**IMPLEMENTASI ALGORITMA *NEURAL NETWORK* PADA SISTEM DETEKSI *MALWARE* ANDROID**

**SKRIPSI**

****

**Oleh :**

**MOCHAMAD FERRIAN MULYADI**

**191011400256**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PAMULANG**

**2023**

# IMPLEMENTASI ALGORITMA *NEURAL NETWORK* PADA SISTEM DETEKSI *MALWARE* ANDROID

SKRIPSI

Diajukan Untuk Melengkapi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer



Oleh:

MOCHAMAD FERRIAN MULYADI

191011400256

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PAMULANG**

**2023**

# LEMBAR PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : MOCHAMAD FERRIAN MULYADI

NIM : 191011400256

Program Studi : Teknik Informatika

Fakultas : Ilmu Komputer

Jenjang Pendidikan : Strata 1

Menyatakan bahwa skripsi yang saya buat dengan judul:

IMPLEMENTASI ALGORITMA *NEURAL* *NETWORK* PADA SISTEM DETEKSI *MALWARE* ANDROID

1. Merupakan hasil karya tulis ilmiah sendiri, bukan merupakan karya yang pernah diajukan untuk memperoleh gelar akademik oleh pihak lain, dan bukan merupakan hasil plagiat.
2. Saya ijinkan untuk dikelola oleh Universitas Pamulang sesuai dengan norma hukum dan etika yang berlaku.

Pernyataan ini saya buat dengan penuh tanggung jawab dan saya bersedia menerima konsekuensi apa pun sesuai aturan yang berlaku apabila dikemudian hari pernyataan ini tidak benar.

Tangerang Selatan, September 2023

Materai 6000

(Mochamad Ferrian Mulyadi)

# LEMBAR PERSETUJUAN

NIM : 191011400256

Nama : MOCHAMAD FERRIAN MULYADI

Program Studi : TEKNIK INFORMATIKA

Fakultas : ILMU KOMPUTER

Jenjang Pendidikan : STRATA 1

Judul Skripsi : IMPLEMENTASI ALGORITMA

NEURAL NETWORK PADA SISTEM

DETEKSI MALWARE ANDROID

Skripsi ini telah diperiksa dan disetujui oleh pembimbing untuk persyaratan sidang skripsi.

Tangerang Selatan, September 2023

|  |
| --- |
| Pembimbing |
|  |
| Nurjaya, S.Kom., M.Kom  NIDN : 0405078502 |
| Mengetahui, |
| Ketua Program Studi |
|  |
| Achmad Udin Zailani, S.Kom., M.Kom  NIDN : 0429058303 |

# LEMBAR PENGESAHAN

NIM : 191011400256

Nama : MOCHAMAD FERRIAN MULYADI

Program Studi : TEKNIK INFORMATIKA

Fakultas : ILMU KOMPUTER

Jenjang Pendidikan : STRATA 1

Judul Skripsi : IMPLEMENTASI ALGORITMA NEURAL NETWORK PADA SISTEM DETEKSI MALWARE ANDROID

Pamulang, September 2023

Penguji I Penguji II

Nurjaya, S.Kom., M.Kom.

Pembimbing

Mengetahui,

Achmad Udin Zailani, S.Kom., M.Kom

Kaprodi Teknik Informatika

# DAFTAR ISI

[LEMEBAR PERSETUJUAN 3](#_Toc144586287)

[LEMBAR PENGESAHAN 4](#_Toc144586288)

[DAFTAR ISI 5](#_Toc144586289)

[DAFTAR GAMBAR 8](#_Toc144586290)

[DAFTAR TABLE 9](#_Toc144586291)

[ABSTRAK 10](#_Toc144586292)

[KATA PENGANTAR 11](#_Toc144586293)

[BAB I PENDAHULUAN 12](#_Toc144586294)

[1.1 Latar belakang masalah 12](#_Toc144586295)

[1.2 Identifikasi masalah 14](#_Toc144586296)

[1.3 Rumusan Masalah 14](#_Toc144586297)

[1.4 Batasan Masalah 15](#_Toc144586298)

[1.5 Tujuan penelitian 15](#_Toc144586299)

[1.6 Manfaat Penelitian 15](#_Toc144586300)

[1.7 Metodologi Penelitian 16](#_Toc144586301)

[1.8 Sistematika Penuliasan 16](#_Toc144586302)

[BAB II TINJUAN PUSTAKA DAN LANDASAN TEORI 18](#_Toc144586303)

[2.1 Tinjauan Pustaka 18](#_Toc144586304)

[2.2 Landasan Teori 22](#_Toc144586305)

[2.2.1 Data *Mining* 22](#_Toc144586306)

[2.2.2 *Machine Learning* 25](#_Toc144586307)

[2.2.3 Python 26](#_Toc144586308)

[2.2.4 Library Python 27](#_Toc144586309)

[2.2.5 Neural Network 29](#_Toc144586310)

[2.2.6 Pengujian Model 30](#_Toc144586311)

[2.2.7 Aplikasi Pendukung 31](#_Toc144586312)

[2.2.8 Kerangka Pemikiran 32](#_Toc144586313)

[2.3 Tinjauan Objek 32](#_Toc144586314)

[2.3.1 *Android* 32](#_Toc144586315)

[2.3.2 *Malware* 33](#_Toc144586316)

[BAB III METODE PEELITIAN 34](#_Toc144586317)

[3.1 Analisa Kebutuhan 34](#_Toc144586318)

[3.1.1 Teknik Analisis 34](#_Toc144586319)

[3.1.2 Pengumpulan Data (*Dataset*) 34](#_Toc144586320)

[3.1.3 Pengolahan Data Awal 52](#_Toc144586321)

[3.2 Perancangan Penelitian 59](#_Toc144586322)

[BAB IV IMPLEMENTASI DAN PENGUJIAN 63](#_Toc144586323)

[4.1 Hasil 63](#_Toc144586324)

[4.1.1 Persiapan Data 63](#_Toc144586325)

[4.1.2 Implementasi Metode 63](#_Toc144586326)

[4.2 Pembahasan 63](#_Toc144586327)

[4.2.1 Pengolahan Data 63](#_Toc144586328)

[4.2.2 Eksplorasi Data 63](#_Toc144586329)

[4.2.3 Pembersihan data 93](#_Toc144586330)

[4.2.4 *Encoding* (*Feature* *Engineering*) 94](#_Toc144586331)

[4.2.5 Membangun Model 95](#_Toc144586332)

[4.2.6 Pengujian Model 97](#_Toc144586333)

[4.3 Evaluasi dan Validasi hasil 98](#_Toc144586334)

[BAB V KESIMPULAN DAN SARAN 99](#_Toc144586335)

[5.1 Kesimpulan 99](#_Toc144586336)

[5.2 Saran 99](#_Toc144586337)

[BAB VI DAFTAR PUSTAKA 100](#_Toc144586338)

[BAB VII DAFTAR RIWAYAT HIDUP 104](#_Toc144586339)

# DAFTAR GAMBAR

[Gambar 2. 1 Kerangka Pemikiran 33](#_Toc144844336)

[Gambar 3. 1 Rancangan Penelitian 61](#_Toc144844399)

[Gambar 4. 1 Korelasi Atribut 79](#_Toc144844407)

[Gambar 4. 2 *Class* 80](#_Toc144844408)

[Gambar 4. 3 Transact 81](#_Toc144844409)

[Gambar 4. 4 *Transact vs Class* 81](#_Toc144844410)

[Gambar 4. 5 *Android.os.binder* 82](#_Toc144844411)

[Gambar 4. 6 Android.os.binder vs class 83](#_Toc144844412)

[Gambar 4. 7 *Onserviceconnected* 84](#_Toc144844413)

[Gambar 4. 8 *Onserviceconnected vs Class* 84](#_Toc144844414)

[Gambar 4. 9 *Bindservice* 85](#_Toc144844415)

[Gambar 4. 10 *Bindservice vs Class* 86](#_Toc144844416)

[Gambar 4. 11 *Serviceconnection* 87](#_Toc144844417)

[Gambar 4. 12 Serviceconnection vs class 87](#_Toc144844418)

[Gambar 4. 13 *Write\_History\_Bookmark* 88](#_Toc144844419)

[Gambar 4. 14 *Write\_History\_Bookmarks vs Class* 89](#_Toc144844420)

[Gambar 4. 15 *Read\_History\_Bookmarks* 90](#_Toc144844421)

[Gambar 4. 16 *Read\_History\_Bookmarks vs Class* 90](#_Toc144844422)

[Gambar 4. 17 Send\_sms 91](#_Toc144844423)

[Gambar 4. 18 *Send\_sms vs Class* 92](#_Toc144844424)

[Gambar 4. 19 Read\_sms 93](#_Toc144844425)

[Gambar 4. 20 *Read\_sms vs Class* 93](#_Toc144844426)

[Gambar 4. 21 *Telephonymanager.getline1number* 94](#_Toc144844427)

[Gambar 4. 22 Telephonymanager.getline1number vs Class 95](#_Toc144844428)

[Gambar 4. 23 Model *Neural Network\* 98](#_Toc144844429)

[Gambar 4. 24 *Confusion* *Matrix* 99](#_Toc144844430)

# DAFTAR TABEL

[Tabel 2. 1 Tinjauan Studi 20](#_Toc144844150)

[Tabel 3. 1 Kebutuhan Perangkat Keras dan Perangkat Lunak 34](#_Toc144844159)

[Tabel 3. 2 Pengumpulan Data 35](#_Toc144844160)

[Tabel 3. 3 Pengolahan Data Awal 52](#_Toc144844161)

[Tabel 3. 4 Pengolahan Data 60](#_Toc144844162)

[Tabel 3. 5 Evaluasi & validasi hasil 62](#_Toc144844163)

[Tabel 4. 1 Persiapan Data 64](#_Toc144844192)

[Tabel 4. 2 Analisa Data 65](#_Toc144844193)

[Tabel 4. 3 *Class* atau label 72](#_Toc144844194)

[Tabel 4. 4 *Transact* 72](#_Toc144844195)

[Tabel 4. 5 *Android.os.binder* 73](#_Toc144844196)

[Tabel 4. 6 *Onserviceconnected* 73](#_Toc144844197)

[Tabel 4. 7 *Bindservice* 74](#_Toc144844198)

[Tabel 4. 8 *Serviceconnection* 74](#_Toc144844199)

[Tabel 4. 9 *Write\_History\_Bookmarks* 74](#_Toc144844200)

[Tabel 4. 10 *Read\_History\_Bookmarks* 75](#_Toc144844201)

[Tabel 4. 11 *Send\_sms* 75](#_Toc144844202)

[Tabel 4. 12 *Read\_sms* 76](#_Toc144844203)

[Tabel 4. 13 *Telephonymanager.getline1number* 76](#_Toc144844204)

[Tabel 4. 14 Statistik Data 77](#_Toc144844205)

[Tabel 4. 15 Korelasi Atribut 77](#_Toc144844206)

[Tabel 4. 16 *Missing Value* 95](#_Toc144844207)

[Tabel 4. 17 Data Duplikat 95](#_Toc144844208)

[Tabel 4. 18 Data *Outlier* 95](#_Toc144844209)

[Tabel 4. 19 Rekayasa Fitur 96](#_Toc144844210)

[Tabel 4. 20 Dataset yang digunakan 96](#_Toc144844211)

[Tabel 4. 21 Split data *training* 96](#_Toc144844212)

[Tabel 4. 22 *Training* Model 97](#_Toc144844213)

[Tabel 4. 23 Pengujian Model Data *Training* 98](#_Toc144844214)

[Tabel 4. 24 Dimensi Data Latih dan Data Uji 99](#_Toc144844215)

[Tabel 4. 25 Pengujian Model Data *Testing* 99](#_Toc144844216)

# ABSTRAK

Sejalan dengan perkembangan teknologi, penggunaan *smartphone* atau *tablet* PC dengan menggunakan sistem operasi android marak digunakan karena multitasking sehingga banyaknya bermunculan aplikasi-aplikasi untuk android, berdampak kepada ancaman keamanan yang semakin besar oleh serangan *malware* (Refhaldo, Budiarto, Sari, & Monica, 2022) dan proses identifikasi *malware* yang semakin rumit

# KATA PENGANTAR

# PENDAHULUAN

## Latar belakang masalah

*Malware* atau *malicious software* merupakan jenis serangan yang terstruktur yang dirancang dengan tujuan jahat untuk merusak sistem komputer, mencuri informasi pribadi, atau melanggar privasi pengguna sehingga serangan *malware* mampu menyebabkan kerusakan pada sistem komputer. Perkembangan teknologi dapat memicu dikembangkannya *malware* baru, sehingga semakin banyak pihak yang dirugikan karena adanya *malware* ini (Novrianda, Kunang, & Shaksono, 2014). Kepala Badan Siber dan Sandi Negara (BSSN) Hinsa Siburian mengatakan, selama tahun 2021 ini tercatat ada 888.711.736 serangan siber. Adapun data tersebut merupakan data yang dihimpun sejak Januari hingga Agustus 2021 dan serangan terbanyak terdapat pada *malware*. Oleh karena itu, Presiden Joko Widodo mengingatkan agar Indonesia selalu bersiap menghadapi serangan siber dan kejahatan penggunaan data (Mashabi, 2021).

Sejalan dengan hal perkembangan teknologi, penggunaan *smartphone* atau *tablet* PC dengan menggunakan sistem operasi android marak digunakan karena multitasking, kemudahan dalam notifikasi hingga banyaknya aplikasi atau software yang dapat dinikimati dengan menggunakan sistem android. banyaknya bermunculan aplikasi-aplikasi untuk android, berdampak kepada ancaman keamanan yang semakin besar oleh serangan *malware* (Refhaldo, Budiarto, Sari, & Monica, 2022) dan proses identifikasi *malware* yang semakin rumit (Hadianto, Novitasari, & Rahmawati, 2019).

Beberapa metode untuk mendeteksi serangan *malware* yang telah dilakukan oleh para peneliti yaitu dengan menggunakan algoritma *neural network* (Asry, Siswanto, & Huda, 2023), *Support Vector Machine* (Rawat, Amrita, & Singh, 2023), *Naive Bayes* (Prasetyo, Suryani, & Anbiya, 2021), *Decision Tree* dan *K-Nearest Neighbors* (Hadiprakoso, Aditya, & Pramitha, 2022)*.*

*Decision Tree* adalah algoritma dalam pembelajaran mesin yang digunakan untuk pengambilan keputusan dan prediksi. Pohon (*tree*) adalah sebuah struktur data yang terdiri dari simpul (*node*) dan rusuk (*edge*). Simpul pada sebuah pohon dibedakan menjadi tiga, yaitu simpul akar (*root node*), simpul percabangan/ internal (*branch/ internal node*) dan simpul daun (*leaf node*) (Eska, 2016). Kekurangan algoritma *Decision Tree* cenderung mempelajari pola yang sangat spesifik pada data pelatihan, yang dapat mengakibatkan overfitting pada dataset tersebut. Ini berarti model akan berkinerja buruk pada data baru yang belum pernah dilihat sebelumnya. *Decision Tree* sensitif terhadap perubahan kecil dalam data pelatihan, sedikit perubahan dalam data dapat menghasilkan struktur pohon yang berbeda, yang pada gilirannya dapat menghasilkan hasil yang berbeda.

