BAB 2
LANDASAN TEORI

2.1 Penelitian Terdahulu

Rangkuman penelitian terdahulu tersebut dapat dilihat pada Tabel 2.1.

Tabel 2.1. Rangkuman Penelitian Terdahulu

No	Penulis dan Tahun	Judul Penelitian	Topik Penelitian	Hasil Penelitian
1	Perdana, Y. (2025)	*Comparative Analysis of Random Forest and XGBoost for Detecting Phishing Websites: A Machine Learning Approach*	Analisis komparasi kinerja *Random Forest* dan *XGBoost* untuk deteksi *phishing*.	*XGBoost* menghasilkan tingkat akurasi sebesar 96,4%, nilai presisi sebesar 95,8%, dan tingkat *recall* sebesar 94,9%. Algoritma *Random Forest* menghasilkan akurasi sebesar 95,2%.
2	Karin, T. R., Sani, R. R., Al Zami, F., dan Rohmani, A. (2024)	*Enhancing Bank Customer Protection Against Phishing Attacks Through XGBoost-Based Feature Analysis*	Seleksi fitur berbasis *weight* pada *XGBoost* untuk proteksi nasabah perbankan.	Seleksi fitur menaikkan akurasi dasar model sebesar 0,6%. Hasil akhir model *XGBoost* mencatatkan nilai *Accuracy* sebesar 95,5%, *Precision* sebesar 95,5%, *Recall* sebesar 95,1%, dan nilai *F1-score* sebesar 95,3%.

Lanjut pada halaman berikutnya

Tabel 2.1 Rangkuman Penelitian Terdahulu (Lanjutan)

No	Penulis dan Tahun	Judul Penelitian	Topik Penelitian	Hasil Penelitian
3	Birthriya, S. K., Ahlawat, P., dan Jain, A. K. (2025)	*Phishing Website Detection with XGBoost and Adaptive Hyperparameter Optimization using the Bat Algorithm*	Optimasi *hyperparameter XGBoost* menggunakan *Bat Algorithm*.	Optimasi parameter *learning rate* dan *max depth* berhasil mencatatkan nilai tingkat akurasi akhir yang stabil sebesar 94,27%.
4	Ramadan, A. R., Hariyadi, M. A., dan Almais, A. T. W. (2025)	*XGBoost Model Optimization Using PCA for Classification of Cyber Attacks on The Internet of Things*	Optimasi model *XGBoost* berbasis PCA dan SMOTE untuk klasifikasi serangan siber.	Kombinasi SMOTE, PCA, dan *XGBoost* mencatatkan performa akhir berupa skor akurasi sebesar 99,22%, nilai *precision* sebesar 99,10%, dan tingkat *recall* sebesar 98,85%.
5	Lucky, V. (2021)	*Rancang Bangun Browser Extension untuk Website Phishing Detector Menggunakan Algoritma XGBoost*	Ekstensi *browser* deteksi *phishing* menggunakan *XGBoost* dan RFE.	Skenario *default* menghasilkan *F1-score* 0,9783 dengan rincian *confusion matrix*: $TP = 925$, $TN = 1238$, $FP = 14$, dan $FN = 34$. Pengujian *XGBoost* + RFE (29 fitur) menghasilkan *F1-score* sebesar 0,9823.
Lanjut pada halaman berikutnya				

Tabel 2.1 Rangkuman Penelitian Terdahulu (Lanjutan)

No	Penulis dan Tahun	Judul Penelitian	Topik Penelitian	Hasil Penelitian
6	Fajar, A., Yazid, S., dan Budi, I. (2024)	*Enhancing Phishing Detection through Feature Importance Analysis and Explainable AI: A Comparative Study of CatBoost, XGBoost, and EBM Models*	Investigasi deteksi *phishing* berbasis RFE dan interpretabilitas model menggunakan *Explainable AI*.	Reduksi fitur RFE hingga 75% mempertahankan akurasi di atas 95%. Model *XGBoost* pada dataset UCI (ds_11055_32) mencatat akurasi sebesar 92,3% dengan durasi komputasi *runtimes* selama 0,6 detik.

