**LAPORAN PROYEK MATA KULIAH  
10S3001 - KECERDASAN BUATAN**

**Network Traffic Analysis for Android Malware Detection using Support Vector Machine (SVM) Classification**

****

**Disusun Oleh :**

|  |  |
| --- | --- |
| 12S21027 | Rebecca Yulyartha Bulawan Sihombing |
| 12S21037 | Immanuella Eklesia Lumbantobing |
| 12S21039 | Widya Indah Sari Manurung |
| 12S21053 | Chesya Ivana J. M. Sitorus |
| 12S21058 | Grace Christina Yohanna Situmorang |

**Tautan GitHub** : <https://github.com/ImmanuellaEL793/PROYEK-CERTAN>

**Tautan Kaggle** : …

|  |  |  |
| --- | --- | --- |
| **PROGRAM STUDI SARJANA SISTEM INFORMASI**  **FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO**  **INSTITUT TEKNOLOGI DEL**  **NOVEMBER 2023** | | |
| Nama Dokumen: LP-PBDSI-23-GG | Tanggal : 28 November 2023 | Jumlah Halaman : 27 |

**DAFTAR ISI**

[DAFTAR ISI 2](#_Toc152056298)

[DAFTAR TABEL 3](#_Toc152056299)

[DAFTAR GAMBAR 4](#_Toc152056300)

[1. Pendahuluan 5](#_Toc152056301)

[1.1 Latar Belakang 5](#_Toc152056302)

[1.2 Tujuan 6](#_Toc152056303)

[1.3 Manfaat 7](#_Toc152056304)

[1.4 Ruang Lingkup 7](#_Toc152056305)

[1.5 Istilah dan Singkatan 8](#_Toc152056306)

[2. Studi Literatur 10](#_Toc152056307)

[3. Metode 15](#_Toc152056308)

[4. Hasil Pengujian 26](#_Toc152056309)

[5. Analisis 28](#_Toc152056310)

[6. Kesimpulan 29](#_Toc152056311)

[7. Pembagian Pekerjaan 30](#_Toc152056312)

[REFERENSI 31](#_Toc152056313)

[LAMPIRAN 34](#_Toc152056314)

**DAFTAR TABEL**

[Tabel 1. Istilah atau Singkatan Proyek 6](#_Toc151554797)

[Tabel 2. Pembagian Tugas Anggota Kelompok 24](#_Toc151554798)

# **DAFTAR GAMBAR**

[Gambar 1. Tahapan Klasifikasi Android Malware Menggunakan SVM Model 14](#_Toc151984510)

[Gambar 2. Hyperplanes di tampilan 2D dan 3D 17](#_Toc151984511)

[Gambar 3. Optimal Hyperplanes 18](#_Toc151984512)

[Gambar 4. Cc 19](#_Toc151984513)

[Gambar 5. Cc 19](#_Toc151984514)

1. **Pendahuluan**
   1. **Latar Belakang**

Di era digital yang terus berkembang, internet menjadi alat utama dalam mempermudah berbagai aktivitas sehari-hari. Namun, sayangnya, semakin banyak orang yang menggunakan internet, semakin banyak juga kejahatan yang terjadi [1]. Salah satu bentuk serangan yang umum adalah penggunaan perangkat lunak jahat, yang lebih dikenal sebagai malware [2]. Hal ini ditandai dengan semakin beragamnya serangan malware yang digunakan oleh para penyerang, seperti *virus, worm, trojan horse, rootkit, spyware, adware,* dan lainnya, memiliki tujuan merusak atau mengganggu sistem perangkat pengguna [3]. Masing-masing jenis malware memiliki cara kerja yang berbeda, tetapi semuanya memiliki tujuan untuk membahayakan pengguna.

Pentingnya keamanan digital tidak bisa diabaikan mengingat dampak negatif serangan malware, seperti risiko pencurian data pribadi, pengambilalihan kendali perangkat, dan penyebaran konten berbahaya [4]. Saat ini, para penyerang malware banyak mencari celah untuk menyembunyikan malware mereka dan menghindari deteksi oleh sistem keamanan. Salah satu cara yang sering digunakan adalah dengan menyembunyikan malware di dalam file atau program lain yang tampak tidak berbahaya, seperti dalam file PDF, dokumen Microsoft Office, atau bahkan file musik [5]. Cara lain yang sering digunakan adalah dengan menggunakan teknik enkripsi untuk menyembunyikan kode malware [6]. Ancaman serangan malware juga dilakukan dengan penyebaran konten berbahaya melalui aplikasi dari sumber yang tidak resmi, menciptakan saluran untuk serangan siber yang lebih luas [7]. Aplikasi-aplikasi ini dapat diunduh dari situs web atau toko aplikasi yang tidak terpercaya dan berisi malware yang dapat menginfeksi perangkat pengguna. Hal ini tentunya menjadi masalah serius yang berdampak pada ekosistem perangkat pengguna. Hal ini terjadi karena diversifikasi malware membuat perangkat lunak jahat semakin sulit dideteksi dikarenakan serangan malware terus berkembang dan lebih luas menyerang perangkat pengguna.

Dalam menghadapi masalah ini, perlu dilakukan analisis data lebih lanjut untuk mengetahui pola dinamika serangan malware yang banyak terjadi. Salah satu caranya dengan menggunakan metode klasifikasi. Metode klasifikasi menjadi esensi penting untuk memahami jenis-jenis malware yang mengancam. Klasifikasi dapat digunakan untuk mengelompokkan malware ke dalam kategori-kategori berdasarkan karakteristik dan perilakunya. Namun, metode klasifikasi mengadaptasi berbagai algoritma untuk mengelompokkan data. Salah satu metode klasifikasi yang paling efektif untuk mengklasifikasi adalah algoritma *Support Vector Machine (SVM).* Algoritma SVM mampu membangun model yang efektif dalam mengenali pola-pola khas dari berbagai jenis malware [8].

SVM merupakan metode klasifikasi yang efektif untuk menyelesaikan masalah pengelompokan, terutama ketika kita berurusan dengan data yang memiliki banyak fitur atau karakteristik [9]. Algoritma ini bukan hanya efektif dalam konteks klasifikasi malware, tetapi juga telah terbukti berhasil dalam berbagai penelitian lainnya. Hal ini dikarenakan algoritma SVM mempunyai akurasi klasifikasi yang lebih konstan dibandingkan dengan algoritma lainnya [10]. Penelitian sebelumnnya dilakukan oleh Maglogiannis dan Zafiropolous (2007) melakukan diagnosis dan prognosis breast cancer dengan menggunakan SVM [4]. Hasil penelitian tersebut menunjukkan bahwa ketepatan klasifikasi menggunakan SVM mencapai 97%. Rachman dan Purnami (2012) melakukan penelitian terkait klasifikasi tingkat keganasan kanker menggunakan metode regresi logistik dan SVM. Hasilnya menunjukkan bahwa penggunaan SVM memberikan tingkat akurasi yang lebih tinggi, mencapai 98,11% [11].

