama, bersama dengan beberapa library seperti pandas, scikit-learn, matplotlib, dan seaborn. Proses penelitian dilakukan menggunakan fitur notebook yang tersedia di website Kaggle. Metode yang digunakan untuk analisis datapada penelitian komparasi ini adalah Decision Tree dan Random Forest.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

4.1.1 Tools yang Digunakan

Kaggle adalah sebuah platform yang menyediakan fasilitas bagi para peneliti untuk mengembangkan dan menguji model machine learning mereka. Platform ini memungkinkan pengguna untuk memanfaatkan sumber daya komputasi yang tersedia di Kaggle, sehingga hanya memerlukan performa CPU yang cukup baik dari komputer pribadi. Disarankan untuk menggunakan prosesor yang kuat agar proses pemodelan dapat berjalan lebih cepat. Berikut adalah langkah-langkah menggunakan Kaggle:

1. Buka halaman website Kaggle di https://www.kaggle.com/.
2. Login ke Kaggle.
3. Pada dashboard, klik 'Models' untuk membuat model machine learning.
4. Klik “New Model”
5. Pada bagian Create Model, pilih “New Model”, isi nama model, lalu “Create Model”
6. Pilih framework scikit-learn pada bagian “Framework”
7. Klik “Go to model detail page”
8. Klik pada pilihan “Code”, klik “Your Work”, dan klik “New Notebook”
9. Tulis kode pada bagian penulisan kode.
10. Unggah dataset sebagai input pada kode.

4.1.2 Pengolahan Dataset

A. Exploratory Data Analysis

Pada tahap ini, dilakukan analisis eksplorasi data dengan langkah-langkah berikut:

1. Persiapan Data

   Library pandas digunakan untuk menyusun dan mengintegrasikan dataset NSL-KDD. Dataset dijadikan variabel ‘df’ untuk memanggil dataset lebih lanjut.

```
columns = (['duration','protocol_type','service','flag','src_bytes','dst_bytes','land',
            'wrong_fragment','urgent','hot','num_failed_logins','logged_in','num_compromised'
            ,'root_shell','su_attempted','num_root','num_file_creations','num_shells'
            ,'num_access_files','num_outbound_cmds','is_host_login','is_guest_login'
            ,'count','srv_count','serror_rate','srv_serror_rate','rerror_rate'
            ,'srv_rerror_rate','same_srv_rate','diff_srv_rate','srv_diff_host_rate'
            ,'dst_host_count','dst_host_srv_count','dst_host_same_srv_rate'
            ,'dst_host_diff_srv_rate','dst_host_same_src_port_rate'
            ,'dst_host_srv_diff_host_rate','dst_host_serror_rate'
            ,'dst_host_srv_serror_rate','dst_host_rerror_rate'
            ,'dst_host_srv_rerror_rate','attack','level'])
```

Sumber: Persiapan Data

Gambar IV.1 Nama Fitur

```
df1 = pd.read_csv('/kaggle/input/kdd-dataset/KDD_dataset.txt'
                  ,header=None,names=columns)
df = pd.concat([df1], ignore_index=True)
len(df.columns)
```

```
43
```

+ Code + Markdown

```
df
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_same_srv_rate	dst_host_diff_srv_rate
0	0	tcp	ftp_data	SF	491	0	0	0	0	0	...	0.17	0.03
1	0	udp	other	SF	146	0	0	0	0	0	...	0.00	0.60
2	0	tcp	private	S0	0	0	0	0	0	0	...	0.10	0.05
3	0	tcp	http	SF	232	8153	0	0	0	0	...	1.00	0.00
4	0	tcp	http	SF	199	420	0	0	0	0	...	1.00	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...
125968	0	tcp	private	S0	0	0	0	0	0	0	...	0.10	0.06
125969	8	udp	private	SF	105	145	0	0	0	0	...	0.96	0.01
125970	0	tcp	smtp	SF	2231	384	0	0	0	0	...	0.12	0.06
125971	0	tcp	klogin	S0	0	0	0	0	0	0	...	0.03	0.05

Sumber: Persiapan Data

Gambar IV.2 Nama Fitur 2

Fitur dataset yang berasal dari NSL-KDD menggunakan penamaan berupa angka. Oleh karena itu, kolom dataset diberi keterangan yang menjelaskan informasi tentang jenis trafik jaringan yang biasanya ditemukan, bisa dilihat pada gambar IV.4.

2. Informasi Dataset

   Menampilkan informasi umum tentang dataset, seperti jumlah baris dan kolom, serta tipe data tiap kolom.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125973 entries, 0 to 125972
Data columns (total 43 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   duration                     125973 non-null  int64
 1   protocol_type                125973 non-null  object
 2   service                      125973 non-null  object
 3   flag                         125973 non-null  object
 4   src_bytes                    125973 non-null  int64
 5   dst_bytes                    125973 non-null  int64
 6   land                         125973 non-null  int64
 7   wrong_fragment               125973 non-null  int64
 8   urgent                       125973 non-null  int64
 9   hot                          125973 non-null  int64
 10  num_failed_logins            125973 non-null  int64
 11  logged_in                    125973 non-null  int64
 12  num_compromised              125973 non-null  int64
 13  root_shell                   125973 non-null  int64
 14  su_attempted                 125973 non-null  int64
 15  num_root                     125973 non-null  int64
 16  num_file_creations           125973 non-null  int64
 17  num_shells                   125973 non-null  int64
 18  num_access_files             125973 non-null  int64
```

Sumber: Persiapan Data

Gambar IV.3 Informasi Dataset

```
 19  num_outbound_cmds            125973 non-null  int64
 20  is_host_login                125973 non-null  int64
 21  is_guest_login               125973 non-null  int64
 22  count                        125973 non-null  int64
 23  srv_count                    125973 non-null  int64
 24  serror_rate                  125973 non-null  float64
 25  srv_serror_rate              125973 non-null  float64
 26  rerror_rate                  125973 non-null  float64
 27  srv_rerror_rate              125973 non-null  float64
 28  same_srv_rate                125973 non-null  float64
 29  diff_srv_rate                125973 non-null  float64
 30  srv_diff_host_rate           125973 non-null  float64
 31  dst_host_count               125973 non-null  int64
 32  dst_host_srv_count           125973 non-null  int64
 33  dst_host_same_srv_rate       125973 non-null  float64
 34  dst_host_diff_srv_rate       125973 non-null  float64
 35  dst_host_same_src_port_rate  125973 non-null  float64
 36  dst_host_srv_diff_host_rate  125973 non-null  float64
 37  dst_host_serror_rate         125973 non-null  float64
 38  dst_host_srv_serror_rate     125973 non-null  float64
 39  dst_host_rerror_rate         125973 non-null  float64
 40  dst_host_srv_rerror_rate     125973 non-null  float64
 41  attack                       125973 non-null  object
 42  level                        125973 non-null  int64
dtypes: float64(15), int64(24), object(4)
memory usage: 41.3+ MB
```

Sumber: Persiapan Data

Gambar IV.4 Informasi Dataset 2

Tabel IV.1
Informasi Fitur

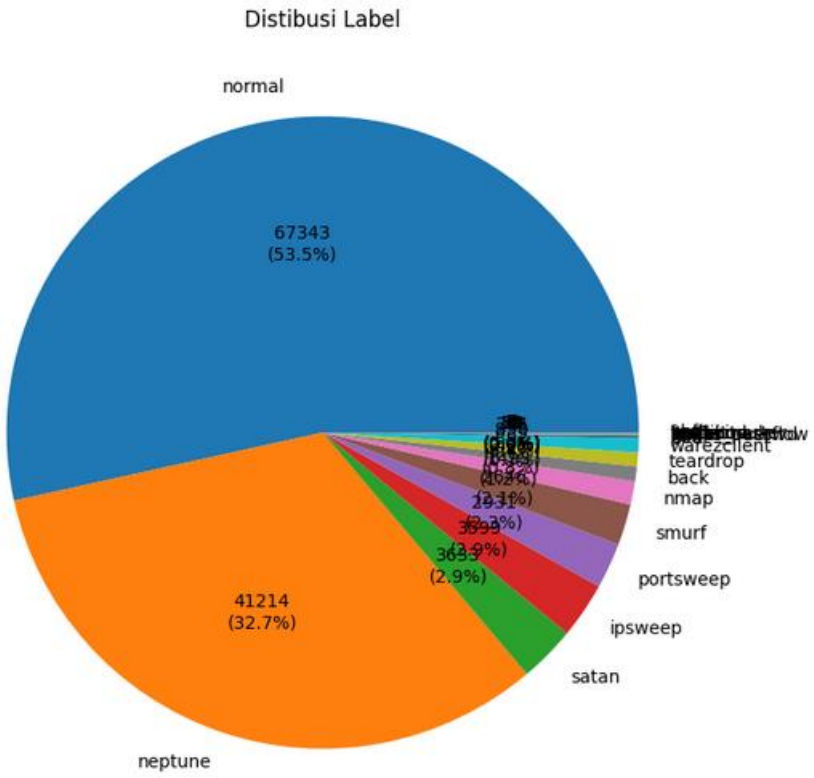
Nama Fitur	Jumlah Data	Tipe Data
duration	125973	int64
protocol_type	125973	object
service	125973	object
flag	125973	object
src_bytes	125973	int64
dst_bytes	125973	int64
land	125973	int64
wrong_fragment	125973	int64
urgent	125973	int64
hot	125973	int64
num_failed_logins	125973	int64
logged_in	125973	int64
num_compromised	125973	int64
root_shell	125973	int64
su_attempted	125973	int64
num_root	125973	int64
num_file_creations	125973	int64
num_shells	125973	int64
num_access_files	125973	int64
num_outbound_cmds	125973	int64
is_host_login	125973	int64

is_guest_login	125973	int64
count	125973	int64
srv_count	125973	int64
serror_rate	125973	float64
srv_serror_rate	125973	float64
rerror_rate	125973	float64
srv_rerror_rate	125973	float64
same_srv_rate	125973	float64
diff_srv_rate	125973	float64
srv_diff_host_rate	125973	float64
dst_host_count	125973	int64
dst_host_srv_count	125973	int64
dst_host_same_srv_rate	125973	float64
dst_host_diff_srv_rate	125973	float64
dst_host_same_src_port_rate	125973	float64
dst_host_srv_diff_host_rate	125973	float64
dst_host_serror_rate	125973	float64
dst_host_srv_serror_rate	125973	float64
dst_host_rerror_rate	125973	float64
dst_host_srv_rerror_rate	125973	float64
attack	125973	object
level	125973	int64

Sumber: Persiapan Data

3. Visualisasi Label

Dengan menggunakan matplotlib, dibuat *pie chart* untuk menggambarkan distribusi label. Berikut *pie chart* label dari fitur ‘attack’ yang terdapat pada dataset NSL-KDD:



Sumber: Visualisasi Label

Gambar IV.5 Distribusi Label

Untuk memperjelas informasi jumlah label pada fitur 'attack', berikut adalah tabel persentase yang dihasilkan dari visualisasi di atas:

