BAB 2
LANDASAN TEORI

2.1. Penelitian Rujukan

Berdasarkan penelitian terdahulu dapat digunakan sebagai acuan dalam mengerjakan penelitian ini agar penulis tidak akan menemukan judul yang identik dengan judul lainnya, serta dapat memperkaya wawasan dan referensi dalam memperkaya kajian pustaka. Tabel 2.1 merupakan daftar penelitian terlebih dahulu untuk menjadi sebuah referensi penulis dalam melakukan penelitian.

Tabel 2. 1 Penelitian Terdahulu.

No	Judul	Penulis dan Tahun	Metode	Hasil Penelitian
1	NATICUSdroid: A malware detection framework for Android using native and custom permissions	Mathur ,dkk. (2021)	KNN,SVM, LR,Extra Trees, AdaBoost, XGBoost, Bagging dan Random forest ,	Penelitian ini berhasil mengembangkan model machine learning bernama NATICUSDROID untuk mendeteksi berbagai jenis malware Android. Model ini menggunakan delapan algoritma machine learning yang berbeda, seperti Random Forest dan Support Vector Machine, serta teknik seleksi fitur Backward Elimination dan Multicollinearity Removal untuk meningkatkan akurasi deteksi.
2	Xgboost-Based Android Malware Detection	Jiong wang, dkk. (2020)	XGBoost, Random Forest	Hasil penelitian menunjukkan bahwa penggunaan model XGBoost dalam deteksi malware Android memberikan efisiensi komputasi yang baik dan akurasi klasifikasi yang tinggi. Dalam penelitian tersebut, eksperimen yang dilakukan menunjukkan bahwa XGBoost mampu mencapai akurasi deteksi sebesar 99% dengan waktu pelatihan yang lebih singkat dibandingkan dengan metode lain seperti Support Vector Machine.

3	An Efficient Android Malware Prediction Using Ensemble machine learning algorithms	Neamat Al Sarah,dkk (2021)	LightGBM dan Random Forest	Hasil penelitian yang dilakukan dengan pendekatan menggunakan algoritma LightGBM, menunjukkan akurasi terbaik sebesar 99,5% dalam memprediksi malware pada aplikasi Android. Penelitian ini juga menggunakan metode Recursive Feature Elimination (RFE) untuk pemilihan fitur, yang berhasil mengurangi jumlah fitur dari 215 menjadi 100, sehingga meningkatkan efisiensi dan efektivitas model dalam mendeteksi malware. RFE dan teknik pemilihan fitur lainnya berperan penting dalam meningkatkan kinerja model dengan mengeliminasi fitur yang tidak relevan, yang dapat mengganggu proses prediksi. Penelitian ini menyoroti pentingnya pemilihan fitur yang tepat dalam pengembangan model deteksi malware yang lebih akurat dan efisien.
4	Android Malware Detection Using Extreme Gradient Boosting Algorithm	A. Meena, dkk. (2023)	XGBoost, LR	Hasil penelitian menunjukkan bahwa model XGBoost Classifier unggul dalam mendeteksi malware Android dibandingkan dengan model Logistic Regression, dengan akurasi tertinggi sebesar 0.947, skor presisi 0.996, dan skor recall 0.998, meskipun Logistic Regression dikenal lebih sederhana dan mudah diinterpretasi, sementara XGBoost lebih kompleks namun efektif dalam menangkap hubungan nonlinier antara fitur dan kelas malware serta dapat mendeteksi malware baru dan yang sebelumnya belum terlihat.
5	Android Malware Detection Based On Logistic Regression And Xgboost	Li Suhuan dkk.(2023)	XGBoost, LR, Random Forest, Adaboost	Hasil penelitian menunjukkan bahwa metode deteksi malware berbasis kombinasi Logistic Regression dan XGBoost berhasil meningkatkan akurasi dan efisiensi deteksi secara signifikan.

				Dengan rata-rata Area Under the Curve (AUC) sebesar 96,18% dan akurasi mencapai 92,86% dalam 5-fold cross-validation, metode ini terbukti lebih efektif dibandingkan dengan metode deteksi malware lain yang telah ada, seperti Random Forest dan Adaboost. Selain itu, metode ini mampu mengurangi waktu pelatihan secara signifikan, dengan hanya membutuhkan 32,41 detik dari proses feature engineering hingga keluaran klasifikasi, tanpa mengorbankan akurasi.