Berdasarkan hasil kontribusi penelitian yang positif dan pemahaman atas masalah yang telah diuraikan sebelumnya, diharapkan proyek ini akan mengembangkan model untuk mengklasifikasi berbagai jenis malware yang mengancam pengguna Android dengan menggunakan algoritma *Support Vektor Machine* (SVM). Dengan demikian, proyek ini diharapkan dapat memberikan solusi yang efektif dalam menganalisis ancaman keamanan digital pada platform Android melalui penerapan SVM dalam mengklasifikasi malware.

* 1. **Tujuan**

Penelitian ini memiliki tujuan, antara lain:

1. Mengembangkan model untuk mengklasifikasi berbagai jenis malware yang mengancam pengguna Android dengan menggunakan algoritma Support Vektor Machine (SVM).
2. Menganalisis keakuratan klasifikasi malware Android menggunakan algoritma SVM.
3. Meningkatkan efisiensi proses klasifikasi malware Android menggunakan algoritma SVM.
   1. **Manfaat**

Penelitian ini diharapkan dapat memberikan kontribusi, diantaranya:

1. Memberikan pemahaman bagi pengguna Android terkait jenis malware yang mengancam keamanan digital pada platform Android.
2. Menyediakan landasan untuk pengembangan sistem keamanan lebih lanjut yang dapat secara proaktif mengatasi ancaman malware Android.
3. Menjadi referensi bagi peneliti lain dalam mengembangkan metode klasifikasi malware yang lebih efektif.
   1. **Ruang Lingkup**

Ruang lingkup proyek ini mencakup beberapa aspek yang akan dibahas dan diimplementasikan. Adapun ruang lingkup proyek ini meliputi:

1. Dataset yang digunakan dalam penelitian ini menggunakan data public yang tersedia melalui halaman website Dataset Kaggle yang diakses melalui link berikut: <https://www.kaggle.com/datasets/xwolf12/network-traffic-android-malware>
2. Algoritma SVM yang digunakan adalah algoritma SVM dengan kernel RBF.
3. Dataset yang digunakan adalah dataset yang bersifat numerik. Fitur-fitur atau kolom-kolom dalam dataset ini berisi nilai numerik yang mencerminkan berbagai aspek aktivitas jaringan pada sistem Android.
4. Pembatasan variabel yang tidak diperlukan adalah yang terdaftar dalam variabel unwanted\_columns**,** seperti kolom duracion, avg\_local\_pkt\_rate, avg\_remote\_pkt\_rate, source\_app\_packets.
5. Pembagian data menjadi set pelatihan dan pengujian dilakukan dengan proporsi tetap (test\_size=0.2).
6. Klasifikasi target hanya terdiri dari dua kelas (“*benign*” dan “*malware*”). Tipe kelas *benign* termasuk ancaman yang tidak mengandung malware. Sebaliknya, *malware* yang paling berbahaya.
7. Evaluasi kinerja agen cerdas akan melibatkan metrik-metrik standar seperti akurasi, presisi, recall, dan F1 score untuk mengukur dampak malware pada jaringan Android.
   1. **Istilah dan Singkatan**

Istilah atau singkatan yang digunakan dalam pengembangan proyek, dijabarkan pada tabel berikut:

**Tabel 1.** Istilah atau Singkatan Proyek

|  |  |
| --- | --- |
| Istilah atau Singkatan | Definisi atau Kepanjangan |
| Agen Cerdas | Entitas pemrosesan yang dapat membuat keputusan cerdas berdasarkan informasi yang diterimanya. |
| Akurasi | Persentase data uji yang diklasifikasikan dengan benar oleh model. |
| Android | Sistem operasi seluler |
| Dataset | Kumpulan data yang digunakan untuk pelatihan model klasifikasi |
| F1 Score | Harmonic mean dari presisi dan recall, memberikan gambaran keseluruhan kinerja model klasifikasi. |
| Kaggle | Platform kompetisi dan sumber daya untuk data ilmiah dan proyek analisis data. |
| Kernel RBF | Kernel yang digunakan dalam algoritma SVM untuk merepresentasikan data |
| Malware | Perangkat lunak berbahaya yang dirancang untuk merusak, mengakses, atau mengambil alih sistem atau data tanpa izin. |
| Metode Klasifikasi | Teknik yang digunakan untuk mengelompokkan data ke dalam kategori tertentu |
| Perangkat Lunak | Serangkaian instruksi atau program komputer yang dirancang untuk melakukan fungsi-fungsi tertentu, seperti pengolah kata dan peramban web, hingga sistem operasi yang mengontrol operasi dasar komputer. |
| Preprocessing Data | Serangkaian langkah untuk membersihkan, mengubah, atau mengorganisir data agar sesuai dengan kebutuhan analisis. |
| Presisi | Rasio antara jumlah *true positive* dan total hasil yang diprediksi sebagai positif oleh model klasifikasi. |
| Recall | Rasio antara jumlah true positive dan total jumlah sampel yang sebenarnya positif. |
| SVM | *Support Vector Machine*, algoritma pembelajaran mesin untuk klasifikasi dan regresi. |

Tabel di atas berisi istilah dan singkatan yang mungkin dijumpai dalam dokumen ini, beserta definisi atau kepanjangannya masing-masing. Hal ini bertujuan untuk memastikan pemahaman yang konsisten dan jelas terhadap terminologi yang digunakan dalam konteks proyek pengklasifikasian malware pada jaringan Android.

1. **Studi Literatur**
   1. **Malware**

Malware, singkatan dari *malicious software*, merujuk pada perangkat lunak berbahaya yang dirancang untuk merusak, mengakses, atau mengambil alih sistem atau data tanpa izin [12]. Jenis malware melibatkan berbagai ancaman, termasuk *virus, worm, trojan horse, rootkit, spyware,* dan *adware.* Masing-masing jenis memiliki karakteristik dan tujuan yang berbeda, tetapi semuanya dapat menimbulkan dampak negatif terhadap perangkat pengguna, seperti pencurian data pribadi, pengambilalihan kendali perangkat, dan penyebaran konten berbahaya (Smith, 2022).

* 1. ***SVM (Support Vector Machine)***

*Support Vector Machine (SVM)* adalah algoritma pembelajaran mesin yang digunakan untuk melakukan klasifikasi dan regresi. SVM bekerja dengan mencari batas keputusan yang optimal, atau hyperplane, yang dapat memisahkan dua kelas data dengan margin maksimal [8]. SVM bekerja dengan membangun model berdasarkan data pelatihan yang dikategorikan. Model ini kemudian digunakan untuk mengklasifikasikan data baru ke dalam salah satu kelas yang telah ditentukan. Pada algoritma *Support Vector Machine* (SVM), hal ini dikenal sebagai optimalisasi *hyperplane*. *Hyperplane* ini berfungsi sebagai batas pemisah antara *support vector* dari kelas satu dan kelas lainnya. Tujuan utamanya adalah mengoptimalkan *support vector*, khususnya yang berdekatan antara kelas satu dan kelas lainnya. *Support vector* ini menjadi acuan untuk menentukan batas klasifikasi agar *hyperplane* yang dibuat menjadi seoptimal mungkin. *Support vector* berasal dari dataset yang telah diubah menjadi nilai vektor setelah proses ekstraksi fitur. Sebagai contoh, dalam dataset pelatihan, kita memiliki pasangan nilai x dan y dalam bentuk {(𝑥1, 𝑦1), … , (𝑥𝑛, 𝑦𝑛)}, di mana x adalah vektor dan y adalah label kelasnya [13].