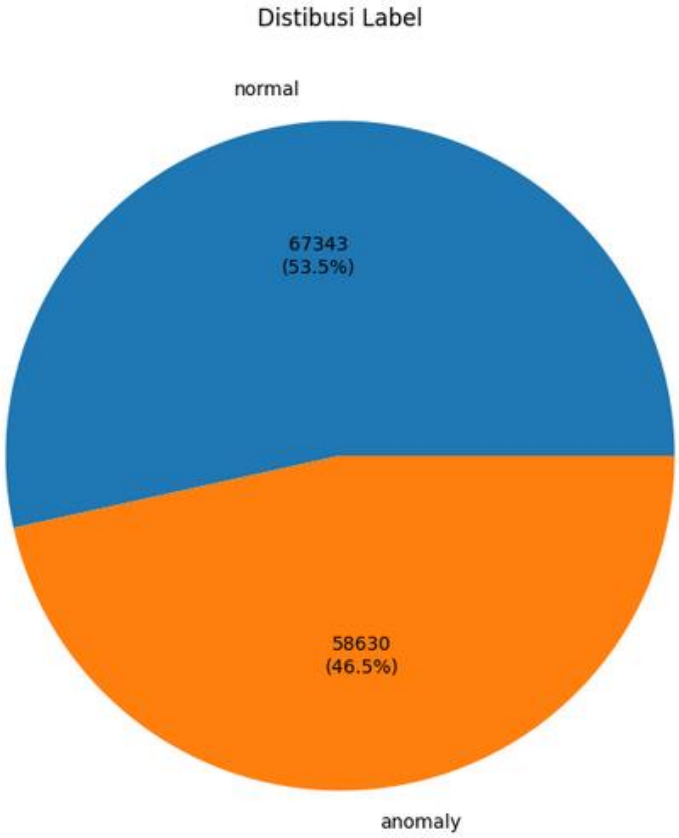
Tabel IV.2
Distribusi Label

Label	Jumlah	Persentase
normal	67343	53.45%
neptune	41214	32.71%
satan	3633	2.88%

ipsweep	3599	2.85%
portsweep	2931	2.32%
smurf	2646	2.10%
nmap	1493	1.18%
back	956	0.75%
teardrop	892	0.70%
warezclient	890	0.70%
pod	201	0.15%
guess_passwd	53	0.04%
buffer_overflow	30	0.02%
warezmaster	20	0.01%
land	18	0.01%
imap	11	0.008%
rootkit	10	0.007%
loadmodule	9	0.007%
ftp_write	8	0.006%
multihop	7	0.005%
phf	4	0.003%
perl	3	0.002%
spy	2	0.001%

Sumber: Visualisasi Label

Semua label selain ‘normal’ merupakan lalu lintas jaringan yang tidak wajar atau tidak sah. Untuk mempermudah kerja model, label-label tersebut dikelompokkan menjadi satu kategori yaitu 'anomaly'.

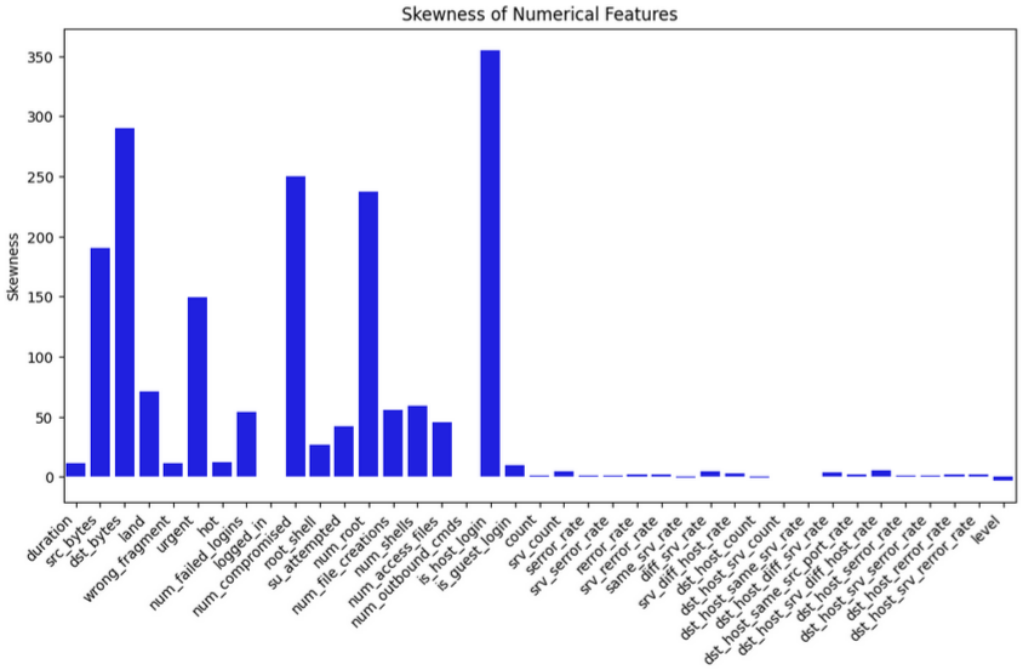


Sumber: Visualisasi Label

Gambar IV.6 Distribusi Label 2

4. Kemiringan (*Skewness*)

Menghitung data skewness untuk setiap kolom numerik. Masih terdapat *skewness* (kemiringan) data pada beberapa fitur yang dapat mempengaruhi kinerja pada model. Walaupun decision tree dan random forest memiliki resistensi terhadap data *skewness*, namun *skewness* yang ekstrim dapat mempengaruhi hasil pada model. Data *skewness* dapat dikaitkan dengan data *outliers* yang bisa menyebabkan model mengalami *overfitting.*



Sumber: Kemiringan (*Skewness*)

Gambar IV.7 Kemiringan Data Pada Setiap Fitur

5. Mendeteksi Data yang Hilang

   Mengidentifikasi data yang hilang untuk memastikan jika ada beberapa kolom yang tidak terdapat data, karena kolom yang tidak memilki data di dalamnya dapat mengakibatkan model tidak bisa berjalan. Pada penelitian ini dataset yang tersedia memiliki data pada setiap fiturnya.

B. Pre-Processing Data

Setelah data di Analisa dan terdapat beberapa fitur pada dataset yang harus diatur untuk memudahkan model untuk bekerja Tahap ini melibatkan proses *pre-processing* data sebelum digunakan dalam model, dengan langkah-langkah sebagai berikut:

1. Seleksi Fitur

   Data training NSL-KDD memiliki beragam fitur dari lalu lintas jaringan. Dalam machine learning, beberapa fitur dapat mempengaruhi kinerja model. Oleh karena itu, penting untuk mengukur ketergantungan antara dua variabel, dalam hal ini antara setiap fitur dan target. Nilai mutual yang diperoleh dapat menentukan root

node dan leaf node pada algoritma decision tree dan random forest. Pemilihan root node dan leaf node yang tepat dapat meningkatkan kinerja model dari kedua algoritma tersebut. Hasil pencarian nilai mutual antara fitur-fitur dan label target adalah sebagai berikut:

Tabel IV.3
Nilai Mutual Fitur

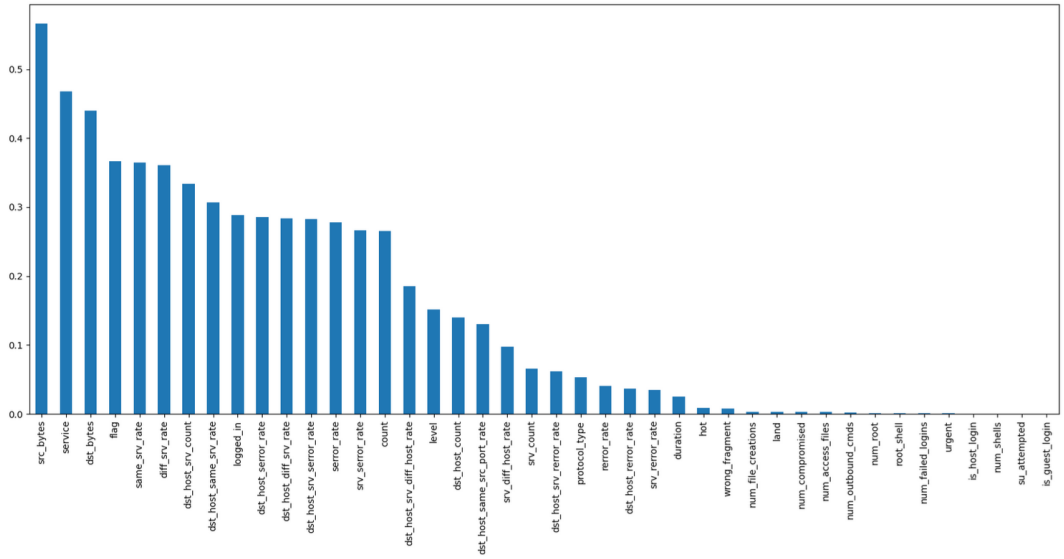
Fitur	Nilai Mutual
src_bytes	0.565895
service	0.467651
dst_bytes	0.439467
flag	0.366597
same_srv_rate	0.364998
diff_srv_rate	0.360907
dst_host_srv_count	0.334046
dst_host_same_srv_rate	0.306693
logged_in	0.287922
dst_host_serror_rate	0.28574
dst_host_diff_srv_rate	0.283622
dst_host_srv_serror_rate	0.282592
serror_rate	0.277501
srv_serror_rate	0.266391
count	0.265377
dst_host_srv_diff_host_rate	0.185037
level	0.151829

dst_host_count	0.139611
dst_host_same_src_port_rate	0.129922
srv_diff_host_rate	0.097037
srv_count	0.065349
dst_host_srv_rerror_rate	0.06154
protocol_type	0.053313
rerror_rate	0.040334
dst_host_rerror_rate	0.036776
srv_rerror_rate	0.034888
duration	0.025107
hot	0.008954
wrong_fragment	0.008086
num_file_creations	0.002912
land	0.002887
num_compromised	0.002462
num_access_files	0.002433
num_outbound_cmds	0.001846
num_root	0.001247
root_shell	0.000822
num_failed_logins	0.000701
urgent	0.000441
is_host_login	0
num_shells	0
su_attempted	0

is_guest_login	0

Sumber: Seleksi Fitur

Visualisasi dari nilai mutual pada setiap fitur:



Sumber: Seleksi Fitur

Gambar IV.8 Visualisasi Nilai Mutual

Semakin tinggi nilai yang diperoleh, semakin besar pengaruh fitur tersebut dalam prediksi target.

Setelah didapatkan hasil mutual atau ketergantungan fitur terhadap target, 25 fitur tertinggi yang berpengaruh dipilih untuk proses model selanjutnya.

25 fitur tersebut yaitu :