Penelitian oleh Mathur, dkk [11] mengembangkan model logika machine learning dengan kerangka kerja baru yang disebut NATICUSDROID. Kerangka ini digunakan untuk menginvestigasi dan mengklasifikasikan aplikasi Android sebagai malware atau aplikasi jinak, dengan menggunakan izin asli dan kustom yang dipilih secara statistik sebagai fitur untuk berbagai algoritma machine learning (ML). Penelitian ini menganalisis izin pada lebih dari 29.000 aplikasi yang dikumpulkan antara tahun 2010–2019 untuk mengidentifikasi izin yang paling signifikan berdasarkan tren. Setelah itu, izin yang teridentifikasi tersebut dikumpulkan, termasuk izin asli dan kustom. Teknik seleksi fitur yang digunakan adalah backward elimination dan multicollinearity removal. Delapan algoritma ML dievaluasi untuk NATICUSDROID dalam membedakan aplikasi jinak dari malware. Hasil menunjukkan bahwa Random Forest menghasilkan kinerja terbaik dengan akurasi 97%, false-positive rate 3,32%, dan f-measure 0,96.

Penelitian yang dilakukan oleh Wang, dkk [8] mengembangkan kerangka kerja machine learning untuk klasifikasi malware menggunakan algoritma GBDT (Gradient Boosting Decision Tree), dengan memanfaatkan XGBoost untuk mendeteksi malware Android. Mereka merancang sistem deteksi otomatis yang mencakup ekstraksi fitur, analisis statis, pemilihan fitur, dan klasifikasi sampel berbahaya. Peneliti mengumpulkan sejumlah besar aplikasi dari berbagai kumpulan data atau situs web yang tersedia dan mengekstrak fitur izin serta panggilan API melalui analisis statis. Setelahnya, algoritma random forest digunakan untuk pemilihan fitur, sementara model XGBoost diterapkan untuk klasifikasi dataset yang berjumlah 2000 data. Hasil akhir menunjukkan bahwa akurasi XGBoost lebih baik atau setara

dengan SVM, dengan masing-masing mencapai akurasi 0,95%. Namun, dari segi waktu pelatihan, SVM memerlukan waktu lebih lama karena dimensi yang lebih besar, terutama dalam pencarian parameter yang optimal, dengan XGBoost memerlukan 0,19 detik dan SVM 0,10 detik.

Penelitian yang dilakukan oleh Sarah, dkk [12] menganalisis prediksi malware Android menggunakan beberapa algoritma ensemble machine learning, salah satunya adalah LightGBM. Tujuan penelitian ini adalah menemukan model terbaik untuk mendeteksi malware Android dengan menerapkan seleksi fitur Recursive Feature Elimination (RFE) guna mengurangi jumlah fitur dan mengoptimalkan waktu serta penggunaan sumber daya. Meskipun penelitian ini tidak menyebutkan asal dataset atau total datanya, diketahui bahwa dataset tersebut memiliki 215 fitur. Penelitian ini menggunakan delapan kategori fitur, dengan jumlah fitur yang dipilih secara otomatis menggunakan RFE dan RFECV. Algoritma yang digunakan meliputi empat algoritma machine learning tradisional (Logistic Regression, Gaussian Naïve Bayes, SVM, Decision Tree) dan empat algoritma ensemble (Random Forest, Gradient Boosting, XGBoost, LightGBM). Hasilnya menunjukkan bahwa LightGBM mencapai akurasi tertinggi (99,5%) dengan 100 fitur, sementara Random Forest memberikan keseimbangan dengan akurasi 99,1% dan hanya menggunakan 55 fitur. LightGBM lebih baik dalam hal akurasi, sedangkan Random Forest lebih efisien jika jumlah fitur yang lebih sedikit diperlukan.