Dalam konteks klasifikasi malware pada platform Android, SVM dapat digunakan untuk mengenali pola-pola khas dari berbagai jenis malware berdasarkan fitur-fitur tertentu, seperti perilaku jaringan, pola akses, dan tanda-tanda khas lainnya [14]. Melalui proses ini, SVM dapat membantu dalam membedakan antara perangkat lunak yang aman dan malware.

* 1. **Kernel RBF**

Dalam menggunakan metode RBF (Radial Basis Function) untuk penyelesaian, terdapat dua parameter penting yaitu gamma dan C. Gamma berperan sebagai penentu seberapa ketatnya batas keputusan dan wilayah keputusan. Sebagai contoh, jika nilai gamma kecil, maka batas keputusan akan menjadi sempit, tetapi wilayah keputusan akan menjadi luas, dan sebaliknya. Oleh karena itu, besarnya nilai gamma yang dihasilkan harus lebih besar dari nol [13]. Sementara itu, parameter C berfungsi sebagai hukuman atau penalti terhadap kesalahan dalam proses klasifikasi. Nilai C ini juga memainkan peran dalam menentukan sejauh mana model SVM mengizinkan kesalahan klasifikasi pada data pelatihan. Nilai C yang lebih tinggi akan mengakibatkan penalti yang lebih besar terhadap kesalahan, menciptakan batas keputusan yang lebih ketat[13].

* 1. **Penelitian Terdahulu**

Beberapa penelitian terdahulu telah dilakukan dalam mengimplementasikan SVM untuk klasifikasi dan deteksi malware. Watson et al. (2015) melakukan penelitian bertujuan untuk mengembangkan pendekatan deteksi malware di dalam infrastruktur *cloud computing* dengan menggunakan algoritma *Support Vector Machines (SVM)* satu kelas, teknik analisis statis dan dinamis, serta pendekatan adaptif untuk mengklasifikasi ancaman baru secara real-time dengan biaya komputasi yang minim. Para peneliti juga melakukan eksperimen dengan sampel malware, seperti Trojan.Kelihos-5, Trojan.Zbot-1433, Trojan.Zbot-1023, Trojan.Zbot-18, dan Trojan.Zbot-385 untuk menganalisis perilaku dan karakteristiknya di lingkungan cloud. Dalam penelitian ini, para peneliti menciptakan cara baru untuk mendeteksi aktivitas mencurigakan di tingkat server cloud menggunakan algoritma *Support Vector Machines (SVM)*. SVM ini seperti detektif pintar yang diajarkan untuk mengenali perangkat lunak jahat dan serangan. Hasil pengujian menunjukkan bahwa SVM ini sangat baik dalam mengenali perangkat lunak jahat dengan tingkat keakuratan di atas 90% [15].

Berdasarkan hasil penelitian yang dilakukan oleh *Deutsche Telekom Laboratories,* metode deteksi malware menggunakan pendekatan *Support Vector Machines (SVM)* satu kelas dijelaskan dengan rinci. SVM dilatih dengan data normal dan mampu mengklasifikasikan anomali secara real-time dengan biaya komputasi yang minim. Metode ini juga melibatkan teknik analisis statis dan dinamis serta seleksi fitur untuk membangun model deteksi malware. Hasil penelitian menunjukkan bahwa sistem ini dapat diandalkan, melatih dengan cepat, dan tahan terhadap kegagalan komponen. Dengan demikian, sistem ini terbukti efisien dalam mendeteksi anomali dalam jaringan seluler dengan keandalan dan ketahanan yang baik terhadap kegagalan komponen [16].

Sejumlah penelitian terdahulu telah menggambarkan aplikasi SVM dalam konteks keamanan siber dan deteksi malware. Sementara, penelitian oleh Turnip et al. (2020) fokus pada deteksi malware Android menggunakan teknik machine learning, termasuk SVM dan penggunaan eXtreme Gradient Boosting (XGBoost) dalam mendeteksi dan mengklasifikasikan malware berbasis Android berdasarkan kategori izin *(permission)* APK. [17]. Model ini digunakan untuk mengklasifikasikan enam kelas malware, seperti Trojan-Ransom, Trojan-SMS, Trojan-Banker, RiskTool, Trojan, dan lainnya. Secara keseluruhan, hasil evaluasi model menunjukkan bahwa model klasifikasi SVM memberikan kinerja lebih baik dengan F1-score sebesar 74,40% [17].

Sementara, penelitian yang dilakukan oleh Moh Yamin pada tahun 2014 membahas klasifikasi tuberkulosis menggunakan metode Support Vector Machine (SVM). Tujuan penelitian tersebut adalah untuk menemukan hyperplane terbaik yang dapat memisahkan dua kelas pada ruang input. Penulis menjelaskan bahaya penyakit tuberkulosis, terutama jika tidak diobati dengan cepat. Hasil klasifikasi menggunakan metode SVM menunjukkan tingkat akurasi sebesar 98%, membuktikan bahwa algoritma SVM memberikan solusi yang sangat baik dalam mengidentifikasi kasus *tuberculosis* [18].

Selanjutnya, penelitian dilakukan oleh Zuriel et.al (2021), mengklasifikasi Support Vector Machine untuk menganalisa sentimen pengguna Twitter terhadap kebijakan PSBB. Hasil penelitian ini menunjukkan bahwa performa model klasifikasi SVM dengan kernel RB. Model klasifikasi dengan kernel RBF ini memberikan mengklasifikasikan 11.764 (52.7%) data tweet ke dalam kelas positif dan 10.571 (47.3%) data tweet ke dalam kelas negative. Hasil tersebut memberikan kesimpulan bahwa pengguna Twitter cenderung bersentimen positif terhadap kebijakan PSBB [19]

Hasil dari penelitian-penelitian tersebut memberikan dukungan untuk penggunaan SVM dalam konteks deteksi malware, termasuk pada platform Android. Dengan merinci cara kerja SVM dan memahami penelitian terdahulu, proyek ini dapat memanfaatkan keunggulan algoritma ini untuk mengidentifikasi dan mengklasifikasikan jenis-jenis malware pada platform Android dengan lebih efektif.

* 1. **Tingkat Keberhasilan *Support Vector Machine Classification***

Implementasi SVM satu kelas dalam konteks deteksi malware telah membuktikan keunggulannya dengan mencapai tingkat keberhasilan yang luar biasa. Dengan pelatihan dan penyesuaian yang cermat, SVM mampu memberikan tingkat akurasi deteksi yang melebihi 90% untuk semua sampel malware, termasuk dalam kondisi analisis statis maupun migrasi. Keberhasilan ini menandakan kemampuan SVM dalam mengenali pola yang kompleks dan bervariasi yang terkait dengan aktivitas malware pada platform Android [14].