Penelitian pertama dilakukan oleh Perdana yang melakukan analisis komparasi kinerja antara algoritma *Random Forest* dan *XGBoost* untuk mengotomatisasi deteksi situs *phishing*. Pengujian ini menggunakan dataset berskala besar yang mencatat rekam jejak aktivitas siber global dengan total anomali trafik mencapai 47.231.390 insiden. Melalui pembagian data menggunakan skema *train-test split*, algoritma *XGBoost* terbukti lebih unggul secara kuantitatif dibandingkan *Random Forest*. Model *XGBoost* sukses mencapai tingkat akurasi sebesar 96,4%, nilai presisi sebesar 95,8%, dan tingkat *recall* sebesar 94,9%, sedangkan algoritma *Random Forest* hanya memperoleh akurasi akhir sebesar 95,2% [17].

Penelitian kedua dilakukan oleh Karin, Sani, Al Zami, dan Rohmani yang berfokus pada analisis fitur berbasis metrik *weight* pada *XGBoost* untuk memproteksi nasabah perbankan dari serangan *zero-day phishing*. Eksperimen ini menggunakan dataset berisi 30 fitur awal URL yang disaring menggunakan teknik seleksi fitur intrinsik dengan menetapkan ambang batas nilai *weight importance* $\geq$ 20. Pemangkasan dimensi tersebut terbukti berhasil menaikkan nilai akurasi dasar model sebesar 0,6%. Hasil akhir pengujian menggunakan algoritma *XGBoost*

mencatatkan performa numerik yang stabil dengan nilai *Accuracy* sebesar 95,5%, *Precision* sebesar 95,5%, *Recall* sebesar 95,1%, dan nilai *F1-score* sebesar 95,3% [16].

Penelitian ketiga dilakukan oleh Birthriya, Ahlawat, dan Jain yang mengajukan optimasi *hyperparameter* pada algoritma *XGBoost* menggunakan *Bat Algorithm* untuk meningkatkan akurasi deteksi *website phishing*. Pengujian dilakukan pada beberapa dataset global dengan fokus melakukan *tuning* otomatis pada parameter tingkat pembelajaran (*learning rate*) dan kedalaman pohon keputusan (*max depth*). Integrasi metode metaheuristik berbasis ekolokasi kelelawar ini terbukti mampu mengoptimalkan struktur percabangan pohon dan mencegah terjadinya masalah *overfitting*. Hasil eksperimen dari pemodelan hibrida ini sukses mendapatkan angka skor akurasi akhir yang sangat stabil sebesar 94,27% [21].

Penelitian keempat dilakukan oleh Ramadan, Hariyadi, dan Almais yang menguji optimasi model *XGBoost* berbasis *Principal Component Analysis* (PCA) untuk klasifikasi serangan siber pada jaringan IoT. Eksperimen ini menghadapi tantangan dimensi data yang tinggi serta ketimpangan sebaran kelas biner yang ekstrem dengan rasio awal sebesar 1:27. Peneliti menerapkan reduksi dimensi matriks fitur menggunakan PCA, lalu melakukan klasifikasi menggunakan *XGBoost* yang dikombinasikan dengan skema pencarian parameter *grid search*. Hasil pengujian performa pada tabel kontingensi mencatatkan angka yang sangat tinggi, yaitu skor akurasi sebesar 99,22%, nilai *precision* sebesar 99,10%, dan tingkat *recall* sebesar 98,85% [19].

Penelitian kelima dilakukan oleh Lucky yang merancang bangun ekstensi *browser* untuk deteksi *phishing* menggunakan algoritma *XGBoost* dan *Recursive Feature Elimination* (RFE). Riset ini menggunakan dataset terstruktur yang memuat 30 fitur utama tautan web. Pada skenario pengujian awal dengan parameter *default*, model menghasilkan skor *F1-score* sebesar 0,9783 dalam waktu 0,4 detik dengan rincian komponen *confusion matrix* berupa nilai $TP = 925$, $TN = 1238$, $FP = 14$, dan $FN = 34$. Skenario terbaik yang diusulkan menggabungkan model *XGBoost* dengan seleksi fitur RFE (29 fitur aktif) serta proses *hyperparameter tuning*, yang sukses mendongkrak *F1-score* hingga mencapai angka tertinggi sebesar 0,9823 dengan total durasi pelatihan selama 3 jam 6 menit 25,96 detik [15].