Penelitian yang dilakukan oleh A. Meena, dkk [13] pada deteksi malware Android menggunakan algoritma Extreme Gradient Boosting (XGBoost). Dalam eksperimen ini, penulis menggunakan dataset besar yang terdiri dari lebih dari 20.000 aplikasi Android, termasuk sampel yang berbahaya dan yang tidak berbahaya. Hasil penelitian menunjukkan bahwa algoritma XGBoost mampu mencapai akurasi sekitar 95% dalam mendeteksi malware Android, menunjukkan keunggulannya dibandingkan dengan Logistic Regression, yang hanya mencapai akurasi sekitar 90%. XGBoost juga unggul dalam menangkap hubungan non-linear antara fitur-fitur yang ada, sehingga lebih efektif dalam mendeteksi varian malware yang baru dan belum pernah terlihat sebelumnya. Meskipun Logistic Regression memiliki keunggulan dalam efisiensi komputasi, XGBoost terbukti lebih efektif dalam konteks deteksi malware Android berkat kemampuannya untuk menangani dataset besar dan hubungan yang kompleks antara fitur dan kelas keluaran. Penelitian ini menunjukkan potensi XGBoost sebagai model yang kuat dalam deteksi malware Android.

Penelitian yang dilakukan oleh Suhuan, dkk [14] mengusulkan metode deteksi malware aplikasi Android berbasis Logistic Regression dan XGBoost. Metode ini menggabungkan fitur probabilistik yang dihasilkan dari model Logistic Regression dengan

fitur statistik dasar, kemudian menginputnya ke dalam XGBoost untuk mendeteksi aplikasi berbahaya. Penelitian ini mengumpulkan 568 sampel malware dari situs web ContagioMinidump dan Androzoo serta 566 sampel aplikasi jinak dari berbagai toko aplikasi utama. Fitur yang digunakan mencakup informasi panggilan API pada level framework Android yang diekstraksi secara non-intrusif menggunakan Xposed Framework dan ZjDroid. Dengan menggunakan model N-gram untuk memodelkan urutan panggilan API dan menghitung Term Frequency-Inverse Document Frequency (TFIDF) dari setiap urutan, Logistic Regression digunakan untuk mengklasifikasikan probabilitas fitur yang kemudian digabungkan dengan fitur statistik. Hasil percobaan menunjukkan bahwa metode yang diusulkan secara efektif meningkatkan akurasi deteksi aplikasi berbahaya Android, dengan AUC rata-rata 96,18% pada validasi silang 5-fold, dan akurasi deteksi sebesar 92,86%. Selain itu, metode ini menunjukkan peningkatan kecepatan, hanya memerlukan 32,41 detik dari tahap rekayasa fitur hingga keluaran hasil klasifikasi.

Dalam hal ini, penelitian berupaya mengintegrasikan dua tahap pemilihan fitur, yaitu Recursive Feature Elimination dan Multicollinearity Removal, ke dalam algoritma XGBoost. Pendekatan ini bertujuan untuk menghasilkan model yang lebih optimal dalam mendeteksi malware Android dengan mempertimbangkan efisiensi dan akurasi sebagai parameter utama dalam evaluasi. Selain itu, penelitian ini juga memberikan kontribusi tambahan pada literatur dengan mengeksplorasi integrasi dua metode pemilihan fitur yang belum banyak dibahas dalam konteks penggunaan algoritma XGBoost

2.2 . Android

Android adalah sistem operasi seluler yang didasarkan pada versi modifikasi dari kernel Linux dan perangkat lunak sumber terbuka lainnya, yang dirancang terutama untuk perangkat seluler layar sentuh seperti ponsel cerdas dan tablet. Sebagai platform perangkat lunak berbasis Linux, Android menyediakan lingkungan komputasi untuk perangkat mobile. Pada inti arsitekturnya, Android memanfaatkan kernel Linux, yang bertanggung jawab atas manajemen perangkat keras, seperti prosesor, memori, dan perangkat jaringan. Kernel Linux memberikan fondasi stabil dan aman, memungkinkan Android untuk mengontrol dan mengelola interaksi dengan perangkat keras secara efisien [15]. Kernel Linux memberikan abstraksi terhadap perangkat keras sehingga lapisan-lapisan perangkat lunak di atasnya, seperti Hardware Abstraction Layer (HAL), dapat berfungsi dengan baik. HAL berperan sebagai antarmuka standar untuk mengakses berbagai komponen perangkat keras, memungkinkan pengembang untuk membuat aplikasi tanpa harus mengetahui detail teknis

