BAB II
KAJIAN LITERATUR

2.1 Phishing

Phishing berasal dari kata "*fishing*" yang berarti memancing, karena penyerang menggunakan umpan berupa pesan menarik untuk memancing korban agar memberikan informasi pribadi yang ingin mereka curi seperti *username*, *password*, atau data kartu kredit [10]. *Phishing* merupakan sebuah strategi penipuan di mana pelaku berpura-pura menjadi individu atau organisasi yang dapat diandalkan, dengan tujuan untuk mendapatkan data pribadi yang penting dari pengguna internet [11]. Penyerang menggunakan berbagai teknik canggih untuk mengecoh pengguna, seperti membuat situs web palsu yang menyerupai situs asli, mengirim pesan yang tampak sah, dan memanfaatkan informasi pribadi yang tersedia di profil pengguna. Oleh karena itu, diperlukan pendekatan yang lebih efektif dalam menanggulangi serangan *phishing*, tidak hanya melalui edukasi dan kesadaran pengguna, tetapi juga dengan implementasi teknologi deteksi berbasis kecerdasan buatan (AI) untuk mengidentifikasi dan mencegah serangan *phishing* secara efektif [12].

Phishing adalah serangan *social engineering* di mana penyerang mencoba memikat pengguna untuk mendapatkan informasi sensitif mereka dengan memanfaatkan organisasi publik atau terpercaya, sehingga pengguna mempercayai pesan tersebut dan mengungkapkan informasi sensitif kepada penyerang. Setiap orang rentan terhadap serangan *phishing* karena penyerang memanfaatkan pemicu psikologis dan emosional individu, seperti rasa ingin tahu dan urgensi, serta kerentanan teknis untuk mencapai tujuannya [13]. Penyerang umumnya membuat situs web palsu yang realistis dengan URL yang tidak mencurigakan untuk menipu korban agar mengungkapkan kredensial login dan informasi sensitif seperti nomor jaminan sosial dan detail kartu kredit. Taktik yang biasa digunakan adalah penyerang mengirimkan tautan *phishing* yang tampak sah kepada korban melalui email, platform media sosial, pesan instan, atau bentuk komunikasi elektronik lainnya dengan menyamar sebagai entitas atau individu terpercaya. Ketika korban mengikuti URL tersebut, mereka diarahkan ke halaman login palsu, di mana mereka memasukkan detail login mereka seperti yang mereka lakukan di halaman login asli, sehingga penyerang memperoleh kredensial korban dan akses ke akun serta informasi pribadi mereka untuk melakukan pencurian identitas, pencurian data, atau pencurian finansial [14]. *Phishing* digambarkan sebagai penipuan

di mana pengguna internet dimanipulasi untuk mengungkapkan informasi pribadi atau rahasia yang dapat digunakan oleh penyerang secara ilegal. Serangan *phishing* dapat mengkompromikan target, baik individu maupun perusahaan, hanya melalui interaksi sosial. Dari perspektif teknis, URL *phishing* memiliki karakteristik tertentu yang membedakannya dari URL yang sah, seperti penggunaan karakter numerik yang tidak umum, karakter khusus seperti '@', jumlah titik yang berlebihan, penggunaan subdomain yang berlebihan, serta kosakata sugestif seperti '*login*', '*banking*', dan '*confirm*' yang sering ditemukan dalam URL *phishing*. Oleh karena itu, fitur-fitur yang diekstraksi dari struktur URL dapat dijadikan indikator yang efektif dalam proses deteksi serangan *phishing* secara otomatis [15]. Karakteristik tersebut meliputi:

1. Panjang URL yang tidak wajar atau berlebihan
2. Penggunaan protokol HTTP yang tidak aman, bukan HTTPS
3. Keberadaan simbol "@" dalam URL yang mengarahkan ke situs berbeda
4. Penggunaan URL shortener untuk menyembunyikan alamat asli
5. Adanya alamat IP sebagai pengganti nama domain
6. Jumlah subdomain yang tidak lazim
7. Adanya tanda hubung (hyphen) yang berlebihan dalam domain
8. Penggunaan kata kunci keamanan seperti "*secure*", "*login*", atau "*verify*" dalam URL Adanya fitur *redirect* yang mencurigakan.

Selain karakteristik tersebut, penggunaan domain yang sengaja dibuat menyerupai situs resmi melalui teknik *typosquatting* atau domain *spoofing* juga merupakan salah satu indikator penting yang kerap ditemukan pada URL *phishing* [16].

