Evaluasi Performa Model Ensemble Learning dalam Deteksi Serangan Jaringan Internet of Things pada Dataset CIC-BCCC-IOT-HCRL-2019

Yudi Raharja[1], Agung Budi Susanto[2], Tukiyat[3]

[1,2,3] *Pascasarjana, Teknik Informatika S-2, Universitas Pamulang*

Jl. Raya Puspitek Buaran Kec. Pamulang Kota Tangerang Selatan - Banten - Indonesia

e-mail: yudiraharja17@gmail.com[1], dosen02860@unpam.ac.id[2], dimastuky@gmail.com[3]

***Abstrak*— Perkembangan pesat perangkat *Internet of Things (IoT)* membawa peningkatan kompleksitas sekaligus risiko pada keamanan jaringan. Studi ini bertujuan untuk menilai performa lima algoritma *Ensemble Learning*, yaitu *Random Forest, AdaBoost, CatBoost, XGBoost, dan LightGBM*, dalam Sistem Deteksi Intrusi (IDS) pada jaringan IoT dengan menggunakan dataset CIC-BCCC-IoT-HCRL-2019. Metode penelitian melibatkan tahap pra-pemrosesan data termasuk penerapan dua teknik normalisasi yaitu *MinMaxScaler* dan *Normalizer*, serta evaluasi model menggunakan validasi silang *5-Fold Cross-Validation* dan pembagian data latih dan uji dengan rasio 80:20. Hasil eksperimen menunjukkan algoritma *boosting* seperti *XGBoost, CatBoost, dan LightGBM* secara konsisten memiliki kinerja lebih baik dibandingkan dengan model *bagging* tradisional seperti *Random Forest. XGBoost* yang dikombinasikan dengan *MinMaxScaler* mencapai akurasi tertinggi sebesar 99,8% sementara *LightGBM* dengan *MinMaxScaler* mencatat waktu pelatihan tercepat yakni 2,54 detik. Temuan ini mengindikasikan bahwa penggunaan teknik boosting bersama normalisasi *MinMaxScaler* dapat secara signifikan meningkatkan akurasi serta efisiensi IDS berbasis IoT.**

***Kata Kunci*— *Internet of Things, Deteksi Intrusi, Machine Learning, Ensemble Learning, Boosting,* Normalisasi Data.**

Abstract - The rapid growth of Internet of Things (IoT) devices has led to increased network complexity and higher risks of cybersecurity attacks. This research assesses the effectiveness of five Ensemble Learning algorithms Random Forest, AdaBoost, CatBoost, XGBoost, and LightGBMin Intrusion Detection Systems (IDS) for IoT networks, using the CIC-BCCC-IoT-HCRL-2019 dataset. The approach includes data preprocessing, implementing two normalization methods (MinMaxScaler and Normalizer), and evaluating models via 5-Fold Cross-Validation alongside an 80:20 train-test split. Results demonstrate that boosting algorithms like XGBoost, CatBoost, and LightGBM consistently outperform the bagging-based Random Forest. Specifically, XGBoost paired with MinMaxScaler reached the highest accuracy at 99,8%, whereas LightGBM combined with MinMaxScaler achieved the quickest training time of 2.54 seconds. This study highlights that integrating boosting algorithms with MinMaxScaler normalization significantly improves both the accuracy and speed of IoT intrusion detection.

Keywords - Internet of Things, Intrusion Detection, Machine Learning, Boosting, Data Normalization.

I. PENDAHULUAN

Transformasi digital yang dibawa oleh *Internet of Things (IoT)* telah merambah berbagai sektor, namun sekaligus menimbulkan tantangan kompleks dalam aspek keamanan siber [1]. Karakteristik perangkat IoT yang beragam dan keterbatasan sumber daya komputasi pada node menjadi faktor kerentanan terhadap beragam serangan siber. Sistem Deteksi Intrusi (IDS) konvensional kerap kali tidak memadai dalam menghadapi variasi dan dinamika serangan yang terus berkembang dalam ekosistem IoT [2]. Oleh karena itu, pendekatan berbasis machine learning, terutama *Ensemble Learning*, dipandang sebagai solusi efektif untuk meningkatkan akurasi dan kemampuan adaptasi IDS [3].