Penelitian keenam dilakukan oleh Fajar, Yazid, dan Budi yang menginvestigasi deteksi *website phishing* dengan penekanan pada seleksi fitur hibrida berbasis RFE serta interpretabilitas model menggunakan *Explainable AI*

(XAI) seperti SHAP. Eksperimen dilakukan secara komprehensif pada sembilan variasi dataset, termasuk dataset UCI (32 fitur) dan dataset ISCX (112 fitur) untuk menguji skalabilitas model. Proses seleksi fitur menggunakan RFE terbukti sangat efisien karena mampu memangkas dimensi fitur hingga 75% dengan tetap mempertahankan nilai tingkat akurasi model di atas 95%. Hasil pengujian performa menempatkan *XGBoost* sebagai algoritma yang paling unggul dari segi kecepatan waktu, di mana pada pengujian dataset UCI (ds_11055_32) model *XGBoost* hanya membutuhkan waktu komputasi runtimes sebesar 0,6 detik dengan capaian nilai akurasi sebesar 92,3% [22].

2.2 Phishing

Phishing merupakan suatu upaya untuk memperoleh informasi sensitif pengguna, seperti username, kata sandi (*password*), dan data pribadi lainnya, dengan cara menduplikasi situs web resmi sehingga terlihat serupa dengan aslinya[23]. Phishing merupakan istilah yang berasal dari kata *fishing* yang berarti memancing, yaitu suatu teknik penipuan yang bertujuan mengelabui individu agar secara tidak sadar mengungkapkan informasi pribadi yang kemudian dimanfaatkan untuk kepentingan kriminal[24]. Tindakan kriminal ini dilakukan dengan cara menyamar sebagai entitas atau institusi tepercaya melalui komunikasi elektronik resmi, seperti surat elektronik (*e-mail*) maupun pesan instan[25].

Serangan *phishing* umumnya dilakukan secara sengaja oleh orang orang yang tidak bertanggung jawab, seperti pihak internal, peretas, maupun pelaku kejahatan siber yang mengeksploitasi kerentanan keamanan pada suatu situs web. Setelah berhasil menembus sistem, pelaku akan menyisipkan atau membuat halaman palsu yang menyerupai tampilan situs aslinya. Selain bertujuan untuk mencuri informasi, situs *phishing* juga sering dimanfaatkan sebagai media penyebaran *malware* serta instrumen penipuan dengan menyerupai identitas suatu website yang sah[26].

Karakteristik situs *phishing* dapat diklasifikasikan ke dalam empat kategori utama berdasarkan fitur-fitur yang diekstraksi [27]:

- *Address Bar-based Features*: Kategori ini berfokus pada manipulasi elemen pada bilah alamat (*address bar*).
- textitAbnormal-based Features: Kategori ini mengidentifikasi keberadaan anomali pada identitas situs.

- *HTML and JavaScript-based Features*: Kategori ini menganalisis pemanfaatan skrip sisi klien (*client-side*) untuk melakukan *obfuskasi* informasi, memalsukan tampilan elemen visual, atau menyembunyikan fungsi berbahaya melalui manipulasi *Document Object Model* (DOM).

- *Domain-based Features*: Kategori ini mengevaluasi aspek kredibilitas nama domain, termasuk usia registrasi domain (*domain age*), informasi *Whois*, serta penggunaan *sub-domain* yang menyerupai entitas resmi untuk membangun kepercayaan palsu.

2.3 Website

Situs web (*website*) didefinisikan sebagai himpunan halaman digital yang saling terintegrasi dan memuat berbagai elemen multimedia, seperti teks, gambar, dan elemen lainnya, yang dikonstruksi menggunakan bahasa pemrograman standar meliputi *HTML, CSS,* dan *JavaScript*. Sebagai manifestasi dari evolusi teknologi informasi, situs web kini berfungsi sebagai instrumen strategis untuk diseminasi informasi, pemasaran digital, serta penyediaan konten yang dapat diakses secara global melalui perangkat peramban web[28].