- src_bytes
- service
- dst_bytes
- flag
- same_srv_rate

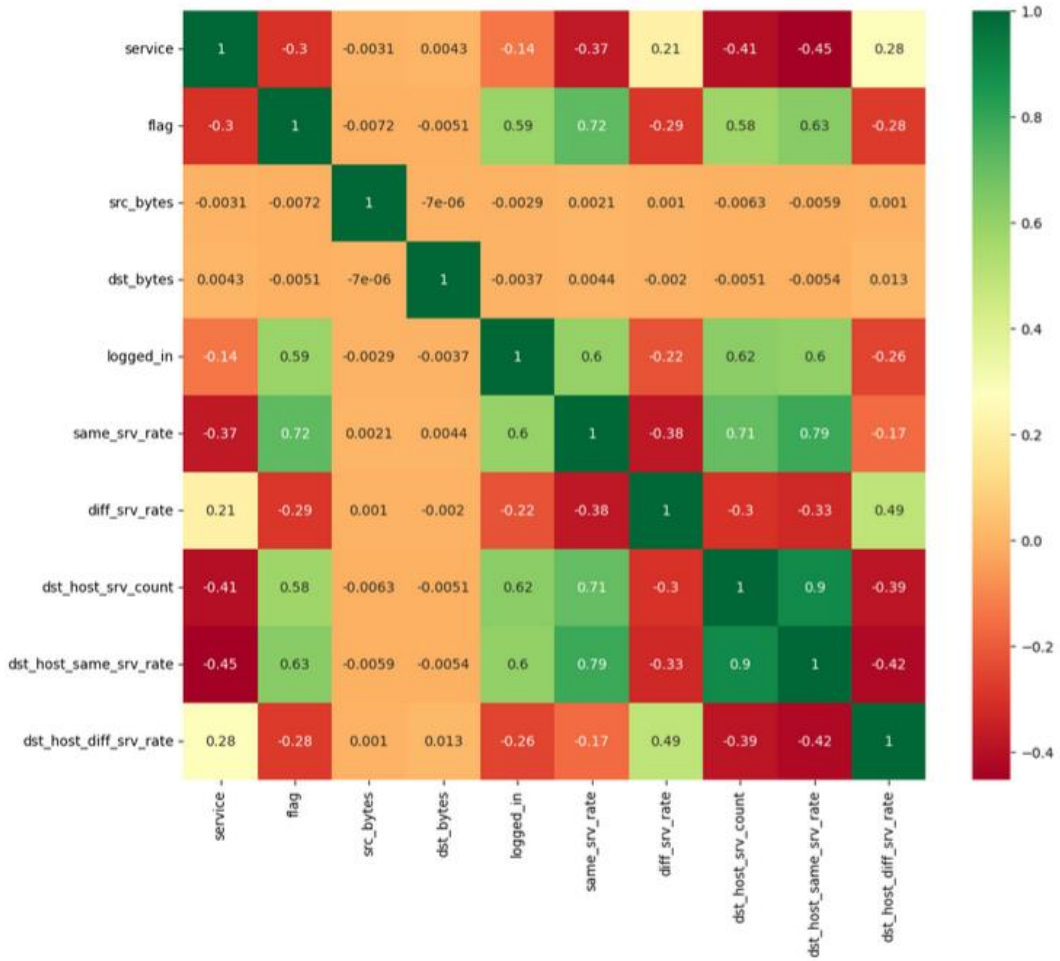
- diff_srv_rate
- dst_host_srv_count
- dst_host_same_srv_rate
- logged_in
- dst_host_serror_rate
- dst_host_diff_srv_rate
- dst_host_srv_serror_rate
- serror_rate
- srv_serror_rate
- count
- dst_host_srv_diff_host_rate
- level
- dst_host_count
- dst_host_same_src_port_rate
- srv_diff_host_rate
- srv_count
- dst_host_srv_rerror_rate
- protocol_type
- rerror_rate
- dst_host_rerror_rate

2. Data Split

   Membagi dataset menjadi data training (70%) dan data testing (30%).

3. Visualisasi *Correlation Matrix*

   Memvisualisasikan *correlation matrix* pada 10 fitur, dari 25 fitur yang telah diseleksi.



Sumber: Visualisasi Correlation Matrix

Gambar IV.9 Correlation Matrix 10 Fitur

4. Normalisasi Fitur

Pada data skew yang telah ditemukan sebelumnya, tentu harus diatur agar model dapat bekerja dengan baik terhadap fitur fitur tersebut, metode normalisasi dipilih dibandingkan standarisasi karena nilai pada fitur merupakan angka angka yang beragam.

```
# Scalling
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train= scaler.fit_transform(X_train)
X_test= scaler.fit_transform(X_test)
```

Sumber: Normalisasi Fitur

Gambar IV.10 Scalling

Normalisasi menggunakan *MinMaxScaler* adalah metode yang menskalakan semua fitur ke dalam rentang skala (0, 1). Rumus *MinMaxScaler* yang akan diterapkan pada semua fitur adalah sebagai berikut::

$$X' = \frac{X - Xmin}{Xmax - Xmin}$$

Di mana *X* adalah nilai asli atau nilai skala, *Xmin* dan *Xmax* adalah nilai minimum dan maksimum dari fitur tersebut. Setelah fitur-fitur dinormalisasi, model machine learning akan berfungsi lebih baik dengan menghindari masalah fitur yang mendominasi fitur lainnya karena skala yang lebih besar.

4.1.3 Pemodelan Decision Tree

Setelah dataset mengalami tahap pre-processing, data kemudian digunakan untuk melakukan pelatihan dengan algoritma decision tree.

```
dt_model = DecisionTreeClassifier(max_depth=3)
dt_model.get_params()
```

```
{'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': 3,
 'max_features': None,
 'max_leaf_nodes': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'random_state': None,
 'splitter': 'best'}
```

Sumber: Pemodelan Decision Tree

Gambar IV.11 Model Decision Tree

Konfigurasi parameter yang diatur pada model decision tree pada Gambar IV.11 dapat dijelaskan sebagai berikut:

a. ccp_alpha: 0.0 (tidak ada pruning)

b. class_weight: None (tidak ada bobot tambahan)

c. criterion: 'gini' (impurity Gini)

d. max_depth: 3 (kedalaman maksimum 3)

e. max_features: None (mempertimbangkan semua fitur)

f. max_leaf_nodes: None (tidak ada batasan pada jumlah node daun)

g. min_impurity_decrease: 0.0 (threshold minimal impurity decrease)

h. min_samples_leaf: 1 (minimal 1 sampel per node daun)

i. min_samples_split: 2 (minimal 2 sampel per node internal)

j. min_weight_fraction_leaf: 0.0 (tidak ada fraksi bobot minimum)

k. random_state: None (pengacakan default)

l. splitter: 'best' (memilih split terbaik)

Node akar *(root node)* dan node daun *(leaf node)* pada pohon decision tree diperoleh menggunakan metode *gini index*, yang juga dikenal sebagai metode kelanjutan dari information gain. Pohon keputusan memiliki kedalaman maksimum hingga 3, agar kedalaman pohon tidak terlalu kompleks sehingga menyebabkan *overfitting*. Hasil dari pelatihan dan testing model dengan proses waktu kurang dari 1 detik menghasilkan hasil berikut:

Tabel IV.4
Hasil Decision Tree

Decision Tree	precision	recall	f1-score	support
0	0.88	0.98	0.93	17656
1	0.98	0.88	0.93	20136

accuracy			0.93	37792
macro avg	0.93	0.93	0.93	37792
weighted avg	0.93	0.93	0.93	37792

Sumber: Pemodelan Decision Tree

Berikut adalah penjelasan dari hasil laporan Random Forest di atas:

A. Precision

Precision mengukur seberapa tepat prediksi model. Precision dihitung sebagai jumlah true positive dibagi dengan jumlah total prediksi positif.

1. Untuk label 0 (anomaly), precision adalah 0.88, yang berarti dari semua prediksi label 88% benar-benar anomaly.
2. Untuk label 1 (normal), precision adalah 0.98, yang berarti dari semua prediksi label 98% di antaranya benar-benar normal.

B. Recall

Recall mengukur seberapa baik model mendeteksi semua instance positif. Recall dihitung sebagai jumlah true positive dibagi dengan jumlah total instance positif sebenarnya.

1. Untuk label 0, recall adalah 0.98, yang berarti model mendeteksi 98% dari semua instance anomaly yang sebenarnya.
2. Untuk label 1, recall adalah 0.88, yang berarti model mendeteksi 88% dari semua instance normal dengan sempurna.

C. F1-Score

F1-Score adalah rata-rata harmonis dari precision dan recall, memberikan keseimbangan antara keduanya.

1. Untuk label 0, F1-Score adalah 0.93 atau 93%
2. Untuk label 1, F1-Score adalah 0.93.atau 93%

D. Support

Support menunjukkan jumlah instance sebenarnya untuk setiap label dalam dataset.

1. Ada 17.656 contoh data dengan label 0 (anomaly) dalam dataset.
2. Ada 20.136 contoh data dengan label 1 (normal) dalam dataset.

E. Accuracy

Accuracy mengukur persentase prediksi yang benar dari keseluruhan prediksi. Dalam hal ini, accuracy adalah 0.93, yang berarti model membuat prediksi yang benar pada 93% instance dalam dataset.

F. Macro Average

Rata-rata yang dihitung secara sederhana tanpa mempertimbangkan proporsi kelas.

1. Macro average precision sebesar 0.93 atau 93%
2. Macro average recall sebesar 0.93 atau 93%
3. Macro average F1-Score sebesar 0.93 atau 93%

G. Weighted Average

Rata-rata yang dihitung dengan mempertimbangkan proporsi dari setiap kelas.

1. Weighted average precision sebesar 0.93 atau 93%
2. Weighted average recall sebesar 0.93 atau 93%
3. Weighted average F1-Score sebesar 0.93 atau 93%