*Naive Bayes* merupakan salah satu algoritma metode pengklasifikasian suatu probabilitas dan statistik yang diperoleh Thomas Bayes seorang ilmuwan Inggris dengan cara melakukan prediksi peluang di masa depan berdasarkan pengalaman pada masa sebelumnya (Bustami, 2014). Kelemahan pada algoritma ini yaitu sensitivitas terhadap data latih, jika jumlah data latih sangat terbatas atau terdapat *outlier* yang signifikan, dapat membuat estimasi probabilitas dapat menjadi tidak akurat. Algoritma *Naive Bayes* cenderung hanya memerhatikan hubungan statistik antara fitur-fitur dalam data latih, tanpa memperhatikan konteks atau makna di balik data tersebut. Ini dapat mengurangi kemampuan algoritma untuk memahami aspek kontekstual dari data.

*Support Vector Machine* (SVM) adalah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur (*feature space*) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning bias* yang berasal dari teori pembelajaran statistik (Munawarah, Soesanto, & Faisal, 2016). Kekurangan dari metode SVM adalah sulit diaplikasikan untuk dataset dengan jumlah dimensi yang sangat besar (Aulia, Aprianti, Supriyanto, & Rozikin, 2022).

*K-Nearest Neighbor* (KNN) merupakan metode klasifikasi terhadap sekumpulan data berdasarkan pembelajaran data yang sudah terklasifikasikan sebelumya (Admojo & Ahsanawati, 2020). Kekurangan pada metode ini yaitu KNN harus menyimpan semua data *training* dalam memori dan melakukan perhitungan jarak dengan semua data tersebut, algoritma ini mungkin menjadi kurang efisien pada dataset yang sangat besar.

*Neural Network* adalah sebuah model komputasi yang terinspirasi oleh struktur dan fungsi jaringan saraf biologis dalam otak manusia, *neural network* mengandung elemen pemrosesan dan pembobotan yang saling terhubung. Setiap lapisan dalam jaringan berisi oleh kelompok elemen pemrosesan (Hadianto, Novitasari, & Rahmawati, 2019), kelebihan dari metode ini yaitu mampu mengenali dan memahami pola-pola yang sangat kompleks dan abstrak dalam data, melalui lapisan-lapisan dan hubungan antara unit-unit pemrosesan, *neural network* dapat memahami representasi data dalam berbagai tingkat abstraksi dan hirarki. Ini berarti mereka dapat menangkap fitur-fitur yang semakin kompleks dan informasi yang semakin berguna pada setiap lapisan

Berdasarkan permasalahan diatas, menerapkan algoritma *Neural Network* pada model *machine learning* yang digunakan dalam mendeteksi *malware* android, agar dapat mempermudah mengidentifikasi jenis dan dapat meminimalisir ancaman serangan *malware.*

## Identifikasi masalah

Berdasarkan latar belakang diatas dapat diidenetifikasi masalah yang dihadapi dalam penelitian ini adalah

1. Banyaknya bermunculan aplikasi-aplikasi untuk android, berdampak kepada ancaman keamanan yang semakin besar oleh serangan *malware* (Refhaldo, Budiarto, Sari, & Monica, 2022)*.*
2. Proses identifikasi *malware* yang semakin rumit (Hadiprakoso, Aditya, & Pramitha, 2022).

## Rumusan Masalah

1. Apakah dengan menerapkan algoritma neural network pada sistem deteksi malware android dapat meminimalisir ancaman serangan malware ?
2. Apakah dengan menerapkan algoritma neural network pada sistem deteksi malware android dapat mempermudah meidentifikasi jenis serangan malware ?

## Batasan Masalah

Untuk memperjelas arah permasalahan yang akan dibahas dari rumusan masalah diatas, maka batasan masalah pada penelitian ini hanya membahas sebagai berikut:

1. Pada penelitian ini data yang digunakan merupakan data dari penelitian sebelumnya, yaitu dataset yang diambil pada situs website *(*[*https://www.kaggle.com/datasets/shashwatwork/android-malware-dataset-for-machine-learning*](https://www.kaggle.com/datasets/shashwatwork/android-malware-dataset-for-machine-learning) *)* dan tidak membahas menggunakan data *real*.
2. Penelitian ini berfokus pada pembuatan model *machine learning* yang digunakan untuk melakukan deteksi *malware* android, tidak membahas tentang implementasi model ke dalam sebuah aplikasi yang berbasis andorid atau yang lainnya.
3. Dalam penelitian ini hanya membahas deteksi *malware* android dengan menggunakan algoritma *neural network* tidak membahas tentang penggunaan algoritma atau metode lainnya*.*

## Tujuan penelitian

Adapun tujuan dari penelitian ini adalah :

1. Menerapkan algoritma neural network pada sistem deteksi malware android agar dapat meminimalisir ancaman serangan malware
2. Menerapkan algoritma neural network pada sistem deteksi malware android agar mempermudah mengidentifikasi jenis serangan malware

## Manfaat Penelitian

Terdapat beberapa manfaat penelitian yang dibuat oleh peneliti, sebagai berikut:

1. Bagi Universitas

Dengan adanya skripsi yang mengangkat topik *data science* dapat dimanfaatkan dan digunakan oleh Universitas sebagai referensi dasar untuk generasi selanjutnya untuk menulis penelitian lainnya.

1. Bagi Peneliti

Dapat mengimplementasikan teori dan ilmu yang sudah didapatkan selama kuliah khususnya dalam bidang *data science*

1. Bagi Masyarakat Umum

Menambah ilmu pengetahuan dan wawasan bagi masyarakat umum yang ingin mempelajari tentang data science khususnya tentang deteksi *malware* android.

## Metodologi Penelitian

Metodologi yang digunakan dalam penelitian ini melalui beberapa tahapan, yaitu :

1. Observasi

Observasi adalah proses pemerolehan data dengan cara melakukan pengamatan secara langsung. Mengamati dataset yang dikumpulakn untuk mengidentifikasi dan mengetahui serangan *malware* android. Metode observasi yang digunakan yaitu observasi non-partisipan dengan mengambil data langsung melalui dataset yang diambil dari *(*[*https://www.kaggle.com/datasets/shashwatwork  
/android-malware-dataset-for-machine-learning*](https://www.kaggle.com/datasets/shashwatwork/android-malware-dataset-for-machine-learning) *).*

1. Studi Pustaka

Tahap ini dilakukan dengan cara mengkaji dan mempelajari literatur dan referensi berupa naskah ilmiah, buku konsep *machine* *learning*, serta cara kerja algoritma *Neural Network* sehingga dapat menunjang metodologi yang akan diterapkan pada penelitian.

## Sistematika Penuliasan

Untuk memudahkan dalam proses penyusunan tugas akhir, penulis menyusun ke dalam lima bab. Setiap bab tersebut secara keseluruhan saling berkaitan satu sama lain, dimana diawali dengan bab pendahuluan dan diakhiri dengan bab penutup yang berupa kesimpulan dan saran. Maka dibuat suatu sistematika penulisan sebagai berikut :

**BAB I PENDAHULUAN**

Bab ini berisi penjelasan secara sistematis mengenasi landasan topik penelitian yang meliputi latar belakang, identifikasi masalah, rumusan masalah dan batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penilitian, dan yang terakhir sistematika penulisan.

**BAB II TINJAUAN PUSTAKA**

Bab ini menguraikan tentang penelitian sebelumnya yang pernah dilakukan dan landasan teori penelitian mengenai data *mining*, *Machine Learning*, dan metode Algoritma *Neural Network* yang berkaitan langsung dengan penelitian ini.

**BAB III METODE PENELITIAN**

Bab ini menjelaskan secara sistematis, bagaimana proses penelitian dilakukan. Penjelasan pada bab ini meliputi Analisa kebutuhan sistem dan perancagan penelitian.

**BAB IV IMPLEMENTASI DAN PENGUJIAN**

Bab ini menjelaskan hasil pengujian yang dilakukan serta analisis dari data yang diperoleh dari hasil pengujian yang dilakukan.

**BAB V KESIMPULAN DAN SARAN**

Bab ini menjelaskan hasil akhir dari penelitian ini dan saran untuk penelitian yang akan datang

# TINJUAN PUSTAKA DAN LANDASAN TEORI

## Tinjauan Pustaka

Penelitian yang dilakukan oleh Nishant Rawat, Amrita, dan Avjeet Singh pada tahun 2023 dengan judul penelitian “*Permission-Based Malware Detection in Android Using Machine Learning*”, penelitian ini menggunakan Algoritma *Support Vector Machine* (SVM), *Random Forest*, dan *Naive Bayes*. Penelitian ini menggunakan pendekatan berbasis multi-level yang melibatkan pengumpulan *Dataset* dari 10.000 aplikasi dan mengidentifikasi berbagai aspek seperti izin, ukuran kecil, dan tingkat izin. Aplikasi kemudian diklasifikasikan sebagai *malware* atau jinak menggunakan algoritma *machine learning*.Tujuan dari penelitian ini adalah mengembangkan sistem deteksi *malware* dengan menganalisis pemintaan izin aplikasi dan mengkategorikannya sebagai jinak atau *malware*. Sedangkan masalah yang dihadapi adalah terancamnya privasi data pengguna karena sifat *open source* dari platform android yg membuat aplikasi lebih rentan terhadap serangan. Hasil yang didapat dari penelitian yaitu akurasi 91,54% untuk *Support Vector Machi*ne, 92,04% untuk *Random Forest*, dan 91,11% untuk model *Naive Bayes.*

Penelitian yang dilakukan oleh Dieta Wahyu Asry, Eko Siswanto, Dendy Kurniawan Haris Ihsanil Huda pada tahun 2023 dengan judul penelitian “Deteksi *Malware* Statis menggunakan *Deep Neural Networks* Pada *Portable Executable*”, penelitian ini mendeteksi *malware* dengan menggunakan metode *Neural Network with Dropout* terhadap model pohon keputusan untuk memeriksa seberapa baik kinerjanya dalam mendeteksi file PE berbahaya yang sebenarnya. Penelitian ini menggunakan *dataset* 900 ribu sampel pelatihan dengan 300 ribu sampel berbahaya, 300 ribu jinak, dan 300 ribu sampel tidak berlabel. Adapun masalah yang dihadapi dalam penelitian ini adalah bagaimana meminimalkan terjadinya false positive saat mengenali malware baru pada berkas portable secara statis. Penelitian ini bertujuan untuk mengusulkan dan mengevaluasi deep neural network untuk menganalisis file portabel secara statis guna mempelajari fitur - fitur dari portable executable malware untuk meminimalkan terjadinya false positive saat mengenali malware baru. Hasil yang didapatkan mencapai AUC sebesar 99,8% dengan 98% *true positive* pada 1% *false positive* pada kurva ROC.

Penelitian yang dilakukan oleh Raden Budiarto Hadiprakoso, Wahyu Rendra Aditya dan Febriora Nevia Pramitha pada tahun 2022 dengan judul penelitian “Analisis Statis Deteksi *Malware* Android Menggunakan Algoritma *Supervised Machine Learning*”, metode yang digunakan pada penelitian ini adalah Algoritma *Support Vector Machine* (SVM), *Naive Bayes*, *Decision Tree* dan *K-Nearest Neighbor*, dan juga berkonsentrasi pada memaksimalkan pencapaian dengan mengevaluasi berbagai algoritma dan menyesuaikan beberapa konfigurasi untuk mendapatkan kombinasi terbaik dari hyper-parameter. Penelitian mengumpulkan fitur analisis statis dari aplikasi aman dan berbahaya (*malware*). *Dataset* yang digunakan pada penelitian adalah *Dataset* *malware* DREBIN yang merupakan *Dataset* *malware* yang tersedia secara publik. Adapun masalah yang dihadapi penelitian ini yaitu karena banyaknya serangan *malware* pada sistem android dan proses identifikasi *malware* yang semakin rumit, maka perlu digunakan machine learning untuk klasifikasi *malware*. Sedangkan tujuan penelitian ini untuk dapat mengidentifikasi *malware* guna menjaga keamanan dan privasi pengguna. Hasil dari penelitian ini menunjukkan bahwa klasifikasi model SVM mendapatkan hasil terbaik dengan mencapai akurasi 96,94% dan nilai AUC (Area Under Curve) 95%.

Penelitian yang dilakukan oleh Muhammad Refhaldo, Eko Budiarto, Putri Anggun Sari dan Sella Monica pada tahun 2022 dengan judul penelitian “Klasifikasi Aplikasi *Malware* Android Menggunakan Algoritma C5.0”, metode yang digunakan pada penelitian ini adalah Algoritma C5.0. Teknik pengujian split validation 80:20 dimana 80% data akan digunakan sebagai data training dan 20% data akan digunakan sebagai data testing. *Dataset* yang digunakan pada penelitian adalah *Dataset* *malware* DREBIN yang merupakan *Dataset* *malware* yang tersedia secara publik. Adapun masalah yang dihadapi penelitian ini yaitu banyaknya bermunculan aplikasi-aplikasi untuk android, berdampak kepada ancaman keamanan yang semakin besar oleh serangan *malware*. Sedangkan tujuan penelitian ini untuk mendapatkan hasil analisis menggunakan metode C5.0 dalam mengklasifikasi sebuah aplikasi yang teridentifikasi sebagai *malware*. Hasil dari penelitian ini menunjukkan bahwa klasifikasi menggunakan algoritma C5.0 mendapatkan hasil akurasi sebesar 94,96% pada data training dan 94,23% pada data testing.