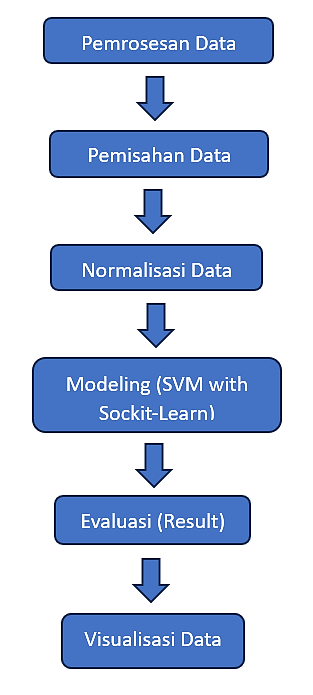
Lebih lanjut, keunggulan SVM tidak hanya terbatas pada akurasi tinggi, namun juga pada kemampuannya untuk mendeteksi anomali secara online dengan biaya waktu minimal. Dalam situasi di mana respons cepat terhadap ancaman malware sangat penting, SVM menunjukkan kinerja yang luar biasa dengan tingkat akurasi deteksi secara keseluruhan yang tetap melampaui 90% dalam kebanyakan kasus [14].

Kemampuan SVM dalam deteksi malware tidak hanya terletak pada tingkat keberhasilan tinggi, tetapi juga pada fleksibilitasnya dalam menangani berbagai kondisi dan metode analisis. SVM mampu mempertahankan kinerjanya yang unggul baik dalam situasi analisis statis, di mana karakteristik statis malware dianalisis tanpa eksekusi, maupun dalam skenario migrasi di mana perilaku malware dipantau selama aktivitas di dalam sistem.

Secara keseluruhan, prestasi SVM dalam mengklasifikasikan dan mendeteksi malware pada platform Android menciptakan landasan yang solid untuk pengembangan sistem keamanan yang andal. Keberhasilannya tidak hanya terletak pada tingkat akurasi tinggi, tetapi juga pada kemampuannya untuk beradaptasi dengan perubahan kondisi dan taktik yang digunakan oleh malware. Dengan mempertimbangkan kecepatan responsnya yang tinggi dan biaya waktu yang minimal dalam mendeteksi anomali secara online, SVM muncul sebagai pilihan yang kuat dalam upaya melawan ancaman malware yang terus berkembang .(Khalil dan Khammas, 2022).

1. **Metode**

Dalam rangka mengklasifikasikan malware ke dalam kategori malware tertentu, penelitian ini merancang sebuah model Machine Learning (ML) yang inovatif. Adapaun model klasifikasi Machine Learning yang digunakan pada penelitian ini adalah menggunakan Support Vector Machine (SVM). Dalam menerapkannya, terdapat tahapan-tahapan utama yang akan dijalankan untuk mengklasifikasikan Malware menggunakan Support Vector Machine (SVM). Adapun tahapan-tahapan tersebut mencakup pemrosesan data, pemisahan data, normalisasi data, SVM modeling *(SVM with scikit-learn),* evaluasi *(result),* dan visualisasi data.



**Gambar 1.** Diagram Klasifikasi Android Malware Menggunakan SVM Model

* 1. **Pemrosesan Data**

Pada tahap ini, data lalu lintas jaringan dari perangkat Android diproses untuk mendapatkan fitur-fitur yang dapat digunakan untuk membedakan antara lalu lintas normal dan lalu lintas malware. Fitur-fitur tersebut dapat berupa jenis protokol port yang digunakan, ukuran paket, dan sebagainya. Pemrosesan data pada tahap awal ini menjadi langkah kritis dalam konstruksi model klasifikasi malware menggunakan algoritma Support Vector Machine (SVM). Tujuan utamanya adalah untuk memastikan bahwa data yang digunakan memiliki kualitas yang baik dan sesuai sebelum diterapkan ke dalam model SVM. Oleh karena itu, dengan melakukan langkah pemrosesan data yang tepat, kita dapat secara signifikan mempengaruhi kualitas dan keandalan penemuan dan keputusan otomatis selanjutnya[20].

Proses pemrosesan data dapat dimulai dari pengambilan dataset yang tersedia dalam format csv. Untuk pengklasifikasian malware pada penelitian ini, dataset diambil dari website Kaggle yang berjudul “android\_traffic.csv”. Setelah itu, informasi dasar mengenai dataset tersebut akan ditampilkan guna memulai penerimaan wawasan awal tentang karakteristik dan struktur data yang akan diolah. Selanjutnya akan dilakukan penghapusan kolom yang tidak relevan atau tidak diinginkan agar dataset tetap terfokus pada fitur-fitur yang benar-benar berkontribusi pada pemodelan SVM. Pada pemrosesan data ini juga akan dilakukan penanganan nilai yang hilang untuk memastikan kebersihan dan integritas data.

* 1. **Pemisahan Data**

Setelah data diproses, data tersebut kemudian dibagi menjadi dua set, yaitu set pelatihan dan set pengujian. Set pelatihan digunakan untuk melatih model SVM, sedangkan set pengujian digunakan untuk mengevaluasi kinerja model. Pemisahan data adalah desain studi yang banyak digunakan dalam pengaturan dimensi tinggi dengan membagi kumpulan data menjadi set pelatihan dan set uji sebagai sarana untuk memperkirakan akurasi klasifikasi. Pengklasifikasi dikembangkan pada set pelatihan dan diterapkan pada setiap sampel dalam set uji [21]. Metode pemisahan yang digunakan untuk membagi data menjadi set pelatihan dan set pengujian pada penelitian ini adalah dengan menggunakan salah satu library pada aplikasi machine learning yang digunakan. Library ini bernama “sklearn.model\_selection”, library ini menyediakan berbagai fungsi untuk membagi data menjadi set pelatihan dan pengujian. Salah satu fungsinya ada fungsi “train\_test\_split”, yang secara acak mempartisi data menjadi dua subset dengan proporsi yang ditentukan oleh parameter “test\_size”. Seperti yang sudah dijelaskan sebelumnya, parameter “test\_size” akan menentukan proporsi data yang akan dialokasikan ke set pengujian. Selain parameter ini, terdapat juga parameter yang bernama “random\_state”. Parameter ini akan menentukan seed acak untuk proses pemisahan data.

Pemisahan data menjadi set pelatihan dan set pengujian tidak hanya terletak pada kemampuan model untuk mempelajari data pelatihan, tetapi juga pada kemampuannya untuk menggeneralisasi dan mengklasifikasikan data yang belum pernah dilihat sebelumnya. Dengan ukuran set pelatihan yang memadai, distribusi kelas yang serupa, dan metode pembagian data yang tepat, model SVM dapat meningkatkan akurasi, menghindari overfitting, dan memberikan evaluasi kinerja yang handal pada set pengujian [22].

* 1. **Normalisasi Data**

Pada tahap ini, data dalam set pelatihan dan set pengujian dinormalisasi agar nilainya berada dalam skala yang sama. Hal ini dilakukan agar model SVM dapat bekerja dengan lebih efektif. Proses normalisasi data ditandai dengan mengubah skala data sehingga semua fitur memiliki nilai yang mirip. Normalisasi ini penting untuk dilakukan karena dapat meningkatkan kinerja model SVM. SVM dapat sensitif terhadap skala data, sehingga jika data tidak ternormalisasi, maka model SVM dapat belajar dengan lebih baik pada fitur-fitur yang memiliki skala yang lebih besar [23]. Ada beberapa metode yang dapat digunakan untuk normalisasi data. Salah satu metode yang paling umum adalah normalisasi Min-Max. Metode ini mengubah nilai setiap fitur sehingga nilainya berada dalam rentang tertentu [24].

