BAB 3
METODOLOGI PENELITIAN

3.1 Gambaran Umum Penelitian

Tahap pertama dalam penelitian ini adalah pengumpulan dataset parameter jaringan komputer dari sumber Kaggle (”Network Anomaly Dataset”). Label klasifikasi ditentukan berdasarkan kondisi kemacetan, di mana 1 menunjukkan ”macet” dan 0 ”tidak macet”. Data yang terkumpul kemudian diolah menjadi struktur tabel dengan fitur-fitur kunci seperti *latency*, *throughput*, *jitter*, dan *packet loss*. Setelah data terkumpul, dilakukan pra-pemrosesan data yang meliputi:

1. Normalisasi numerik menggunakan *MinMaxScaler* untuk menyamakan skala fitur.
2. Rekayasa fitur tambahan (misal: *latency_per_connection, loss_to_throughput_ratio*) untuk menangkap konteks jaringan secara lebih mendalam.
3. Pembagian *dataset* dengan dua skema:
   (a) Skema A: 60% training, 20% validation, 20% testing.
   (b) Skema B: 80% training, 20% testing.

Tahap inti penelitian adalah pelatihan model *random forest* dengan optimasi *hyperparameter* menggunakan *GridSearchCV*. Parameter yang diuji mencakup *n_estimators*, *max_depth*, dan *min_samples_split*. Model kemudian dikalibrasi dengan *CalibratedClassifierCV* untuk meningkatkan akurasi probabilitas. Proses akhir penelitian ini dilakukan dengan mengevaluasi model menggunakan data *testing* dengan metrik:

1. Akurasi, *F1-score*, *Recall*, dan *Precision* untuk mengukur kinerja klasifikasi.
2. *Threshold tuning* (optimalisasi ambang batas klasifikasi) berdasarkan kurva ROC dan *Precision-Recall*.

3.2 Spesifikasi Perangkat yang Digunakan

Dalam melakukan penelitian kali ini, perangkat yang digunakan adalah laptop pribadi dengan spesifikasi sebagai berikut:

1. *Operating System*: Windows 10
2. *Memory*: 16GB (8+8)
3. *Storage*: 512GB
4. *Processor*: AMD Ryzen 7 5800H *with Radeon Graphics*
5. *Graphic Card*: NVDIA GeForce RTX 3050Ti

3.3 Telaah Literatur

Pada tahap ini, akan dilakukan kajian lebih lanjut terhadap berbagai jurnal ilmiah yang membahas mengenai klasifikasi kemacetan jaringan atau deteksi kemacetan jaringan menggunakan model *machine learning*. Studi literatur ini bertujuan untuk memahami landasan teoritis dan penelitian terkait topik yang diambil. Pencarian literatur ini didasarkan pada situs-situs *website* dan jurnal-jurnal publikasi dari kampus, atau perorangan terkait dengan topik yang dibahas pada penelitian ini. Pada telaah literatur juga akan dilakukan pengidentifikasian metrik jaringan yang berpengaruh dalam kemacetan jaringan seperti *throughput*, *packet loss*, *latency*, *jitter*, dan jumlah koneksi aktif.

3.4 Pengumpulan dan Pengolahan Data

Dataset yang digunakan berasal dari Kaggle berjudul "*Network Anomaly Dataset*". *Dataset* ini menyediakan metrik performa jaringan seperti *throughput*, *latency*, *jitter*, *packet loss*, dan *bandwidth usage*.

3.4.1 Pembuatan Label

Label 'status_macet' yang digunakan dalam penelitian ini sudah tersedia dari sumber *dataset* pada Kaggle, dengan modifikasi kecil pada penamaan kolom untuk menyesuaikan konteks lokal, yaitu menjadi 'macet' (1) dan 'tidak macet' (0). Jumlah *dataset* yang digunakan adalah 1002 baris.