Penelitian yang dilakukan oleh Yitshak Wanli Sitorus, Parman Sukarno, dan Satria Mandala pada tahun 2021 dengan judul penelitian “Analisis Deteksi *Malware* Android menggunakan metode *Support Vector Machine* & *Random Forest*”, metode yang digunakan yaitu Algoritma *Support Vector Machine* (SVM) dan *Random Forest*. Sedangkan masalah yang dihadapi dalam penelitian ini adalah melakukan perbandingan antara metode *Support Vector Machine* (SVM) dengan metode *Random Forest* dalam melakukan klasifikasi data, serta membandingkan hasil akurasi dengan penelitian sebelumnya. Penelitian ini bertujuan membandingkan dua algoritma *Support Vector Machine* (SVM) dan *Random Forest*. Proses klasifikasi menggunakan metode *Support Vector Mach*ine (SVM) menghasilkan nilai *precision* 97%, nilai *Recall* 97%, dan nilai *f1-score* 97%, dan akurasi 96,23%, pada metode *Random Forest* menghasilkan nilai *precision* 99%, nilai *Recall* 99%,nilai *f1-score* 99%, dan akurasi 98,99%. Hasil percobaan ini, metode *Random Forest* lebih unggul dari metode *Support Vector Machine* (SVM) dan pendekatan yang diusulkan di penelitian ini memiliki hasil performansi matrik di atas 95% yang lebih baik daripada penelitian sebelumnya.

Penelitian yang dilakukan oleh Bangkit Prasetyo, Vera Suryani, dan Dhika Rizki Anbiya pada tahun 2021 dengan judul penelitian “Analisis Deteksi *Malware* pada Aplikasi Android *Fintech* berdasarkan *Permissions* dengan menggunakan *Naive Bayes* dan *Random Forest*”, penelitian ini mendeteksi *malware* pada aplikasi android fintech berdasarkan permission dengan menggunakan metode klasifikasi naive bayes dan random forest. Penelitian ini menggunakan 160 *Dataset* dari sampel aplikasi *fintech* dengan 80 sampel aplikasi fintech yang terdaftar di OJK dan 80 sampel aplikasi *fintech* yang tidak terdaftar di OJK. Adapun masalah yang dihadapi dalam penelitian ini adalah bagaimana aplikasi android *fintech* dapat mendeteksi *malware.* Penelitian ini bertujuan untuk deteksi *malware* aplikasi fintech berdasarkan permissionnya untuk *Dataset* diambil dari ekstraksi fitur permission. Hasil yang didapatkan *random forest* memiliki akurasi yang lebih tinggi yaitu sebesar 80,1% dibandingkan dengan *naive bayes* yang mendapatkan akurasi sebesar 78%.

Tabel 2. 1 Tinjauan Studi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Peneliti** | **Tahun** | **Topik Penelitian** | **Metode** | **Hasil** |
| Nishant Rawat, Amrita, dan Avjeet Singh | 2023 | *Permission-Based Malware Detection in Android Using Machine Learning* | Algoritma *Support Vector Machine* (SVM), *Random Forest*, dan *Naive Bayes*. | akurasi SVM  91,54%,  RF 92,04% dan *Naive Bayes.*  91,11%.*.* |
| Dieta Wahyu Asry, Eko Siswanto, Dendy Kurniawan Haris Ihsanil Huda | 2023 | Deteksi *Malware* Statis menggunakan *Deep Neural Networks* Pada *Portable Executable* | *Deep Neural Networks* | AUC sebesar 99,8% dengan 98% *true positive* pada 1% *false positive* pada kurva ROC |
| Raden Budiarto Hadiprakoso, Wahyu Rendra Aditya dan Febriora Nevia Pramitha | 2022 | Analisis Statis Deteksi *Malware* Android Menggunakan Algoritma *Supervised Machine Learning* | Algoritma *Support Vector Machine* (SVM), *Naive Bayes*, *Decision Tree* dan *K-Nearest Neighbor* | Akurasi SVM 96,94% dan nilai AUC (Area Under Curve) 95% |
| Refhaldo, Eko Budiarto, Putri Anggun Sari dan Sella Monica | 2022 | Klasifikasi Aplikasi *Malware* Android Menggunakan Algoritma C5.0 | Algoritma C5.0 | akurasi 94,96%  data training dan 94,23% data testing. |
| Yitshak Wanli Sitorus, Parman Sukarno, dan Satria Mandala | 2021 | Analisis Statis Deteksi *Malware* Android Menggunakan Algoritma *Supervised Machine Learning* | Algoritma *Support Vector Machine* (SVM) dan *Random Forest* | SVM *Accuracy 96,23%, Precision 97%, f1-score 97%, & Recall 97%* dan untuk *Random Forest Accuracy 96,23%, Precision 97%, f1-score 97%, & Recall 97%* |
| Bangkit Prasetyo, Vera Suryani, dan Dhika Rizki Anbiya | 2021 | Analisis Deteksi *Malware* pada Aplikasi Android *Fintech* berdasarkan *Permissions* dengan menggunakan *Naive Bayes* dan *Random Forest* | *Naive Bayes* dan *Random Forest* | RF 80,1%  Dan NB 78% |
| Mochamad Ferrian Mulyadi | 2023 | Implementasi Algorima *Neural Network* Pada Sistem Deteksi Malware Android | Algorima *Neural Network* | ? |

## Landasan Teori

### **Data *Mining***

Data *mining* dapat diartikan sebagai pengekstrakan informasi baru yang diambil dari bongkahan data besar yang membantu dalam pengambilan keputusan. Istilah data mining kadang disebut juga *knowledge discovery* (Husain, 2018). Sedangkan menurut (Noviyanto, 2020)Data *mining* adalah metoda yang digunakan untuk mengekstraksi informasi prediktif tersembunyi pada database, ini adalah teknologi yang sangan potensial bagi perusahaan yang sangat potensial bagi perusahaan dalam memberdayakan data *warehouser.*

secara analogi, data *mining* seharusnya lebih tepat disebut "penambangan pengetahuan dari data," yang sayangnya agak panjang. Namun, istilah yang lebih pendek, penambangan pengetahuan, mungkin tidak mencerminkan penekanan pada penambangan dari sejumlah besar data. Meskipun demikian, penambangan adalah istilah yang hidup menggambarkan proses yang menemukan seperangkat kecil inti berharga dari banyak bahan mentah. Dengan demikian, kesalahan penamaan seperti itu yang membawa kedua "data" dan "penambangan" menjadi pilihan popular (Han, Kamber, & Pei, 2012).

Dari beberapa pengertian tersebut, dapat disimpulkan bahwa data *mining* adalah proses ekstraksi informasi yang berharga, tersembunyi, dan sebelumnya tidak diketahui dari kumpulan data besar. Tujuan dari data mining adalah untuk menemukan pola, keteraturan, dan wawasan yang dapat membantu dalam pengambilan keputusan dan prediksi di berbagai aspek.

#### Tahap-tahap Data *Mining*

Ada Tujuh Tahapan dari data mining sebagai berikut (Romadhon & Kodar, 2020):

1. Pembersihan Data, Proses untuk membuang data yang tidak valid dan yang tidak sesuai untuk digunakan.
2. Integrasi Data, Merupakan proses penggabungan sumber-sumber data.
3. Seleklsi Data, Proses pengambilan data-data yang sesuai untuk analisis.
4. Transformasi Data, Pengubahan data menjadi bentuk yang tepat sesuai data mining.
5. Proses Mining, Proses awal pengkajian metode.
6. Evaluasi Data, Proses utuk mengidentifikasi pola yang menarik untuk mewakili pengetahuan dari data yang tersembunyi.
7. Presentasi pengetahuan, merupaka proses presentasi pengetahuan dan Teknik visualisasi dapat membantu memberitahukan hasil dari data mining.

#### Metode Data Mining

Metode data minig merupakan suatu proses utama yang digunakan saat metode diterapkan untuk menemukan pengetahuan berharga dari data, ada beberapa teknik dan sifat analisa yang dapat digolongkan dalam *data mining* (Muslim, et al., 2019) yaitu;

1. *Classification*

Klasifikasi merupakan teknik yang digunakan untuk menemukan model agar dapat menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Metode klasifikasiyang sering digunakan yaitu*, Support Vector Machine, Multilayer Perceptron, Naive bayes, ID3, Ensemble Methode, dll.*

1. *Association*

Association rule mining adalah teknik data mining yang berfokus untuk menemukan aturan kesamaan dalam suatu kejadian. Contoh aturan asosiasi yang sering dijumpai adalah proses pembelian barang dagangan pada pusat perbelanjaan. Metode asosiasi yang umum digunakan adalah FP-Growth, *Coefficient* *of* *Correlation*, Chi Square, A Priori, dll.

1. *Clustering*

*Clustering* merupakan teknik dengan cara mengelompokkan data secara otomatis tanpa diberitahukan label kelasnya. *Clustering* dapat digunakan untuk memberikan label pada kelas data yang belum diketahui, karena *clustering* sering digolongkan sebagai metode *unsupervised* *learning*. Metode *clustering* yang sering digunakan yaitu, K-*Medoids*, K-*Means*, *Fuzzy* C-*Means*, *Self*-*Organizing* *Map* (SOM), dll.

1. *Prediction*

memperkirakan suatu nilai di masa mendatang, misalnya memprediksi stok barang tiga tahun ke depan. Yang termasuk fungsi ini antara lain metode Neural Network, Decision Tree, dan k –Nearest Neighbor.

1. *Regression*

Digunakan untuk mencari pola dan menentukan nilai numerik. Metode yang sering digunakan adalah linear regression, logistic regression, support vector regression, dll.

1. *Description*

memberi gambaran secara ringkas terhadap sejumlah data yang bersakala besar dan memiliki banyak jenis. Termasuk di dalamnya metode *Decision* *Tree*, *Exploratory* Data *Analysis* dan *Neural* *Network*.

### ***Machine Learning***

*Machine Learning* (ML) atau Mesin Pembelajar adalah cabang dari AI yang fokus belajar dari data (*learn* from data), yaitu fokus pada pengembangan sistem yang mampu belajar secara “mandiri" tanpa harus berulang kali diprogram manusia. ML membutuhkan Data yang valid sebagai bahan belajar (ketika proses *training*) sebelum digunakan ketika *testing* untuk hasil *output* yang optimal (Cholissodin, Sutrisno, Soebroto, Hasanah, & Febiola, 2020).

Menurut (Awad & Khanna, 2015) *Machine learning* adalah cabang kecerdasan buatan yang menerapkan algoritma secara sistematis mensintesis hubungan yang mendasari antara data dan informasi. *Machine* *learning* dapat didefinisikan sebagai aplikasi komputer dan algoritma matematika yang diadopsi dengan cara pembelajaran yang berasal dari data dan menghasilkan prediksi di masa yang akan datang (Goldberg & Holland, 1988). Adapun proses pembelajaran yang dimaksud adalah suatu usaha dalam memperoleh kecerdasan yang melalui dua tahap antara lain latihan (*training*) dan pengujian (*testing*) (huang, G.-B., Zhu, Q.-Y., & Siew, 2006).

*Machine* *learning* dibagi menjadi dua, yakni *unsupervised* *learning* dan *supervised* *learing*. *Supervised* *learning* merupakan pemnelajaran menggunakan data yang mengandung input serta *output* yang diinginkan. Klasifikasi dan Regresi merupakan bentuk dari *supervised learing*. *Unsupervised* learning merupakan pembelajaran menggunakan data yang hanya memiliki input tetapi tidak mempunyai output spesifik. Fungsi utama dari Unsupervised learning adalah menemukan fitur atau pettern dan akan mengkategorikan hasilya sebagai prediksi (Sanjaya, Renata, Budiman, Anderson, & Ayub, 2020). Secara umum, proses *machine learning* melibatkan langkah-langkah berikut:

1. Pengumpulan data: Data yang relevan dan representatif dikumpulkan, baik dalam bentuk numerik maupun non-numerik, untuk melatih dan menguji model *machine learning*.
2. Pra-pemrosesan data: Data yang dikumpulkan sering kali memerlukan pra-pemrosesan untuk membersihkan, menghilangkan nilai yang hilang, mengubah format, atau melakukan normalisasi agar dapat digunakan oleh model machine learning.
3. Pemilihan model: Berdasarkan tugas yang ingin diselesaikan, pemilihan model machine learning yang tepat sangat penting. Beberapa jenis model *machine learning* populer termasuk regresi linear, pohon keputusan, jaringan saraf tiruan, dan mesin vektor dukungan (SVM).
4. Pelatihan model: Pada langkah ini, model *machine learning* dipelajari menggunakan data pelatihan yang tersedia. Proses ini melibatkan pencocokan model ke data pelatihan untuk menemukan pola atau hubungan yang ada.
5. Evaluasi model: Setelah melatih model, penting untuk mengevaluasi kinerja model menggunakan data uji atau validasi. Hal ini dapat melibatkan pengukuran metrik evaluasi seperti akurasi, presisi, *recall*, atau metrik khusus lainnya tergantung pada jenis masalah yang dipecahkan.
6. Penyempurnaan model: Jika kinerja model belum memuaskan, perlu dilakukan penyempurnaan melalui iterasi berulang pada langkah-langkah sebelumnya, termasuk pemilihan fitur yang lebih baik, pengaturan parameter yang lebih baik, atau eksplorasi model yang berbeda.
7. Penerapan model: Setelah model yang baik telah dikembangkan, model tersebut dapat digunakan untuk membuat prediksi atau mengambil keputusan pada data baru yang tidak digunakan selama pelatihan.

*Machine learning* adalah cabang dari kecerdasan buatan yang fokus pada pengembangan algoritma dan model komputer yang dapat belajar dan mengambil keputusan atau melakukan prediksi berdasarkan data yang diberikan.

### **Python**

Python merupakan bahasa pemerograman tingkat tinggi (*hight level language*) yang dikembangkan Oleh Guldo van Rossum pada tahun 1989 dan diperkenalkan untuk pertama kalinra pada tahun 1991. *Python* lahir atas dasar keinginan untuk mempermudah scorang programer dalam menyelesaikan tugas-tugasnya dengan cepat. *Python* dirancang untuk memberikan kemudahan yang sangat luar biasa kepada programer baik dari segi efisiensi waktu, maupun kemudahan dalam pengembangan program khususnya dalam hal kompatibiliras dengan sistem (Wadi, 2015). Selain itu, *Python* juga sering digunakan sebagai bahasa *scripting* atau sebagai bahasa penghubung untuk mengintegrasikan komponen yang ada (Hilpisch, 2015).

*Python* adalah bahasa pemrograman tingkat tinggi yang sering digunakan dalam berbagai bidang, termasuk pengembangan perangkat lunak, analisis data, kecerdasan buatan, pemrosesan bahasa alami, pengembangan *web*, dan sebagainya. Meskipun kinerja bahasa yang di tafsirkan seperti *Python* untuk tugas komputasi intensif lebih rendah daripada bahasa pemrograman tingah rendah, *library* ekstensi *NumPy* dan *SciPy* telah dikembangkan yang dibangun di atas implementasi Fortran dan C lapisan bawah untuk operasi cepat dan vektorisasi pada *array* multidimensi (VanderPlas, 2017).