dari setiap perangkat. Selanjutnya, Android Runtime (ART) bertanggung jawab untuk menjalankan dan mengelola kode aplikasi. ART menggunakan kompilasi Ahead-of-Time (AOT) dan Just-In-Time (JIT) untuk mengubah kode Java menjadi instruksi mesin yang dapat dijalankan oleh prosesor. Di atas ART, terdapat Android Framework yang menyediakan API dan komponen penting untuk pengembangan aplikasi. Framework ini mencakup modul-modul seperti manajemen jendela, manajemen sumber daya, dan layanan sistem, yang memungkinkan aplikasi berinteraksi secara efisien dengan sistem. Pada lapisan teratas, aplikasi pengguna berjalan di lingkungan yang disediakan oleh lapisan aplikasi, di mana pengembang dapat menggunakan bahasa pemrograman seperti Java dan Kotlin [16].

2.3 Malware

Malicious software atau sering disebut dengan malware merupakan perangkat lunak yang dirancang untuk merusak sistem komputer atau perangkat lainnya dengan niat jahat, seperti mencuri data, menyebabkan kerusakan, atau memperoleh akses tidak sah. Saat ini, malware dapat dengan mudah menghindari perangkat perlindungan yang beroperasi dalam mode kernel, seperti firewall dan antivirus [17]. Malware sendiri ada beberapa jenis yaitu sebagai berikut.

1. Virus: Virus adalah jenis malware yang menggandakan dirinya dengan menyisipkan salinan diri ke dalam program atau file lain. Saat file atau program yang terinfeksi dijalankan, virus dapat menyebar dan merusak sistem dengan memodifikasi atau menggantikan file yang ada. Virus sering memanfaatkan celah keamanan untuk menyebar dan dapat menyebabkan kerusakan yang signifikan pada sistem.
2. Worm: Worm adalah malware yang dapat menyebar secara otomatis melalui jaringan tanpa memerlukan interaksi pengguna. Worm memanfaatkan kerentanan sistem atau celah keamanan untuk menginfeksi komputer lain yang terhubung dalam jaringan yang sama. Mereka dapat mengirim salinan diri mereka melalui jaringan, mengonsumsi sumber daya, dan mengganggu operasi sistem.
3. Trojan: Trojan, atau Trojan horse, adalah malware yang menyembunyikan dirinya dalam program atau file yang tampak sah atau bermanfaat. Ketika program tersebut dijalankan, Trojan dapat membuka pintu belakang pada sistem yang terinfeksi, memungkinkan penyerang untuk mengendalikan komputer dari jarak jauh, mencuri informasi sensitif, atau menambahkan malware lainnya.
4. Spyware: Spyware adalah malware yang dirancang untuk memantau dan mengumpulkan informasi tentang aktivitas pengguna tanpa izin. Spyware dapat

mencuri data pribadi seperti kata sandi, informasi keuangan, atau riwayat penelusuran internet dan mengirimkannya kepada pihak yang tidak berwenang. Biasanya, spyware tersembunyi dalam perangkat lunak atau aplikasi yang tampaknya sah.

5. Ransomware: Ransomware adalah jenis malware yang mengenkripsi file pada sistem komputer dan meminta tebusan untuk memberikan kunci dekripsi. Ransomware dapat mencegah pengguna mengakses data mereka dan menuntut pembayaran dalam bentuk mata uang digital sebagai imbalan untuk kunci dekripsi. Serangan ransomware dapat menyebabkan kerugian finansial dan kerusakan data yang signifikan.
6. Adware: Adware adalah malware yang menghasilkan iklan yang tidak diinginkan pada sistem komputer atau perangkat pengguna. Iklan ini sering muncul secara agresif dan mengganggu pengalaman pengguna. Adware sering terpasang bersamaan dengan perangkat lunak atau aplikasi gratis yang diunduh dari sumber yang tidak tepercaya.
7. Keylogger: Keylogger adalah malware yang merekam setiap ketikan yang dilakukan oleh pengguna pada sistem komputer atau perangkat lain. Keylogger dapat mencatat kata sandi, informasi pribadi, dan data sensitif lainnya yang diketik oleh pengguna, dan kemudian mengirimkan data tersebut kepada penyerang untuk tujuan jahat.
8. Botnet: Botnet adalah jaringan komputer yang terinfeksi malware dan dikendalikan oleh penyerang dari jarak jauh. Komputer dalam botnet, yang disebut bot, dapat digunakan untuk menjalankan serangan terkoordinasi seperti DDoS (Distributed Denial of Service), spam, atau aktivitas ilegal lainnya tanpa sepengetahuan pemilik komputer.