Ensemble Learning mengintegrasikan beberapa model dasar sehingga prediksi yang dihasilkan lebih andal dan akurat dibandingkan satu model tunggal [4]. Metode bagging, seperti *Random Forest* [5], serta teknik *boosting*, antara lain *AdaBoost* [6], *XGBoost* [7], *CatBoost* [8], dan *LightGBM* [9], telah menunjukkan efektivitas pada klasifikasi yang kompleks, termasuk deteksi anomali dalam jaringan. Namun, evaluasi komprehensif mengenai kinerja algoritma *boosting* modern dalam konteks IDS IoT, terutama terkait pengaruh teknik pra-pemrosesan data seperti normalisasi, masih perlu dilakukan secara mendalam.

Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja lima algoritma *Ensemble Learning* tersebut dalam mendeteksi serangan pada jaringan IoT menggunakan dataset CIC-BCCC-IoT-HCRL-2019 [1]. Fokus kajian meliputi pengaruh teknik normalisasi, yakni *MinMaxScaler* dan *Normalizer*, terhadap metrik performa model seperti akurasi, presisi, *recall*, *F1-score*, dan efisiensi waktu komputasi.

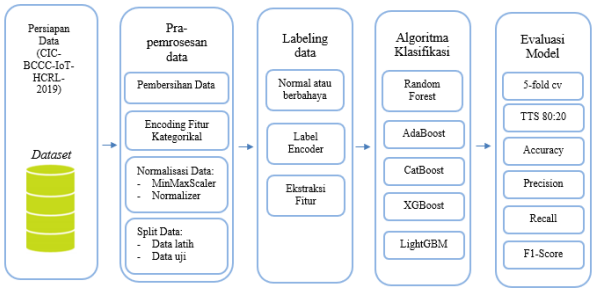
II. METODOLOGI PENELITIAN

2.1. *Analisis Penelitian*

Penelitian ini menggunakan metode deskriptif kuantitatif yang memanfaatkan pendekatan *machine learning*. Fokus dari analisis kebutuhan adalah menentukan spesifikasi yang dibutuhkan dalam pengembangan IDS untuk jaringan IoT, mencakup pemilihan algoritma yang sesuai, proses pengolahan data, serta teknik evaluasi model guna menjamin keefektifan pendeteksian beragam jenis serangan.

2.2. *Perancangan Penelitian*

Penelitian dirancang mengikuti langkah-langkah sistematis yang mencakup pengumpulan dataset, pembersihan data, pembuatan model *ensemble*, penerapan algoritma, serta penilaian kinerja model secara mendalam.



Gbr. 1 Rancangan Penelitian

2.3. *Dataset*

Penelitian ini memanfaatkan dataset CIC-BCCC-IoT-HCRL-2019, yang mencakup rekaman lalu lintas jaringan IoT asli serta simulasi beberapa jenis serangan siber seperti *Denial-of-Service (DoS)*, *Mirai*, dan *Scanning* [1]. Dataset tersebut berisi 99.702 data sampel dengan berbagai fitur jaringan yang beragam. Pemilihan dataset ini didasarkan pada kemampuannya untuk merepresentasikan lingkungan IoT masa kini secara menyeluruh, serta variasi serangan yang ada, sebagaimana diaplikasikan dalam penelitian lain.

2.4. *Pra-Pemrosesan Data*

Tahapan pra-pemrosesan meliputi beberapa langkah, yaitu: pertama, membersihkan data dengan cara menghilangkan kolom yang berisi data non-numerik serta menangani nilai yang hilang. Kedua, melakukan *encoding* pada label serangan agar menjadi format angka menggunakan Label *Encoder*. Selanjutnya, dilakukan normalisasi data menggunakan dua metode berbeda secara terpisah, yaitu *MinMaxScaler* untuk mengubah nilai fitur ke rentang antara 0 hingga 1, dan *Normalizer* untuk menormalkan tiap sampel secara individual. Terakhir, dataset dipisahkan menjadi dua bagian, di mana 80% data digunakan untuk pelatihan dan 20% untuk pengujian.