### **Library Python**

#### Pandas

*Pandas* adalah *library open source* pada *python* yang sering digunakan untuk melakukan operasi numerik dan komputasi ilmiah. Selain itu pandas juga memproses data yang meliputi pembersihan data, manipulasi data, hingga melakukan analisis data yang nantinya dapat menyimpan data dalam berbagai format seperti CSV, Excel, dan SQL (Nelli, 2015). *Pandas* menyediakan struktur data dan fungsi *high-level* untuk membuaut pekerjaan dengan data terstruktur/tabular lebih cepat, mudah, dan ekspresif (Rohman, 2019).

#### Numpy

*NumP*y adalah pustaka fungsi yang memungkinkan melakukan banyak tugas manipulasi data umum dengan *Python*. Banyak interaksi antara *array NumPy* dan *Python* sangat mirip dengan apa yang akan dilakukan dengan variabel Python biasa. Untuk tugas yang lebih lanjut, *NumPy* juga memiliki fungsi yang bekerja dengan aljabar linier, transformasi *Fourier,* dan matriks (Putra, Kartini, Suyitno, Sugiarta, & Puspita, 2023).

Numpy (Numerical Python) adalah paket dasar untuk komputasi ilmiah dengan Pyhton dan khusnya unutk analisa data. Faktanya, ini library adalah dasar dari sejumlah besar paket Python matematika dan ilmiah, dan diantaranya sebagai anda akan melihat nanti di buku, perpustakaan pandas. Perpustakaan ini sepenuhnya dikhususkan untuk analis data sepenuhnya dikembangkan mengggunakan konsep yang diperkenalkan oleh NumPy. Bahkan, alat bawaan disediakan oleh standar pustaka Python mungkin terlalu sederhana atau tidak memadai untuk sebagian besar perhitungan dalam analisis data (Nelli, 2015).

#### *Matplotlib*

*Matplotlib* merupakan sebuah pustaka lengkap yang digunakan untuk menciptakan visualisasi dalam bahasa pemrograman *Python*. Dengan *Matplotlib*, pengguna dapat menghasilkan gambar-gambar berkualitas tinggi dalam format cetak maupun dalam lingkungan interaktif pada berbagai *platform*. Pustaka ini dapat digunakan dalam skrip *Python*, *shell Python/IPython*, server aplikasi web, serta berbagai *toolkit* antarmuka pengguna grafis (Hunter, 2023).

#### *Seaborn*

*Seaborn* adalah sebuah pustaka *Python* yang digunakan untuk visualisasi data berbasis statistik. Pustaka ini dibangun di atas pustaka *matplotlib* dan dirancang khusus untuk membuat visualisasi data yang lebih menarik dan informatif dengan lebih sedikit kode, menurut (Waskom, 2021) *Seaborn* adalah *library* visualisasi data *Python* berdasarkan *matplotlib*. Ini menyediakan antarmuka tingkat tinggi untuk menggambar grafik statistik yang menarik dan informatif.

*Seaborn* menyediakan berbagai jenis plot yang telah diatur dengan baik dan memiliki gaya yang estetis secara default. Hal ini memungkinkan pengguna untuk dengan mudah membuat plot seperti scatter plot, line plot, bar plot, histogram, heatmap, dan banyak jenis plot lainnya.

#### Scikit-Learn

*Scikit-Learn* adalah modul *Python* yang mengintegrasikan banyak algoritma *machine learning*. Pustaka ini awalnya dikembangkan oleh Cournapeu pada tahun 2007, namun rilis pertama yang sebenarnya terjadi pada tahun 2010 (Nelli, 2015). Terdapat beberapa pustaka *Python* yang menyediakan implementasi yang solid dari berbagai algoritma machine learning. Salah satu yang paling terkenal adalah *Scikit-Learn*, sebuah paket yang menyediakan versi efisien dari sejumlah besar algoritma umum. *Scikit-Learn* ditandai dengan API yang bersih, seragam, dan efisien, serta dokumentasi *online* yang sangat berguna dan lengkap. Keuntungan dari keseragaman ini adalah setelah Anda memahami penggunaan dasar dan sintaksis *Scikit-Learn* untuk satu jenis model, beralih ke model atau algoritma baru akan sangat mudah (VanderPlas, 2017).

#### Keras

Keras adalah pustaka pembelajaran mesin (*machine* *learning*) yang populer dan kuat yang awalnya ditulis dalam bahasa pemrograman *Python*. Pustaka ini dirancang untuk memudahkan pembuatan, pelatihan, dan evaluasi model jaringan saraf tiruan (*neural networks*) serta model pembelajaran mesin lainnya.

Keras dapat menyederhanakan *script* *TensorFlow* dan memberikan solusi yang jelas untuk permasalahan dalam mengembangkan model *deep* *learning* (Syarovy & Sutiarso, 2023). Keras banyak digunakan oleh berbagai organisasi ilmiah di seluruh dunia, termasuk CERN, NASA, dan NIH (Gulli & Pal, 2017).

#### Tensorflow

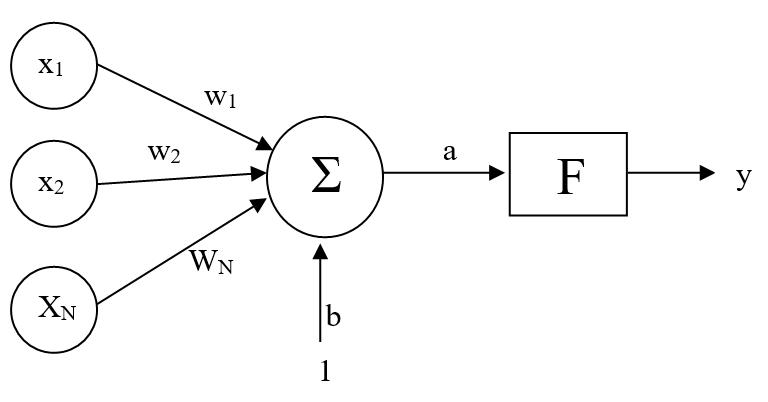
*TensorFlow* adalah pustaka yang digunakan untuk komputasi numerik yang berfokus pada *machine learning* dan *artificial intelligence,* *TensorFlow* menggunakan struktur data yang disebut "*Tensor*" untuk merepresentasikan data numerik multidimensi, yang merupakan dasar dari operasi matematika yang kompleks yang dilakukan oleh model pembelajaran mesin.

TensorFlow memiliki ekosistem, pustaka, dan komunitas yang kaya serta fleksibel yang memungkinkan peneliti merancang, membangun, dan menerapkan aplikasi berbasis machine learning dan deep learning (Syarovy & Sutiarso, 2023).

### **Neural Network**

*Neural Network* adalah model matematika yang terinspirasi dari cara kerja jaringan saraf dalam otak manusia. Ini adalah salah satu komponen inti dalam *machine learning* dan *artificial intelligence*. *Neural Network* merupakan salah satu metode dalam data mining yang memiliki keunggulan tingkat akurasi yang lebih baik atau optimal dibandingkan dengan metode lainnya (Handayani, Riandini, & Situmorang, 2022)

*Neural Network* adalah sebuah metode yang terinspirasi oleh struktur jaringan syaraf otak manusia, yang dirancang untuk meniru cara otak manusia melakukan pemrosesan dan penyimpanan informasi (Ramdhan, 2019). *Neural Network* terdiri dari *node* yang menggabungkan inputnya, yang bisa berupa variabel dari database atau output dari node lainnya. *Node-node* ini dapat dikelompokkan menjadi tiga lapisan yang sederhana, yaitu lapisan input, lapisan output, dan lapisan tersembunyi (Bhakti, 2019). Berikut adalah gambar jaringan syaraf sederhana dengan fungsi aktivasi F.

i

Gambar 2. 1 Fungsi Aktivasi Jaringan Syaraf Sederhana

Pada gambar tersebut sebuah *neuron* akan mengolah N input (x1, x2, …, xN) yang masing – masing memiliki bobot w1, w2, …, wN dan bobot bias b, dengan rumus :

Kemudian fungsi aktivasi F akan mengaktivasi a menjadi output jaringan y (Kusumadewi, 2004).

#### Arsitektur Jaringan

Menurut (Drs. Jong Jek Siang, 2005) terdapat tiga arsitektur jaringan yang sering dipakai dalam *neural network* antara lain :

1. *Single layer Neural Network*

Dalam jaringan jenis ini memiliki koneksi pada *input* nya secara langsung ke jaringan *output*. Sehingga jaringan ini sangatlah terbatas, dan hanya digunakan pada kasus yang sederhana.

1. *Multilayer Neural Network*

Jenis *neural network* ini memiliki layer yang dinamakan “*hidden*”, ditengah layer *input* dan *output.* *Hidden* ini bersifat *variable*, dapat digunakan lebih dari satu *hidden* *layer*.

1. *Reccurent Neural Network*

Model jaringan ini mirip dengan *single layer* dan *multilayer* *neural* *network*, yang membedakan yaitu, ada *neuron* *output* yang memberikan sinyal pada unit *input* (*feedback loop*).

#### *Backpropagation*

*Backpropagation* adalah metode yang digunakan dalam pelatihan jaringan saraf tiruan (*neural* *network*) untuk mengoptimalkan bobot dan bias agar jaringan dapat belajar dari data pelatihan dan menghasilkan prediksi yang akurat. *Backpropagation* merupakan metode pelatihan terawasi (*supervised learning*), dalam artian mempunyai target yang akan dicari (Utomo W. , 2015). *backpropagation* terdiri dari beberapa langkah utama (Drs. Jong Jek Siang, 2005):

1. ***Forward Pass***

Data pelatihan dimasukkan ke dalam jaringan neural. Data tersebut mengalir melalui jaringan dari lapisan input hingga mencapai lapisan output. Setiap neuron dalam jaringan menghitung nilai keluaran berdasarkan bobot dan bias yang terhubung dengannya dan menghasilkan prediksi atau output jaringan untuk data pelatihan tertentu.

1. **Perbandingan dengan Target**

Hasil prediksi yang dihasilkan oleh jaringan dibandingkan dengan target yang sebenarnya. Sehingga menghasilkan nilai kesalahan (*error*) yang mencerminkan seberapa besar perbedaan antara prediksi jaringan dan target yang diharapkan.

1. ***Backward Pass***

Selama tahap ini, setiap neuron menghitung gradien kesalahan terhadap bobot dan bias yang terhubung dengannya. Gradien ini menggambarkan bagaimana kesalahan akan berubah jika bobot atau bias tersebut diubah sedikit.

1. **Optimisasi**

Optimasi diperlukan untuk memperbarui bobot dan bias jaringan dengan tujuan untuk mengurangi kesalahan secara bertahap sehingga jaringan dapat belajar dari data pelatihan.

1. **Iterasi**

Seluruh proses ini diulangi untuk setiap data pelatihan dalam set pelatihan. Data pelatihan diacak sebelum setiap iterasi (*epoch*) untuk memastikan bahwa jaringan belajar dari berbagai contoh dengan baik.

1. **Konvergensi**

Proses iterasi dilanjutkan hingga jaringan mencapai tingkat kesalahan yang dapat diterima atau hingga kriteria berhenti tertentu terpenuhi

### **Pengujian Model**

#### *F1*-*Score*

*F1*-*score* adalah salah satu metrik evaluasi yang umum digunakan dalam bidang klasifikasi dalam pembelajaran mesin untuk memberikan gambaran yang lebih komprehensif tentang kinerja model dalam memprediksi kelas yang benar *.*

*F1-score* digunakan untuk mengukur kombinasi hasil *precision* dan *recall*, sehingga menjadi satu nilai pengukuran (Istighfarizky, et al., 2022). *F-1 score* membantu mengetahui kemampuan model dalam mengenali *false negative* dan *false positive* (Abdurrohman, Dini, & Muharram, 2018).

*F1* = *2* ×

#### *Recall*

*Recall* adalah adalah metrik evaluasi yang digunakan dalam bidang klasifikasi untuk mengukur sejauh mana model dapat mengidentifikasi semua instance positif yang sebenarnya. *recall* menunjukkan berapa persen data kategori positif yang terklasifikasikan dengan benar oleh *system* (Istighfarizky, et al., 2022), Dalam kata lain, *recall* mengukur kemampuan model untuk menemukan atau "mengingat" semua contoh yang benar dari kelas positif. Persamaan *recall* perbandingan antara *true positive* terhadap total contoh yang benar-benar *positive* (Nasution & Hayaty, 2019):

*Recall* =

#### Akurasi

Akurasi menggambarkan seberapa akurat system dapat mengklasifikasikan data secara benar.Dengan kata lain, nilai akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan keseluruhan data (Istighfarizky, et al., 2022)

Akurasi=

#### *Confusion* *Matrix*

*Confusion matrix* adalah alat yang digunakan untuk menganalisis kemampuan model klasifikasi dalam mengenali berbagai *tuple* data yang berbeda. (Nasution & Hayaty, 2019). *Confusion Matrix* adalah tabel dengan 4 kombinasi berbeda dari nilai prediksi dan nilai aktual. Ada empat istilah yang merupakan representasi hasil proses klasifikasi pada confusion matrix yaitu *True Positif, True Negatif, False Positif,* dan *False Negatif* (*University*, 2023).