Website dapat diakses secara daring oleh individu maupun sekumpulan organisasi untuk mempercepat perolehan informasi serta memudahkan interaksi dan promosi bagi pemilik *platform* tersebut. Secara esensial, *website* bertindak sebagai representasi digital yang bertujuan menyediakan layanan sekaligus memfasilitasi komunikasi antarpengguna secara *global*. Setiap situs *website* memiliki *homepage* yang berfungsi sebagai tampilan standar dan gerbang utama bagi pengunjung sebelum mengeksplorasi konten lebih mendalam. Proses navigasi antarhalaman didukung oleh penggunaan *hyperlink*, yang memungkinkan aktivitas *browsing* atau penjelajahan informasi dilakukan secara sistematis dan terintegrasi[29].

2.4 Website Scraping

Web scraping merupakan teknik pengumpulan data secara otomatis dari halaman web dengan cara mengekstraksi informasi yang ditampilkan pada situs menggunakan program atau *script* tertentu. Teknik ini banyak digunakan dalam penelitian karena mampu memperoleh data dalam jumlah besar secara cepat dan efisien dari berbagai sumber di internet. Menurut Brown dkk. (2025),*web scraping* tidak hanya berfungsi sebagai metode akuisisi data, tetapi juga harus

memperhatikan aspek legal, etika, institusional, dan validitas ilmiah dari data yang dikumpulkan. Peneliti perlu memastikan bahwa proses pengambilan data dilakukan sesuai dengan kebijakan situs, menjaga privasi pengguna, serta mempertimbangkan kualitas dan representativitas data yang diperoleh agar hasil penelitian tetap valid dan dapat dipertanggungjawabkan. Dengan demikian, *web scraping* menjadi metode yang efektif untuk memperoleh data penelitian, namun penggunaannya harus disertai dengan pertimbangan hukum, etika, dan metodologi yang memadai. [30]

2.5 Machine Learning

Machine Learning (ML) atau pembelajaran mesin merupakan cabang dari kecerdasan buatan (*Artificial Intelligence*) yang memfokuskan pada pengembangan sistem yang mampu belajar dari data, mengidentifikasi pola, dan membuat keputusan tanpa perlu diprogram secara eksplisit untuk setiap aturan spesifik. Secara fundamental, *Machine Learning* menggunakan algoritma statistik untuk membangun model matematika berdasarkan data sampel, yang dikenal sebagai ”data latih” (*training data*), guna membuat prediksi atau keputusan pada data baru (*testing data*). Dalam domain klasifikasi keamanan siber, pendekatan *Supervised Learning* (pembelajaran terawasi) menjadi metode yang paling umum digunakan, di mana model dilatih menggunakan dataset yang telah memiliki label (misalnya: label ”*phishing*” atau ”sah”) untuk memetakan input ke output yang sesuai [31].

Penerapan *Machine Learning* dalam deteksi *phishing* dikembangkan untuk mengatasi berbagai keterbatasan yang dimiliki oleh metode tradisional, seperti pendekatan berbasis daftar hitam (*blacklist*) dan heuristik statis yang bersifat reaktif[32]. Selain itu, algoritma *Machine Learning* memiliki kemampuan untuk melakukan generalisasi terhadap pola serangan baru (*zero-day attacks)* yang belum pernah dikenali sebelumnya[33]. Dengan mengekstraksi fitur-fitur kompleks dari *URL* dan konten halaman, model *Machine Learning* dapat beradaptasi terhadap perubahan taktik penyerang secara dinamis. Efektivitas ini menjadikan *Machine Learning* sebagai komponen vital dalam arsitektur keamanan modern, mengingat volume data ancaman yang terlalu besar untuk dianalisis secara manual oleh tenaga ahli keamanan [34].

2.6 Confution Matrix

Confusion matrix merupakan matriks berukuran $N \times N$, dengan N sebagai jumlah kelas keluaran, di mana setiap baris merepresentasikan jumlah instansi dari kelas prediksi (*predicted class*) dan setiap kolom menunjukkan jumlah instansi dari kelas aktual (*actual class*)[35]. *Confusion matrix* menggambarkan performa model dengan menyajikan perbandingan antara hasil klasifikasi pada kelas aktual di setiap baris dan kelas prediksi di setiap kolom, di mana nilai pada diagonal utama merepresentasikan frekuensi data yang terklasifikasi dengan benar, sementara nilai di luar diagonal tersebut menunjukkan kesalahan prediksi[36]. matriks ini merupakan instrumen evaluasi pada model klasifikasi yang digunakan untuk memvisualisasikan serta mengukur kinerja prediksi dengan menyajikan perbandingan komprehensif antara nilai aktual dan nilai prediksi guna memberikan gambaran akurat mengenai sejauh mana model mampu mengidentifikasi kelas target secara tepat. Representasi visual dari komponen-komponen *confusion matrix* secara umum dapat dilihat pada Tabel 2.2 [37].