2.5. *Algoritma dan Evaluasi Model*

Lima metode *Ensemble Learning* yang digunakan meliputi *Random Forest* [5], *AdaBoost* [6], *CatBoost* [8], *XGBoost* [7], dan *LightGBM* [9]. Setiap model diuji dengan teknik *5-Fold Cross-Validation* [10] serta pemisahan data *Train-Test* dengan rasio 80:20. Evaluasi dilakukan menggunakan metrik Akurasi, Presisi, *Recall*, *F1-Score*, dan waktu yang dibutuhkan untuk pelatihan. Adapun model algoritma yang digunakan sebagai berikut:

Random Forest adalah sebuah algoritma *ensemble* yang menggunakan metode *bagging* untuk membentuk beberapa pohon keputusan secara acak [5]. Prediksi akhir ditentukan melalui voting mayoritas dari seluruh pohon yang telah dibuat. Secara matematis, kelas diprediksi menggunakan metode ini berdasarkan hasil voting tersebut.

$$\hat{y} = \text{mode}(h_1(x), h_2(x), ..., h_m(x))$$

a. (h1 (x): prediksi pohon keputusan ke-1
b. M: jumlah pohon dalam model
c. y^: kelas hasil prediksi

AdaBoost meningkatkan performa model secara bertahap dengan fokus lebih pada data yang sebelumnya salah klasifikasi dengan memberikan bobot lebih besar [6]. Hasil akhir berupa gabungan berbobot dari beberapa learner yang lemah. Secara matematis, prediksi akhir dapat dinyatakan sebagai:

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

a. ht(x): weak learner pada iterasi ke-t
b. αt: bobot learner berdasarkan akurasinya
c. T: jumlah iterasi

CatBoost mengadopsi metode gradient boosting yang dilengkapi dengan teknik *ordered boosting* untuk mengurangi pergeseran prediksi dan mengelola fitur kategori secara efektif [8]. Pembaruan model dilakukan melalui fungsi sebagai berikut:

$$F_t(x) = F_{t-1}(x) + \eta \cdot h_t(x)$$

a. Ft(x): model pada iterasi ke-t
b. η: learning rate
c. ht(x): model pohon pada iterasi ke-t

XGBoost mengembangkan teknik *boosting* dengan menambahkan regularisasi dan menggunakan optimasi berdasarkan *gradient tree boosting*. Fungsi tujuan yang digunakan adalah:

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

a. l: fungsi loss
b. Ω: fungsi regularisasi
c. fk: tree ke-k dalam model
d. K: jumlah pohon

LightGBM adalah framework *gradient boosting* yang menerapkan pertumbuhan pohon secara *leaf-wise*,

membuatnya lebih cepat dan efisien terutama untuk dataset besar [9]. Model diperbarui dengan menggunakan metode sebagai berikut:

$$F_t(x) = F_{t-1}(x) + \eta \cdot f_t(x)$$

a. ft(x): pohon keputusan yang dihasilkan pada iterasi ke-t
b. η: learning rate

Penilaian model dilakukan untuk menilai kinerja berbagai algoritma klasifikasi *(Random Forest, AdaBoost, CatBoost, XGBoost, LightGBM)* dalam menguji kemampuan *ensemble learning* pada deteksi serangan jaringan *Internet of Things* menggunakan dataset cic-bccc-iot-hcrl-2019. Proses pengujian memakai metode *5-Fold Cross-Validation* [10] guna menilai kestabilan kinerja model serta pembagian data *Train-Test (80:20)* untuk mengukur performa pada data uji. Metrik yang dipakai dalam evaluasi meliputi *5-Fold CV*, pembagian data *TTS 80:20*, *Accuracy, Precision, Recall, F1-Score,* dan *Confusion Matrix* sebagai pelengkap analisis kinerja.