### **Aplikasi Pendukung**

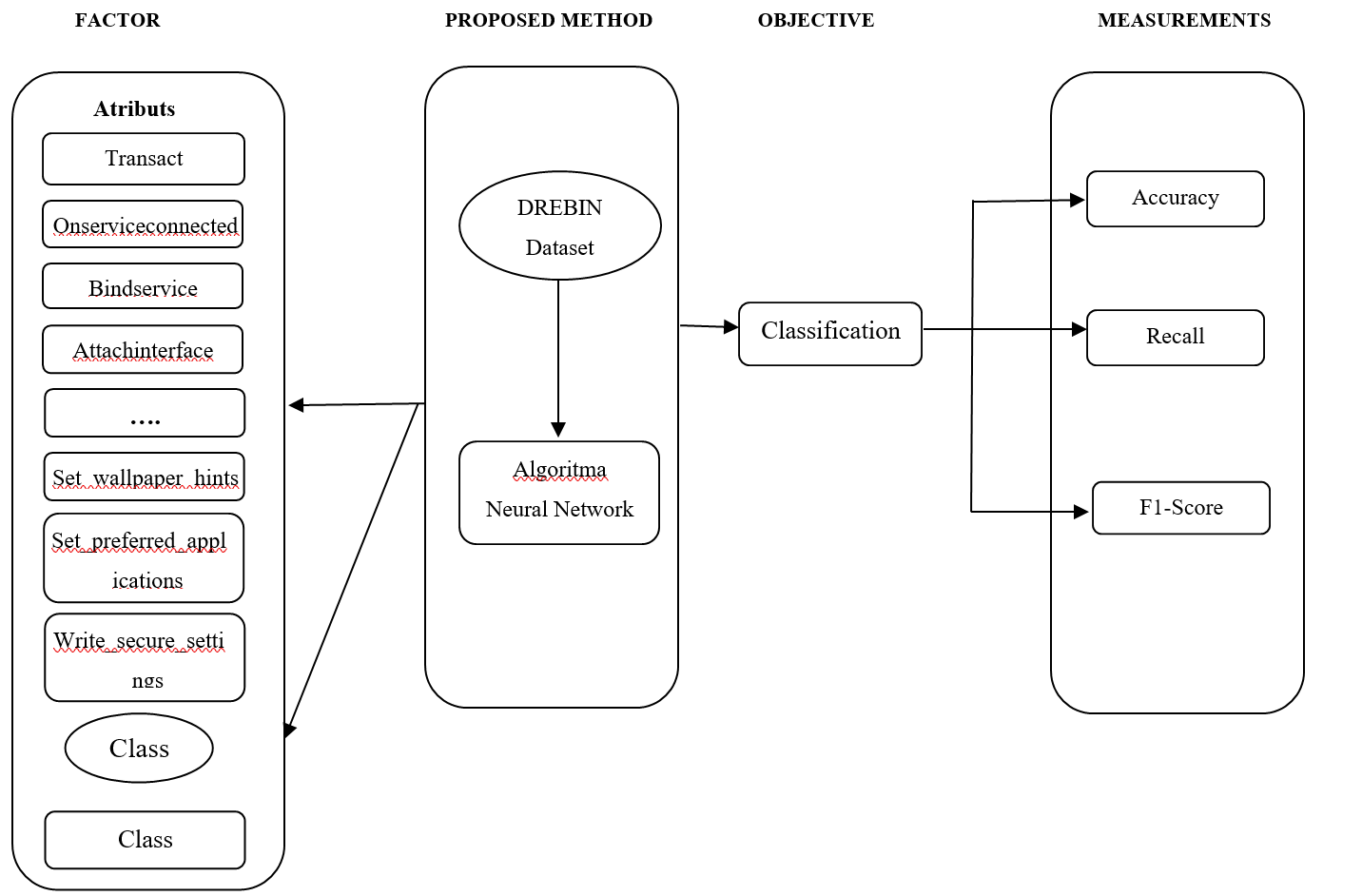
#### *Anaconda*

*Anaconda* merupakan salah satu aplikasi yang berfungsi sebagai distribusi bahasa pemrograman Python dan R yang memiliki sifat *open source*. Python banyak dimanfaatkan untuk berbagai perhitungan ilmiah, yang di dalamnya berupa machine learning, pengolahan data dengan ukuran besar, analisis prediksi, dan lain sebagainya. Anaconda memiliki tujuan untuk dapat menyederhanakan berbagai proses manajemen package ataupun deployment. Anaconda memiliki jumlah distribusi lebih dari 1500 package yang popular dan dapat diakses oleh berbagai platform sistem operasi seperti halnya Windows, Linux, dan MacOS (Efanntyo & Mitra, 2021).

#### *Jupyter*

*Jupyter Notbook* biasa juga di sebut *jupyter* adalah dokumen yang dapat dibagikan yang menggabungkan kode komputer, bahasa sederhana deskripsi, data, visualisasi kaya seperti model 3D, bagan, grafik, dan angka, dan kontrol interaktif. Buku catatan, bersama dengan editor (seperti *JupyterLab*), menyediakan lingkungan interaktif yang cepat untuk pembuatan prototipe dan menjelaskan kode, mengeksplorasi dan memvisualisasikan data, dan berbagi ide dengan Lain (Collonoval, Dafna, Ivanov, & Eric, 2023).

### **Kerangka Pemikiran**



Gambar 2. 2 Kerangka Pemikiran

Dalam kerangka pemiikiran diatas dijelaskan bahwa penelitian ini ditunjukan untuk mengetahui seberapa tinggi tingkat akurasi *Neural Network* dalam mendeteksi serangan *android* *malware* dan menghasilkan nilai akurasi , *F1*-*score*, dan *Recall.*

## Tinjauan Objek

### ***Android***

Android merupakan sistem operasi berbasis Linux yang dikembangkan oleh Google untuk perangkat mobile, seperti smartphone, tablet, smartwatch, dan perangkat lainnya. Sistem operasi dapat diilustrasikan sebagai ‘jembatan’ antara peranti (device) dan penggunanya, sehingga pengguna dapat berintraksi dengan devicenya dan menjalankan aplikasi-aplikasi yang tersedia pada *device* (Alfa Syahputra & Eva Maulina, 2016)*.*

Sistem android memiliki keunggulan, seperti sistem operasi bersifat open source, multitasking, kemudahan dalam notifikasi hingga banyaknya aplikasi atau software yang dapat dinikimati dengan menggunakan sistem android (Novrianda, Kunang, & Shaksono, 2014), Fitur utama Android termasuk notifikasi yang kuat, integrasi Google, keamanan, dan pembaruan sistem yang dirilis secara berkala, dan pengguna dapat mengharapkan peningkatan dalam hal fitur, keamanan, dan kinerja

### ***Malware***

*Malware* adalah singkatan dari "*malicious software*" yang merujuk kepada perangkat lunak berbahaya yang dirancang dengan tujuan jahat. Ini dapat menyusup ke dalam perangkat komputer atau digital lainnya tanpa izin pengguna dan dapat merusak data, mencuri informasi, atau merusak perangkat keras.

*Malware* merupakan sebuah software yang dapat menyusup ke sistem operasi sehingga dapat merusak sistem dan juga dapat mencuri file-file penting yang ada pada sistem. Malware atau juga disebut software perusak mencakup Virus Komputer, Trojan Horse, perangkat pengintai (spyware), perangkat iklan (adware) yang tidak jujur, perangkat jahat (crimeware) dan perangkat lunak lainnya yang berniat jahat dan tidak diinginkan (Novrianda, Kunang, & Shaksono, 2014).

Ada beberapa jenis malware yang perlu diketahui (Utomo, Setia Juli Irzal Ismail S.T., & Tafta Zani S.T., 2018), antara lain:

1. Trojan merupakan jenis malware yang disisipkan pada aplikasi. Tujuan trojan biasanya untuk mencuri data yang terdapat pada perangkat tanpa di ketahui pengguna.
2. Worm merupakan jenis malware yang menyerang lewat jaringan dengan cara mengirimkan file yang telah tersisipkan worm ini kepada penerima lewat jaringan. Worm dapat menggandakan dirinya sendiri pada sistem komputer.
3. Spyware merupakan jenis malware yang bekerja sebagai mata-mata untuk memata-matai pengguna komputer dan mengirimkan informasipada pihak yang mengirimkan spyware tersebut.
4. Adware merupakan jenis malware yang disisipkan kedalam aplikasi, malware ini akan menampilkan pop-up iklan terus menerus.
5. Virus merupakan jenis malware yang menyerang file dan menggandakan dirinya sendiri.

# METODE PEELITIAN

## Analisa Kebutuhan

Dalam penelitian ini pengujian data akan menggunakan aplikasi dengan Bahasa pemprograman python dengan tujuan untuk mendeteksi *Malware* dengan menggunakan algoritma *Neural Network*.

Untuk menunjang eksperimen dalam penelitian ini dibutuhkan satu unit laptop dalam melakukan simulasi dengan spesifikasi dibawah ini :

Tabel 3. 1 Kebutuhan Perangkat Keras dan Perangkat Lunak

|  |  |
| --- | --- |
| Perangkat Keras | Spesifikasi |
| *Processor* | Core i3 atau setara |
| *Hard Disk Drive* | Minimum 240 Gb |
| *Memory* | Minimum 8 Gb |
| Perangkat Lunak | Spesifikasi­ |
| Sistem Operasi | Windows 10 atau setara |
| Aplikasi Simulator | Pemrograman Python 3.x |
| Aplikasi pengolah kata | Minimum Microsoft Excell 2013 |
| *Library* python | *Pandas, Numpy,* Keras*, Tensorflow, Seaborn, Matplotlib, Scikit Learn* |

### **Teknik** **Analisis**

### **Pengumpulan Data (*Dataset*)**

Data yang di gunakan adalah *Dataset* drebin yang tersedia secara public di situs kaggle.com, dan dan telah banyak digunakan oleh para peneliti untuk mengevaluasi kinerja deteksi serangan, yang memiliki atribut sebagai berikut :