Dalam mengenali pentingnya normalisasi data, metode normalisasi Min-Max sering kali menjadi pilihan yang umum. Metode ini mengubah nilai setiap fitur sehingga nilainya terdapat dalam rentang tertentu, contohnya dari 0 hingga 1. Langkah-langkah konkret untuk menerapkan normalisasi Min-Max melibatkan perhitungan nilai minimum dan maksimum untuk setiap fitur, kemudian mengurangkan setiap nilai fitur dengan nilai minimumnya, dan membagi setiap nilai fitur dengan selisih nilai maksimum dan minimumnya.

(1)

Keterangan:

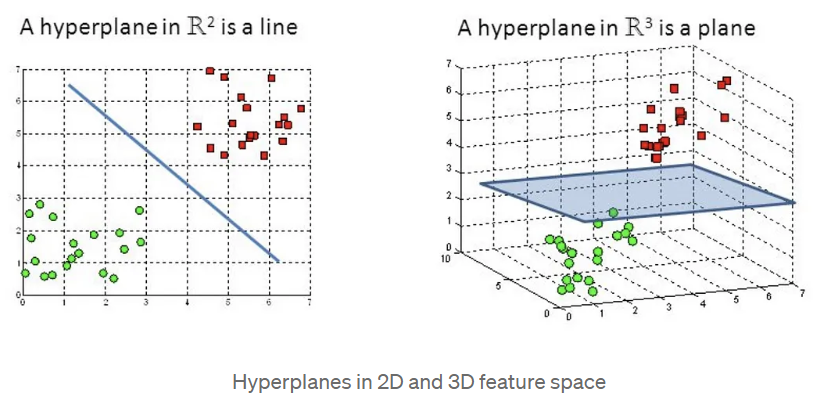
|  |  |  |
| --- | --- | --- |
| Real value | : | Nilai asli dari variabel yang akan dinormalisasi |
| Min value | : | Nilai minimum dari variabel |
| Max value | : | Nilai maksimum dari variable |
| Normalisasi value | : | Nilai data yang telah dinormalisasi |

* 1. **Model *Support Vector Machine (SVM)***
     1. **Penentuan Rentang Nilai Parameter untuk Penalaan Hiperparameter**

Dalam membangun model Support Vector Machine (SVM), langkah pertama yang kritis adalah menentukan rentang nilai parameter yang akan diuji untuk penalaan hiperparameter. Pada tahap ini, model SVM dilatih menggunakan set pelatihan. Model SVM akan mempelajari hubungan antara fitur-fitur data dan kelasnya (normal atau malware). Model SVM yang digunakan dalam penelitian ini adalah Support Vector Machine (SVM) dengan kernel *Radial Basis Function (RBF).* Hiperparameter pada SVM melibatkan elemen-elemen seperti berikut ini :

* 1. *Hyperplane*

Untuk memahami logika SVM, pemahaman akan *Hyperplane* sangat ditekankan. Adapun demikian, Hyperplane dalam ruang Euclidean berdimensi n adalah himpunan bagian datar berdimensi n-1 dari ruang tersebut yang membagi ruang menjadi dua bagian yang tidak terhubung [x]. Untuk dua dimensi dapat dilihat bahwa garis pemisahnya adalah *hyperplane*. Demikian pula, untuk tiga dimensi, sebuah bidang dengan dua dimensi membagi ruang 3d menjadi dua bagian dan dengan demikian bertindak sebagai bidang hiper. Jadi untuk ruang berdimensi n akan dimiliki bidang hiper berdimensi n-1 yang memisahkannya menjadi dua bagian.



Gambar 2. Hyperplanes di tampilan 2D dan 3D

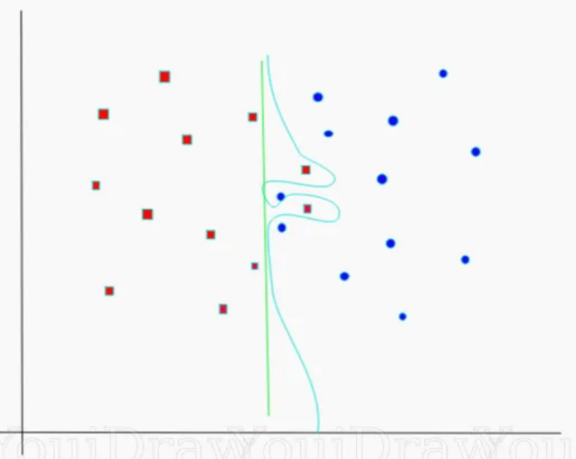
Sumber: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>

* 1. Turning Parameters

Parameter adalah argumen yang digunakan untuk membuat pengklasifikasian. Terdapat 2 jenis parameter yaitu :

1. C

Parameter C berfungsi untuk mengontrol *trade off* antara batas keputusan yang mulus dan mengklasifikasikan poin pelatihan dengan benar. Nilai c yang besar berarti model akan mendapatkan lebih banyak poin pelatihan dengan benar.



Gambar 3. Batas Keputusan dan Klasifikasi Poin Pelatihan Hyperplane

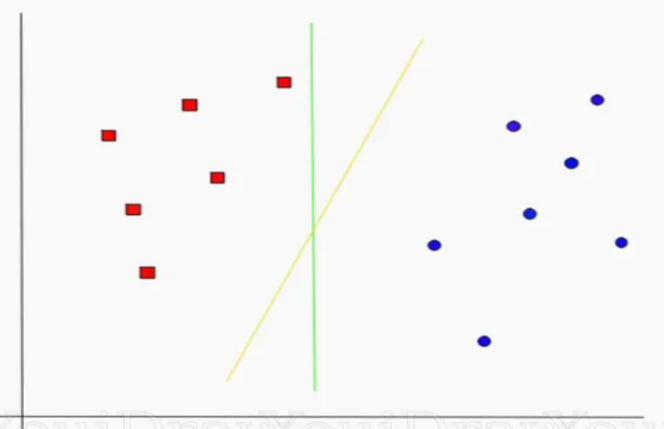
Sumber: <https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989>

1. Gamma

Parameter Gamma berfungsi untuk menentukan seberapa jauh pengaruh satu contoh pelatihan. Jika nilainya rendah berarti setiap titik mempunyai jangkauan yang jauh dan sebaliknya nilai gamma yang tinggi berarti setiap titik mempunyai jangkauan yang dekat. Dengan kata lain, jika nilai gamma yang dihasilkan rendah menunjukkan bahwa sampel data yang jauh dari garis pemisah memiliki pengaruh yang signifikan dalam perhitungan untuk menentukan posisi garis pemisah. Sebaliknya, jika nilai Gamma tinggi, sampel data yang dekat dengan garis pemisah akan memiliki pengaruh yang lebih besar dalam proses perhitungan untuk menentukan garis pemisah yang optimal. Besarnya nilai Gamma yang dihasilkan memengaruhi seberapa ketat atau luas garis pemisah dalam klasifikasi data [25].