III. HASIL DAN PEMBAHASAN

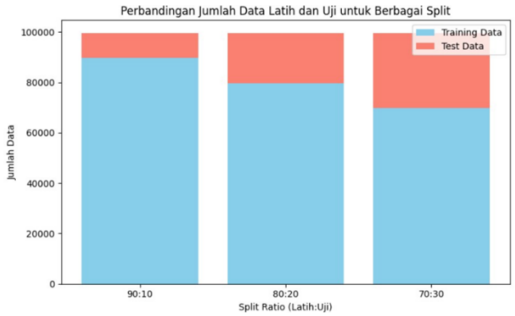
3.1. Proses dan Prosedur Pemrosesan Data

Dalam penelitian ini, dataset yang digunakan adalah CIC-BCCC-IoT-HCRL-2019 yang berisi data trafik jaringan IoT dan jenis serangannya. Proses pra-pemrosesan data meliputi beberapa tahap yakni pembersihan data, *encoding* fitur kategorikal, normalisasi data menggunakan metode *MinMaxScaler* dan *Normalizer*, serta pembagian data menjadi set pelatihan dan pengujian. Tiga skema pembagian data digunakan dengan rasio berbeda, yaitu model pertama dengan 89.731 data latih dan 9.971 data uji, model kedua dengan 79.761 data latih dan 19.941 data uji, serta model ketiga dengan 69.791 data latih dan 29.911 data uji. Perbandingan dari ketiga skema tersebut dijabarkan pada tabel berikut. Normalisasi dengan *MinMaxScaler* berguna agar nilai fitur berada dalam rentang yang sama, sehingga mencegah bias akibat perbedaan skala variabel. Perbandingan ketiga skema tersebut ditampilkan pada tabel sebagai berikut:

TABEL I
MODEL SPLIT DATA

Model	Komposisi Pembagian Data	Traning Data	Test Data	Total Data
Model 1	90:10	89731	9971	99702
Model 2	80:20	79761	19941	99702
Model 2	70:30	69791	29911	99702

Tabel I menunjukkan Split data dan dapat dapat di visualisasikan sebagai berikut:



Gbr. 2 Grafik Batang Perbandingan Split Data

Rasio pembagian data 80:20 memberikan keseimbangan yang ideal antara data untuk pelatihan dan pengujian. Dengan menggunakan 79.761 data untuk melatih model dan 19.941 data untuk pengujian, model dapat memperoleh pengetahuan dari jumlah data yang besar, sementara data uji yang cukup signifikan memastikan evaluasi performa seperti akurasi, presisi, dan *recall* dilakukan dengan representasi yang memadai dan hasil yang valid secara statistik.

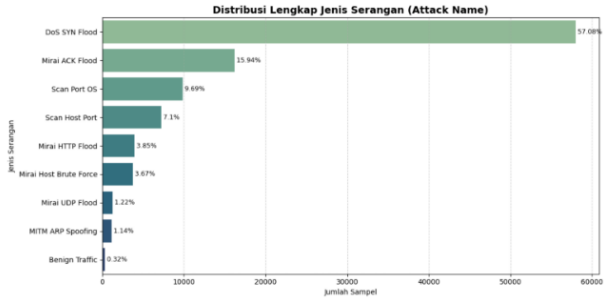
3.2. Perfoma Jenis Serangan (Attact Name)

Dataset disusun dengan mengambil data dari populasi berdasarkan proporsi yang terdapat dalam data asli. Data yang diperoleh ditampilkan dalam bentuk tabel, serta disajikan visualisasi sederhana yang menggambarkan distribusi label serangan *(Attack Name)* pada dataset tersebut.