2.4 Machine Learning

Machine learning merupakan salah satu cabang dari kecerdasan buatan (AI) yang berfokus pada penggunaan algoritma dan data untuk membuat prediksi serta meningkatkan kinerja sistem berdasarkan pengalaman atau observasi sebelumnya, teknik-teknik dalam machine learning memungkinkan otomatisasi prediksi melalui pola dari data masa lalu, meniru cara manusia belajar secara bertahap dengan tujuan untuk meningkatkan akurasinya seiring waktu [18]. Selain itu, terdapat empat jenis utama algoritma machine learning, yaitu:

1. Supervised Learning: Dalam supervised learning, mesin dilatih menggunakan contoh data. Operator menyediakan algoritma machine learning dengan kumpulan data yang telah diketahui, termasuk input dan output yang diinginkan. Algoritma kemudian mempelajari cara mengaitkan input dengan output tersebut. Operator mengetahui jawaban yang benar untuk setiap masalah, dan algoritma bertugas menemukan pola

dalam data, belajar dari pengamatan tersebut, serta membuat prediksi. Prediksi yang dihasilkan oleh algoritma akan dikoreksi oleh operator, dan proses ini berlanjut sampai algoritma mencapai tingkat akurasi yang tinggi. Supervised learning ini mencakup beberapa teknik, seperti klasifikasi, regresi, dan peramalan.

2. Semi-Supervised Learning: Semi-supervised learning adalah pendekatan yang mirip dengan supervised learning, namun menggunakan kombinasi data berlabel dan tidak berlabel. Data berlabel adalah informasi yang memiliki tag atau penanda yang dapat dipahami oleh algoritma, sementara data tidak berlabel tidak memiliki informasi tersebut. Algoritma machine learning menggunakan kombinasi data ini untuk mempelajari bagaimana memberi label pada data yang tidak berlabel.
3. Unsupervised Learning: Dalam unsupervised learning, algoritma machine learning memproses data untuk menemukan pola-pola tertentu tanpa adanya kunci jawaban atau instruksi dari operator manusia. Algoritma ini menentukan korelasi dan hubungan dalam data yang dianalisis. Proses ini memungkinkan algoritma untuk menginterpretasikan kumpulan data besar dan mengelompokkan atau menyusun data tersebut berdasarkan struktur yang teridentifikasi. Metode ini mencakup teknik seperti pengelompokan (clustering) dan pengurangan dimensi (dimension reduction).
4. Reinforcement Learning: Reinforcement learning berfokus pada proses pembelajaran yang terstruktur, di mana algoritma dilengkapi dengan serangkaian tindakan, parameter, dan tujuan akhir. Dengan mendefinisikan aturan-aturan ini, algoritma kemudian mencoba berbagai opsi dan kemungkinan, memantau dan mengevaluasi setiap hasil untuk menentukan pilihan yang paling optimal. Dalam reinforcement learning, mesin belajar melalui trial and error, mengadaptasi pendekatannya berdasarkan pengalaman sebelumnya untuk mencapai hasil yang terbaik