Tabel 3. 2 Pengumpulan Data

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Atribut** | **Keterangan** |
| 1 | *Transact* | Transaksi |
| 2 | *Onserviceconnected* | Metode koneksi ke layanan Android |
| 3 | *Bindservice* | Komponen untuk berinteraksi dengan layanan yang berjalan di latar belakang |
| 4 | *Attachinterface* | Kelas internal dalam Android yang digunakan untuk menghubungkan antara Binder dan antarmuka |
| 5 | *Serviceconnection* | Sebuah antarmuka yang digunakan untuk menghubungkan komponen aplikasi dengan layanan dalam Android |
| 6 | *Android.os.Binder* | Mengimplementasikan mekanisme komunikasi antar proses (IPC) dan menghubungkan objek layanan dengan klien |
| 7 | *Send\_sms* | *Permission* (izin) yang dapat diberikan kepada aplikasi Android untuk mengirim SMS |
| 8 | *Ljava.lang.Class.getcanonicalname* | Untuk mengembalikan sebuah string yang berisi nama kanonikal dari sebuah kelas |
| 9 | *Ljava.lang.Class.getmethods* | Untuk mengembalikan array yang berisi semua objek Method yang mewakili metode-metode publik yang dapat diakses dari kelas yang terkait |
| 10 | *Ljava.lang.Class.cast* | Untuk melakukan pengecoran objek menjadi tipe yang ditentukan oleh kelas yang terkait |
| 11 | *Ljava.net.urldecoder* | Untuk mendekode URL yang telah dienkripsi atau dikodekan menggunakan metode pengodean URL |
| 12 | *Android.content.pm.Signature* | Merepresentasikan tanda tangan digital |
| 13 | *Android.telephony.smsmanager* | Mengirim pesan SMS melalui perangkat telepon |
| 14 | *Read\_phone\_state* | Izin membaca informasi tentang status perangkat telepon |
| 15 | *Getbinder* | Mengembalikan objek IBinder yang mewakili antarmuka komunikasi yang dapat digunakan untuk berinteraksi dengan layanan |
| 16 | *Classloader* | Memuat kelas-kelas dan sumber daya dalam aplikasi Java |
| 17 | *Landroid.content.Context.registerreceiver* | Mendaftarkan sebuah objek *BroadcastReceiver* agar dapat menerima dan menangani pesan siaran |
| 18 | *Ljava.lang.Class.getfield* | Merepresentasikan sebuah *field* (variabel) publik dari suatu kelas |
| 19 | *Landroid.content.Context.unregisterreceiver* | Metode untuk membatalkan pendaftaran sebuah objek *BroadcastReceiver* yang sebelumnya telah didaftarkan menggunakan metode *registerReceiver*() |
| 20 | *Get\_accounts* | Izin untuk mengakses daftar akun yang ada di perangkat |
| 21 | *Receive\_sms* | Izin ini untuk menerima dan membaca pesan SMS |
| 22 | *Ljava.lang.Class.getdeclaredfield* | Metode untuk mendapatkan objek *Field* yang merepresentasikan sebuah *field* dalam suatu kelas, termasuk *field* yang bersifat *private* |
| 23 | *Read\_sms* | Izin untuk membaca konten dari pesan SMS yang ada di perangkat |
| 24 | *Getcallinguid* | Metode untuk mendapatkan ID UID (*User* ID) dari pemanggil (*caller*) saat ini dalam konteks Android. |
| 25 | *Ljavax.crypto.spec.secretkeyspec* | Untuk menyimpan kunci rahasia yang akan digunakan dalam algoritma kriptografi |
| 26 | *Android.intent.action.BOOT\_COMPLETED* | konstanta untuk menandakan bahwa sistem Android telah selesai melakukan proses *booting* dan aplikasi dapat menerima pesan siaran yang dikirim pada saat boot selesai |
| 27 | *Use\_credentials* | Izin memberikan akses ke akun pengguna yang disimpan di perangkat untuk otentikasi atau otorisasi |
| 28 | *Manage\_accounts* | Izin aplikasi untuk mengelola akun pengguna yang disimpan di perangkat |
| 29 | *Android.content.pm.packageinfo* | Kelas untuk mendapatkan informasi paket aplikasi |
| 30 | *Keyspec* | superkelas dari berbagai spesifikasi kunci rahasia yang digunakan dalam kriptografi |
| 31 | *Telephonymanager.getline1number* | Metode untuk mendapatkan nomor telepon dari perangkat jika tersedia |
| 32 | *Dexclassloader* | Kelas untuk memuat kelas dari file DEX (Dalvik *Executable*) eksternal |
| 33 | *Httpget.init* | Merupakan konstruktor untuk membuat objek HttpGet dalam pustaka Apache HttpClient |
| 34 | *Secretkey* | Merepresentasikan kunci rahasia dalam kriptografi |
| 35 | *Ljava.lang.Class.getmethod* | Metode untuk mendapatkan objek Method yang merepresentasikan sebuah metode dalam suatu kelas berdasarkan nama metode dan tipe parameter yang ditentukan |
| 36 | *System.loadlibrary* | Metode untuk memuat pustaka dinamis ke dalam JVM (Java *Virtual Machine*) |
| 37 | *Android.intent.action.SEND* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* untuk mengirim data ke aplikasi lain |
| 38 | *Ljavax.crypto.Cipher* | kelas untuk melakukan operasi enkripsi dan dekripsi dalam kriptografi |
| 39 | *Write\_sms* | Izin untuk menulis pesan SMS ke penyimpanan SMS di perangkat |
| 40 | *Read\_sync\_settings* | Izin untuk membaca pengaturan sinkronisasi yang terkait dengan akun pengguna di perangkat |
| 41 | *Authenticate\_accounts* | Izin untuk mengotentikasi akun pengguna yang terdaftar di perangkat |
| 42 | *Android.telephony.gsm.smsmanager* | Kelas untuk mengirim pesan SMS menggunakan teknologi GSM |
| 43 | *Write\_history\_bookmarks* | Izin aplikasi untuk menulis data bookmark ke penyimpanan bookmark di perangkat |
| 44 | *Telephonymanager.getsubscriberid* | Metode untuk mendapatkan ID *subscriber* (IMSI - *International Mobile Subscriber Identity*) dari kartu SIM yang terpasang di perangkat |
| 45 | *Mount* | Proses pemasangan atau pemasangan ulang sistem file yang terhubung ke perangkat |
| 46 | *Install\_packages* | Izin aplikasi untuk menginstal paket aplikasi baru di perangkat |
| 47 | *Runtime.getruntime* | metode untuk mendapatkan objek *runtime* yang merupakan representasi *runtime environment* |
| 48 | *Camera* | Izin aplikasi untuk mengakses dan menggunakan kamera perangkat untuk mengambil foto atau merekam video |
| 49 | *Ljava.lang.Object.getclass* | Metode untuk mendapatkan objek *Class* yang merepresentasikan kelas dari objek yang dipanggil |
| 50 | *Write\_sync\_settings* | Izin aplikasi untuk menulis pengaturan sinkronisasi yang terkait dengan akun pengguna di perangkat |
| 51 | *Read\_history\_bookmarks* | Izin aplikasi untuk membaca data riwayat bookmark dari penyimpanan bookmark di perangkat. |
| 52 | *Ljava.lang.Class.forname* | Metode untuk mendapatkan objek *Class* berdasarkan nama kelas yang diberikan sebagai argument |
| 53 | *Internet* | Izin aplikasi untuk mengakses jaringan internet |
| 54 | *Android.intent.action.PACKAGE\_REPLACED* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat sebuah paket aplikasi diinstal ulang pada perangkat |
| 55 | *Binder* | kelas untuk mengimplementasikan mekanisme komunikasi antar proses dan menghubungkan objek layanan dengan klien |
| 56 | *Android.intent.action.SEND\_MULTIPLE* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* untuk mengirim data ke aplikasi lain dalam bentuk multiple |
| 57 | *Record\_audio* | Izin ini memungkinkan aplikasi untuk merekam audio melalui mikrofon perangkat |
| 58 | *Ibinder* | antarmuka dalam Android untuk mewakili objek yang dapat dikirim antar proses (IPC) |
| 59 | *Android.os.ibinder* | unuk mengirim dan menerima pesan antar proses (IPC) antara proses yang berbeda di Android |
| 60 | *Createsubprocess* | pembuatan subprocess baru dari suatu program |
| 61 | *Nfc* | Izin untuk mengaktifkan dan mengakses fitur NFC pada perangkat |
| 62 | *Access\_location\_extra\_commands* | Izin aplikasi untuk mengakses perintah tambahan yang terkait dengan lokasi pada perangkat |
| 63 | *Urlclassloader* | Kelas memungkinkan aplikasi untuk memuat kelas yang tidak ada dalam paket aplikasi utama dari sumber yang berasal dari URL yang diberikan |
| 64 | *Write\_apn\_settings* | Izin aplikasi untuk mengubah pengaturan akses titik akses jaringan pada perangkat |
| 65 | *Abortbroadcast* | Metode untuk membatalkan atau menghentikan penyebaran pesan siaran yang diterima oleh BroadcastReceiver |
| 66 | *Bind\_remoteviews* | Izin aplikasi untuk menggunakan mekanisme RemoteViews untuk mengirim tampilan antarmuka pengguna yang dihasilkan dari aplikasi lain. |
| 67 | *Android.intent.action.TIME\_SET* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat pengaturan waktu di perangkat berubah. |
| 68 | *Read\_profile* | Izin aplikasi untuk membaca profil pengguna yang terkait dengan perangkat. |
| 69 | *Telephonymanager.getdeviceid* | Metode untuk mendapatkan ID perangkat unik (IMEI) dari perangkat telepon. |
| 70 | *Modify\_audio\_settings* | Izin aplikasi untuk mengubah pengaturan audio pada perangkat |
| 71 | *Getcallingpid* | Untuk mendapatkan ID PID (Process ID) dari pemanggil (*caller*) saat ini dalam konteks Android |
| 72 | *Read\_sync\_stats* | Izin aplikasi untuk membaca statistik sinkronisasi yang terkait dengan akun pengguna di perangkat |
| 73 | *Broadcast\_sticky* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat melakukan siaran sticky broadcast di Android |
| 74 | *Android.intent.action.PACKAGE\_REMOVED* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat sebuah paket aplikasi dihapus dari perangkat |
| 75 | *Android.intent.action.TIMEZONE\_CHANGED* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat pengaturan zona waktu pada perangkat berubah |
| 76 | *Wake\_lock* | Izin aplikasi untuk mempertahankan perangkat dalam keadaan terjaga agar tidak masuk ke mode tidur meskipun layar mati atau perangkat dalam keadaan tidak aktif |
| 77 | *Receive\_boot\_completed* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat perangkat Android selesai melakukan proses *booting* dan semua komponen sistem telah dimuat |
| 78 | *Restart\_packages* | Izin aplikasi untuk memulai kembali paket aplikasi lainnya. |
| 79 | *Ljava.lang.Class.getpackage* | Metode untuk mendapatkan objek Package yang merepresentasikan paket dari kelas yang diberikan. |
| 80 | *Chmod* | Fungsi mengubah mode akses file atau direktori pada sistem operasi berbasis Unix, termasuk Android |
| 81 | *Ljava.lang.Class.getdeclaredclasses* | Metode untuk mendapatkan daftar objek *Class* yang merepresentasikan kelas-kelas yang dideklarasikan secara langsung di dalam kelas yang diberikan. |
| 82 | *Android.intent.action.ACTION\_POWER\_DISCONNECTED* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat perangkat Android terputus dari sumber daya listrik |
| 83 | *Android.intent.action.PACKAGE\_ADDED* | Sebuah konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat sebuah paket aplikasi baru ditambahkan ke perangkat. |
| 84 | *Pathclassloader* | Kelas dalam Android yang digunakan untuk memuat kelas dari jalur file sistem |
| 85 | *Telephonymanager.getsimserialnumber* | Metode untuk mendapatkan nomor serial kartu SIM yang terpasang di perangkat. |
| 86 | *Runtime.load* | Metode untuk memuat pustaka dinamis pada saat *runtime* |
| 87 | *Telephonymanager.getcallstate* | Metode ini digunakan untuk mendapatkan status panggilan telepon saat ini |
| 88 | *Bluetooth* | Izin aplikasi untuk mengakses fitur Bluetooth pada perangkat |
| 89 | *Read\_calendar* | Izin aplikasi untuk membaca data kalender yang tersimpan di perangkat. |
| 90 | *Read\_call\_log* | Izin ini memungkinkan aplikasi untuk membaca riwayat panggilan yang tersimpan di perangkat |
| 91 | *Subscribed\_feeds\_write* | Izin aplikasi untuk menulis data ke penyimpanan yang digunakan oleh aplikasi pembaca berita untuk mengelola langganan dan berlangganan umpan berita |
| 92 | *Read\_external\_storage* | Izin aplikasi untuk membaca data dari penyimpanan eksternal |
| 93 | *Telephonymanager.getsimcountryiso* | Metode untuk mendapatkan kode ISO dari negara tempat kartu SIM terpasang |
| 94 | *Sendmultiparttextmessage* | Metode untuk mengirim pesan teks (SMS) dalam bentuk multipart |
| 95 | *Packageinstaller* | Kelas dalam Android untuk menginstal paket aplikasi baru di perangkat |
| 96 | *Vibrate* | Izin aplikasi untuk mengakses fitur getar pada perangkat |
| 97 | *Remount* | Fungsi untuk mengubah status per-mount file sistem yang terhubung ke perangkat |
| 98 | *Android.intent.action.ACTION\_SHUTDOWN* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat perangkat Android akan dimatikan |
| 99 | *Senddatamessage* | Metode untuk mengirim pesan data (SMS) dalam format PDUs yang dikirimkan melalui jaringan telepon |
| 100 | *Access\_network\_state* | Izin aplikasi untuk mengakses informasi tentang status jaringan, seperti koneksi internet yang sedang aktif atau tidak |
| 101 | *Chown* | Fungsi untuk mengubah kepemilikan file atau direktori |
| 102 | *Httppost.init* | Untuk melakukan permintaan HTTP POST |
| 103 | *Ljava.lang.Class.getclasses* | Metode untuk mendapatkan daftar objek *Class* yang merepresentasikan kelas-kelas yang dideklarasikan secara publik di dalam kelas yang diberikan |
| 104 | *Subscribed\_feeds\_read* | Izin ini untuk membaca (mengakses) data langganan dan berlangganan umpan berita dari penyimpanan yang digunakan oleh aplikasi pembaca berita |
| 105 | *Telephonymanager.isnetworkroaming* | Metode untuk memeriksa apakah perangkat sedang berada dalam kondisi roaming jaringan |
| 106 | *Change\_wifi\_multicast\_state* | Izin aplikasi untuk mengubah status multicast Wi-Fi pada perangkat |
| 107 | *Write\_calendar* | Izin aplikasi untuk menulis (mengubah) data kalender yang tersimpan di perangkat |