TABEL II
JENIS-JENIS SERANGAN *(ATTACK NAME)*

No	Nama Serangan (*Attack Name*)	Jumlah Serangan	Persentase (%)
1	*DoS SYN Flood*	57979	57.08
2	*Mirai ACK Flood*	16186	15.94
3	*Scan Port OS*	9839	9.69
4	*Scan Host Port*	7212	7.10
5	*Mirai HTTP Flood*	3908	3.85
6	*Mirai Host Brute Force*	3725	3.67
7	*Mirai UDP Flood*	1242	1.22
8	*MITM ARP Spoofing*	1153	1.14
9	*Benign Traffic*	327	0.32

Tabel II memperlihatkan distribusi berbagai jenis serangan dalam dataset. Serangan yang paling banyak terjadi adalah *DoS SYN Flood* dengan frekuensi 57.979 kali, atau setara dengan 57,08% dari keseluruhan serangan yang terdeteksi. Sebaliknya, jenis serangan dengan frekuensi paling rendah adalah *Benign Traffic*, yang hanya tercatat sebanyak 327 kejadian atau 0,32% dari total data serangan. Distribusi jenis trafik serangan ini juga dapat divisualisasikan sebagai berikut:



Gbr. 3 Grafik Batang Distribusi Lengkap Jenis Serangan (Attack Name)

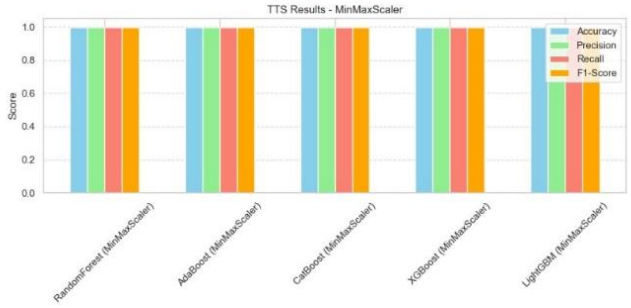
3.3. *Analisis Algoritma Ensemle Learning Performa Model dengan MinMaxScaler vs. Normalizer*

MinMaxScaler memberikan hasil performa yang lebih tinggi dan stabil dibandingkan *Normalizer* pada hampir seluruh algoritma. Hal ini kemungkinan karena *MinMaxScaler* mampu mengatur skala fitur numerik yang bervariasi dalam dataset jaringan secara lebih optimal. Eksperimen yang dilakukan memperlihatkan performa terbaik dari setiap algoritma sebagai berikut:

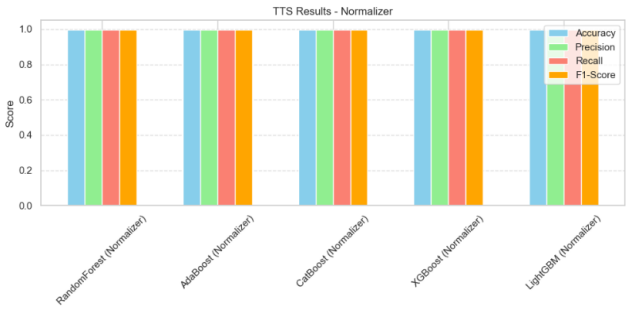
TABEL III
KLASIFIKASI KINERJA *BOOSTING*

Object	*Model*	*TTS 80:20*					*5 Fold CV*		
		Acc	*Pre*	*Rec*	*F1-S*	*Train Time (s)*	*Mean Accuracy*	*Std*	*Train Time (s)*
MinMaxScaler	*Random Forest*	0.9976	0.9975	0.9976	0.9971	4.7582	0.9977	0.0002	4.7582
	AdaBoost	0.9972	0.9966	0.9972	0.9964	32.2843	0.9972	0.0004	32.2843
	CatBoost	0.9977	0.9976	0.9977	0.9971	15.2186	0.9977	0.0002	15.2186
	XGBoost	0.9979	0.9976	0.9979	0.9976	6.1229	0.9980	0.0002	6.1229
	LightGBM	0.9978	0.9976	0.9978	0.9976	2.5388	0.9973	0.0004	2.5388
Normalizer	*Random Forest*	0.9974	0.9971	0.9974	0.9968	7.7216	0.9975	0.0003	7.7216
	AdaBoost	0.9972	0.9968	0.9972	0.9963	54.6604	0.9972	0.0002	54.6604
	CatBoost	0.9974	0.9972	0.9974	0.9966	13.8576	0.9975	0.0001	13.8576
	XGBoost	0.9975	0.9971	0.9975	0.9970	6.6034	0.9975	0.0002	6.6034
	LightGBM	0.9967	0.9960	0.9967	0.9962	3.0071	0.9968	0.0003	3.0071