* + 1. **Pembuatan Model SVM dan Penggunaan Grid Search**

Setelah rentang nilai parameter ditentukan, langkah selanjutnya adalah membangun model SVM dan menggunakan Grid Search untuk menemukan kombinasi parameter terbaik. Grid Search adalah metode pencarian sistematis yang mengevaluasi kombinasi berbagai nilai hiperparameter untuk menemukan kombinasi yang memberikan kinerja optimal. Pada tahap ini, kita dapat menggunakan modul GridSearchCV dari pustaka Scikit-learn untuk secara otomatis melakukan pencarian parameter terbaik.

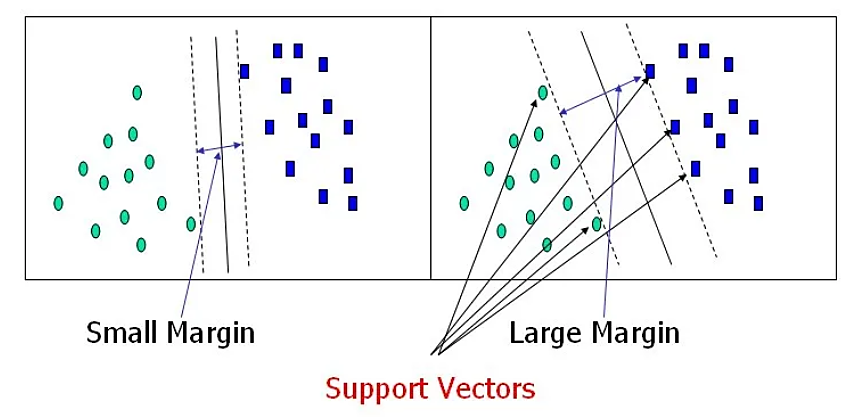


Gambar 4. Visual Pembagian data Dimensi tinggi

Sumber: <https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989>

* + 1. **Inisialisasi dan Pelatihan Model SVM dengan Parameter Terbaik**

Setelah Grid Search selesai, kita dapat mengakses parameter terbaik yang ditemukan dan menginisialisasi ulang model SVM menggunakan parameter tersebut. Langkah ini memastikan bahwa model yang dibangun mengoptimalkan kinerja berdasarkan parameter terbaik yang diidentifikasi.



Gambar 5. Optimal Hyperplane using the SVM Algorithm

Sumber: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>

* 1. **Evaluasi (Results)**

Selanjutnya, kinerja model SVM dievaluasi menggunakan berbagai metrik, seperti akurasi, presisi, recall, dan f1-score.

* + 1. **Kriteria Evaluasi Model**

Setelah berhasil melatih model SVM dengan parameter terbaik yang diidentifikasi melalui *Grid Search*, langkah selanjutnya adalah mengevaluasi kinerjanya. Evaluasi model adalah tahap kritis untuk memahami seberapa baik model dapat menggeneralisasi informasi dari data pelatihan ke data pengujian. Beberapa metrik evaluasi yang dapat digunakan mencakup akurasi, matriks kebingungan, dan laporan klasifikasi.

* + 1. **Pengukuran Akurasi**

Pada tahap ini dilakukan pengukuran akurasi untuk mengukur sejauh mana model dapat memprediksi kelas dengan benar. Dengan kata lain, mengindikasikan sejauh mana suatu teknik klasifikasi dapat memberikan hasil yang tepat. Semakin tinggi nilai akurasi klasifikasi, semakin baik pula kinerja teknik klasifikasi tersebut secara keseluruhan. Dengan kata lain, tingkat akurasi mencerminkan seberapa tepat suatu metode dapat mengklasifikasikan data, dan semakin tinggi nilai akurasi, semakin baik pula performansi teknik klasifikasi yang digunakan [26]. Akurasi dihitung sebagai rasio prediksi yang benar (positif dan negatif) terhadap jumlah total data. Namun, untuk masalah klasifikasi yang tidak seimbang, akurasi saja mungkin tidak mencerminkan kinerja yang sebenarnya.

* + 1. **Matriks Kebingungan *(Confusion Matrix)***

Ketika hasil pengukuran data dan pemodelan SVM telah dilakukan, maka selanjutnya akan diidentifikasi *confusion matrix.* Matriks kebingungan ini menyajikan informasi tentang seberapa baik model dapat memprediksi setiap kelas [13]. Matriks ini terdiri dari empat elemen: *True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN)*. Dengan menggunakan matriks kebingungan, kita dapat menghitung metrik seperti presisi, recall, dan tingkat kesalahan tipe I dan II pada klasifikasi malware berdasarkan implementasi algoritma SVM.

* + 1. **Evaluasi Klasifikasi**

Laporan klasifikasi memberikan ringkasan komprehensif dari berbagai metrik evaluasi untuk setiap kelas. Ini mencakup presisi, recall, dan F1-score,

1. Presisi adalah metrik yang mengukur sejauh mana prediksi positif yang dibuat oleh model adalah benar.

Rumus presisi:

(2)

Keterangan:

|  |  |  |
| --- | --- | --- |
| True Positives | : | Jumlah instance yang benar-benar termasuk dalam kelas yang diinginkan dan diprediksi dengan benar oleh model. |
| False Positives | : | Jumlah instance yang sebenarnya bukan termasuk dalam kelas yang diinginkan, tetapi diprediksi sebagai anggota kelas tersebut oleh model. |

Presisi memberikan informasi tentang seberapa baik model mengidentifikasi instance yang sebenarnya termasuk dalam kelas tersebut.

1. *Recall* mengukur sejauh mana model mampu mengidentifikasi semua *instance* positif yang sebenarnya.

Rumus recall

(3)

Keterangan:

|  |  |  |
| --- | --- | --- |
| True Positives | : | Jumlah instance yang benar-benar termasuk dalam kelas yang diinginkan dan diprediksi dengan benar oleh model. |
| False Negatives | : | Jumlah instance yang sebenarnya bukan termasuk dalam kelas yang diinginkan, tetapi diprediksi sebagai anggota kelas tersebut oleh model. |

Recall memberikan informasi tentang seberapa baik model dapat menangkap semua instance positif yang sebenarnya.

1. F1-*score* adalah metrik gabungan yang menggabungkan presisi dan recall dalam satu nilai.

Rumus F1-s*core*

(4)

Keterangan:

|  |  |  |
| --- | --- | --- |
| Precision | : | Metrik yang mengukur sejauh mana prediksi positif yang dibuat oleh model adalah benar. |
| Recall | : | Metrik yang mengukur sejauh mana model mampu mengidentifikasi semua instance positif yang sebenarnya. |

F1-score memberikan keseluruhan ukuran kinerja model dengan mempertimbangkan kedua *false positives* dan *false negatives*. F1-score merupakan harmonic mean dari presisi dan recall. Karena menggunakan harmonic mean, F1-score cenderung memberikan bobot lebih besar pada nilai yang lebih rendah dari kedua metrik tersebut. Oleh karena itu, F1-score baik digunakan ketika keseimbangan antara presisi dan recall diinginkan, terutama dalam situasi di mana ada ketidakseimbangan antara kelas positif dan negatif.