Tabel 2.2. Representasi Umum *Confusion Matrix*

Confusion Matrix		**Kelas Aktual**	
		Positif	**Negatif**
Kelas Prediksi	**Positif**	*True Positive* (TP)	*False Positive* (FP)
	Negatif	*False Negative* (FN)	*True Negative* (TN)

Berikut adalah penjelasan komponen-komponen dalam *Confusion matrix* :

- *True Positive* (TP): Kondisi ketika model menebak benar pada data yang memang aslinya bernilai positif.
- *True Negative* (TN): Kondisi ketika model menebak benar pada data yang memang aslinya bernilai negatif.
- *False Positive* (FP): Kondisi ketika model salah menebak data negatif sebagai data positif.
- *False Negative* (FN): Kondisi ketika model salah menebak data positif sebagai data negatif

Untuk mengukur kinerja algoritma klasifikasi, digunakan beberapa metrik evaluasi yang dihitung berdasarkan komponen *confusion matrix*. Metrik tersebut meliputi *Accuracy*, *Precision*, *Recall*, dan *F1-Score* [38].

- *Accuracy*: Digunakan untuk mengukur tingkat ketepatan pengklasifikasi dalam memprediksi data secara keseluruhan pada dua kelas yang dievaluasi. *Accuracy* menjadi acuan performa yang tepat jika jumlah *False Negative* dan *False Positive* dalam kumpulan data bersifat simetris. Rumus untuk menghitung *accuracy* dapat dilihat pada Rumus 2.1

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

- *Precision*: Merupakan rasio perbandingan antara data yang diprediksi secara akurat dengan total seluruh data yang diprediksi sebagai kelas positif. rumus dari *precision* bisa dilihat pada Rumus 2.2

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

- *Recall*: Merupakan rasio perbandingan antara data yang diprediksi secara akurat dengan total jumlah data aktual pada kelas tersebut yang rumusnya dapat dilihat pada Rumus 2.3.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

- *F1-Score*: Merupakan perbandingan rata-rata tertimbang (*weighted average*) antara *precision* dan *recall*. Metrik ini digunakan sebagai referensi utama apabila hasil evaluasi menunjukkan angka *precision* dan *recall* yang tidak seimbang. Rumus dari F1 - Score dapat dilihat pada Rumus 2.4

$$F1\text{-}Score = \frac{2 \times (Recall \times Precision)}{Recall + Precision} \tag{2.4}$$

2.7 XGBoost

Extreme Gradient Boosting atau *XGBoost* merupakan salah satu algoritma *machine learning* berbasis *ensemble learning* yang menggunakan struktur pohon

keputusan (*decision tree*)[39]. Algoritma ini sangat populer karena kemampuannya dalam menghasilkan prediksi yang akurat dengan proses komputasi yang sangat cepat[40]. Secara prinsip, *XGBoost* bekerja dengan teknik *boosting*, yaitu menggabungkan beberapa model lemah (*weak learners*) secara berurutan untuk membentuk satu model yang kuat (*strong learner*) demi meminimalkan kesalahan prediksi dari model sebelumnya secara bertahap[41]. Model prediksi dibentuk secara aditif, di mana hasil akhirnya merupakan akumulasi dari kontribusi seluruh pohon keputusan yang dibangun secara berurutan, dengan setiap pohon baru bertugas mempelajari dan memperbaiki kesalahan (*error*) yang tersisa dari pohon-pohon sebelumnya[42].