2.2 Deteksi Phishing

Deteksi halaman web *phishing* dapat dilakukan dengan dua metode, yaitu melibatkan pengguna dalam mengidentifikasi dan melaporkan URL sebagai *phishing*, serta deteksi otomatis yang membangun model berdasarkan fitur-fitur yang dapat diidentifikasi pada halaman web *phishing*, di mana fitur-fitur tersebut menjadi faktor penentu dalam mengklasifikasikan halaman web sebagai *phishing* atau sah [17]. *Phishing* tidak hanya menyebabkan kerugian finansial secara langsung, tetapi juga menimbulkan kerugian tidak langsung seperti hilangnya kepercayaan pelanggan, kerusakan reputasi, serta biaya yang terkait dengan pemulihan dan tindakan hukum bagi individu maupun organisasi [18]. Secara manual, individu dapat mengidentifikasi

kemungkinan *phishing* dengan berbagai cara. Salah satunya adalah dengan menggerakkan kursor ke atas tautan tanpa mengkliknya untuk memeriksa URL sebenarnya dan memastikan bahwa URL tersebut sesuai dengan konten yang ditampilkan. Selain itu, pengguna juga perlu berhati-hati terhadap lampiran yang meminta aktivasi makro, karena mengaktifkan makro dapat memungkinkan malware berjalan pada perangkat pengguna. Indikator lain yang perlu diperhatikan adalah kualitas tampilan yang buruk, kesalahan ejaan atau tata bahasa, serta ketidaksesuaian antara alamat email pengirim dengan nama organisasi yang diklaim [19]. Oleh karena itu, diperlukan sistem deteksi *phishing* yang lebih akurat dan efisien untuk mengidentifikasi ancaman ini, dengan memanfaatkan algoritma klasifikasi dalam machine learning guna mengenali pola dan karakteristik situs web yang berpotensi berbahaya [20]. Adapun tujuan dari deteksi *phishing* adalah untuk mendeteksi dan mengklasifikasikan serangan *phishing*, yang merupakan hal krusial dalam mengidentifikasi pola-pola penting dari serangan tersebut, sehingga teknik berbasis machine learning dapat membantu proses deteksi dan klasifikasi *phishing* secara lebih efektif [21].

Beragam pendekatan telah diimplementasikan untuk menangani *phishing*, termasuk teknik berbasis aturan (*rule-based*) dan analisis URL yang didesain untuk mengidentifikasi karakteristik *phishing* yang sering digunakan, di mana analisis URL memanfaatkan pemeriksaan fitur seperti usia domain, panjang URL, dan keberadaan kata kunci tertentu untuk membedakan situs yang sah dari situs *phishing* [22]. Namun, metode deteksi konvensional seperti blacklist maupun inspeksi manual saat ini dinilai kurang efektif karena sifatnya yang statis, sehingga sering kali gagal mengenali pola serangan baru yang terus berevolusi dan semakin canggih [23]. Sistem berbasis aturan tradisional seringkali kesulitan beradaptasi dengan teknik *phishing* yang terus muncul, sementara model machine learning memberikan solusi yang lebih fleksibel dan terukur sebagai alternatif yang lebih efektif dalam mendeteksi pola serangan yang terus berkembang [24]. Secara umum, terdapat tiga kategori utama dalam mendeteksi *phishing*, yaitu: (a) pendekatan berbasis daftar hitam (*blacklist*) dan daftar putih (*whitelist*); (b) pendekatan kemiripan visual halaman web (*web-page-based visual similarity*); dan (c) pendekatan ekstraksi fitur berbasis URL dan konten web (*URL- and web-content-based feature extraction*) [25].

Dalam beberapa tahun terakhir, ancaman siber mengalami pertumbuhan yang sangat pesat (*exponential rise*), di mana serangan *phishing* melalui e-mail telah mengakibatkan kerugian besar baik secara finansial maupun identitas. Para pelaku kejahatan siber terus berusaha menemukan cara-cara baru untuk menipu pengguna, sehingga perang antara pelaku kejahatan dan upaya

perlindungan akan terus berlanjut hingga salah satu pihak mampu mengalahkan yang lain [26]. Ketidakseimbangan kelas (*class imbalance*) merupakan salah satu tantangan terbesar dalam mendeteksi *phishing*, di mana jumlah situs web legitim jauh melebihi jumlah situs *phishing* dalam kumpulan data. Kondisi ini cenderung menyebabkan bias model terhadap kelas mayoritas dan meningkatkan risiko *false negative*, yaitu ketika situs *phishing* justru diklasifikasikan sebagai situs yang normal [27]. Serangan *phishing* terus berkembang dengan semakin canggih (*increasing sophistication*) untuk melewati langkah-langkah keamanan tradisional (*bypass traditional security measures*), sehingga teknik-teknik *phishing* yang terus berevolusi ini menjadikannya ancaman yang serius. Kondisi ini menuntut model pembelajaran mesin untuk mampu menangani data berdimensi tinggi (*high-dimensional*) dan pola yang sering kali tidak seimbang (*imbalanced datasets*) dalam mendeteksi serangan *phishing* [28].