| 108 | *Android.intent.action.PACKAGE\_DATA\_CLEARED* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat data paket aplikasi dihapus |
| 109 | *Master\_clear* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat melakukan reset pabrik pada perangkat Android |
| 110 | *Httpurirequest* | Merupakan superkelas dari semua jenis permintaan HTTP, seperti HttpPost atau HttpGet |
| 111 | *Update\_device\_stats* | Izin aplikasi untuk memperbarui statistik perangkat yang terkait dengan penggunaan daya, penggunaan jaringan, atau statistik perangkat keras lainnya |
| 112 | *Write\_call\_log* | Izin aplikasi untuk menulis (mengubah) riwayat panggilan yang tersimpan di perangkat. |
| 113 | *Delete\_packages* | Izin plikasi untuk menghapus paket aplikasi lainnya dari perangkat |
| 114 | *Get\_tasks* | Izin aplikasi untuk mendapatkan informasi tentang tugas-tugas yang sedang berjalan di perangkat |
| 115 | *Global\_search* | Izin aplikasi untuk mengakses fitur pencarian global yang terdapat di perangkat |
| 116 | *Delete\_cache\_files* | Izin aplikasi untuk menghapus file-file cache yang dihasilkan oleh aplikasi lain di perangkat |
| 117 | *Write\_user\_dictionary* | Izin aplikasi untuk menulis (mengubah) kamus pengguna yang digunakan untuk saran pengetikan pada perangkat |
| 118 | *Android.intent.action.PACKAGE\_CHANGED* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat ada perubahan pada sebuah paket aplikasi di perangkat |
| 119 | *Android.intent.action.NEW\_OUTGOING\_CALL* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat melakukan panggilan keluar baru |
| 120 | *Reorder\_tasks* | Izin ini memungkinkan aplikasi untuk mengubah urutan tugas-tugas yang sedang berjalan di perangkat |
| 121 | *Write\_profile* | Izin aplikasi untuk menulis (mengubah) profil pengguna yang terkait dengan perangkat |
| 122 | *Set\_wallpaper* | Izin aplikasi untuk mengatur *wallpaper* pada perangkat |
| 123 | *Bind\_input\_method* | Izin aplikasi untuk terhubung dengan metode *input* yang ada di perangkat, seperti *keyboard virtual* |
| 124 | *Dividemessage* | Metode untuk membagi pesan teks yang terlalu panjang menjadi beberapa pesan yang lebih pendek secara otomatis. |
| 125 | *Read\_social\_stream* | Izin aplikasi untuk membaca data aliran sosial dari penyedia layanan sosial yang terintegrasi di perangkat. |
| 126 | *Read\_user\_dictionary* | Izin aplikasi untuk membaca kamus pengguna yang digunakan untuk saran pengetikan pada perangkat |
| 127 | *Process\_outgoing\_calls* | Izin aplikasi untuk memproses panggilan keluar yang dilakukan dari perangkat |
| 128 | *Call\_privileged* | Izin aplikasi untuk melakukan panggilan telepon menggunakan izin khusus yang memiliki akses lebih tinggi dari panggilan biasa. |
| 129 | *Runtime.exec* | metode untuk menjalankan perintah atau proses baru di sistem operasi yang menjalankan aplikasi. |
| 130 | *Bind\_wallpaper* | Izin aplikasi untuk terhubung dengan wallpaper yang ada di perangkat, untuk mengatur dan mengelola tampilan latar belakang. |
| 131 | *Receive\_wap\_push* | Izin aplikasi untuk menerima pesan WAP *Push* yang dikirim melalui protokol *Wireless Application Protocol* pada perangkat |
| 132 | *Dump* | Izin aplikasi untuk mengakses informasi debug sistem yang tidak tersedia untuk aplikasi biasa |
| 133 | *Battery\_stats* | Izin aplikasi untuk mengakses informasi tentang penggunaan baterai perangkat |
| 134 | *Access\_coarse\_location* | Izin aplikasi untuk mengakses informasi lokasi kasar (koordinat geografis) perangkat menggunakan sumber lokasi yang kurang presisi, seperti jaringan seluler atau Wi-Fi |
| 135 | *Set\_time* | Izin aplikasi untuk mengatur waktu pada perangkat |
| 136 | *Android.intent.action.SENDTO* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat mengirim pesan ke alamat tertentu |
| 137 | *Write\_social\_stream* | Izin aplikasi untuk menulis (mengubah) data aliran sosial yang terkait dengan pengguna di perangkat |
| 138 | *Write\_settings* | Izin aplikasi untuk menulis (mengubah) pengaturan sistem pada perangkat |
| 139 | *Reboot* | Izin aplikasi untuk memulai kembali perangkat |
| 140 | *Bluetooth\_admin* | Izin aplikasi untuk mengakses dan mengelola fitur Bluetooth pada perangkat |
| 141 | *Telephonymanager.getnetworkoperator* | Metode untuk mendapatkan kode operator jaringan yang terhubung dengan perangkat |
| 142 | */System/bin* | Jalur ke direktori /bin dalam sistem operasi Android |
| 143 | *Messengerservice* | Kelas Android untuk mengimplementasikan komunikasi antara komponen aplikasi menggunakan konsep pengiriman pesan |
| 144 | *Bind\_device\_admin* | Izin aplikasi untuk terhubung ke administrator perangkat, yang memberikan akses khusus untuk mengendalikan beberapa aspek keamanan perangkat |
| 145 | *Write\_gservices* | Izin ini memungkinkan aplikasi untuk menulis (mengubah) data layanan Google terkait dengan perangkat, |
| 146 | *Iremoteservice* | Antarmuka untuk berinteraksi dengan layanan yang dijalankan di proses terpisah, memungkinkan komunikasi antara komponen yang berjalan di proses yang berbeda |
| 147 | *Kill\_background\_processes* | Izin aplikasi untuk menghentikan proses-proses latar belakang yang sedang berjalan di perangkat |
| 148 | *Set\_alarm* | Izin ini memungkinkan aplikasi untuk membuat pengaturan alarm pada perangkat. |
| 149 | *Account\_manager* | komponen Android untuk mengelola akun pengguna pada perangkat |
| 150 | */System/app* | jalur ke direktori /app dalam sistem operasi Android |
| 151 | *Android.intent.action.CALL* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat melakukan panggilan telepon |
| 152 | *Status\_bar* | Terdapat di bagian atas layar pada perangkat Android |
| 153 | *Telephonymanager.getsimoperator* | Metode untuk mendapatkan kode operator SIM yang terhubung dengan perangkat |
| 154 | *Persistent\_activity* | Izin aplikasi untuk menjalankan aktivitas secara persisten |
| 155 | *Change\_network\_state* | Izin aplikasi untuk mengubah status jaringan |
| 156 | *Onbind* | Metode yang digunakan dalam komponen layanan di Android |
| 157 | *Process.start* | Untuk menjalankan proses eksternal di Android |
| 158 | *Android.intent.action.SCREEN\_ON* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat layar perangkat dihidupkan |
| 159 | *Context.bindservice* | Metode untuk mengikat komponen klien ke layanan yang sedang berjalan di perangkat |
| 160 | *Receive\_mms* | Izin aplikasi untuk menerima dan memproses pesan MMS yang diterima oleh perangkat |
| 161 | *Set\_time\_zone* | Izin aplikasi untuk mengatur zona waktu pada perangkat |
| 162 | *Android.intent.action.BATTERY\_OKAY* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat baterai perangkat telah mencapai level yang cukup untuk menghidupkan perangkat |
| 163 | *Control\_location\_updates* | Izin aplikasi untuk mengendalikan pembaruan lokasi pada perangkat |
| 164 | *Broadcast\_wap\_push* | Izin aplikasi untuk menyiar pesan WAP *Push* yang diterima oleh perangkat |
| 165 | *Bind\_accessibility\_service* | Izin aplikasi untuk terhubung ke layanan aksesibilitas yang ada di perangkat |
| 166 | *Add\_voicemail* | Izin aplikasi untuk menambahkan pesan suara baru ke aplikasi suara bawaan pada perangkat |
| 167 | *Call\_phone* | Izin aplikasi untuk melakukan panggilan telepon langsung dari perangkat. |
| 168 | *Processbuilder* | Kelas yang digunakan untuk membuat dan mengelola proses eksternal |
| 169 | *Bind\_appwidget* | Izin aplikasi untuk terhubung ke yang ada di perangkat, untuk mengatur dan mengelola tampilan *widget* |
| 170 | *Flashlight* | Izin khusus untuk mengontrol atau mengakses fitur lampu kilat pada perangkat |
| 171 | *Read\_logs* | Izin aplikasi untuk membaca log sistem yang terkait dengan perangka |
| 172 | *Ljava.lang.Class.getresource* | Metode untuk mendapatkan URL yang mengidentifikasi lokasi sumber daya yang terkait dengan kelas tersebut |
| 173 | *Defineclass* | Metode untuk mendefinisikan (membaca) definisi kelas dari data biner yang diberikan |
| 174 | *Set\_process\_limit* | Izin aplikasi untuk mengatur batas proses pada perangkat |
| 175 | *Android.intent.action.PACKAGE\_RESTARTED* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat sebuah paket aplikasi di-*restart* |
| 176 | *Mount\_unmount\_filesystems* | Izin aplikasi untuk mengakses dan mengontrol proses pemasangan dan pencabutan sistem berkas |
| 177 | *Bind\_text\_service* | Izin aplikasi untuk terhubung ke layanan teks yang ada di perangkat |
| 178 | *Install\_location\_provider* | Izin aplikasi untuk menginstal penyedia lokasi baru di perangkat |
| 179 | *Android.intent.action.CALL\_BUTTON* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat tombol panggilan pada perangkat ditekan |
| 180 | *Android.intent.action.SCREEN\_OFF* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat layar perangkat dimatikan |
| 181 | *Findclass* | Metode untuk mencari dan memuat kelas dengan nama yang diberikan dalam sistem kelas |
| 182 | *System\_alert\_window* | Izin aplikasi untuk menampilkan jendela penggembalaan sistem di atas aplikasi lain |
| 183 | *Mount\_format\_filesystems* | Izin aplikasi untuk mengakses dan mengontrol proses pemformatan sistem berkas pada perangkat |
| 184 | *Change\_configuration* | Izin ini memungkinkan aplikasi untuk mengubah konfigurasi perangkat |
| 185 | *Clear\_app\_user\_data* | Izin ini memungkinkan aplikasi untuk menghapus data pengguna yang terkait dengan aplikasi tersebut |
| 186 | *Intent.action.RUN* | Proses dalam konteks aplikasi atau sistem tertentu |
| 187 | *Android.intent.action.SET\_WALLPAPER* | Konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat mengatur gambar latar belakang pada perangkat |
| 188 | *Change\_wifi\_state* | Izin aplikasi untuk mengubah status Wi-Fi pada perangkat |
| 189 | *Read\_frame\_buffer* | Izin aplikasi untuk membaca isi framebuffer yang digunakan untuk menampilkan tampilan layar pada perangkat |
| 190 | *Access\_surface\_flinger* | Izin aplikasi untuk mengakses SurfaceFlinger, |
| 191 | *Runtime.loadlibrary* | Metode untuk memuat pustaka native dari file dengan ekstensi .so (*shared object*) ke dalam aplikasi yang sedang berjalan. |
| 192 | *Broadcast\_sms* | Izin aplikasi untuk menyiar pesan SMS diterima oleh perangkat |
| 193 | *Expand\_status\_bar* | Izin aplikasi untuk mengembangkan bilah status pada perangkat, sehingga menampilkan notifikasi dan ikon yang ada di dalamnya. |
| 194 | *Internal\_system\_window* | Izin aplikasi untuk menampilkan jendela sistem internal di atas aplikasi lain |
| 195 | *Android.intent.action.BATTERY\_LOW* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat level baterai perangkat mencapai batas rendah |
| 196 | *Set\_activity\_watcher* | Izin aplikasi menyediakan layar yang digunakan pengguna untuk berinteraksi guna melakukan sesuatu |
| 197 | *Write\_contacts* | Izin ini memungkinkan aplikasi untuk menulis (mengubah) data kontak pada perangkat |
| 198 | *Android.intent.action.ACTION\_POWER\_CONNECTED* | konstanta string yang digunakan sebagai tipe aksi dalam *Intent* yang dikirim saat perangkat terhubung ke daya |
| 199 | *Bind\_vpn\_service* | Izin aplikasi untuk terhubung ke layanan VPN yang ada di perangkat |
| 200 | *Disable\_keyguard* | Izin aplikasi untuk menonaktifkan kunci layar pada perangkat, |
| 201 | *Access\_mock\_location* | Izin aplikasi untuk mengakses informasi lokasi palsu yang diberikan oleh pengguna |
| 202 | *Get\_package\_size* | Izin aplikasi untuk mendapatkan ukuran paket aplikasi lain di perangkat |
| 203 | *Modify\_phone\_state* | Izin aplikasi untuk mengubah status telepon pada perangkat |
| 204 | *Change\_component\_enabled\_state* | Izin aplikasi untuk mengubah status komponen yang diaktifkan atau dinonaktifkan dalam aplikasi lain |
| 205 | *Clear\_app\_cache* | Izin aplikasi untuk membersihkan cache aplikasi |
| 206 | *Set\_orientation* | Izin aplikasi untuk mengubah orientasi layar pada perangkat |
| 207 | *Read\_contacts* | Izin aplikasi untuk membaca data kontak yang ada di perangkat |
| 208 | *Device\_power* | Izin aplikasi untuk mengelola power otomatis |
| 209 | *Hardware\_test* | Izin aplikasi untuk mengakses fitur tes perangkat keras pada perangkat |
| 210 | *Access\_wifi\_state* | Izin aplikasi untuk mengakses informasi status dan konfigurasi Wi-Fi pada perangkat |
| 211 | *Write\_external\_storage* | Izin aplikasi untuk menulis (mengubah) data pada penyimpanan eksternal perangkat |
| 212 | *Access\_fine\_location* | Izin aplikasi untuk mengakses informasi lokasi yang sangat tepat dari perangkat |
| 213 | *Set\_wallpaper\_hints* | Izin aplikasi untuk memberikan petunjuk tentang preferensi pengaturan latar belakang kepada sistem |
| 214 | *Set\_preferred\_applications* | Izin aplikasi untuk mengatur aplikasi yang dipilih sebagai aplikasi default untuk tugas-tugas tertentu |
| 215 | *Write\_secure\_settings* | Izin aplikasi untuk menulis (mengubah) pengaturan keamanan yang terkait dengan perangkat |
| 216 | *Class* | Merepresentasikan sebuah objek kelas informasi tentang variabel, B = *Benign* S=*Malware* |