Tabel III menampilkan perbandingan performa lima *classifier* berbeda *(Random Forest, AdaBoost, CatBoost, XGBoost, LightGBM)* yang diuji dengan dua metode *preprocessing (MinMaxScaler* dan *Normalizer)* pada dua skenario pembagian data, yaitu *Train-Test Split 80:20* dan *5 Folds Cross Validation*. Evaluasi dilakukan berdasarkan akurasi *(Acc)* dan waktu eksekusi (dalam detik) untuk setiap kombinasi *classifier* dan teknik *preprocessing*. Analisis mendalam terhadap kelima algoritma *ensemble learning* menunjukkan bahwa *XGBoost* dengan *MinMaxScaler* memberikan hasil terbaik secara keseluruhan, dengan nilai rata-rata akurasi tertinggi sebesar 0.9980 dan tingkat stabilitas sangat baik dengan Std 0.0002. Di sisi lain, *LightGBM* yang menggunakan *MinMaxScaler* terbukti paling efisien, mencapai akurasi mendekati dengan *F1-Score* 0.9976 serta waktu pelatihan paling singkat yaitu 2.5388 detik . Secara umum, penggunaan *MinMaxScaler* memberikan keunggulan signifikan pada model *boosting* seperti *XGBoost, LightGBM,* dan *AdaBoost*, baik dari sisi akurasi maupun efisiensi waktu pelatihan jika dibandingkan dengan *Normalizer*. Meskipun *Normalizer* menunjukkan nilai Std terbaik pada *CatBoost* sebesar 0.0001 dan waktu eksekusi paling cepat pada *CatBoost* dan *AdaBoost*, keunggulan performa dan efisiensi yang ditunjukkan oleh model *boosting* saat menggunakan *MinMaxScaler* menegaskan bahwa kombinasi tersebut adalah pendekatan terbaik untuk klasifikasi data. Perbandingan skor hampir sempurna (1.0) untuk kelima model *ensemble learning* dengan *MinMaxScaler* pada skenario *train-test split* divisualisasikan sebagai berikut:



Gbr. 4 Grafik Batang Batang Perbandingan TTS *MinMaxScaler*



Gbr. 5 Grafik Batang Batang Perbandingan TTS *Normalizer*

Hasil percobaan mengindikasikan kinerja optimal dari setiap algoritma:

TABEL IV
PERBANDINGAN KINERJA ALGORITMA

Algoritma	**Akurasi (CV)**	**F1-Score (TTS)**	**Waktu Pelatihan (s)**
XGBoost	0.9980	0.9976	6.12
LightGBM	0.9973	0.9976	2.54
CatBoost	0.9977	0.9971	15.22
Random Forest	0.9977	0.9971	4.76
AdaBoost	0.9972	0.9964	32.28

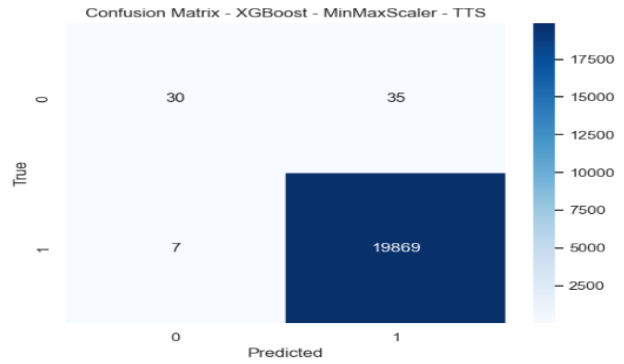
Tabel IV memperlihatkan perbandingan performa algoritma, dimana *XGBoost* mencapai akurasi tertinggi sebesar 0,9980 dan *LightGBM* menunjukkan waktu pelatihan tercepat yaitu 2,54 detik. Temuan ini menegaskan bahwa algoritma *boosting* modern sangat efektif dalam klasifikasi IDS pada IoT.