Pada penelitian ini juga menggunakan standar *Quality of Service* (QoS) seperti ITU-T G.114 dan Cisco QoS yang akan dijadikan acuan untuk memahami tingkat keparahan suatu jaringan (QoS level), namun tidak digunakan sebagai dasar utama dalam pelabelan. Hal ini dikarenakan kemacetan jaringan tidak dapat ditentukan hanya dari satu metrik seperti *jitter* atau *latency* saja, melainkan kondisi kompleks yang melibatkan interaksi berbagai parameter jaringan.

3.4.2 Ringkasan Dataset Akhir

Setelah dilakukan tahapan *preprocessing* seperti pembersihan data, rekayasa fitur, normalisasi menggunakan *MinMaxScaler*, dan pelabelan berdasarkan aturan kualitas layanan (*Quality of Service*/QoS), diperoleh *dataset* akhir yang siap digunakan untuk proses pelatihan dan pengujian model.

Dataset akhir terdiri dari 1001 entri data jaringan, masing-masing dengan sejumlah fitur numerik yang merepresentasikan kondisi jaringan seperti *latency*, *jitter*, *packet loss*, dan *throughput*, serta beberapa fitur hasil rekayasa (*feature engineering*).

Label klasifikasi dibagi menjadi dua kelas:

1. Label 0 – Tidak Macet: 582 data (58.1%)
2. Label 1 – Macet: 419 data (41.9%)

Distribusi label tergolong cukup seimbang, sehingga tidak diperlukan teknik penanganan ketidakseimbangan data. Hal ini penting agar model tidak mengalami bias terhadap salah satu kelas.

Tabel 3.1. Ringkasan Dataset Akhir Setelah Preprocessing

Atribut	**Nilai**
Jumlah data	1001
Jumlah fitur total	12
Label “Tidak Macet” (0)	582 (58.1%)
Label “Macet” (1)	419 (41.9%)
Skala data	0 – 1 (Normalisasi)

3.4.3 Pengolahan Data

Penelitian ini melakukan pengolahan data salah satunya dengan cara menyesuaikan kolom pada *dataset* yang digunakan, yang bertujuan agar kolom pada *dataset* sesuai dengan model yang dibangun. Pada proses ini juga akan dilakukan *drop* kolom yang tidak digunakan seperti *time stamp*, setelah itu akan dilakukan beberapa tahap pengolahan data lagi seperti normalisasi, pembuatan *feature engineering*, *feature selection*, dan analisis korelasi.

A Normalisasi Data

Seluruh fitur numerik dalam *dataset* dinormalisasi menggunakan *MinMaxScaler* untuk mengubah skala menjadi rentang [0, 1]. Meskipun *random forest* tidak sensitif terhadap skala, normalisasi tetap digunakan untuk konsistensi dan keperluan evaluasi model lain di luar lingkup penelitian ini. Dalam penelitian ini, digunakan *MinMaxScaler* atau teknik normalisasi dibandingkan dengan standarisasi karena:

1. Fitur-fitur jaringan memiliki skala yang sangat berbeda,
2. Distribusi fitur cenderung tidak normal (*skewed*),
3. Adanya potensi *outlier* ekstrem pada fitur *latency* dan *packet loss*,
4. *MinMaxScaler* mempertahankan bentuk distribusi asli data dan tidak terpengaruh oleh asumsi distribusi,
5. Berdasarkan studi empiris [44], pemilihan teknik *scaling* yang tepat dapat berdampak besar terhadap performa model.