4.1.4 Pemodelan Random Forest

Pembuatan model dengan algoritma random forest dibuat untuk mengukur algoritma decision tree yang telah dilakukan. Berikut adalah penjelasan dari gambar dibawah:

```
rf_model = RandomForestClassifier()
rf_model.get_params()
```

```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': None,
 'max_features': 'sqrt',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

Sumber: Pemodelan Random Forest

Gambar IV.12 Model Random Forest

Konfigurasi parameter yang diatur pada model random forest pada Gambar IV.12 dapat dijelaskan sebagai berikut:

a. bootstrap: True (menggunakan bootstrap sampel)

b. ccp_alpha: 0.0 (tidak ada pruning)

c. class_weight: None (tidak ada bobot tambahan)

d. criterion: 'gini' (impurity Gini)

e. max_depth: None (tidak ada batasan kedalaman)

f. max_features: 'sqrt' (akar kuadrat dari jumlah total fitur)

g. max_leaf_nodes: None (tidak ada batasan pada jumlah node daun)

h. max_samples: None (menggunakan semua sampel)

i. min_impurity_decrease: 0.0 (threshold minimal impurity decrease)

j. min_samples_leaf: 1 (minimal 1 sampel per node daun)

k. min_samples_split: 2 (minimal 2 sampel per node internal)

l. min_weight_fraction_leaf: 0.0 (tidak ada fraksi bobot minimum)

m. n_estimators: 100 (jumlah pohon dalam hutan)

n. n_jobs: None (menggunakan 1 thread)

o. oob_score: False (tidak menggunakan sampel di luar tas)

p. random_state: None (pengacakan default)

q. verbose: 0 (tidak ada log)

r. warm_start: False (hutan akan dibangun kembali pada setiap panggilan fit)

Node akar (root node) dan node daun (leaf node) pada setiap pohon decision tree yang akan dibuat menjadi sekitar 100 pohon diperoleh menggunakan metode *gini index*, yang juga dikenal sebagai metode information gain. Proses pelatihan dan pengujian model ini memakan waktu 8 detik dan menghasilkan hasil seperti pada Tabel.

Tabel IV.5
Hasil Model Random Forest

Random Forest	precision	recall	f1-score	support
0	0.99	0.99	0.99	17656
1	0.99	1.00	0.99	20136
accuracy			0.99	37792
macro avg	0.99	0.99	0.99	37792
weighted avg	0.99	0.99	0.99	37792

Sumber: Pemodelan Raandom Forest

Berikut adalah penjelasan dari hasil laporan Random Forest di atas:

A. Precision

Precision mengukur seberapa tepat prediksi model. Precision dihitung sebagai jumlah true positive dibagi dengan jumlah total prediksi positif.

1. Untuk label 0 (anomaly), precision adalah 0.99, yang berarti hampir semua prediksi label 0 (anomaly) 99% benar-benar anomaly.
2. Untuk label 1 (normal), precision adalah 0.99, yang berarti hampir semua prediksi label 1 (normal), 99% benar-benar normal.

B. Recall

Recall mengukur seberapa baik model mendeteksi semua instance positif. Recall dihitung sebagai jumlah true positive dibagi dengan jumlah total instance positif sebenarnya.

1. Untuk label 0, recall adalah 0.99, yang berarti model mendeteksi 99% dari semua instance anomaly yang sebenarnya.

2. Untuk label 1, recall adalah 1.00, yang berarti model mendeteksi semua instance normal dengan sempurna.

C. F1-Score

F1-Score adalah rata-rata harmonis dari precision dan recall, memberikan keseimbangan antara keduanya.

1. Untuk label 0, F1-Score adalah 0.99 atau 99%.
2. Untuk label 1, F1-Score adalah 0.99 atau 99%.

D. Support

Support menunjukkan jumlah instance sebenarnya untuk setiap label dalam dataset.

1. Ada 17.656 contoh data dengan label 0 (anomaly) dalam dataset.
2. Ada 20.136 contoh data dengan label 1 (normal) dalam dataset.

E. Accuracy

Accuracy mengukur persentase prediksi yang benar dari keseluruhan prediksi. Dalam hal ini, accuracy adalah 0.99, yang berarti model membuat prediksi yang benar pada 99% instance dalam dataset.

F. Macro Average

Rata-rata yang dihitung secara sederhana tanpa mempertimbangkan proporsi kelas.

1. Macro average precision sebesar 0.99 atau 99%
2. Macro average recall sebesar 0.99 atau 99%
3. Macro average F1-Score sebesar 0.99 atau 99%

G. Weighted Average

Rata-rata yang dihitung dengan mempertimbangkan bagian setiap kelas.

1. Weighted average precision sebesar 0.99 atau 99%

2. Weighted average recall sebesar 0.99 atau 99%
3. Weighted average F1-Score sebesar 0.99 atau 99%

4.1.5 Hasil Pengujian

Dalam penelitian ini, kita akan membahas hasil pengujian dari dua algoritma yaitu Decision Tree dan Random Forest dalam mendeteksi intrusi jaringan menggunakan dataset NSL-KDD. Pengujian dilakukan untuk mengevaluasi performa kedua algoritma berdasarkan beberapa metrik evaluasi seperti precision, recall, f1-score, dan accuracy. Selain itu, kita juga akan membahas keefektifan waktu pelatihan dan pengujian dari kedua model tersebut.

A. Precision

Decision Tree lebih rendah dibandingkan dengan Random Forest untuk kelas normal, yaitu 0.93 dibandingkan dengan 0.99, dan untuk kelas anomaly, yaitu 0.88 dibandingkan dengan 0.99.

B. Recall

Recall untuk kelas normal pada algoritma Decision Tree lebih rendah dibandingkan dengan Random Forest yang menghasilkan nilai sempurna, 0.88 dibandingkan dengan 1.00, tetapi Decision Tree memiliki recall yang hampir sama dengan Random Forest pada kelas anomaly yaitu 0.98 dibandingkan dengan 0.99.

C. F1-Score

Random Forest memiliki F1-Score yang lebih baik untuk kedua label kelas 0.99 dibandingkan dengan Decision Tree yang menghasilkan kedua label kelas sebesar 0.93.

D. Accuracy

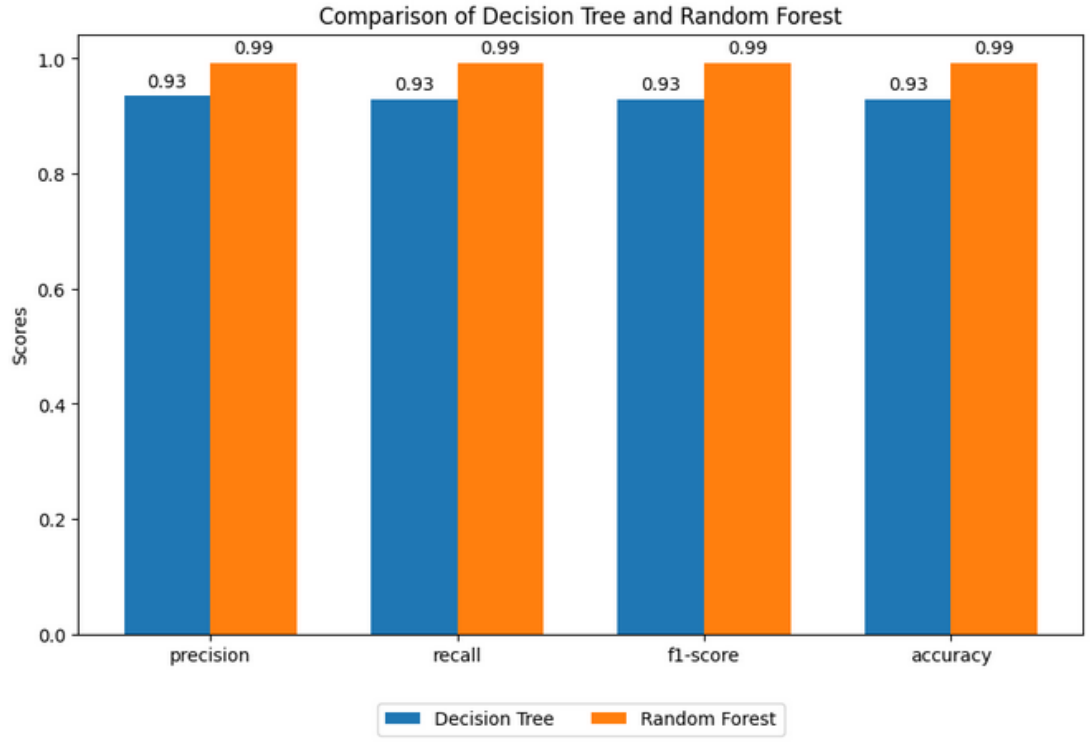
Random Forest sedikit lebih unggul dalam akurasi 0.99 dibandingkan dengan Decisio Tree 0.89.

E. Keefektifan Waktu

Decision Tree lebih cepat dalam pelatihan dan pengujian dengan lama waktu kurang dari 1 detik atau 127 *milisecond* dibandingkan Random Forest dengan lama waktu sekitar 8 detik atau 7.826 *milisecond*. Hal ini karena Random Forest adalah kombinasi dari 100 pohon keputusan, sehingga membutuhkan lebih banyak waktu untuk membangun model.

F. Kestabilan dan Akurasi

Random Forest cenderung memberikan hasil yang lebih stabil dan akurat karena menggabungkan banyak pohon keputusan, yang mengurangi risiko overfitting dibandingkan dengan Decision Tree. Hal ini terbukti dari rata-rata setiap hasil (weighted average) dari metrik seperti *precision, recall, f1-score,* dan *accuracy*.



Sumber: Kestabilan dan Akurasi

Gambar IV.13 Perbandingan Model

1. Perbandingan precision menunjukkan bahwa decision tree mencapai rata-rata 0.93, sedangkan random forest mencapai rata-rata 0.99.
2. Perbandingan recall menunjukkan bahwa decision tree mencapai rata-rata 0.93, sedangkan random forest mencapai rata-rata 0.99.
3. Perbandingan f1-score menunjukkan bahwa decision tree mencapai rata-rata 0.93, sedangkan random forest mencapai rata-rata 0.99.
4. Perbandingan accuracy menunjukkan bahwa decision tree mencapai rata-rata 0.93, sedangkan random forest mencapai rata-rata 0.99.

Berdasarkan hasil rata-rata di atas, dapat disimpulkan bahwa performa random forest lebih unggul dibandingkan model decision tree. Namun, waktu penggunaan random forest sedikit lebih lama karena harus membuat 100 pohon keputusan dari decision tree dan menyimpulkan hasil rata-rata dari hasil setiap jawaban pohon.

BAB V

PENUTUP

5.1 Kesimpulan

Dalam penelitian komparasi model Decision Tree dan Random Forest ini, dataset trafik jaringan NSL-KDD digunakan dengan kriteria pencarian node menggunakan information gain. Dataset yang memiliki jumlah data 125973, dibagi menjadi data training sebanyak 88.181 (70%) dan data testing sebanyak 37.792 (30%).

Berdasarkan pembahasan dan hasil penelitian yang telah dilakukan, maka kesimpulan yang dapat diambil adalah sebagai berikut:

1. Algoritma Random Forest menjadi algoritma yang lebih efektif dalam mendeteksi intrusi jaringan karena menghasilkan hasil evaluasi yang lebih tinggi daripada Decision Tree.
2. Dataset yang lebih besar dapat meningkatkan kinerja kedua algoritma, dataset yang lebih besar dapat membantu mengurangi overfitting, meningkatkan akurasi, dan memungkinkan model mempelajari pola yang lebih kompleks dan beragam.
3. Hasil *accuracy, precision, recall, dan f1-score* yang diperoleh oleh Decision Tree lebih rendah dibandingkan dengan Random Forest. Hal ini disebabkan karena Decision Tree hanya membangun satu pohon keputusan saja, sementara Random Forest menggunakan parameter ‘n_estimators=100’ atau pembuatan 100 pohon keputusan.
4. Random Forest membutuhkan waktu sedikit lebih lama dibandingkan Decision Tree, yaitu 7.826 *milisecond* atau sekitar 8 detik dibandingkan dengan 127 *milisecond* atau kurang dari 1 detik. Namun, dengan membangun beberapa pohon

keputusan, model terhindar dari overfitting dan menghasilkan hasil yang lebih baik.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, terdapat beberapa saran untuk penelitian selanjutnya dalam pembahasan yang sama, sebagai berikut:

1. Menambah jumlah dataset atau menggunakan data nyata agar model lebih sering belajar terhadap trafik jaringan yang semakin baru, sehingga dapat menghasilkan nilai akurasi yang lebih baik.
2. Membandingkan Random Forest dengan algoritma lain atau algoritma tingkat lanjut seperti Gradient Boost, AdaBoost, dan lainnya terhadap dataset NSL-KDD untuk mendapatkan wawasan yang lebih luas.
3. Hasil dari pengklasifikasian trafik jaringan yang sudah dilakukan model diharapkan dapat digunakan untuk teknologi *Intrusion Detection System* (IDS).

DAFTAR PUSTAKA

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1). https://doi.org/10.1002/ett.4150

Alex Khang, D. A. K. S. K. G. S. R. (2023). Smart Cities IoT Technologies, Big Data Solutions, Cloud Platforms, and Cybersecurity Techniques. CRC Press.

Ashwini Pathak, & Sakshi Pathak. (2020). Study on Decision Tree and KNN Algorithm for Intrusion Detection System. International Journal of Engineering Research And, V9(05). https://doi.org/10.17577/IJERTV9IS050303

Brij B. Gupta, A. D. (2021). Distributed Denial of Service (DDoS) Attacks Classification, Attacks, Challenges and Countermeasures. CRC Press.

Camana Acosta, M. R., Ahmed, S., Garcia, C. E., & Koo, I. (2020). Extremely Randomized Trees-Based Scheme for Stealthy Cyber-Attack Detection in Smart Grid Networks. IEEE Access, 8, 19921–19933. https://doi.org/10.1109/ACCESS.2020.2968934

Cybellium Ltd. (2023). Mastering Network Security. Cybellium Ltd.

Dedik Kurniawan. (2023). Ilmu Hacking. Elex Media Komputindo.

Ettore Galluccio, E. C. G. L. (2020). SQL Injection Strategies Practical Techniques to Secure Old Vulnerabilities Against Modern Attacks. Packt Publishing.

Fiqri, K. G. (2020). ANALISIS PERBANDINGAN KLASIFIER DECISION TREE, RANDOM FOREST, DAN ADABOOST DALAM MENDETEKSI SERANGAN SIBER. https://api.semanticscholar.org/CorpusID:212818622

Guezzaz, A., Benkirane, S., Azrour, M., & Khurram, S. (2021). A Reliable Network Intrusion Detection Approach Using Decision Tree with Enhanced Data Quality. Security and Communication Networks, 2021, 1–8. https://doi.org/10.1155/2021/1230593

Irfan Ardiansah, R. H. P. (2023). Memulai Python Belajar Python dari Nol. CV. Cendekia Press.

Janiesch, C. and Z. P. and H. K. (2021). Machine learning and deep learning. Electronic Markets, 31, 685–695.

Jason Brownlee. (2016). Machine Learning Algorithms From Scratch with Python. Machine Learning Mastery.

Kasmara, B. N., Handayani, E. T. E., & Nathasia, N. D. (2024). Perbandingan Kinerja Algoritma K-Nearest Neighbors (K-NN) Dan Decision Tree dalam Deteksi Paket

Malis pada Jaringan. STRING (Satuan Tulisan Riset Dan Inovasi Teknologi), 8(3), 320. https://doi.org/10.30998/string.v8i3.22362

Kevin Jolly. (2018). Machine Learning with Scikit-learn Quick Start Guide Classification, Regression, and Clustering Techniques in Python. Packt Publishing.

Khan, M. Y., Qayoom, A., Nizami, M. S., Siddiqui, M. S., Wasi, S., & Raazi, S. M. K.-R. (2021). Automated Prediction of Good Dictionary EXamples (GDEX): A Comprehensive Experiment with Distant Supervision, Machine Learning, and Word Embedding-Based Deep Learning Techniques. Complexity, 2021, 1–18. https://doi.org/10.1155/2021/2553199

Kong, Q. and S. T. and B. A. (2020). Python programming and numerical methods: A guide for engineers and scientists. Academic Press.

Kurniabudi, K., Harris, A., & Veronica, V. (2022). Komparasi Performa Tree-Based Classifier Untuk Deteksi Anomali Pada Data Berdimensi Tinggi dan Tidak Seimbang. JURNAL MEDIA INFORMATIKA BUDIDARMA, 6(1), 370. https://doi.org/10.30865/mib.v6i1.3473

Liu, C., Gu, Z., & Wang, J. (2021). A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning. IEEE Access, 9, 75729–75740. https://doi.org/10.1109/ACCESS.2021.3082147

Luthfansa, Z. M., & Rosiani, U. D. (2021). Pemanfaatan Wireshark untuk Sniffing Komunikasi Data Berprotokol HTTP pada Jaringan Internet. Journal of Information Engineering and Educational Technology, 5(1), 34–39. https://doi.org/10.26740/jieet.v5n1.p34-39

Mindara, C. L. and Z. A. and U. H. P. and H. T. and M. G. P. (2023). Deteksi Intrusi Untuk Klasifikasi Serangan Jaringan Dengan Penerapan Algoritma Convolutional Neural Network. Jurnal ICT: Information Communication \& Technology, 23, 517–522.

Muh. Sahal, S. Kom. (2021). Administrasi Infrastruktur Jaringan SMK/MAK Kelas XII. Gramedia Widiasarana indonesia.

Mulawarman dkk. (2020). Problematika Penggunaan Internet.

Partha Majumdar. (2023). Mastering Classification Algorithms for Machine Learning Learn how to Apply Classification Algorithms for Effective Machine Learning Solutions. BPB Publications.

Purba Daru Kusuma. (2020). Machine Learning Teori, Program, Dan Studi Kasus. Deepublish.

Saranya, T., Sridevi, S., Deisy, C., Chung, T. D., & Khan, M. K. A. A. (2020). Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review. Procedia Computer Science, 171, 1251–1260. https://doi.org/10.1016/j.procs.2020.04.133

Savargiv, M., Masoumi, B., & Keyvanpour, M. R. (2021). A New Random Forest Algorithm Based on Learning Automata. Computational Intelligence and Neuroscience, 2021, 1–19. https://doi.org/10.1155/2021/5572781

T.N. Nguyen, J. I. (2022). Advances in Parallel Computing Algorithms, Tools and Paradigms (J. I. T. N. N. D.J. Hemanth, Ed.). IOS Press.

Wang, X., Chen, S., & Su, J. (2020). Real Network Traffic Collection and Deep Learning for Mobile App Identification. Wireless Communications and Mobile Computing, 2020, 1–14. https://doi.org/10.1155/2020/4707909

Wu, T., Fan, H., Zhu, H., You, C., Zhou, H., & Huang, X. (2022). Intrusion detection system combined enhanced random forest with SMOTE algorithm. EURASIP Journal on Advances in Signal Processing, 2022(1), 39. https://doi.org/10.1186/s13634-022-00871-6

DAFTAR RIWAYAT HIDUP

I. Biodata Mahasiswa

NIM	:	15200333
Nama Lengkap	:	Aqshal Farhan Maulana
Tempat/ Tanggal Lahir	:	Jakarta, 27 Januari 2002
Alamat Lengkap	:	Jl.Mawar No.20, RT 003, RW 001, Kelurahan Aren Jaya, Kecamatan Bekasi Timur. Bekasi, Provinsi Jawa Barat, Kode Pos 17111

II. Pendidikan

a. Formal

1. SDIT Al Fatah, lulus tahun 2014
2. SMP Al Muhadjirin Bekasi, lulus tahun 2017
3. SMK Karya Guna 2 Bekasi, lulus tahun 2020
4. S1 Ilmu Komputer Universitas Bina Sarana Informatika (Sekarang)

III. Riwayat Pekerjaan

1. Assembling Operator di PT. Howsanindo Industry, Mei 2018 – Juli 2018
2. IT Support di Bintang Delapan Group, Agt 2023 – Mei 2024

Jakarta, 28 Juni 2024

Aqshal Farhan Maulana

LEMBAR KONSULTASI SKRIPSI

UNIVERSITAS BINA SARANA INFORMATIKA

NIM : 15200333
Nama Lengkap : Aqshal Farhan Maulana
Dosen Pembimbing : Rian Septian Anwar, M.Kom
Judul Skrispi : Komparasi Algoritma Decision Tree dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet

No.	Tanggal Bimbingan	Pokok Bahasan	Paraf Dosen Pembimbing
1.	5 April 2024	ACC Judul	
2.	3 Mei 2024	Pengajuan Bab I	
3.	17 Mei 2024	Revisi Bab I dan Pengajuan Bab II	
4.	31 Mei 2024	Revisi Bab II dan Pengajuan Bab III	
5.	7 Juni 2024	Revisi Bab III dan Pengajuan Bab IV	
6.	14 Juni 2024	Revisi Bab IV dan Pengajuan Bab V	
7.	21 Juni 2024	Pengajuan Power Point	
8.	28 Juni 2024	ACC Keseluruhan	

Catatan untuk Dosen Pembimbing.
Bimbingan Skripsi
Dimulai pada tanggal : 5 April 2024
Diakhiri pada tanggal : 28 Juni 2024
Jumlah pertemuan bimbingan : 8 Kali Bimbingan

Disetujui oleh,
Dosen Pembimbing

(Rian Septian Anwar, M.Kom)

SURAT PERNYATAAN KEBENARAN/KEABSAHAN DATA HASIL RISET UNTUK KARYA ILMIAH

Yang bertanda tangan di bawah ini, saya:

Nama	: Aqshal Farhan Maulana
NIM	: 15200333
Jenjang	: Sarjana (S1)
Program Studi	: Ilmu Komputer
Fakultas	: Teknik dan Informatika
Perguruan Tinggi	: Universitas Bina Sarana Informatika

Dengan ini menyatakan bahwa data dan atau informasi yang saya gunakan dalam penulisan karya ilmiah Penulis dengan judul "**Komparasi Algoritma Decision Tree dan Random Forest Dalam Mendeteksi Intrusi Jaringan Internet**" merupakan data dan atau informasi yang saya peroleh melalui hasil penelitian sendiri dan tidak didasarkan pada data atau informasi hasil riset dari perusahaan/instansi/lembaga manapun.

Saya bersedia untuk bertanggung jawab secara pribadi, tanpa melibatkan pihak **Universitas Bina Sarana Informatika**, atas materi/isi karya ilmiah tersebut, termasuk bertanggung jawab atas dampak atau kerugian yang timbul dalam bentuk akibat tindakan yang berkaitan dengan data dan atau informasi yang terdapat pada karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Bekasi

Mengetahui,
Dosen Pembimbing

Rian Septian Anwar, M.Kom.

Yang menyatakan,

Aqshal Farhan Maulana

LAMPIRAN

Lampiran 1 Memanggil Library