3.4. Analisis Confusion Matrix

Analisis *Confusion Matrix* pada semua model mengungkapkan pola yang konsisten, yaitu performa sangat baik dalam mengenali kelas mayoritas (serangan), namun sensitivitas terhadap kelas minoritas (lalu lintas normal) masih

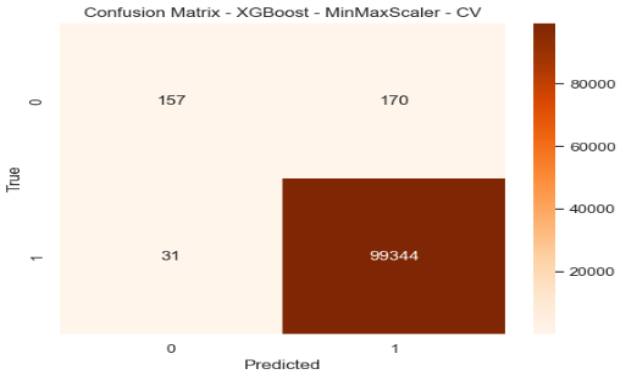
rendah. Hal ini menunjukkan adanya masalah ketidakseimbangan kelas pada dataset.

```
--- Confusion Matrix (XGBoost - MinMaxScaler - TTS) ---
      0      1
0    30     35
1     7  19869
```



Gbr 6 *Confusion Matrix – XGBoost – MinMaxScaler - TTS*

```
=== Confusion Matrix (XGBoost - MinMaxScaler - CV) ===
      0      1
0   157    170
1    31  99344
```



Gbr 7 *Confusion Matrix – XGBoost - MinMaxScaler – CV*

Confusion matrix dari model *XGBoost* dengan prapemrosesan *MinMaxScaler consistently* menunjukkan performa sangat baik dalam mengklasifikasikan kelas mayoritas (kelas 1), tetapi menghadapi kesulitan cukup besar dalam mengenali kelas minoritas (kelas 0). Perbandingan hasil antara metode *Train-Test Split (TTS)* dan *Cross-Validation (CV)* pada model *XGBoost* dengan *MinMaxScaler* yang didapat adalah sebagai berikut:

a. Pada skema TTS, model menunjukkan hasil *True Positive (TP)* sangat tinggi, yaitu 19869, dengan *False Negative (FN)* sangat rendah, hanya 7 kasus. Ini mencerminkan recall yang sangat baik untuk Kelas 1. Namun, untuk Kelas 0, performa agak kurang optimal: model hanya mampu mengklasifikasikan 30 kasus dengan benar sebagai *True Negative (TN)*, sementara 35 kasus salah diprediksi sebagai Kelas 1 *(False Positive/FP)*. Hal ini menandakan model cenderung bias ke Kelas 1 dan kurang efektif mengenali kelas negatif.
b. Pada skema *Cross-Validation*, pola serupa juga muncul ketika diuji pada data lebih besar, di mana *True Positive (TP)* mencapai 99344 dengan *False Negative (FN)* tetap rendah sebanyak 31. Namun, pada identifikasi Kelas 0, model masih mengalami kesulitan dengan *True Negative (TN)* hanya 157 dan *False Positive (FP)* sebesar 170.

Keseluruhan, model *XGBoost* terbukti sangat akurat dalam mengenali Kelas 1, menghasilkan tingkat akurasi yang tinggi. Namun, model ini menunjukkan kelemahan dalam menangkap recall untuk Kelas 0.

IV. PENUTUP

A. *Kesimpulan*

Berdasarkan hasil eksperimen, dapat disimpulkan bahwa:

1. Algoritma *boosting* dalam *ensemble learning* terbaru menunjukkan hasil yang lebih baik untuk mendeteksi serangan pada perangkat IoT.
2. Penggunaan teknik normalisasi *MinMaxScaler* meningkatkan kinerja model secara signifikan.
3. Kombinasi *XGBoost* dengan normalisasi *MinMaxScaler* memberikan hasil akurasi tertinggi yaitu 99,8% di antara konfigurasi lainnya.
4. Untuk aplikasi yang membutuhkan waktu proses cepat, *LightGBM* yang dipadukan dengan *MinMaxScaler* menjadi pilihan yang optimal.
5. Namun, performa model dalam mengenali kelas minoritas masih menjadi masalah yang perlu perhatian lebih lanjut.