Pada data jaringan yang mengandung fitur dengan skala berbeda (misal: *latency* dalam milidetik dan *throughput* dalam Gbps), normalisasi mencegah fitur besar mendominasi perhitungan jarak. Alternatif seperti *StandardScaler* (standardisasi Z-*score*) kurang cocok untuk data *sparse* atau *outlier* ekstrem. Maka dari itu, normalisasi dipilih karena teknik ini tidak mengubah bentuk distribusi data, sehingga cocok digunakan ketika tidak ada asumsi distribusi normal [45]. Normalisasi sangat direkomendasikan jika:

1. *Dataset* memiliki skala yang sangat berbeda antar fitur (contoh: *latency* dalam ms vs *throughput* dalam Gbps),

2. Distribusi data tidak normal (*non-Gaussian*),

3. Model yang digunakan sensitif terhadap skala seperti KNN, SVM, *Logistic Regression*, dan ANN.

Pada penelitian ini *dataset* khususnya pada kolom *latency* sebagian besar *latency* berada di kisaran 10–40 ms, tapi ada beberapa nilai ekstrem seperti 300 ms. Fitur *packet loss* umumnya 0–1%, tetapi pada *dataset* ditemukan nilai 30–50% karena *error* jaringan ekstrem, hal ini membuat distribusinya menjadi *right-skewed*.

Distribusi seperti hal di atas tidak cocok untuk standarisasi karena rata-rata dan standar deviasi akan "ditarik" ke nilai ekstrem, menyebabkan mayoritas nilai menjadi terlalu kecil atau terlalu besar secara proporsi. Sehingga Normalisasi yang dipilih dibandingkan dengan standarisasi.

B Feature Engineering

Rekayasa fitur dilakukan untuk meningkatkan kemampuan model dalam memahami struktur data. Empat fitur baru yang ditambahkan adalah:

1. `loss_to_throughput_ratio` adalah rasio antara *packet loss* dan *throughput*, yang menunjukkan tingkat kehilangan data relatif terhadap kapasitas transmisi.

2. `latency_per_connection` adalah *latency* dibagi dengan jumlah koneksi aktif, hal ini bertujuan untuk menggambarkan *delay* per pengguna.

3. `jitter_variability` adalah *jitter* dibagi *latency*, bertujuan untuk menunjukkan ketidakstabilan relatif.

4. `bandwidth_per_connection` adalah *bandwidth usage* dibagi jumlah koneksi aktif, bertujuan untuk mengindikasikan beban *bandwidth* per koneksi.

Fitur-fitur ini dirancang berdasarkan pemahaman domain jaringan dan terbukti meningkatkan performa model.