```
import os
import glob
import pandas as pd
import numpy as np

columns = (['duration'
,'protocol_type'
,'service'
,'flag'
,'src_bytes'
,'dst_bytes'
,'land'
,'wrong_fragment'
,'urgent'
,'hot'
,'num_failed_logins'
,'logged_in'
,'num_compromised'
,'root_shell'
,'su_attempted'
,'num_root'
,'num_file_creations'
,'num_shells'
,'num_access_files'
,'num_outbound_cmds'
,'is_host_login'
,'is_guest_login'
,'count'
,'srv_count'
,'serror_rate'
,'srv_serror_rate'
,'rerror_rate'
,'srv_rerror_rate'
,'same_srv_rate'
,'diff_srv_rate'
,'srv_diff_host_rate'
,'dst_host_count'
,'dst_host_srv_count'
,'dst_host_same_srv_rate'
,'dst_host_diff_srv_rate'
,'dst_host_same_src_port_rate'
,'dst_host_srv_diff_host_rate'
,'dst_host_serror_rate'
,'dst_host_srv_serror_rate'
,'dst_host_rerror_rate'
,'dst_host_srv_rerror_rate'
,'attack'
,'level'])
```

Lampiran 2 Membaca dan Menyatukan Dataset

```
df1 = pd.read_csv('/kaggle/input/kdd-dataset/KDD_dataset.txt',header=None,names=columns)
df2 = pd.read_csv('/kaggle/input/dataset-skripsi/2022.06.13.csv')
df3 = pd.read_csv('/kaggle/input/dataset-skripsi/2022.06.14.csv')
```

```
df = pd.concat([df1], ignore_index=True)
len(df.columns)
```

```
43
```

```
df
```

dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_srv_diff_host_rate	dst_host_serror_rate	dst_host_srv_serror_rate	dst_host_
0.03	0.17	0.00	0.00	0.00	0.05
0.60	0.88	0.00	0.00	0.00	0.00
0.05	0.00	0.00	1.00	1.00	0.00
0.00	0.03	0.04	0.03	0.01	0.00
0.00	0.00	0.00	0.00	0.00	0.00
...	...	...	...	...	...
0.06	0.00	0.00	1.00	1.00	0.00
0.01	0.01	0.00	0.00	0.00	0.00
0.06	0.00	0.00	0.72	0.00	0.01
0.05	0.00	0.00	1.00	1.00	0.00
0.03	0.30	0.00	0.00	0.00	0.00

125973 rows × 43 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125973 entries, 0 to 125972
Data columns (total 43 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   duration                     125973 non-null  int64
 1   protocol_type                125973 non-null  object
 2   service                      125973 non-null  object
 3   flag                         125973 non-null  object
 4   src_bytes                    125973 non-null  int64
 5   dst_bytes                    125973 non-null  int64
 6   land                         125973 non-null  int64
 7   wrong_fragment               125973 non-null  int64
 8   urgent                       125973 non-null  int64
 9   hot                          125973 non-null  int64
 10  num_failed_logins            125973 non-null  int64
```

Lampiran 3 Membaca dan Menyatukan Dataset II

```
 11  logged_in                    125973 non-null  int64
 12  num_compromised              125973 non-null  int64
 13  root_shell                   125973 non-null  int64
 14  su_attempted                 125973 non-null  int64
 15  num_root                     125973 non-null  int64
 16  num_file_creations           125973 non-null  int64
 17  num_shells                   125973 non-null  int64
 18  num_access_files             125973 non-null  int64
 19  num_outbound_cmds            125973 non-null  int64
 20  is_host_login                125973 non-null  int64
 21  is_guest_login               125973 non-null  int64
 22  count                        125973 non-null  int64
 23  srv_count                    125973 non-null  int64
 24  serror_rate                  125973 non-null  float64
 25  srv_serror_rate              125973 non-null  float64
 26  rerror_rate                  125973 non-null  float64
 27  srv_rerror_rate              125973 non-null  float64
 28  same_srv_rate                125973 non-null  float64
 29  diff_srv_rate                125973 non-null  float64
 30  srv_diff_host_rate           125973 non-null  float64
 31  dst_host_count               125973 non-null  int64
 32  dst_host_srv_count           125973 non-null  int64
 33  dst_host_same_srv_rate       125973 non-null  float64
 34  dst_host_diff_srv_rate       125973 non-null  float64
 35  dst_host_same_src_port_rate  125973 non-null  float64
 36  dst_host_srv_diff_host_rate  125973 non-null  float64
 37  dst_host_serror_rate         125973 non-null  float64
 38  dst_host_srv_serror_rate     125973 non-null  float64
 39  dst_host_rerror_rate         125973 non-null  float64
 40  dst_host_srv_rerror_rate     125973 non-null  float64
 41  attack                       125973 non-null  object
 42  level                        125973 non-null  int64
dtypes: float64(15), int64(24), object(4)
memory usage: 41.3+ MB
```

```
df.describe()
```