Mengingat ketidakefisienan teknologi keamanan yang ada saat ini, para peneliti mulai banyak memperhatikan penerapan metode pembelajaran mesin (*machine learning*/ML) untuk mendeteksi *phishing*. ML menawarkan keunggulan berupa kemampuan adaptasi dan kemampuan untuk mempelajari pola dari data, yang sangat dibutuhkan dalam menghadapi taktik *phishing* yang dinamis dan terus berkembang [29]. Keputusan untuk menggunakan *machine learning* dilatarbelakangi oleh meningkatnya jumlah serangan *phishing* yang tercatat serta keterbatasan metode konvensional berbasis aturan (*rule-based methods*). Pendekatan berbasis *machine learning* merupakan pendekatan yang lazim digunakan untuk mendeteksi situs web *phishing*, di mana pengklasifikasi berbasis *machine learning* terbukti mampu mencapai akurasi lebih dari 99% dan menjadi metode yang paling efektif [30]. Mengembangkan fitur semantik yang berfokus pada URL dan Identitas Domain, Fitur Berdasarkan Anomali, Fitur yang Berbasis HTML dan JavaScript, serta Fitur yang Berdasarkan Domain, sehingga jumlah fitur yang dihasilkan lebih sedikit daripada penelitian lainnya dan meningkatkan kecepatan klasifikasi. Berdasarkan perbandingan 16 algoritma pembelajaran mesin pada dua dataset, *GradientBoostingClassifier* dan *RandomForestClassifier* memperoleh tingkat akurasi tertinggi sekitar 97%, sedangkan *GaussianNB* dan *SGD Classifier* mencatat akurasi terendah masing-masing 84% dan 81% [31]. Sementara itu, penelitian [32] yang membandingkan delapan algoritma pembelajaran mesin dengan enam metode seleksi fitur menemukan bahwa Random Forest yang menggunakan seleksi fitur Chi-2 memperoleh tingkat akurasi tertinggi sebesar 96,99% dalam mengidentifikasi situs *phishing*, menjadikannya pilihan terbaik untuk deteksi *phishing*.

2.3 Information Gain

Information gain merupakan salah satu metode yang umum digunakan dalam algoritma filter untuk memperkuat korelasi antara fitur dan kelas, sekaligus mengurangi korelasi antar fitur itu sendiri. Dengan demikian, metode ini bertujuan memperkuat hubungan antara fitur dengan kelas sekaligus mengurangi pengaruh fitur yang kurang relevan, sehingga fitur dengan nilai Information Gain yang tinggi dianggap paling informatif dan memiliki kemampuan yang lebih baik dalam membedakan setiap kelas. Kelemahan yang menjadi perhatian utama dalam pengembangan metode seleksi fitur berbasis Information Gain, karena penetapan *threshold* yang tidak tepat dapat menghasilkan subset fitur yang kurang optimal untuk proses klasifikasi. Oleh karena itu, banyak penelitian yang mengintegrasikan Information Gain dengan metode pencarian lain seperti *Particle Swarm Optimization* (PSO), *Genetic Algorithm* (GA), maupun pendekatan hibrida lainnya untuk menghasilkan subset fitur yang lebih optimal dan meningkatkan akurasi klasifikasi secara signifikan [33].