Nilai F1-score berkisar antara 0 dan 1, di mana nilai 1 menunjukkan kinerja model yang sempurna, dan nilai 0 menunjukkan kinerja yang sangat buruk. F1-score berguna untuk menyatukan informasi presisi dan recall menjadi satu nilai yang dapat memberikan gambaran holistik tentang kemampuan model dalam tugas klasifikasi.

* + 1. **Analisis Hasil dan Penyempurnaan Model**

Setelah mendapatkan metrik evaluasi, langkah berikutnya adalah melakukan analisis mendalam terhadap hasilnya. Dengan mendetail dalam menganalisis hasil evaluasi, dapat diambil kesimpulan untuk memahami kekuatan dan kelemahan model SVM yang telah dikembangkan. Ini memberikan landasan untuk mengambil tindakan selanjutnya, baik itu peningkatan model atau penerapannya dalam skenario produksi. Evaluasi yang cermat adalah kunci untuk memastikan bahwa model yang dikembangkan dapat memberikan solusi yang andal dan efektif terhadap masalah klasifikasi malware.

* 1. **Visualisasi Data**

Pada tahap ini, data divisualisasi untuk membantu memahami karakteristik data dan kinerja model SVM. Visualisasi data dapat dilakukan dengan menggunakan berbagai teknik, seperti grafik, histogram, dan sebagainya.Visualisasi data dapat membantu peneliti untuk memahami pola yang ada dalam data dan untuk menganalisis kinerja model SVM.

* + 1. **Matriks Kebingungan *(Confusion Matrix)***

Matriks konfusi merupakan alat evaluasi kinerja dalam pembelajaran mesin, yang mewakili keakuratan model klasifikasi. Matriks ini menampilkan jumlah positif benar, negatif benar, positif palsu, dan negatif palsu. Model matriks ini akan menganalisis kinerja model, mengidentifikasi kesalahan klasifikasi, dan meningkatkan akurasi prediksi (Aniruddha,2023) [x].

Matriks Kebingungan dapat divisualisasikan untuk memberikan gambaran yang lebih intuitif tentang kinerja model. Visualisasi ini dapat membantu dalam memahami sejauh mana model mampu membedakan antara kelas-kelas tertentu. Misalnya, visualisasi ini dapat menggunakan pustaka seperti `seaborn` dan `matplotlib` untuk membuat heatmap matriks kebingungan:

import seaborn as sns

import matplotlib.pyplot as plt

# Membuat heatmap matriks kebingungan

sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues', cbar=False,

xticklabels=['Prediksi Negatif', 'Prediksi Positif'],

yticklabels=['Aktual Negatif', 'Aktual Positif'])

plt.xlabel('Prediksi')

plt.ylabel('Aktual')

plt.title('Matriks Kebingungan')

plt.show()

Visualisasi ini memberikan representasi grafis tentang seberapa baik model dapat mengklasifikasikan data ke dalam kategori positif dan negatif.

* + 1. **Kurva ROC *(Receiver Operating Characteristic)***

Kurva karakteristik operasi penerima (ROC) merupakan grafik yang memetakan sensitivitas uji terhadap spesifisitas 1 atau tingkat positif palsu (FPR) terhadap tingkat negatif palsu sebagai koordinat x. ROC digunakan sebagai metode yang efektif dalam mengevaluasi kinerja uji diagnostik. (Park, 2004).

Kurva Karakteristik Operasi Penerima (ROC) adalah alat visualisasi yang digunakan untuk menggambarkan kinerja model pada berbagai tingkat ambang batas *(threshold)* untuk klasifikasi. ROC menggambarkan trade-off antara Tingkat Positif Benar *(True Positive Rate)* dan Tingkat Positif Salah *(False Positive Rate).* Kurva ROC memberikan gambaran visual tentang seberapa baik model dapat membedakan antara kelas positif dan negatif. Area di bawah kurva ROC (AUC) adalah metrik yang berguna semakin besar nilai AUC, semakin baik kinerja model.

from sklearn.metrics import roc\_curve, auc

# Menghitung nilai prediksi probabilitas positif

y\_prob = best\_svm\_model.predict\_proba(X\_test)[:, 1]

# Menghitung nilai False Positive Rate (FPR) dan True Positive Rate (TPR) untuk berbagai ambang batas

fpr, tpr, thresholds = roc\_curve(y\_test, y\_prob)

# Menghitung area di bawah kurva ROC (AUC)

roc\_auc = auc(fpr, tpr)

# Membuat kurva ROC

plt.figure(figsize=(8, 6))

plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'AUC = {roc\_auc:.2f}')

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate (FPR)')

plt.ylabel('True Positive Rate (TPR)')

plt.title('Kurva ROC')

plt.legend(loc="lower right")

plt.show()

1. **Hasil Pengujian**

<Jabarkan hasil pengujian terhadap metode yang telah Anda implementasikan.>.

|  |
| --- |
| ## Necessary Imports  import matplotlib.pyplot as plt  import pandas as pd  import numpy as np  import matplotlib.pyplot as plt |

|  |
| --- |
| ## About Android Traffic Dataset |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

1. **Analisis**

<Jabarkan hasil analisis Anda terhadap hasil pengujian>.

1. **Kesimpulan**

<Tuliskan apakah sistem kecerdasan buatan selesai dibangun dan apakah dapat menyelesaikan masalah yang ditulis pada latar belakang. Tuliskan saran pengembangan.>

1. **Pembagian Pekerjaan**

**Tabel 2.** Pembagian Tugas Anggota Kelompok

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **NIM** | **Nama** | **Peran** | **Tugas** |
| 1. | 12S21027 | Rebecca Yulyartha Bulawan Sihombing | Ketua | Bab III Metode Penelitian, Bab IV Implementasi |
| 2. | 12S21037 | Immanuella Eklesia Lumbantobing | Anggota | Bab IV Implementasi |
| 3. | 12S21039 | Widya Indah Sari Manurung | Anggota | Bab III Metode Penelitian |
| 4. | 12S21053 | Chesya Ivana J.M Sitorus | Anggota | Bab II Studi Literatur, Bab IV Analisis Hasil Pengujian |
| 5. | 12S21058 | Grace Christina Yohanna Situmorang | Anggota | Bab I Pendahuluan, Bab IV Implementasi |

**REFERENSI**

[1] G. R. Kanagachidambaresan, A. Ruwali, D. Banerjee, and K. B. Prakash, “Klasifikasi Malware Menggunakan MetodeRecurrent Neural Network,” *EAI/Springer Innov. Commun. Comput.*, vol. 23, no. 3, pp. 53–61, 2021.

[2] Hafiz and Meidi Dwi, “Visualisasi dan Klasifikasi Malware Menggunakan Metode K-Nearest Neighbor (K-NN). ,” *Indralaya Univ. Sriwij.*, 2020, [Online]. Available: https://repository.unsri.ac.id/39918/18/RAMA\_56201\_09011281520097\_0 003047905\_01\_front\_ref.pdf.

[3] Y. Ilhamdi and Y. N. Kunang, “Analisis Malware Pada Sistem Operasi Windows Menggunakan Teknik Forensik,” *Bina Darma Conf. Comput. Sci.*, vol. 3, pp. 256–264, 2021, [Online]. Available: https://conference.binadarma.ac.id/index.php/BDCCS/article/view/2124.