:

	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_log
count	125973.00000	1.259730e+05	1.259730e+05	125973.000000	125973.000000	125973.000000	125973.000000	125973.000000
mean	287.14465	4.556674e+04	1.977911e+04	0.000198	0.022687	0.000111	0.204409	0.001222
std	2604.51531	5.870331e+06	4.021269e+06	0.014086	0.253530	0.014366	2.149968	0.045239
min	0.00000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.00000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.00000	4.400000e+01	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.00000	2.760000e+02	5.160000e+02	0.000000	0.000000	0.000000	0.000000	0.000000
max	42908.00000	1.379964e+09	1.309937e+09	1.000000	3.000000	3.000000	77.000000	5.000000

8 rows × 39 columns

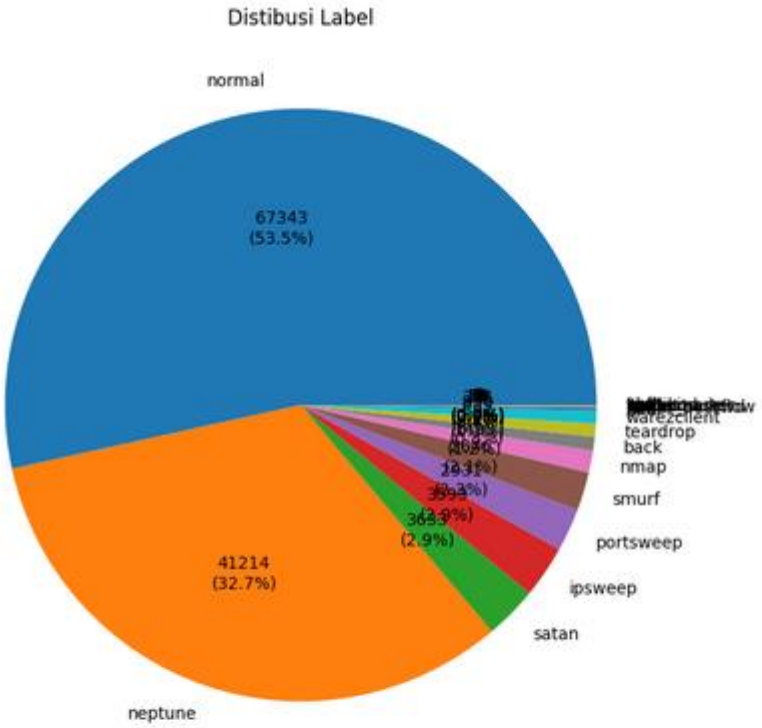
Lampiran 4 Membaca dan Menyatukan Dataset III

```
df['attack'].value_counts()
```

```
attack
normal             67343
neptune            41214
satan               3633
ipsweep             3599
portsweep           2931
smurf               2646
nmap                1493
back                 956
teardrop             892
warezclient          890
pod                  201
guess_passwd          53
buffer_overflow       30
warezmaster           20
land                  18
imap                  11
rootkit               10
loadmodule             9
ftp_write              8
multihop               7
phf                    4
perl                   3
spy                    2
Name: count, dtype: int64
```

```
import seaborn as sns
import matplotlib.pyplot as plt

# Count the occurrences of each label
label_counts = df['attack'].value_counts()

# Plot using Matplotlib
plt.figure(figsize=(8, 8))
plt.pie(label_counts, labels=label_counts.index, autopct=lambda p: '{:.0f}\n({:.1f}%)'.format(p *
sum(label_counts) / 100, p))

# Show the plot
plt.title('Distibusi Label')
plt.show()
```

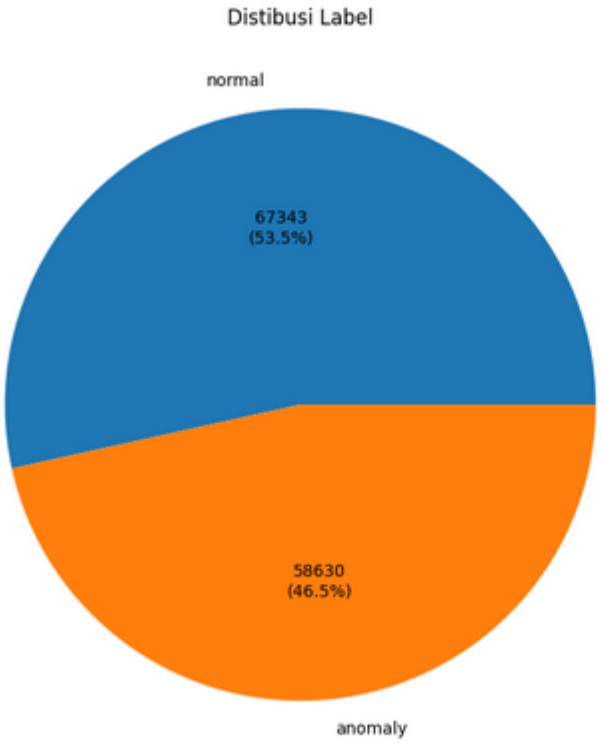
Lampiran 5 Explarotory Data Analysis



Lampiran 6 Explarotory Data Analysis

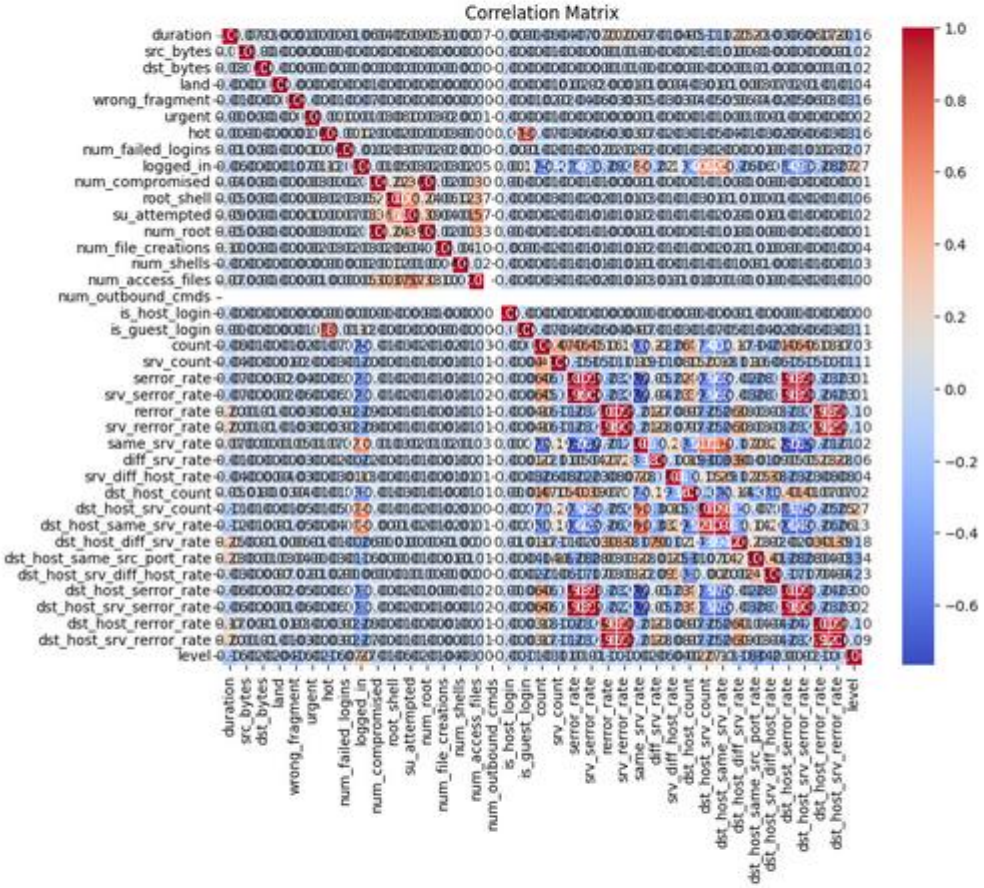
```
df['attack'] = df['attack'].replace(['neptune','satan','ipsweep','portsweep','smurf', 'nmap','bac
k','teardrop','warezclient','pod','guess_passwd','buffer_overflow','warezmaster','land','imap','ro
otkit','loadmodule','ftp_write','multihop','phf','perl','spy'], 'anomaly')
```

```
import seaborn as sns
import matplotlib.pyplot as plt

# Count the occurrences of each label
label_counts = df['attack'].value_counts()

# Plot using Matplotlib
plt.figure(figsize=(8, 8))
plt.pie(label_counts, labels=label_counts.index, autopct=lambda p: '{:.0f}\n({:.1f}%)'.format(p *
sum(label_counts) / 100, p))

# Show the plot
plt.title('Distibusi Label')
plt.show()
```



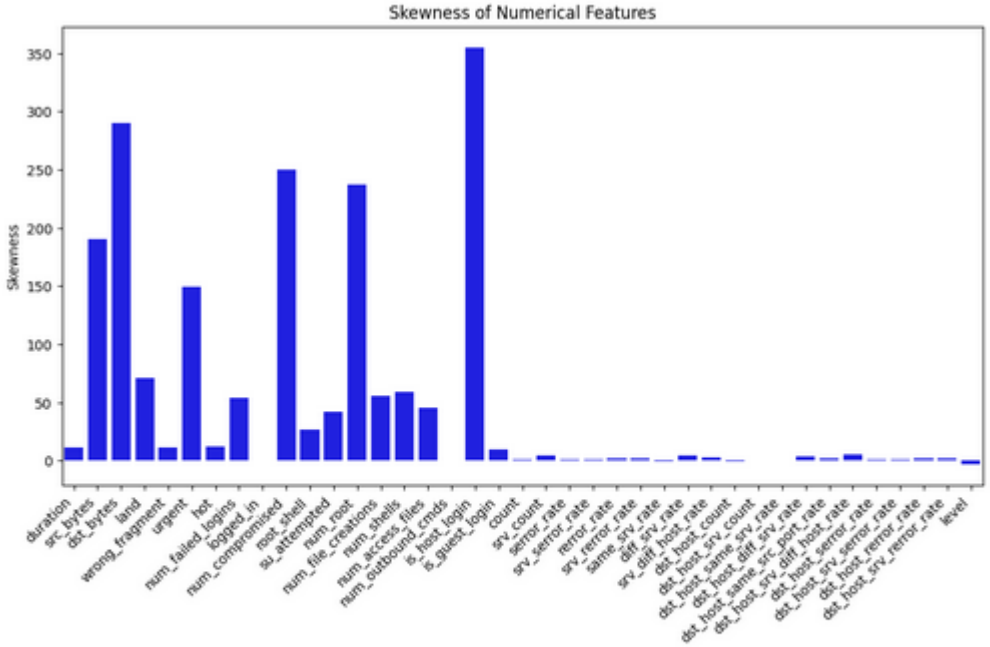
Lampiran 7 Explarotory Data Analysis II

```
df_dropped = df.drop(columns=['attack', 'protocol_type', 'service', 'flag'])

# Create correlation matrix
correlation_matrix = df_dropped.corr()

# Create a heatmap using Seaborn
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```




```
# Calculate skewness for each numerical column
skewness = df.drop(columns=['attack', 'protocol_type', 'service', 'flag']).skew()