Information gain tidak hanya berfungsi sebagai alat pemilihan fitur, tetapi juga berperan dalam proses reduksi entropi data sebelum dan sesudah pemisahan, sehingga membantu meningkatkan efisiensi proses klasifikasi secara keseluruhan. Terkait penetapan *threshold*, beberapa penelitian menggunakan nilai yang berbeda-beda menjelaskan bahwa beberapa penelitian menggunakan nilai *threshold* tetap, sementara penelitian lain menggunakan standar deviasi sebagai dasar penetapan *threshold*. Untuk mengatasi kelemahan tersebut mengusulkan penggunaan nilai median dari hasil transformasi Information Gain sebagai dasar penentuan *threshold* yang lebih objektif. Dalam pendekatan ini, nilai Information Gain ditransformasikan terlebih dahulu menggunakan *Fast Fourier Transform* (FFT) untuk kemudian diambil nilai rilnya sebagai dasar perhitungan median. Fitur yang memiliki nilai IG lebih besar atau sama dengan nilai median tersebut akan dipilih sebagai fitur yang relevan. Pendekatan ini terbukti memberikan hasil yang lebih baik dibandingkan metode *threshold* konvensional karena *threshold* ditentukan secara otomatis berdasarkan karakteristik data itu sendiri, bukan berdasarkan nilai yang ditetapkan secara subjektif [34]. Terlepas dari metode penetapan *threshold* yang digunakan, proses inti information gain tetap bekerja dengan cara yang sama, yaitu mengevaluasi setiap atribut dalam dataset akan dievaluasi menggunakan metode Information Gain untuk menentukan fitur yang paling relevan. Proses ini diawali dengan menghitung nilai entropy dari masing-masing fitur, yang digunakan

untuk mengukur ketidakpastian data. Nilai information gain kemudian dihitung berdasarkan selisih antara entropy keseluruhan dan *entropy* bersyarat dari setiap atribut. Atribut dengan nilai information gain tertinggi dianggap paling berpengaruh dan diprioritaskan dalam proses klasifikasi, karena mampu memberikan pemisahan data yang paling optimal, dibawah ini adalah rumus untuk *entropy* dan information gain yang dapat dilihat sebagai berikut ini [7].

Entropy (S) = $\sum_{i=1}^{n} -p\,i * log_2\, p\,i$ …… (1)

Keterangan:

- S = dataset awal
- $p\,i$ = probabilitas kelas ke-i dalam dataset
- n = jumlah kelas

Gain (S, A) = Entropy(S) − $\sum_{i=1}^{n} \frac{|s_i|}{|s|}$ Entropy (s_i) ……..… (2)

Keterangan:

- S = dataset awal
- A = atribut yang digunakan untuk pemisahan
- s_i = subset data
- $|s_i|$ = jumlah data pada subset
- $|s|$ = jumlah data pada dataset awal

Cara kerja Information Gain dalam proses seleksi fitur dilakukan melalui beberapa tahapan berikut. Pertama, dihitung nilai *entropy* dari keseluruhan dataset S menggunakan persamaan (1) untuk mengukur tingkat ketidakpastian awal sebelum dilakukan pemisahan data. Kedua, untuk setiap atribut A, dataset dibagi menjadi beberapa subset berdasarkan nilai-nilai yang dimiliki atribut tersebut, kemudian dihitung nilai *entropy* pada masing-masing subset tersebut. Ketiga, nilai Information Gain dihitung menggunakan persamaan (2), yaitu selisih antara *entropy* keseluruhan dengan rata-rata tertimbang *entropy* dari setiap subset. Keempat, proses ini diulang untuk setiap atribut dalam dataset, dan atribut dengan nilai Information Gain yang melebihi nilai *threshold* yang ditetapkan akan dipilih sebagai fitur yang relevan dan signifikan untuk digunakan dalam pembangunan model klasifikasi.

2.4 Machine Learning untuk Deteksi Phishing

Machine learning (ML) adalah bidang dalam kecerdasan buatan yang memberi kesempatan pada sistem untuk belajar dan mengoptimalkan performanya berdasarkan data tanpa perlu diprogram secara langsung [35]. Metode deteksi *phishing* konvensional seperti pendekatan berbasis daftar hitam (*blacklist*) dan berbasis aturan (*rule-based*) terbukti kesulitan mengimbangi teknik *phishing* yang terus berkembang dengan pesat. Selain itu, metode berbasis heuristik juga sering gagal mendeteksi evolusi serangan *phishing* yang semakin canggih. Sebagai respons terhadap tantangan ini, pembelajaran mesin (*machine learning*) menawarkan solusi yang menjanjikan, meskipun model individual masih memiliki keterbatasan dalam hal generalisasi dan akurasi. Oleh karena itu, sistem deteksi *phishing* hibrid berbasis URL yang menggabungkan teknik pembelajaran terawasi (*supervised*) dan tidak terawasi (*unsupervised*) diusulkan untuk meningkatkan keandalan dan efisiensi mekanisme deteksi *phishing* [36]. Machine learning merupakan metode yang efektif untuk mendeteksi *phishing* karena mampu mengeliminasi kelemahan-kelemahan yang dimiliki oleh metode-metode sebelumnya [37].