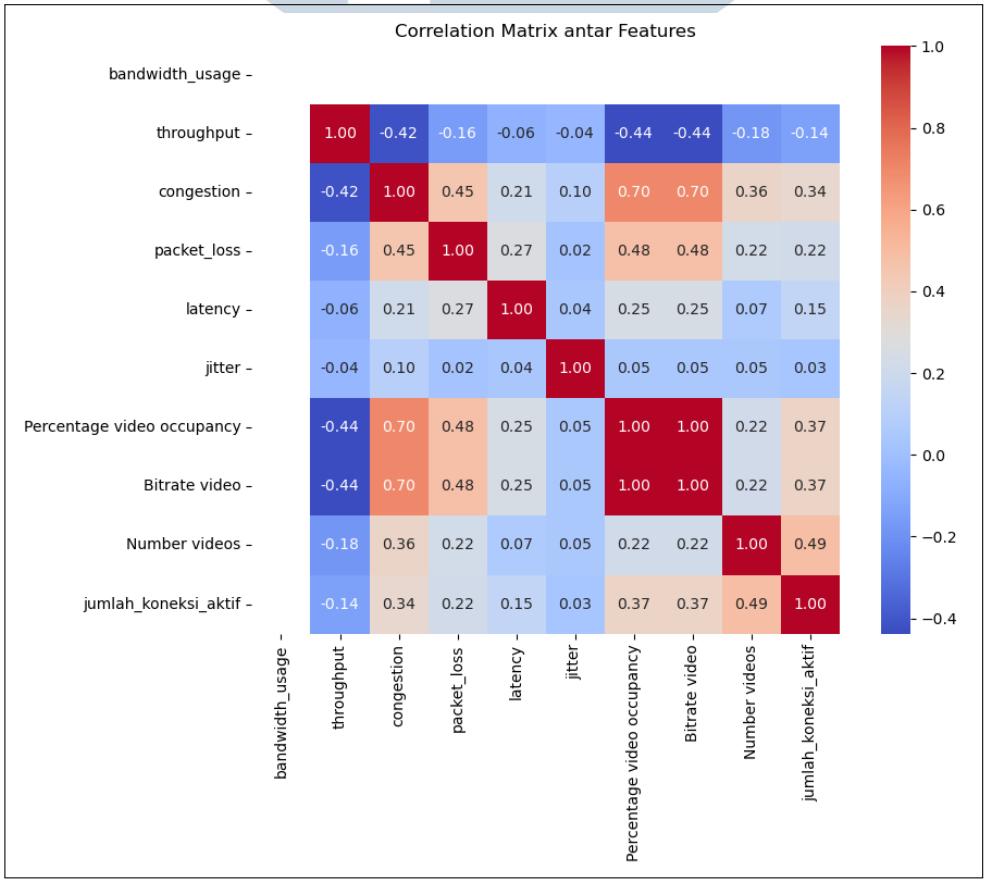
C Feature Selection

Proses seleksi fitur dilakukan menggunakan metode *Recursive Feature Elimination* (RFE) dengan estimator `RandomForestClassifier`. *Recursive*

feature elimination digunakan untuk mengurangi redundansi fitur dan bertujuan untuk meningkatkan efisiensi model, terutama dalam skenario dengan banyak variabel. Hasil seleksi digunakan dalam model 60/20/20, sedangkan pada model 80/20 seluruh fitur tetap digunakan karena tidak ditemukan korelasi tinggi antar fitur.

D Analisis Korelasi Antar Fitur

Analisis korelasi diperlukan untuk mengidentifikasi apakah terdapat hubungan linear antar fitur yang digunakan dalam model. Hal ini bertujuan jika ada fitur yang saling berkorelasi maka fitur tersebut bisa dihilangkan karena jika terdapat korelasi yang tinggi antar fitur dapat menyebabkan redundansi informasi dan berisiko terhadap masalah multikolinearitas, terutama pada model yang sensitif seperti regresi linear atau *naive bayes*. Namun, dalam konteks model seperti *random forest*, korelasi rendah antar fitur justru merupakan kondisi yang ideal karena setiap fitur menyumbangkan informasi unik.



Gambar 3.1. Grafik Matriks Analisis Korelasi Antar Fitur

3.4.4 Tabel Korelasi Antar Enam Fitur Utama

Dari Gambar 3.1 dapat diinterpretasikan ke dalam tabel berdasarkan aturan dari tabel 2.2, berikut adalah tabel korelasi antar enam fitur utama yang digunakan dalam penelitian:

Tabel 3.2. Koefisien Korelasi Antar Fitur Utama

Fitur 1	**Fitur 2**	**Koefisien Korelasi (r)**
throughput	packet_loss	-0.16
throughput	jitter	-0.04
throughput	latency	-0.06
throughput	jumlah_koneksi_aktif	-0.14
packet_loss	latency	0.27
packet_loss	jumlah_koneksi_aktif	0.22
latency	jumlah_koneksi_aktif	0.15
jitter	semua fitur lainnya	< 0.05
bandwidth_usage	semua fitur lainnya	< 0.2

3.4.5 Analisis dan Interpretasi

Dari Tabel 3.2, dapat dilihat bahwa tidak ada pasangan fitur yang memiliki korelasi tinggi (di atas 0.50). Dari hasil di atas dapat dilihat bahwa sebagian besar korelasi berada pada rentang yang sangat lemah hingga lemah, seperti pada pasangan berikut:

1. `throughput` dengan `jitter` ($r = -0.04$)
2. `packet_loss` dengan `jumlah_koneksi_aktif` ($r = 0.22$)
3. `latency` dengan `jumlah_koneksi_aktif` ($r = 0.15$)

Kondisi ini mengindikasikan bahwa keenam fitur utama bersifat saling melengkapi dan tidak redundan. Hal ini sangat penting dalam pengembangan model *machine learning*, khususnya *random forest*, karena model tidak akan mengalami masalah seperti multikolinearitas. Multikolinearitas sendiri adalah suatu kondisi pada analisis regresi di mana adanya hubungan linear yang kuat antar variabel independen dalam model, sehingga akan membuat proses pelatihan menjadi lebih stabil.