# Create a bar plot or a histogram of skewness values
plt.figure(figsize=(12, 6))
sns.barplot(x=skewness.index, y=skewness.values, color='blue')
plt.xticks(rotation=45, ha='right')
plt.title('Skewness of Numerical Features')
plt.ylabel('Skewness')
plt.show()
```

Lampiran 8 Explarotory Data Analysis III




```
missing_values = df.isnull().sum()

print(missing_values)
```

```
duration                     0
protocol_type                0
service                      0
flag                         0
src_bytes                    0
dst_bytes                    0
land                         0
wrong_fragment               0
urgent                       0
hot                          0
num_failed_logins            0
logged_in                    0
num_compromised              0
root_shell                   0
su_attempted                 0
num_root                     0
num_file_creations           0
num_shells                   0
num_access_files             0
num_outbound_cmds            0
is_host_login                0
is_guest_login               0
count                        0
srv_count                    0
```

Lampiran 9 Explarotory Data Analysis IV

```
serror_rate                  0
srv_serror_rate              0
rerror_rate                  0
srv_rerror_rate              0
same_srv_rate                0
diff_srv_rate                0
srv_diff_host_rate           0
dst_host_count               0
dst_host_srv_count           0
dst_host_same_srv_rate       0
dst_host_diff_srv_rate       0
dst_host_same_src_port_rate  0
dst_host_srv_diff_host_rate  0
dst_host_serror_rate         0
dst_host_srv_serror_rate     0
dst_host_rerror_rate         0
dst_host_srv_rerror_rate     0
attack                       0
level                        0
dtype: int64
```

Lampiran 10 Preprocessing Dataset

```
df.select_dtypes(['object']).columns
```

```
Index(['protocol_type', 'service', 'flag', 'attack'], dtype='object')
```

```
# Label Encoder
from sklearn import preprocessing
le=preprocessing.LabelEncoder()
clm=['protocol_type', 'service', 'flag', 'attack']
for x in clm:
    df[x]=le.fit_transform(df[x])
```

```
df.select_dtypes(['object']).columns
```

```
Index([], dtype='object')
```

```
df
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_same_srv_ra
0	0	1	20	9	491	0	0	0	0	0	...	0.17
1	0	2	44	9	146	0	0	0	0	0	...	0.00
2	0	1	49	5	0	0	0	0	0	0	...	0.10
3	0	1	24	9	232	8153	0	0	0	0	...	1.00
4	0	1	24	9	199	420	0	0	0	0	...	1.00
...	...	...	...	...	...	...	...	...	...	...	...	...
125968	0	1	49	5	0	0	0	0	0	0	...	0.10
125969	8	2	49	9	105	145	0	0	0	0	...	0.96
125970	0	1	54	9	2231	384	0	0	0	0	...	0.12
125971	0	1	30	5	0	0	0	0	0	0	...	0.03
125972	0	1	20	9	151	0	0	0	0	0	...	0.30

125973 rows × 43 columns

```
from sklearn.model_selection import train_test_split
# from sklearn.preprocessing import StandardScaler, MinMaxScaler

# Separate features (X) and target variable (y)
X = df.drop('attack', axis=1)
y = df['attack']

# numerical_features = X.select_dtypes(include=['float64', 'int64']).columns

# Log-transform numerical columns
# X[numerical_features] = np.log1p(X[numerical_features])

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=17)
```

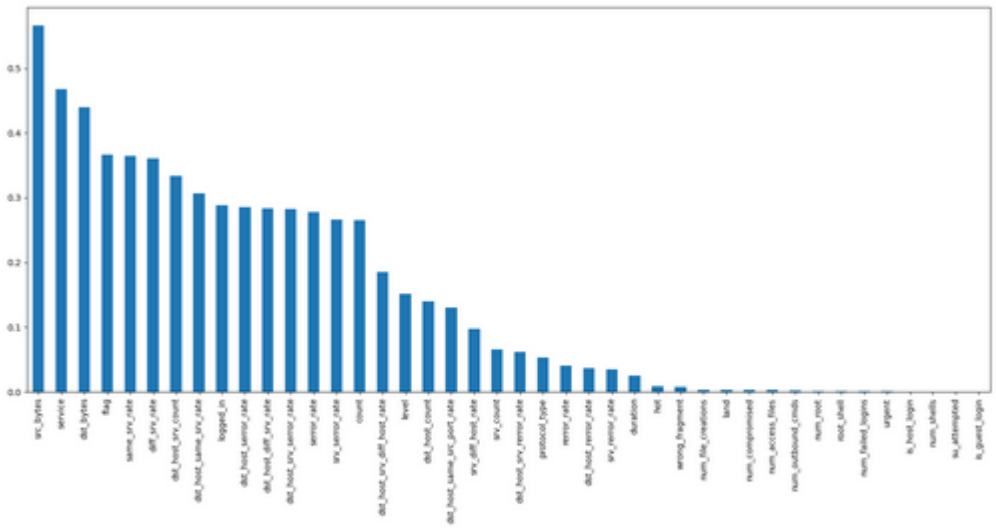
Lampiran 11 Preprocessing Dataset II

```
from sklearn.feature_selection import mutual_info_classif
mutual_info = mutual_info_classif(X_train, y_train)
mutual_info = pd.Series(mutual_info)
mutual_info.index = X_train.columns
mutual_info.sort_values(ascending=False)
```

```
src_bytes                    0.565895
service                      0.467651
dst_bytes                    0.439467
flag                         0.366597
same_srv_rate                0.364998
diff_srv_rate                0.360907
dst_host_srv_count           0.334046
dst_host_same_srv_rate       0.306693
logged_in                    0.287922
dst_host_serror_rate         0.285740
dst_host_diff_srv_rate       0.283622
dst_host_srv_serror_rate     0.282592
serror_rate                  0.277501
srv_serror_rate              0.266391
count                        0.265377
dst_host_srv_diff_host_rate  0.185037
level                        0.151829
dst_host_count               0.139611
dst_host_same_src_port_rate  0.129922
srv_diff_host_rate           0.097037
srv_count                    0.065349
dst_host_srv_rerror_rate     0.061540
protocol_type                0.053313
rerror_rate                  0.040334
dst_host_rerror_rate         0.036776
srv_rerror_rate              0.034888
duration                     0.025107
hot                          0.008954
wrong_fragment               0.008086
num_file_creations           0.002912
land                         0.002887
num_compromised              0.002462
num_access_files             0.002433
num_outbound_cmds            0.001846
num_root                     0.001247
root_shell                   0.000822
num_failed_logins            0.000701
urgent                       0.000441
is_host_login                0.000000
num_shells                   0.000000
su_attempted                 0.000000
is_guest_login               0.000000
dtype: float64
```

Lampiran 12 Preprocessing Dataset III

```
mutual_info.sort_values(ascending=False).plot.bar(figsize=(20, 8));
```



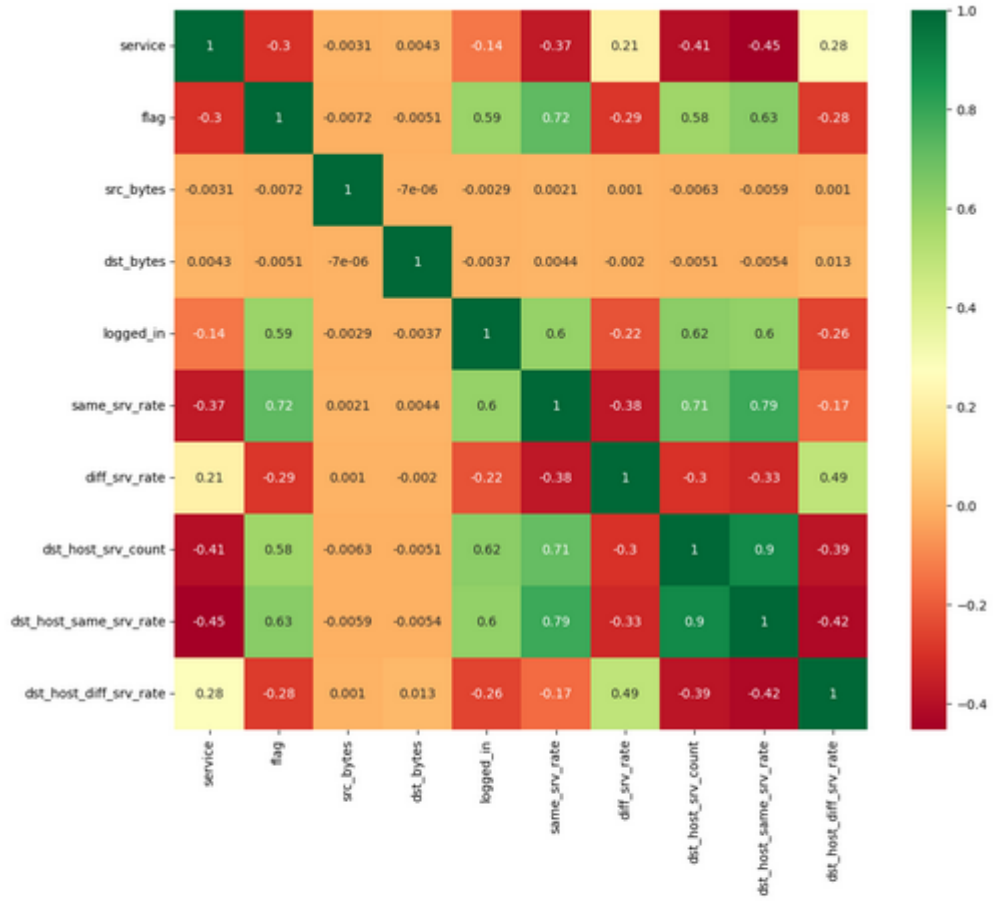
```
# memilih 25 fitur yang berpengaruh terhadap fitur label
from sklearn.feature_selection import SelectKBest
sel_five_cols = SelectKBest(mutual_info_classif, k=25)
sel_five_cols.fit(X_train, y_train)
X_train.columns[sel_five_cols.get_support()]
```

```
Index(['protocol_type', 'service', 'flag', 'src_bytes', 'dst_bytes',
       'logged_in', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate',
       'rerror_rate', 'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate',
       'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',
       'dst_host_same_srv_rate', 'dst_host_diff_srv_rate',
       'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate',
       'dst_host_serror_rate', 'dst_host_srv_serror_rate',
       'dst_host_srv_rerror_rate', 'level'],
      dtype='object')
```

```
col=['service', 'flag', 'src_bytes', 'dst_bytes', 'logged_in',
       'same_srv_rate', 'diff_srv_rate', 'dst_host_srv_count',
       'dst_host_same_srv_rate', 'dst_host_diff_srv_rate']
X_train=X_train[col]
X_test=X_test[col]
```

```
plt.figure(figsize=(12,10))
p=sns.heatmap(X_train.corr(), annot=True,cmap ='RdYlGn')
```

Lampiran 13 Preprocessing Dataset IV




```
# Scalling
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train= scaler.fit_transform(X_train)
X_test= scaler.fit_transform(X_test)
```

Lampiran 14 Preprocessing Dataset V

```
X_train
```

```
array([[[7.10144928e-01, 9.00000000e-01, 7.24656644e-10, ...,
         3.37254902e-01, 3.40000000e-01, 9.00000000e-02],
        [5.21739130e-01, 5.00000000e-01, 0.00000000e+00, ...,
         3.52941176e-02, 4.00000000e-02, 7.00000000e-02],
        [2.75362319e-01, 9.00000000e-01, 2.44933946e-07, ...,
         1.72549020e-01, 1.80000000e-01, 3.00000000e-02],
        ...,
        [3.47826087e-01, 9.00000000e-01, 2.23194246e-07, ...,
         1.00000000e+00, 1.00000000e+00, 0.00000000e+00],
        [4.05797101e-01, 5.00000000e-01, 0.00000000e+00, ...,
         1.56862745e-02, 2.00000000e-02, 7.00000000e-02],
        [8.40579710e-01, 5.00000000e-01, 0.00000000e+00, ...,
         3.13725490e-02, 3.00000000e-02, 7.00000000e-02]]])
```

```
y_train
```

```
95044     0
37827     0
5019      1
2986      1
6741      1
         ..
98710     0
112006    1
34959     1
64753     0
76399     0
Name: attack, Length: 88181, dtype: int64
```