Berdasarkan parameter masukan yang digunakan, solusi deteksi *phishing* otomatis dapat diklasifikasikan ke dalam tiga kategori utama, yaitu metode berbasis alamat web (*web address-based methods*), solusi berbasis konten halaman web (*webpage content-based solutions*), dan pendekatan hibrida (*hybrid approaches*). Lebih lanjut, metode berbasis alamat web diklasifikasikan menjadi pendekatan berbasis daftar (*list-based*), berbasis aturan heuristik (*heuristic rule-based*), dan berbasis pembelajaran (*learning-based*), sedangkan pendekatan berbasis konten web dibagi menjadi solusi berbasis aturan dan berbasis pembelajaran mesin [38]. Deteksi *phishing* berbasis pembelajaran mesin pada dasarnya merupakan permasalahan klasifikasi biner (*binary classification problem*), di mana sistem harus membedakan antara website yang bersifat menipu (*malicious*) dan website yang sah (*legitimate*). Berbagai teknik, teknologi, dan metodologi analisis digunakan dalam proses ini, dan pembelajaran mesin merupakan pendekatan yang paling umum digunakan [39]. Prosedur ekstraksi fitur merupakan langkah untuk mengubah data kasar menjadi atribut numerik yang dapat dikelola sambil tetap menjaga informasi yang terkandung dalam dataset asli, sehingga menghasilkan performa yang lebih optimal dibandingkan dengan langsung menerapkan teknik pembelajaran mesin pada data mentah. Dalam jurnal ini, fitur URL diekstrak dalam bentuk *stemmed words* menggunakan *Snowball Stemmer***,** yang mereduksi kata-kata ke bentuk dasarnya agar dataset dapat dilatih secara lebih efektif [40].

Setelah semua fitur berhasil diekstrak, model pembelajaran mesin kemudian dilatih menggunakan delapan algoritma klasifikasi, yaitu Naive Bayes (NB), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Multi-Layer Perceptron (MLP), Gradient Boosting (GB), dan Logistic Regression (LR) untuk mengklasifikasikan URL sebagai *phishing* atau *benign* [41]. Metode *machine learning* untuk mengidentifikasi situs web penipuan (*phishing*) telah banyak diteliti. Proses deteksinya umumnya dirumuskan sebagai masalah klasifikasi biner. Berbagai jenis algoritma pembelajaran mesin telah digunakan untuk menangani masalah ini, seperti SVM, KNN, Random Forest, Decision Tree, XGBoost, Logistic Regression, CNN, dan Deep Learning, yang menghasilkan model untuk mengklasifikasikan halaman web sebagai sah (*legitimate*) atau penipuan (*phishing*) berdasarkan serangkaian fitur yang dipilih [42].

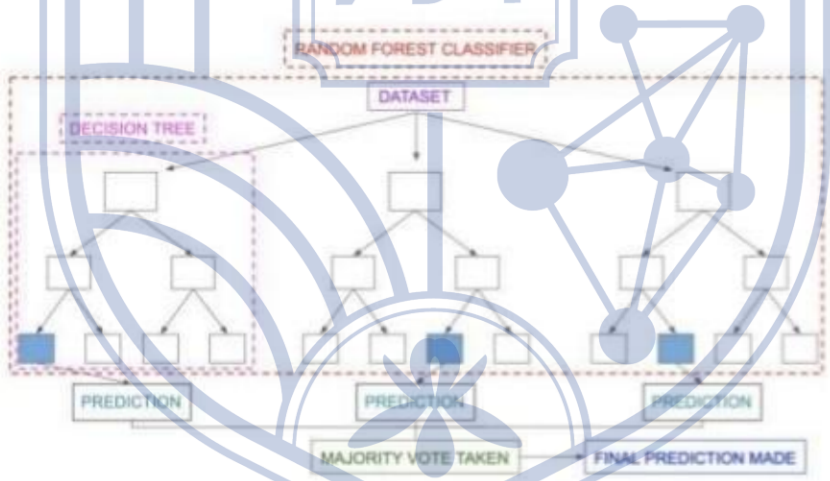
Berbagai algoritma pembelajaran mesin seperti Random Forest, SVM, XGBoost, Regresi Logistik, dan jaringan saraf telah dievaluasi untuk mendeteksi email *phishing* yang dihasilkan oleh AI. Hasil evaluasi menunjukkan bahwa jaringan saraf mencapai akurasi tertinggi sebesar 99,78%, diikuti oleh SVM sebesar 99,20% dan Regresi Logistik sebesar 99,03%. Meskipun demikian, Regresi Logistik dipilih sebagai model yang paling optimal karena menawarkan keseimbangan terbaik antara akurasi tinggi dan kemudahan interpretasi hasil klasifikasi [43]. Penelitian ini mengevaluasi tujuh algoritma pembelajaran mesin, termasuk Decisiom Tree (DT), Random Forest (RF), Peningkatan Gradien (GB), XGBoost (XGB), AdaBoost, Multi Layer Perceptron (MLP), dan pengklasifikasi Voting pada dua dataset *phishing*. Temuan menunjukkan bahwa pengklasifikasi Voting mencatatkan akurasi tertinggi sebesar 97,8% dengan precision 0,975, recall 0,987, dan F1-score 0,981 pada dataset pertama, mengungguli semua algoritma lainnya. Sementara itu, pada dataset kedua, seluruh algoritma menghasilkan nilai sempurna pada keempat metrik evaluasi, yang mengindikasikan bahwa masing-masing algoritma mampu melaksanakan proses prediksi secara efektif [44].