[4] B. Guc *et al.*, “Semantic annotation of GPS trajectories,” *11th Agil. Int. Conf. Geogr. Inf. Sci. 2008*, vol. 38, no. January 2008, pp. 1–9, 2008.

[5] F. C. Venna, “Implementasi Steganografi Audio pada File Wav dengan metode Redundant Pattern Encoding (RPE) Berbasis Sndroid,” *Repository.Uinjkt.Ac.Id*, 2019, [Online]. Available: http://repository.uinjkt.ac.id/dspace/handle/123456789/47958.

[6] M. F. Syawal, D. C. Fikriansyah, and N. Agani, “Implementasi Teknik Steganografi Menggunakan Algoritma Vigenere Cipher Dan Metode LSB,” *J. TICOM*, vol. 4, no. 3, pp. 91–99, 2016.

[7] N. K. Gyamfi, N. Goranin, D. Ceponis, and H. A. Čenys, “Automated System-Level Malware Detection Using Machine Learning: A Comprehensive Review,” *Appl. Sci.*, vol. 13, no. 21, p. 11908, 2023, doi: 10.3390/app132111908.

[8] E. S. Lamdompak Sistem Komputer and F. Ilmu Komputer, “Klasifikasi Malware Trojan Ransomware Dengan Algoritma Support Vector Machine (SVM),” vol. 2, no. 1, pp. 122–127, 2016, [Online]. Available: http://ars.ilkom.unsri.ac.id.

[9] D. I. Pushpita Anna Octaviani, Yuciana Wilandari, “Penerapan Metode SVM Pada Data Akreditasi Sekolah Dasar Di Kabupaten Magelang,” *J. Gaussian*, vol. 3, no. 8, pp. 811–820, 2014.

[10] D. Zhang, H. Huang, Q. Chen, and Y. Jiang, “A Comparison Study of Credit Scoring Models,” *Proc. - Third Int. Conf. Nat. Comput. ICNC 2007*, vol. 1, no. June, pp. 15–18, 2007, doi: 10.1109/ICNC.2007.15.

[11] F. Rachman and Santi Wulan Purnami, “Perbandingan Klasifikasi Tingkat Keganasan Breast Cancer Dengan Menggunakan Regresi Logistik Ordinal Dan Support Vector Machine (SVM),” *Chest*, vol. 73, no. 2 suppl., pp. 293–299, 1978, doi: 10.1378/chest.73.2\_supplement.293.

[12] E. V. Tjahjadi and B. Santoso, “Klasifikasi Malware Menggunakan Teknik Machine Learning,” *J. Ilm. Ilmu Komput.*, vol. 2, no. 1, pp. 60–70, 2023.

[13] A. S. Nugraha and K. K. Purnamasari, “Penerapan Metode Support Vector Machine Pada Part of Speech Tag Bahasa Indonesia,” no. 112, 2019.

[14] N. A. Khalil and B. M. Khammas, “an Effective and Efficient Features Vectors for Ransomware Detection Via Machine Learning Technique,” *Iraqi J. Inf. Commun. Technol.*, vol. 5, no. 3, pp. 23–33, 2022, doi: 10.31987/ijict.5.3.205.

[15] O. Article, “Malware Detection in Cloud Computing Infrastructures Manuscript,” *Environ. Educ. Res.*, 2011.

[16] A. S. Shamili, C. Bauckhage, and T. Alpcan, “Malware detection on mobile devices using distributed machine learning,” *Proc. - Int. Conf. Pattern Recognit.*, pp. 4348–4351, 2010, doi: 10.1109/ICPR.2010.1057.

[17] T. N. Turnip, A. Situmorang, A. Lumbantobing, J. Marpaung, and S. I. G. Situmeang, “Android malware classification based on permission categories using extreme gradient boosting,” *ACM Int. Conf. Proceeding Ser.*, no. November, pp. 190–194, 2020, doi: 10.1145/3427423.3427427.

[18] M. Y. Darsyah, “KLASIFIKASI TUBERKULOSIS DENGAN PENDEKATAN METODE SUPPORTS VECTOR MACHINE (SVM),” *J. Kesehat.*, vol. 10, no. 3, p. 405, 2019, doi: 10.26630/jk.v10i3.1479.

[19] H. P. P. Zuriel and A. Fahrurozi, “Implementasi Algoritma Klasifikasi Support Vector Machine Untuk Analisa Sentimen Pengguna Twitter Terhadap Kebijakan Psbb,” *J. Ilm. Inform. Komput.*, vol. 26, no. 2, pp. 149–162, 2021, doi: 10.35760/ik.2021.v26i2.4289.

[20] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, “A survey on data preprocessing for data stream mining: Current status and future directions,” *Neurocomputing*, vol. 239, pp. 39–57, 2017, doi: 10.1016/j.neucom.2017.01.078.

[21] K. K. Dobbin and R. M. Simon, “Optimally splitting cases for training and testing high dimensional classifiers,” *BMC Med. Genomics*, vol. 4, 2011, doi: 10.1186/1755-8794-4-31.

[22] B. Raharjo, *Pembelajaran Mesin (Machine Learning)*. YAYASAN PRIMA AGUS TEKNIK, 2016.

[23] P. Adytia, W. Wahyuni, K. Sussolaikah, and Y. Satria, “Klasifikasi Penggunaan Data Trafik Internet Menggunakan Algoritma Support Vector Machine,” *J. Komput. dan Inform.*, vol. 11, no. 1, pp. 96–102, 2023, doi: 10.35508/jicon.v11i1.10039.

[24] W. Agustina, M. T. Furqon, and B. Rahayudi, “Implementasi Metode Support Vector Machine (SVM) Untuk Klasifikasi Rumah Layak Huni (Studi Kasus: Desa Kidal Kecamatan Tumpang Kabupaten Malang),” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 10, pp. 3366–3372, 2018, [Online]. Available: http://j-ptiik.ub.ac.id.

[25] D. D. D. Kemala, “PERBANDINGAN EMPIRIS FUNGSI KERNEL PADA METODE KLASIFIKASI SUPPORT VECTOR MACHINE DAN PENERAPANNYA PADA DATA PENDERITA PENYAKIT JANTUNG,” pp. 1–14, 2023, [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK558907/.

[26] D. H. Anto Satriyo Nugroho, Arief Budi Witarto, “Support Vector Machine - Teori dan Aplikasinya dalam Bioinformatika1,” *Proc. 2011 Chinese Control Decis. Conf. CCDC 2011*, pp. 842–847, 2011, doi: 10.1109/CCDC.2011.5968300.

[x] Rushiskesh Pupale (2018, Juli 16). *Support Vector Machines(SVM) An Overview* [Towards Data Science]. Available: <https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989>

[x] R. Pupale (2018, Juli 16). *Support Vector Machines (SVM) An Overview* [Online]. Available: <https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989>

[x] A. Bhandari (2023, November 2*). Understanding & Interpreting Confusion Matrix in Machine Learning* [Online]. Available: <https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/>

**LAMPIRAN**

<Opsional.>