B. Saran

Saran yang dapat diberikan untuk pengembangan penelitian selanjutnya berdasarkan hasil *Train-Test Split (TTS)* dan *5-Fold Cross Validation (CV)* adalah:

1. Menangani Ketidakseimbangan Data
   Penelitian selanjutnya disarankan untuk menerapkan metode penanganan data yang tidak seimbang, seperti *SMOTE (Synthetic Minority Over-sampling Technique)* atau penggunaan algoritma dengan penyesuaian bobot kelas *(class weight adjustment)*, agar dapat meningkatkan nilai *Recall* dan sensitivitas model khususnya pada Kelas 0 (minoritas).
2. Penyempurnaan *Hyperparameter* pada *XGBoost* dan *LightGBM*
   Direkomendasikan untuk melakukan *tuning hyperparameter* yang lebih mendetail pada model *XGBoost* dan *LightGBM*, mengingat kedua algoritma ini memiliki keunggulan dalam hal akurasi dan efisiensi waktu. Tujuannya adalah untuk melihat apakah prediksi terhadap Kelas 0 dapat ditingkatkan lebih baik lagi.
3. Penggunaan Metrik Evaluasi yang Lebih Tepat
   Untuk evaluasi di masa depan, sebaiknya menggunakan metrik yang tahan terhadap masalah ketidakseimbangan data, seperti F1-Score khusus untuk Kelas 0 dan *Area*

Under the Curve pada ROC (AUC-ROC), agar hasil evaluasi mampu memberikan gambaran performa model secara lebih menyeluruh.

REFERENSI

[1] Kikissagbe, B. R., & Adda, M. *Machine Learning-Based Intrusion Detection Methods in IoT.* Electronics, vol 13, no 18, pp. 3601, 2024. https://doi.org/10.3390/electronics13183601.

[2] Mohammed, M. S., & Talib, H. A. *Using Machine Learning Algorithms in Intrusion Detection Systems: A Review.* Tikrit Journal of Pure Science Vol. 29 no 3, pp. 63-74, 2024. https://doi.org/10.25130/tjps.v29i3.1553.

[3] Nassreddine, G., Nassereddine, M., & Al-Khatib, O. *Ensemble Learning for Network Intrusion Detection Based on Correlation and Embedded Feature Selection Techniques.* Computers vol 14, no 3, pp. 82, 2025. https://doi.org/10.3390/computers14030082.

[4] Zhou, Z. H. *Ensemble Methods Foundations and Algorithm (2nd ed).* A Chapman & Hall Book, 2025.

[5] Breiman, L. *Random Forest.* Machine Learning, vol 45, no 1, pp. 5-32, 2001 https://doi.org/10.1023/A:1010933404324.

[6] Battais, P. B. *Overview of AdaBoost: Reconciling its views to better*. arXiv:2310.18323v1 [cs.LG] 06 Oct 2023.

[7] Brownlee, J. *A Gentle Introduction to XGBoost for Applied Machine Learning.* https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/ . Retrieved November 05, 2025.

[8] Hancock, J. T., & Khoshgoftaar, T. M. *CatBoost for big data: an interdisciplinary review.* Survey Paper, Vol 7, pp 94, 2020.

[9] Ileri, K. *Comparative analysis of CatBoost, LightGBM, XGBoost, RF, and DT methods optimised with PSO to estimate the number of k-barriers for intrusion detection in wireless sensor networks.* International Journal of Machine Learning and Cybernetics vol 16, pp. 6937–6956, 2025. https://doi.org/10.1007/s13042-025-02654-5.

[10] Wijayanto, Sopingi, Pradana, A. I., & Atina, V. *Teknik K-Fold Cross Validation untuk Mengevaluasi Performa Mahasiswa.* Jurnal Algoritma vol 21, no 1, pp. 239-248, 2024. https://doi.org/10.33364/algoritma/v.21-1.1618.