2.5 Random Forest

Random Forest merupakan salah satu algoritma dalam machine learning yang termasuk dalam metode *ensemble learning*, yaitu teknik yang menggabungkan banyak pohon keputusan (decision tree) untuk menghasilkan prediksi yang lebih optimal. Algoritma ini bekerja dengan membangun sejumlah decision tree menggunakan teknik *bootstrap sampling*, yaitu pengambilan

sampel data secara acak dengan pengembalian dari dataset pelatihan. Setiap pohon keputusan dilatih menggunakan subset fitur pada setiap pemisahan simpul (*node split*) dari data hasil *bootstrap* tersebut, kemudian digunakan untuk melakukan prediksi. Hasil akhir ditentukan melalui mekanisme voting, di mana kelas dengan suara terbanyak dari seluruh pohon keputusan dijadikan sebagai prediksi akhir [45].

Secara umum, proses kerja algoritma Random Forest terdiri dari beberapa tahapan utama, yaitu *bootstrap aggregating* (bagging), pembangunan pohon keputusan, serta penggabungan hasil prediksi dari seluruh pohon yang terbentuk. Pada tahap pertama, data pelatihan diambil secara acak membentuk beberapa subset yang masing-masing digunakan untuk membangun satu pohon keputusan *(*decision tree*)* yang berbeda. Dengan demikian, terbentuk sekumpulan pohon keputusan yang secara kolektif menghasilkan prediksi. Hasil prediksi dari setiap pohon kemudian digabungkan menggunakan metode majority voting untuk menentukan hasil akhir. Gambar berikut menunjukkan proses kerja algoritma Random Forest, di mana dataset dibagi menjadi beberapa bagian untuk membangun sejumlah decision tree yang kemudian menghasilkan prediksi akhir melalui mekanisme voting [46].



Gambar 2.1 Random Forest [45].

Setelah proses bootstrap sampling dilakukan, tahap berikutnya adalah pembangunan pohon keputusan (decision tree) pada setiap subset data yang telah terbentuk. Setiap decision tree melakukan proses pemisahan node dengan menggunakan ukuran impurity, seperti gini index, untuk menentukan fitur terbaik dalam membagi data. Proses ini dilakukan dengan memilih atribut yang mampu memisahkan data menjadi beberapa kelompok yang lebih homogen. Pada tahap ini, algoritma Random Forest tidak menggunakan seluruh fitur yang tersedia dalam dataset untuk

menentukan pemisahan pada setiap node, melainkan hanya memilih sebagian fitur secara acak. Teknik ini dikenal sebagai random *feature selection* yang bertujuan untuk mengurangi korelasi antar pohon keputusan sehingga meningkatkan keragaman model yang dihasilkan, dibawah ini adalah rumus untuk gini index yang dapat dilihat sebagai berikut ini. [47].

$Gini = 1 - \sum_{i=1}^{c} p\frac{2}{i}$ ..... (3)

Keterangan:

- p i = probabilitas kelas ke-i pada node
- c = jumlah kelas dalam dataset

Setelah seluruh pohon keputusan selesai dibangun, setiap pohon akan menghasilkan prediksi terhadap data yang diberikan. Hasil prediksi dari masing-masing pohon kemudian digabungkan untuk menentukan hasil akhir. Pada permasalahan klasifikasi, Random Forest menggunakan metode majority voting, yaitu kelas yang paling banyak diprediksi oleh seluruh pohon keputusan akan dipilih sebagai hasil akhir. Sementara itu, pada permasalahan regresi, hasil akhir diperoleh dengan menghitung nilai rata-rata dari seluruh prediksi pohon keputusan yang terbentuk, dibawah ini adalah rumus untuk prediction yang dapat dilihat sebagai berikut ini [48].