Keenam fitur utama yang digunakan dalam model yakni *jitter*, *latency*, *throughput*, *packet_loss*, *bandwidth_usage*, dan jumlah_koneksi_aktif tidak menunjukkan korelasi tinggi satu sama lain. Oleh karena itu, seluruh fitur tetap digunakan dalam pelatihan model tanpa perlu adanya penghapusan salah satu fitur.

3.5 Perancangan Desain Model

Selain penggunaan skema pembagian data 80% untuk pelatihan dan 20% untuk pengujian sebagai model utama, penelitian ini juga menggunakan skema 60/20/20, yaitu 60% untuk pelatihan, 20% untuk validasi, dan 20% untuk pengujian. Penggunaan skema ini bertujuan untuk melakukan evaluasi lebih mendalam terhadap generalisasi model dan mengidentifikasi kemungkinan *overfitting*.

Dengan menambahkan data validasi terpisah, model dapat diuji selama pelatihan tanpa melihat data uji akhir, yang membuat evaluasi lebih objektif. Selain itu, skema ini umum digunakan dalam skenario ketika *threshold tuning* dan *calibration* diperlukan, karena proses tersebut membutuhkan data validasi terpisah dari data pelatihan maupun pengujian. Perbandingan antara model 80/20 dan 60/20/20 memberikan gambaran yang lebih menyeluruh tentang bagaimana perubahan proporsi data latih memengaruhi performa model dan apakah model masih mampu mempertahankan performa yang stabil. Berikut adalah dua skema yang digunakan:

1. Skema A (60/20/20) yang terdiri dari 60% data untuk pelatihan, 20% untuk validasi, dan 20% untuk pengujian akhir. Kalibrasi dilakukan dengan menggunakan `CalibratedClassifierCV` dan parameter `cv='prefit'` (*Platt Scaling*).
2. Skema B (80/20) terdiri dari 80% data untuk pelatihan dan 20% untuk pengujian. Kalibrasi dilakukan menggunakan `cv=3` dan metode `sigmoid`, lalu untuk evaluasi akhir menggunakan *cross validation* (`cv=5`) dengan target metrik adalah *F1-Score*.

3.6 Evaluasi Model

Evaluasi performa model dilakukan menggunakan:

1. *Accuracy*

2. *Precision*
3. *Recall*
4. *F1-Score*
5. *Confusion Matrix*
6. ROC *Curve* dan AUC
7. *Precision-Recall Curve*

Threshold tuning dilakukan berdasarkan *F1-score* terbaik pada data validasi (60/20/20) atau data uji (80/20), dengan mengatur ambang probabilitas klasifikasi agar meningkatkan keseimbangan antara *precision* dan *recall*.

3.7 Eksperimen dan Pengujian

1. *Hyperparameter Tuning:* dilakukan menggunakan `GridSearchCV` dengan parameter seperti `n_estimators`, `max_depth`, `min_samples_leaf`, dan `max_features`.
2. *Threshold Tuning:* pencarian ambang klasifikasi optimal berdasarkan *F1-score* terbaik menggunakan iterasi *threshold* 0.1–0.9.
3. *Cross Validation:* digunakan 3-*fold cross validation* untuk kalibrasi model (80/20), dan validasi eksplisit pada skema 60/20/20. Validasi tambahan menggunakan 5-*fold CV* dapat digunakan untuk evaluasi generalisasi.