Extreme Gradient Boosting atau XGBoost membentuk model prediksi secara aditif, di mana hasil akhirnya adalah penjumlahan dari kontribusi seluruh pohon keputusan yang dibangun secara berurutan. Setiap pohon baru bertugas untuk mempelajari dan memperbaiki kesalahan (*error*) yang tersisa dari pohon-pohon sebelumnya . Fungsi prediksi akhir dari model aditif ini dirumuskan pada Rumus 2.5:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{2.5}$$

Keterangan komponen:

- K : Jumlah total pohon keputusan yang dibangun .
- $f_k(x_i)$: Hasil prediksi dari pohon ke-k terhadap input data (x_i) .
- $\hat{y}_i$: Hasil akhir nilai akumulasi prediksi untuk data ke-i .

Salah satu keunggulan utama XGBoost dibandingkan algoritma *boosting* konvensional lainnya adalah adanya mekanisme regularisasi bawaan yang efektif untuk mencegah *overfitting*, yaitu suatu kondisi di mana model terlalu fokus menghafal data pelatihan sehingga gagal memberikan prediksi yang akurat pada data baru . Model ini dioptimalkan secara matematis dengan meminimalkan sebuah fungsi objektif (*objective function*) yang menggabungkan fungsi kerugian (*loss function*) dan fungsi regularisasi , seperti yang tertera pada Rumus 2.6:

$$\text{obj}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.6}$$

Penjelasan komponen fungsi objektif:

- $\sum_{i=1}^{n} l(y_i, \hat{y}_i)$: Mengukur seberapa besar selisih jarak antara label aktual y_i dengan hasil prediksi $\hat{y}_i$. Pada kasus regresi biasanya menggunakan *squared error*, sedangkan pada kasus klasifikasi biner menggunakan *log loss*.

- $\Omega(f_k)$: Fungsi regularisasi yang bertugas mengontrol kompleksitas struktural pohon agar model tetap stabil.

Rumus formal untuk fungsi regularisasi $\Omega(f_k)$ tersebut dijabarkan pada Rumus 2.7:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{2.7}$$

Keterangan komponen penalti regularisasi:

- T : Jumlah daun (*terminal nodes*) yang terbentuk dalam satu pohon keputusan.

- w_j : Nilai bobot skor *output* nyata pada daun ke-j.

- γ (gamma) dan λ (lambda) : Parameter penalti regularisasi untuk mengendalikan jumlah daun dan bobot skor agar arsitektur pohon tidak terlalu kompleks [43].

Di samping optimalisasi fungsi penalti regularisasi struktural tersebut [44], XGBoost juga mendukung fitur pelatihan paralel (*parallel computing*) yang memungkinkannya memproses *dataset* bervolume besar secara efisien melalui mekanisme manajemen sumber daya perangkat keras [42]. Karakteristik arsitektur komputasi ini memberikan XGBoost keunggulan skalabilitas yang sangat tinggi serta performa klasifikasi yang terbukti lebih unggul dan *robust* saat diuji pada karakteristik data nyata di lapangan [45].

2.8 Feature Selection

Seleksi fitur (*feature selection*) merupakan tahapan krusial dalam siklus pengembangan *machine learning* yang bertujuan untuk mengidentifikasi dan memilih subset atribut paling relevan dari data mentah untuk digunakan dalam proses pelatihan model[46]. Tujuan utama dari *feature selection* adalah untuk meningkatkan kemampuan generalisasi model dengan menghilangkan fitur yang

bersifat redundan atau tidak memiliki korelasi signifikan terhadap variabel target[47]. Implementasi teknik ini tidak hanya mampu mempercepat proses komputasi secara efisien, tetapi juga secara efektif mengurangi risiko *overfitting* yang sering terjadi pada *dataset* berdimensi tinggi[48]. Dalam konteks deteksi *phishing*, seleksi fitur memungkinkan algoritma seperti XGBoost untuk berfokus pada karakteristik leksikal dan struktural URL yang paling diskriminatif, sehingga meningkatkan metrik akurasi dan presisi model secara keseluruhan [20].