Lampiran 15 Model Decision Tree dan Random Forest

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
```

```

```

```
dt_model = DecisionTreeClassifier(max_depth=3)
dt_model.get_params()
```

```
{'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': 3,
 'max_features': None,
 'max_leaf_nodes': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'random_state': None,
 'splitter': 'best'}
```

```
dt_model.fit(X_train, y_train)
```

```
DecisionTreeClassifier(max_depth=3)
```

```
rf_model = RandomForestClassifier()
rf_model.get_params()
```

```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': None,
 'max_features': 'sqrt',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

Lampiran 16 Model Decision Tree dan Random Forest II

```
rf_model.fit(X_train, y_train)
```

```
RandomForestClassifier()
```

Lampiran 17 Hasil

```
from sklearn.metrics import classification_report
```

```
dt_pred = dt_model.predict(X_test)
report = classification_report(y_test, dt_pred)
print(f'Decision Tree Report:\n {report}')
```

```
Decision Tree Report:
              precision    recall  f1-score   support

           0       0.88      0.98      0.93     17656
           1       0.98      0.88      0.93     20136

    accuracy                           0.93     37792
   macro avg       0.93      0.93      0.93     37792
weighted avg       0.93      0.93      0.93     37792
```

```
rf_pred = rf_model.predict(X_test)
report = classification_report(y_test, rf_pred)
print(f'Random Forest Report:\n {report}')
```

```
Random Forest Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     17656
           1       0.99      1.00      0.99     20136

    accuracy                           0.99     37792
   macro avg       0.99      0.99      0.99     37792
weighted avg       0.99      0.99      0.99     37792
```

Lampiran 18 Hasil II

```
from sklearn import metrics
print("Decision Tree")
print("Accuracy :",metrics.accuracy_score(y_test, dt_pred))
print("Precision:",metrics.precision_score(y_test, dt_pred))
print("Recall   :",metrics.recall_score(y_test, dt_pred))
print("F1-Score :",metrics.f1_score(y_test, dt_pred))
print(" ")
print("Random Forest")
print("Accuracy :",metrics.accuracy_score(y_test, rf_pred))
print("Precision:",metrics.precision_score(y_test, rf_pred))
print("Recall   :",metrics.recall_score(y_test, rf_pred))
print("F1-Score :",metrics.f1_score(y_test, rf_pred))
```

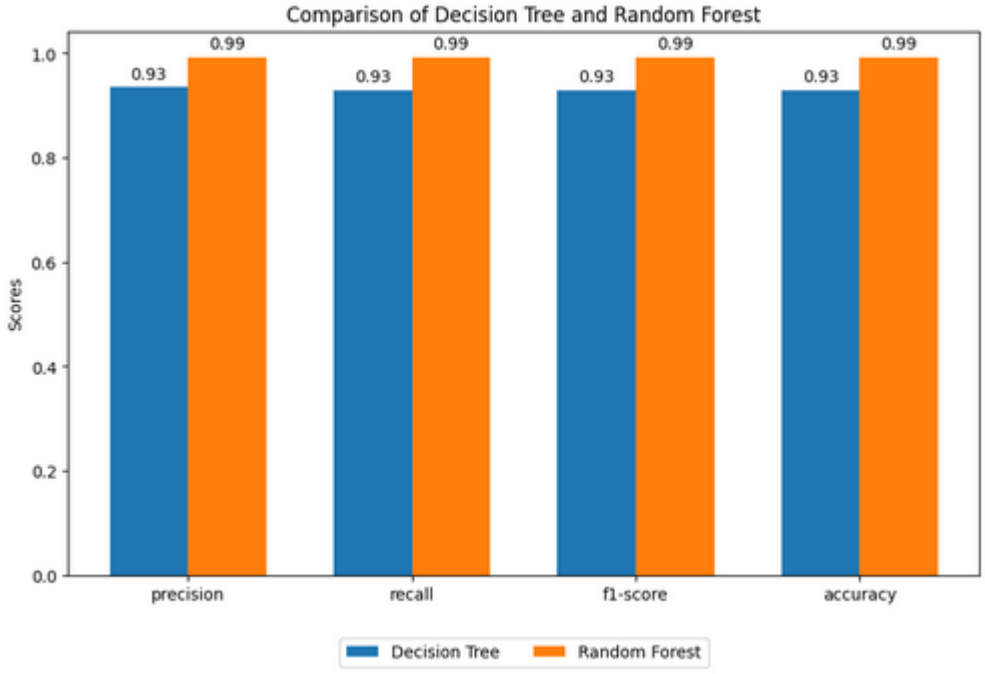
```
Decision Tree
Accuracy : 0.9293501270110076
Precision: 0.9809450379997797
Recall   : 0.8845848232022249
F1-Score : 0.9302762834908863

Random Forest
Accuracy : 0.9925910245554614
Precision: 0.9906107926467681
Recall   : 0.9955303933253874
F1-Score : 0.9930645001486179
```

Lampiran 19 Hasil III

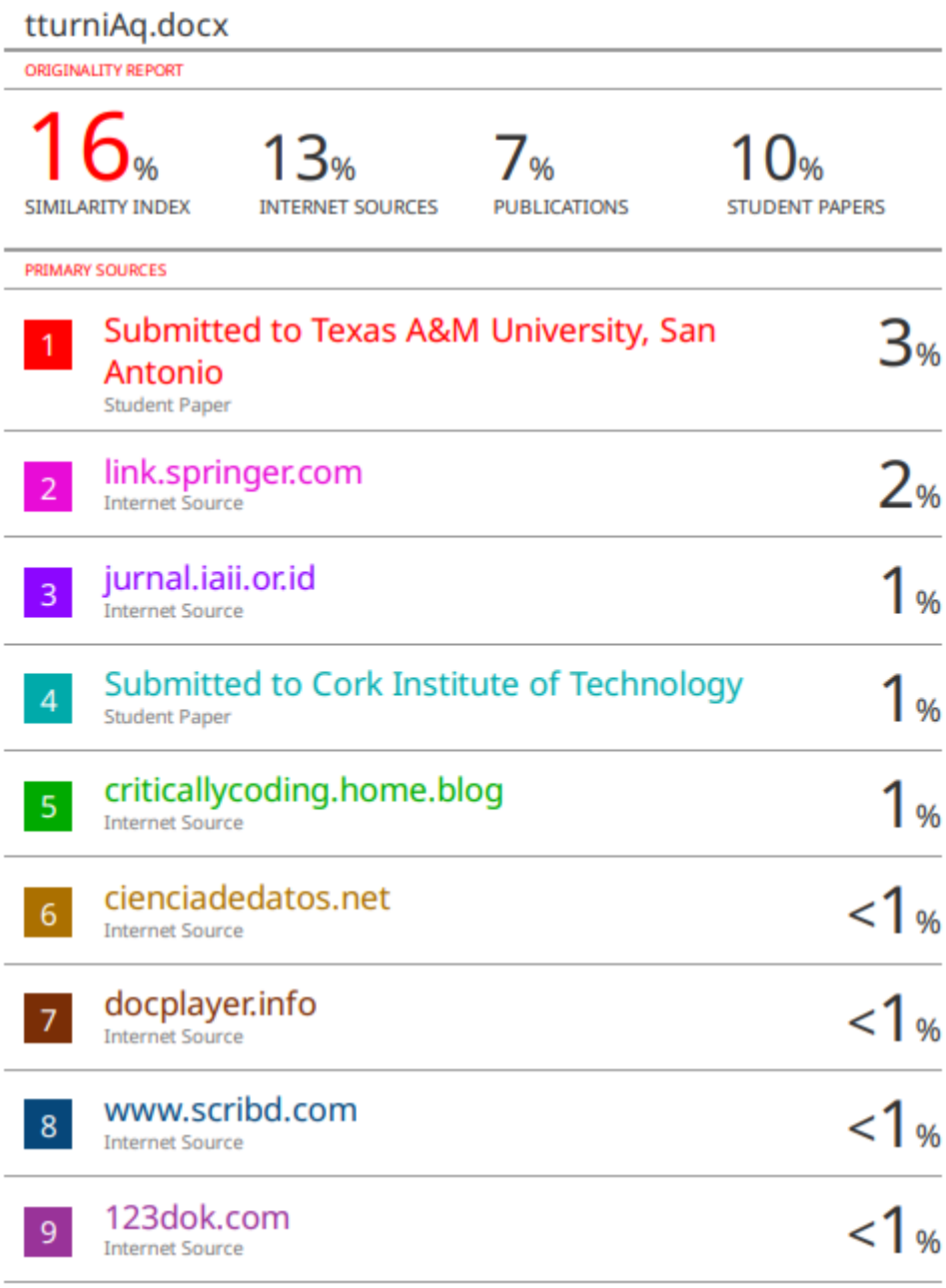
```
from sklearn.metrics import classification_report, precision_score, recall_score, f1_score, accura
cy_score
import matplotlib.pyplot as plt
import numpy as np

# Predictions and reports
dt_pred = dt_model.predict(X_test)
rf_pred = rf_model.predict(X_test)

dt_report = classification_report(y_test, dt_pred, output_dict=True)
rf_report = classification_report(y_test, rf_pred, output_dict=True)

# Extract metrics
metrics = ['precision', 'recall', 'f1-score', 'accuracy']
dt_metrics = {
    'precision': precision_score(y_test, dt_pred, average='weighted'),
    'recall': recall_score(y_test, dt_pred, average='weighted'),
    'f1-score': f1_score(y_test, dt_pred, average='weighted'),
    'accuracy': accuracy_score(y_test, dt_pred)
}

rf_metrics = {
    'precision': precision_score(y_test, rf_pred, average='weighted'),
    'recall': recall_score(y_test, rf_pred, average='weighted'),
    'f1-score': f1_score(y_test, rf_pred, average='weighted'),
    'accuracy': accuracy_score(y_test, rf_pred)
}

# Prepare data for plotting
labels = metrics
dt_values = [dt_metrics[m] for m in metrics]
rf_values = [rf_metrics[m] for m in metrics]

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

# Plotting
fig, ax = plt.subplots(figsize=(10, 6))
bars1 = ax.bar(x - width/2, dt_values, width, label='Decision Tree')
bars2 = ax.bar(x + width/2, rf_values, width, label='Random Forest')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores')
ax.set_title('Comparison of Decision Tree and Random Forest')
ax.set_xticks(x)
ax.set_xticklabels(labels)
```

Lampiran 20 Hasil IV

```
# Function to add labels on bars
def add_labels(bars):
    for bar in bars:
        height = bar.get_height()
        ax.annotate(f'{height:.2f}',
                    xy=(bar.get_x() + bar.get_width() / 2, height),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom')

add_labels(bars1)
add_labels(bars2)

# Adjust legend position to be below the plot
ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.1), ncol=2)

plt.show()
```



Lampiran 21 Hasil Turnitin



tturniAq.docx

ORIGINALITY REPORT

16%	13%	7%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Texas A&M University, San Antonio Student Paper	3%
2	link.springer.com Internet Source	2%
3	jurnal.iaii.or.id Internet Source	1%
4	Submitted to Cork Institute of Technology Student Paper	1%
5	criticallycoding.home.blog Internet Source	1%
6	cienciadedatos.net Internet Source	<1%
7	docplayer.info Internet Source	<1%
8	www.scribd.com Internet Source	<1%
9	123dok.com Internet Source	<1%