Prediction = Mode (T1, T2, T3, ...,T_n) ..... (4)

Keterangan:

- T_n = hasil prediksi setiap decision tree

Random Forest menggabungkan konsep bootstrap sampling (bagging) dan random feature selection, di mana setiap pohon dilatih menggunakan sampel acak yang berbeda dari data training dan hanya mempertimbangkan subset variabel independen secara acak pada setiap node. Dengan menggabungkan hasil prediksi banyak pohon melalui skema majority voting (untuk klasifikasi) atau rata-rata (untuk regresi), Random Forest mampu menghasilkan model dengan generalisasi yang lebih baik dan mengurangi masalah overfitting serta variance yang tinggi dibandingkan dengan decision tree tunggal. Oleh karena itu, algoritma ini telah menjadi salah satu benchmark dalam bidang prediksi dan banyak diaplikasikan untuk berbagai tipe variabel respons, baik kontinu, biner, maupun kategorikal. Random Forest sebagai metode ensemble learning dirancang untuk mengurangi risiko overfitting yang umum terjadi pada decision tree tunggal, yang memiliki kecenderungan terhadap overfitting dan variansi tinggi. Hal ini dicapai melalui dua mekanisme

utama, yaitu *bootstrap sampling* dan random feature selection. Bootstrap sampling memastikan bahwa setiap pohon keputusan dilatih menggunakan subset data yang berbeda tidak ada satu pohon pun yang melihat seluruh data pelatihan secara lengkap. Sementara itu, random feature selection membatasi jumlah fitur yang dipertimbangkan pada setiap pemisahan node, sehingga tujuan dari dua langkah randomisasi ini adalah untuk *decorrelate* antar pohon dan mereduksi variansi, sekaligus menghasilkan heterogenitas yang luas yang umumnya meningkatkan performa model. Dengan menggabungkan prediksi dari banyak pohon melalui mekanisme majority voting pada masalah klasifikasi, Random Forest mampu mengompensasi kesalahan individual setiap pohon, sehingga menghasilkan prediksi akhir yang lebih tergeneralisasi dengan baik. Meskipun demikian, decision tree maupun Random Forest tetap sensitif terhadap data pelatihan yang tidak seimbang, di mana satu kelas mendominasi kelas lainnya, sehingga diperlukan strategi penanganan tambahan seperti teknik resampling [49].

Overfitting merupakan kondisi di mana suatu model machine learning dilatih terlalu baik pada data pelatihan sehingga menjadi terlalu spesifik terhadap data tersebut, yang mengakibatkan lemahnya kemampuan generalisasi terhadap data baru yang belum pernah dilihat sebelumnya. Model yang mengalami overfitting cenderung menghafal pola-pola spesifik dalam data pelatihan, termasuk noise dan anomali, sehingga menghasilkan performa yang sangat baik pada data pelatihan namun menurun secara signifikan ketika diuji pada data pengujian. Hal ini khususnya terlihat pada decision tree yang tidak dipangkas, di mana kedalaman pohon dan jumlah node yang sangat tinggi menjadi indikator jelas terjadinya overfitting. Fenomena ini menjadi tantangan utama dalam pengembangan model machine learning yang andal dan mampu bekerja secara efektif pada kondisi dunia nyata. Secara teoritis, overfitting pada decision tree terjadi ketika pohon dibiarkan tumbuh tanpa pembatasan, sehingga model menjadi terlalu spesifik terhadap data pelatihan dan tidak mampu menggeneralisasi pola pada data baru. Kondisi ini muncul karena decision tree cenderung membagi ruang fitur menjadi wilayah yang semakin murni terhadap variabel target, menghasilkan aturan yang sangat spesifik namun gagal berlaku pada data yang belum pernah dilihat sebelumnya. Tingkat overfitting dapat diidentifikasi dari kesenjangan yang signifikan antara akurasi data pelatihan dan data pengujian, misalnya ketika akurasi pelatihan mendekati 100% sementara akurasi pengujian jauh lebih rendah, sebagaimana ditunjukkan pada eksperimen dataset *Telco Customer Churn* dengan akurasi pelatihan 99,84% namun akurasi pengujian hanya 77–79% [50].