Salah satu metode seleksi fitur yang terbukti efektif dalam mereduksi dimensi data adalah *Recursive Feature Elimination* (RFE). Metode ini bekerja secara iteratif dengan memanfaatkan metrik *feature importance* dari model untuk mengeliminasi atribut yang memiliki kontribusi paling rendah terhadap prediksi secara bertahap. Penerapan RFE dikombinasikan dengan analisis nilai kepentingan fitur global mampu memangkas hingga 75% dimensi atribut pada *dataset phishing* berdimensi tinggi tanpa mendegradasi performa model secara signifikan. Melalui pendekatan ini, atribut krusial seperti *length url*, *time domain activation*, dan *Page rank* dapat diidentifikasi secara konsisten sebagai indikator leksikal dan struktural yang paling diskriminatif. Bagi algoritma *gradient boosting* seperti XGBoost, pemanfaatan metrik *feature importance* (seperti *Gain*) melalui mekanisme RFE ini tidak hanya mempercepat *runtime* komputasi secara drastis, tetapi juga menjaga stabilitas metrik akurasi model tetap berada di atas 95% pada skala *dataset* yang besar[22].

2.9 Pembagian Data (Data Splitting)

Pembagian data (*data splitting*) merupakan tahapan krusial untuk menguji kapabilitas generalisasi model terhadap data baru (*unseen data*). Menggunakan metode *Holdout Validation*, *dataset* dipisahkan secara ketat menjadi *training set* untuk mengekstraksi pola secara iteratif dan *testing set* untuk mengevaluasi performa objektif model tanpa risiko *data leakage*.

Dalam implementasi data berskala besar, rasio pembagian 80% untuk *training set* dan 20% untuk *testing set* menjadi standar empiris yang sangat direkomendasikan. Berdasarkan penelitian oleh Rácz dkk. (2021), rasio 80/20 merupakan titik keseimbangan (*sweet spot*) terbaik, khususnya pada algoritma berbasis *tree-based ensemble* seperti *Extreme Gradient Boosting* (XGBoost). Menempatkan porsi data pelatihan sebesar 80% menjamin ketersediaan sampel yang representatif bagi XGBoost untuk menyusun struktur histogram pembelahan

pohon secara optimal tanpa mengalami *underfitting*, sedangkan alokasi data pengujian sebesar 20% terbukti secara statistik mampu memetakan sebaran varians secara konsisten dan meminimalkan *error* estimasi performa.[49]

2.10 Hyperparameter Tuning

Dalam pengembangan model prediktif menggunakan algoritma *Extreme Gradient Boosting* (XGBoost) untuk pendeteksian *website phishing*, penentuan nilai *hyperparameter* bawaan (*default*) sering kali memicu hambatan berupa *overfitting* dan *bottleneck* pada performa model. Oleh karena itu, skema pencarian parameter dinamis diposisikan sebagai komponen inti untuk mengadaptasi sifat data URL *phishing* yang sensitif dan memiliki dimensi tinggi. Penelitian ini menerapkan metode *RandomizedSearchCV* untuk mengoptimasi kombinasi variabel krusial XGBoost, meliputi *learning rate*, *max depth*, *n estimators*, *gamma*, *subsample*, dan *colsample bytree*. Metode *RandomizedSearchCV* dipilih karena mengevaluasi kombinasi ruang parameter secara acak berdasarkan distribusi tertentu, sehingga mampu memangkas waktu komputasi secara signifikan jika dibandingkan dengan pencarian menyeluruh pada *GridSearchCV*, namun tetap terbukti menghasilkan arsitektur model baseline yang kuat, reproduktif, serta kompetitif dalam menekan angka kesalahan deteksi (*false positive*)[50]

Berdasarkan Hasil pengujian yang dilakukan oleh alamsyah dkk. , proses *hyperparameter tuning* ini berhasil meningkatkan akurasi dan kemampuan generalisasi dari model XGBoost. Model yang dioptimasi menggunakan *RandomizedSearchCV* mampu menghasilkan penurunan nilai *Mean Squared Error* (MSE) menjadi 0,0210 dan peningkatan nilai *R-squared* (R^2) hingga mencapai 0,9820 dengan menggunakan 5-*fold cross-validation*. Konfigurasi parameter yang optimal terbukti dapat menyeimbangkan tingkat kompleksitas model sehingga tidak mengalami *overfitting*. Keberhasilan optimasi ini juga didukung oleh grafik kurva pembelajaran (*learning curve*) yang menunjukkan bahwa performa model semakin stabil and konsisten seiring bertambahnya data pelatihan (*training data*). Dengan demikian, penerapan *RandomizedSearchCV* terbukti efektif dalam meminimalkan *error* serta meningkatkan keandalan model XGBoost pada lingkungan nyata (*real-world environment*).[51]