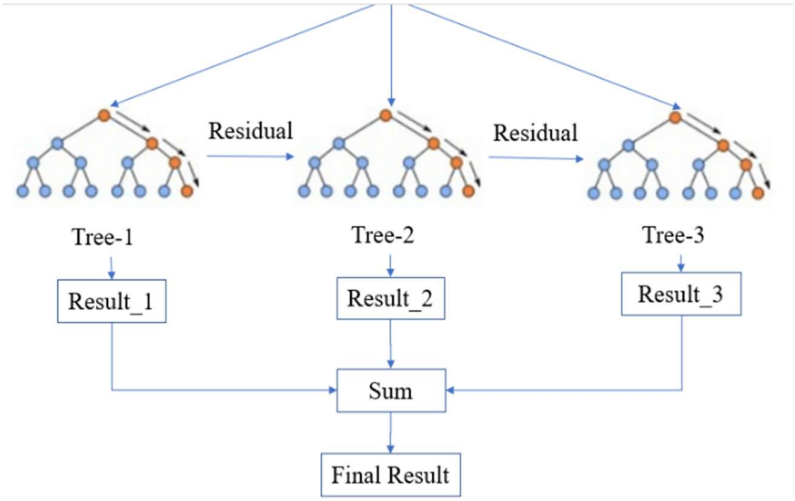
2.5 Klasifikasi

Klasifikasi adalah proses yang digunakan untuk menemukan model atau fungsi yang dapat menggambarkan serta membedakan kelas atau konsep data. Model ini bertujuan untuk digunakan dalam memprediksi kelas dari objek yang belum diketahui berdasarkan karakteristik yang diamati. Dalam statistika, metode klasifikasi tradisional seperti Analisis Diskriminan dan Regresi Logistik sering digunakan. Namun, dengan berkembangnya era data, khususnya big data, kebutuhan akan alat analisis yang kuat dan fleksibel menjadi semakin penting untuk mengungkap informasi yang bernilai dan mengorganisir pengetahuan dari kumpulan data yang besar. Sejalan dengan kemajuan dalam teknologi kecerdasan buatan, metode pembelajaran

mesin telah berkembang pesat. Metode ini memungkinkan mesin untuk belajar secara mandiri tanpa intervensi pengguna, dengan menggunakan prinsip-prinsip dari berbagai disiplin ilmu seperti statistika, matematika, dan penambangan data. Beberapa metode klasifikasi yang umum digunakan dalam pembelajaran mesin meliputi pohon keputusan (Classification and Regression Trees/CART), Random Forest, Naïve Bayes, Support Vector Machines (SVM), dan lain-lain. Metode-metode ini dirancang untuk mengidentifikasi pola dalam data dan membuat prediksi yang akurat, menjadikannya alat yang sangat berguna dalam berbagai aplikasi, mulai dari pengenalan pola hingga analisis risiko.

2.6 XGBoost

XGBoost (Extreme Gradient Boosting) adalah salah satu algoritma pembelajaran mesin yang efektif dan sering digunakan dalam berbagai aplikasi seperti klasifikasi, regresi, dan ranking. Algoritma ini menggunakan teknik ensemble learning, di mana beberapa model digabungkan untuk menghasilkan prediksi yang lebih akurat. Salah satu keunggulan utama XGBoost adalah kemampuannya untuk menangani data besar dan kompleks dengan waktu komputasi yang relatif cepat. Selain itu, algoritma ini mampu mengatasi masalah overfitting melalui proses tuning parameter, seperti learning rate, maximum depth, minimum child weight, dan subsample. XGBoost memulai proses dengan prediksi konstan, kemudian membangun serangkaian decision tree secara iteratif untuk memperbaiki kesalahan prediksi dari langkah sebelumnya, hingga mencapai hasil yang optimal. Dalam konteks klasifikasi dengan ketidakseimbangan kelas, seperti deteksi malware Android, XGBoost memiliki keunggulan dalam memberikan bobot yang lebih besar pada kelas minoritas. Algoritma ini memaksimalkan fungsi kerugian dengan menggunakan gradien dari fungsi tersebut, yang menyebabkan kesalahan pada kelas minoritas menghasilkan gradien yang lebih besar, sehingga memberikan perhatian lebih pada kelas tersebut. Dengan demikian, XGBoost dapat meningkatkan akurasi pada kelas dengan sampel yang lebih sedikit. Namun, penting untuk mempertimbangkan potensi overfitting terhadap kelas minoritas dan menyeimbangkan kinerja model menggunakan teknik seperti oversampling, undersampling, serta evaluasi metrik seperti precision, recall, atau F1-score [19].



Gambar 2. 1 Arsitektur XGBoost