Resiko overfitting semakin besar ketika dataset yang digunakan bersifat tidak seimbang (*imbalanced dataset*), yaitu kondisi di mana jumlah sampel pada kelas minoritas jauh lebih sedikit dibandingkan kelas mayoritas. Pada kondisi ini, model machine learning cenderung bias terhadap kelas mayoritas akibat distribusi kelas yang miring, sehingga kemampuan generalisasinya menurun dan menghambat penerapannya dalam situasi nyata. Meskipun akurasi keseluruhan tampak tinggi, metrik tersebut bisa menyesatkan karena terlalu menekankan performa pada kelas mayoritas. Dalam konteks deteksi *phishing*, ketidakseimbangan kelas ini menyebabkan tingginya risiko false negative, di mana situs *phishing* salah diklasifikasikan sebagai situs *legitimate* sebuah konsekuensi yang sangat berbahaya. Oleh karena itu, diperlukan teknik khusus seperti resampling untuk mengatasi ketidakseimbangan ini dan meningkatkan sensitivitas model terhadap kelas minoritas [25].

2.6 Evaluasi Kinerja Model

Evaluasi efektivitas model klasifikasi dilakukan menggunakan empat metrik utama, yaitu akurasi, presisi, recall, dan F1-score, yang dihitung berdasarkan confusion matrix dengan empat komponen True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Berikut adalah rumus yang dapat dihitung beberapa ukuran evaluasi seperti akurasi, presisi, recall, dan F1-score dengan persamaan berikut:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{.....(5)}$$

$$precision = \frac{TP}{TP+FP} \quad \text{..... (6)}$$

$$recall = \frac{TP}{TP+FN} \quad \text{..... (7)}$$

$$F1\ score = 2 \times \frac{precision \times recall}{precision+recall} \quad \text{.....(8)}$$

Akurasi menunjukkan proporsi sampel yang diprediksi dengan benar dari keseluruhan sampel. Presisi mengukur keakuratan prediksi positif model, yaitu rasio TP terhadap total prediksi positif (TP + FP). Recall mengukur kemampuan model dalam mengidentifikasi seluruh instans positif yang relevan di antara semua positif aktual (TP + FN). F1-score merupakan rata-rata harmonik dari presisi dan recall yang memberikan keseimbangan antara kedua ukuran tersebut [19].

Metrik presisi menilai keakuratan prediksi model ketika mengidentifikasi suatu URL sebagai *phishing*, yaitu seberapa banyak URL yang diprediksi sebagai *phishing* benar-benar merupakan *phishing* (mengendalikan *false positive*). Namun, presisi harus diinterpretasikan bersama dengan

recall, karena presisi yang tinggi tidak selalu menandakan model terbaik apabila recall menurun. Dalam skenario keamanan siber, melewatkan kasus *phishing* (*false negative*) umumnya lebih berisiko dibandingkan menandai URL yang sah sebagai *phishing* (*false positive*). Oleh karena itu, model dengan *recall* sempurna dan presisi yang konsisten tinggi cenderung lebih diutamakan. [51]. *Recall* mengukur kemampuan model dalam mengidentifikasi seluruh instans positif yang relevan, yaitu dihitung sebagai rasio TP terhadap total kasus positif aktual (TP + FN). F1-score merupakan rata-rata harmonik dari presisi dan recall yang memberikan keseimbangan antara kedua ukuran tersebut keempat metrik ini akurasi, presisi, recall, dan F1-score digunakan bersama untuk mengevaluasi performa model klasifikasi *phishing* secara komprehensif [52].

Penelitian ini menggunakan beberapa ukuran performa sekaligus, yaitu precision, recall, F1-score, dan kurva ROC, untuk mengevaluasi model deteksi *phishing* secara komprehensif, tidak hanya mengandalkan akurasi. Dari lima algoritma yang diuji, Random Forest menjadi algoritma dengan performa terbaik. Pada dataset PhishTank, Random Forest mencapai akurasi tertinggi sebesar 98,80% setelah hyperparameter tuning, dengan nilai precision, recall, dan F1-score masing-masing mencapai 99%, serta AUC-ROC sebesar 99,89%. Penelitian ini juga menerapkan teknik feature selection menggunakan *Recursive Feature Elimination* (RFE) untuk memilih fitur yang paling relevan, yang terbukti meningkatkan performa model secara bertahap dibandingkan hanya menggunakan *K-fold cross-validation* [53]. Mengusulkan pendekatan anti *phishing* untuk sistem IoT di *fog networks* yang memanfaatkan sebelas algoritma machine learning tradisional di antaranya Random Forest, LightGBM, SVM, dan Neural Network yang dikombinasikan dengan teknik Deep Learning menggunakan *optimizer Nadam*. Hasil eksperimen pada tiga dataset yang berbeda menunjukkan tingkat akurasi tertinggi sebesar 99,37% dalam mendeteksi dan memitigasi serangan *phishing* pada sistem IoT, khususnya ketika menggunakan dataset DS2 [54].