### **Pengolahan Data Awal**

Dalam *Dataset* ini memiliki lima kategori utama yaitu API *call signature* memiliki 72 kategori, *Class* memiliki 2 kategori, *Commands signature* memiliki 6 kategori, *Intent* memiliki 23 kategori dan *Manifest Permission* memiliki 113 kategori.

Pada pengelolahan data awal atau tahap *preprocessin*g yang akan digunakan dalam penelitian ini sebelumnya sebanyak 15036 data dengan 215 atribut dan 1 kelas menjadi 7259 data 215 atribut dan 1 kelas di karenakan terdapat data duplikat sehingga perlu di hapus. Adapun rincian data yang digunakan seperti tabel di bawah ini :

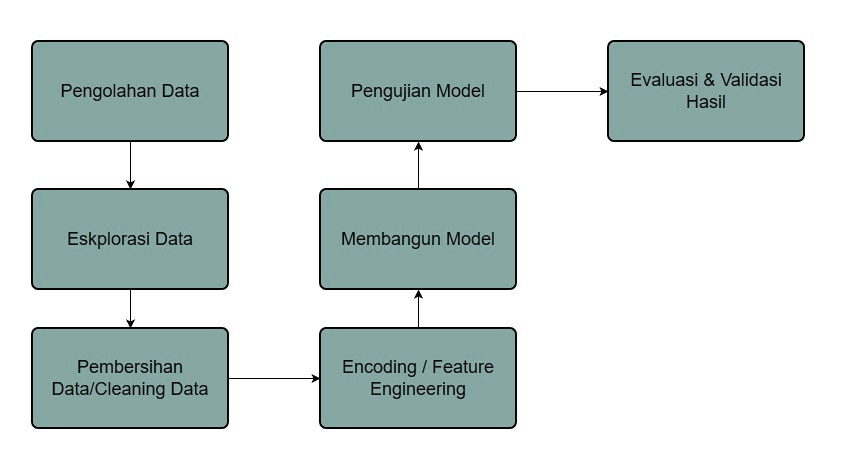
Tabel 3. 3 Pengolahan Data Awal

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Atribut** | **Tipe Data** | **Nilai** |
| 1 | *Transact* | Int | 0-1 |
| 2 | *Onserviceconnected* | Int | 0-1 |
| 3 | *Bindservice* | Int | 0-1 |
| 4 | *Attachinterface* | Int | 0-1 |
| 5 | *Serviceconnection* | Int | 0-1 |
| 6 | *Android.os.binder* | Int | 0-1 |
| 7 | *Send\_sms* | Int | 0-1 |
| 8 | *Ljava.lang.class.getcanonicalname* | Int | 0-1 |
| 9 | *Ljava.lang.class.getmethods* | Int | 0-1 |
| 10 | *Ljava.lang.class.cast* | Int | 0-1 |
| 11 | *Ljava.net.urldecoder* | Int | 0-1 |
| 12 | *Android.content.pm.signature* | Int | 0-1 |
| 13 | *Android.telephony.smsmanager* | Int | 0-1 |
| 14 | *Read\_phone\_state* | Int | 0-1 |
| 15 | *Getbinder* | Int | 0-1 |
| 16 | *Classloader* | Int | 0-1 |
| 17 | *Landroid.content.context.registerreceiver* | Int | 0-1 |
| 18 | *Ljava.lang.class.getfield* | Int | 0-1 |
| 19 | *Landroid.content.context.unregisterreceiver* | Int | 0-1 |
| 20 | *Get\_accounts* | Int | 0-1 |
| 21 | *Receive\_sms* | Int | 0-1 |
| 22 | *Ljava.lang.class.getdeclaredfield* | Int | 0-1 |
| 23 | *Read\_sms* | Int | 0-1 |
| 24 | *Getcallinguid* | Int | 0-1 |
| 25 | *Ljavax.crypto.spec.secretkeyspec* | Int | 0-1 |
| 26 | *Android.intent.action.boot\_completed* | Int | 0-1 |
| 27 | *Use\_credentials* | Int | 0-1 |
| 28 | *Manage\_accounts* | Int | 0-1 |
| 29 | *Android.content.pm.packageinfo* | Int | 0-1 |
| 30 | *Keyspec* | Int | 0-1 |
| 31 | *Telephonymanager.getline1number* | Int | 0-1 |
| 32 | *Dexclassloader* | Int | 0-1 |
| 33 | *Httpget.init* | Int | 0-1 |
| 34 | *Secretkey* | Int | 0-1 |
| 35 | *Ljava.lang.class.getmethod* | Int | 0-1 |
| 36 | *System.loadlibrary* | Int | 0-1 |
| 37 | *Android.intent.action.send* | Int | 0-1 |
| 38 | *Ljavax.crypto.cipher* | Int | 0-1 |
| 39 | *Write\_sms* | Int | 0-1 |
| 40 | *Read\_sync\_settings* | Int | 0-1 |
| 41 | *Authenticate\_accounts* | Int | 0-1 |
| 42 | *Android.telephony.gsm.smsmanager* | Int | 0-1 |
| 43 | *Write\_history\_bookmarks* | Int | 0-1 |
| 44 | *Telephonymanager.getsubscriberid* | Int | 0-1 |
| 45 | *Mount* | Int | 0-1 |
| 46 | *Install\_packages* | Int | 0-1 |
| 47 | *Runtime.getruntime* | Int | 0-1 |
| 48 | *Camera* | Int | 0-1 |
| 49 | *Ljava.lang.object.getclass* | Int | 0-1 |
| 50 | *Write\_sync\_settings* | Int | 0-1 |
| 51 | *Read\_history\_bookmarks* | Int | 0-1 |
| 52 | *Ljava.lang.class.forname* | Int | 0-1 |
| 53 | *Internet* | Int | 0-1 |
| 54 | *Android.intent.action.package\_replaced* | Int | 0-1 |
| 55 | *Binder* | Int | 0-1 |
| 56 | *Android.intent.action.send\_multiple* | Int | 0-1 |
| 57 | *Record\_audio* | Int | 0-1 |
| 58 | *Ibinder* | Int | 0-1 |
| 59 | *Android.os.ibinder* | Int | 0-1 |
| 60 | *Createsubprocess* | Int | 0-1 |
| 61 | *Nfc* | Int | 0-1 |
| 62 | *Access\_location\_extra\_commands* | Int | 0-1 |
| 63 | *Urlclassloader* | Int | 0-1 |
| 64 | *Write\_apn\_settings* | Int | 0-1 |
| 65 | *Abortbroadcast* | Int | 0-1 |
| 66 | *Bind\_remoteviews* | Int | 0-1 |
| 67 | *Android.intent.action.time\_set* | Int | 0-1 |
| 68 | *Read\_profile* | Int | 0-1 |
| 69 | *Telephonymanager.getdeviceid* | Int | 0-1 |
| 70 | *Modify\_audio\_settings* | Int | 0-1 |
| 71 | *Getcallingpid* | Int | 0-1 |
| 72 | *Read\_sync\_stats* | Int | 0-1 |
| 73 | *Broadcast\_sticky* | Int | 0-1 |
| 74 | *Android.intent.action.package\_removed* | Int | 0-1 |
| 75 | *Android.intent.action.timezone\_changed* | Int | 0-1 |
| 76 | *Wake\_lock* | Int | 0-1 |
| 77 | *Receive\_boot\_completed* | Int | 0-1 |
| 78 | *Restart\_packages* | Int | 0-1 |
| 79 | *Ljava.lang.class.getpackage* | Int | 0-1 |
| 80 | *Chmod* | Int | 0-1 |
| 81 | *Ljava.lang.class.getdeclaredclasses* | Int | 0-1 |
| 82 | *Android.intent.action.action\_power\_disconnected* | Int | 0-1 |
| 83 | *Android.intent.action.package\_added* | Int | 0-1 |
| 84 | *Pathclassloader* | Int | 0-1 |
| 85 | *Telephonymanager.getsimserialnumber* | Int | 0-1 |
| 86 | *Runtime.load* | Int | 0-1 |
| 87 | *Telephonymanager.getcallstate* | Int | 0-1 |
| 88 | *Bluetooth* | Int | 0-1 |
| 89 | *Read\_calendar* | Int | 0-1 |
| 90 | *Read\_call\_log* | Int | 0-1 |
| 91 | *Subscribed\_feeds\_write* | Int | 0-1 |
| 92 | *Read\_external\_storage* | Int | 0-1 |
| 93 | *Telephonymanager.getsimcountryiso* | *Object* | 0,1, ?,10 |
| 94 | *Sendmultiparttextmessage* | Int | 0-1 |
| 95 | *Packageinstaller* | Int | 0-1 |
| 96 | *Vibrate* | Int | 0-1 |
| 97 | *Remount* | Int | 0-1 |
| 98 | *Android.intent.action.action\_shutdown* | Int | 0-1 |
| 99 | *Senddatamessage* | Int | 0-1 |
| 100 | *Access\_network\_state* | Int | 0-1 |
| 101 | *Chown* | Int | 0-1 |
| 102 | *Httppost.init* | Int | 0-1 |
| 103 | *Ljava.lang.class.getclasses* | Int | 0-1 |
| 104 | *Subscribed\_feeds\_read* | Int | 0-1 |
| 105 | *Telephonymanager.isnetworkroaming* | Int | 0-1 |
| 106 | *Change\_wifi\_multicast\_state* | Int | 0-1 |
| 107 | *Write\_calendar* | Int | 0-1 |
| 108 | *Android.intent.action.package\_data\_cleared* | Int | 0-1 |
| 109 | *Master\_clear* | Int | 0-1 |
| 110 | *Httpurirequest* | Int | 0-1 |
| 111 | *Update\_device\_stats* | Int | 0-1 |
| 112 | *Write\_call\_log* | Int | 0-1 |
| 113 | *Delete\_packages* | Int | 0-1 |
| 114 | *Get\_tasks* | Int | 0-1 |
| 115 | *Global\_search* | Int | 0-1 |
| 116 | *Delete\_cache\_files* | Int | 0-1 |
| 117 | *Write\_user\_dictionary* | Int | 0-1 |
| 118 | *Android.intent.action.package\_changed* | Int | 0-1 |
| 119 | *Android.intent.action.new\_outgoing\_call* | Int | 0-1 |
| 120 | *Reorder\_tasks* | Int | 0-1 |
| 121 | *Write\_profile* | Int | 0-1 |
| 122 | *Set\_wallpaper* | Int | 0-1 |
| 123 | *Bind\_input\_method* | Int | 0-1 |
| 124 | *Dividemessage* | Int | 0-1 |
| 125 | *Read\_social\_stream* | Int | 0-1 |
| 126 | *Read\_user\_dictionary* | Int | 0-1 |
| 127 | *Process\_outgoing\_calls* | Int | 0-1 |
| 128 | *Call\_privileged* | Int | 0-1 |
| 129 | *Runtime.exec* | Int | 0-1 |
| 130 | *Bind\_wallpaper* | Int | 0-1 |
| 131 | *Receive\_wap\_push* | Int | 0-1 |
| 132 | *Dump* | Int | 0-1 |
| 133 | *Battery\_stats* | Int | 0-1 |
| 134 | *Access\_coarse\_location* | Int | 0-1 |
| 135 | *Set\_time* | Int | 0-1 |
| 136 | *Android.intent.action.sendto* | Int | 0-1 |
| 137 | *Write\_social\_stream* | Int | 0-1 |
| 138 | *Write\_settings* | Int | 0-1 |
| 139 | *Reboot* | Int | 0-1 |
| 140 | *Bluetooth\_admin* | Int | 0-1 |
| 141 | *Telephonymanager.getnetworkoperator* | Int | 0-1 |
| 142 | */System/bin* | Int | 0-1 |
| 143 | *Messengerservice* | Int | 0-1 |
| 144 | *Bind\_device\_admin* | Int | 0-1 |
| 145 | *Write\_gservices* | Int | 0-1 |
| 146 | *Iremoteservice* | Int | 0-1 |
| 147 | *Kill\_background\_processes* | Int | 0-1 |
| 148 | *Set\_alarm* | Int | 0-1 |
| 149 | *Account\_manager* | Int | 0-1 |
| 150 | */System/app* | Int | 0-1 |
| 151 | *Android.intent.action.call* | Int | 0-1 |
| 152 | *Status\_bar* | Int | 0-1 |
| 153 | *Telephonymanager.getsimoperator* | Int | 0-1 |
| 154 | *Persistent\_activity* | Int | 0-1 |
| 155 | *Change\_network\_state* | Int | 0-1 |
| 156 | *Onbind* | Int | 0-1 |
| 157 | *Process.start* | Int | 0-1 |
| 158 | *Android.intent.action.screen\_on* | Int | 0-1 |
| 159 | *Context.bindservice* | Int | 0-1 |
| 160 | *Receive\_mms* | Int | 0-1 |
| 161 | *Set\_time\_zone* | Int | 0-1 |
| 162 | *Android.intent.action.battery\_okay* | Int | 0-1 |
| 163 | *Control\_location\_updates* | Int | 0-1 |
| 164 | *Broadcast\_wap\_push* | Int | 0-1 |
| 165 | *Bind\_accessibility\_service* | Int | 0-1 |
| 166 | *Add\_voicemail* | Int | 0-1 |
| 167 | *Call\_phone* | Int | 0-1 |
| 168 | *Processbuilder* | Int | 0-1 |
| 169 | *Bind\_appwidget* | Int | 0-1 |
| 170 | *Flashlight* | Int | 0-1 |
| 171 | *Read\_logs* | Int | 0-1 |
| 172 | *Ljava.lang.class.getresource* | Int | 0-1 |
| 173 | *Defineclass* | Int | 0-1 |
| 174 | *Set\_process\_limit* | Int | 0-1 |
| 175 | *Android.intent.action.package\_restarted* | Int | 0-1 |
| 176 | *Mount\_unmount\_filesystems* | Int | 0-1 |
| 177 | *Bind\_text\_service* | Int | 0-1 |
| 178 | *Install\_location\_provider* | Int | 0-1 |
| 179 | *Android.intent.action.call\_button* | Int | 0-1 |
| 180 | *Android.intent.action.screen\_off* | Int | 0-1 |
| 181 | *Findclass* | Int | 0-1 |
| 182 | *System\_alert\_window* | Int | 0-1 |
| 183 | *Mount\_format\_filesystems* | Int | 0-1 |
| 184 | *Change\_configuration* | Int | 0-1 |
| 185 | *Clear\_app\_user\_data* | Int | 0-1 |
| 186 | *Intent.action.run* | Int | 0-1 |
| 187 | *Android.intent.action.set\_wallpaper* | Int | 0-1 |
| 188 | *Change\_wifi\_state* | Int | 0-1 |
| 189 | *Read\_frame\_buffer* | Int | 0-1 |
| 190 | *Access\_surface\_flinger* | Int | 0-1 |
| 191 | *Runtime.loadlibrary* | Int | 0-1 |
| 192 | *Broadcast\_sms* | Int | 0-1 |
| 193 | *Expand\_status\_bar* | Int | 0-1 |
| 194 | *Internal\_system\_window* | Int | 0-1 |
| 195 | *Android.intent.action.battery\_low* | Int | 0-1 |
| 196 | *Set\_activity\_watcher* | Int | 0-1 |
| 197 | *Write\_contacts* | Int | 0-1 |
| 198 | *Android.intent.action.action\_power\_connected* | Int | 0-1 |
| 199 | *Bind\_vpn\_service* | Int | 0-1 |
| 200 | *Disable\_keyguard* | Int | 0-1 |
| 201 | *Access\_mock\_location* | Int | 0-1 |
| 202 | *Get\_package\_size* | Int | 0-1 |
| 203 | *Modify\_phone\_state* | Int | 0-1 |
| 204 | *Change\_component\_enabled\_state* | Int | 0-1 |
| 205 | *Clear\_app\_cache* | Int | 0-1 |
| 206 | *Set\_orientation* | Int | 0-1 |
| 207 | *Read\_contacts* | Int | 0-1 |
| 208 | *Device\_power* | Int | 0-1 |
| 209 | *Hardware\_test* | Int | 0-1 |
| 210 | *Access\_wifi\_state* | Int | 0-1 |
| 211 | *Write\_external\_storage* | Int | 0-1 |
| 212 | *Access\_fine\_location* | Int | 0-1 |
| 213 | *Set\_wallpaper\_hints* | Int | 0-1 |
| 214 | *Set\_preferred\_applications* | Int | 0-1 |
| 215 | *Write\_secure\_settings* | Int | 0-1 |
| 216 | *Class* | *Object* | B, S |

## Perancangan Penelitian

Dalam penelitian ini menggunakan metode *Neural Network* dalam mendeteksi serangan *Malware*, dimana data yang diambil merupakan *Dataset* Drebin yang tersedia secara publik di situs kaggle.com.

Tahapan tahapan yang dibutuhkan dalam penelitian ini adalah : Pengolahan data, Eskplorasi data, Pembersihan data, *Encoding*, kemudian Membangun model, lalu pengujian model dan terkahir evaluasi & validasi hasil seperti terlihat pada gambar dibawah ini :



Gambar 3. 1 Rancangan Penelitian

Berdasarkan gambar diatas, dapat dijelaskan bahwa tahapan dalam penelitian ini adalah:

1. Pengolahan Data

Data yang digunakan dalam penelitian ini menggunakan *Dataset* drebin terdiri dari 215 atribut dan 1 *ouput* dan memiliki dimensi 15036 baris dan 216 kolom, sebagai berikut :

Tabel 3. 4 Pengolahan Data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***Index*** | ***Transact*** | ***Onserviceconnected*** | ***Bindservice*** | **…** | ***Set\_preferred\_applications*** | ***Write\_secure\_settings*** | ***Class*** |
| 0 | 0 | 0 | 0 | … | 0 | 0 | S |
| 1 | 0 | 0 | 0 | … | 0 | 0 | S |
| 2 | 0 | 0 | 0 | … | 0 | 0 | S |
| 3 | 0 | 0 | 0 | … | 0 | 0 | S |
| … | … | … | … | … | … | … | … |
| 15032 | 0 | 0 | 0 | … | 0 | 0 | B |
| 15033 | 0 | 0 | 0 | … | 0 | 0 | B |
| 15034 | 1 | 1 | 1 | … | 0 | 0 | B |
| 15035 | 1 | 1 | 1 | … | 0 | 0 | B |

1. Eksplorasi Data

Pada tahap ini, penulis melakukan pengecekan data yaitu berupa pencarian isi, jumlah dan tipe data pada setiap atribut, mencari apa terdapat nilai kosong dan mencari data yang duplikat.

1. Pembersihan Data

Pada fase penelitian ini dilakukan pembersihan data yang terdapat dalam kumpulan data untuk memastikan kualitas dan kevalidan data yang akan digunakan dengan membersihkan data yang duplikat, *outlier*, dan *missing value*.

1. *Encoding* (*Feature Engineering)*

*Encoding* mengacu pada proses mengubah data dari format atau representasi satu ke format atau representasi lainnya. Ini sering dilakukan untuk mengubah data kategorikal menjadi bentuk numerik yang dapat diproses oleh algoritma pembelajaran mesin atau model statistik.

1. Membangun Model

Selanjutnya yaitu membangun model dengan algoritma *Neural Network*. Ini diimplementasikan dengan menggunakan *library* scikit-learn, Keras, Tensorflow, Seaborn, Matplotlib. Scikit-learn atau Sklearn adalah library berbasis Python untuk membangun model pembelajaran mesin. Ia menyediakan banyak algoritma pembelajaran untuk regresi, pengelompokan, dan klasifikasi. Sklearn kompatibel dengan NumPy dan SciPy.

1. Pengujian Model

Pengujian model adalah proses mengevaluasi kinerja dan keefektifan model pembelajaran mesin atau statistik yang telah dibangun. Tujuan pengujian model adalah untuk memahami sejauh mana model dapat melakukan prediksi atau klasifikasi yang akurat dan andal terhadap data yang tidak dikenal.

Ada beberapa metrik dan teknik yang digunakan dalam pengujian model, termasuk:

threshold = 0.5

# Mengonversi nilai-nilai berkelanjutan menjadi biner (0 atau 1)

1. y\_pred\_binary = np.where(y\_pred > threshold, 1, 0)dengan membandingkan prediksi model dengan nilai sebenarnya dari data pengujian.
2. *Recall* (*Recall*): Juga dikenal sebagai sensitivitas, menunjukkan seberapa baik model dapat mengidentifikasi data positif. Ini mengukur proporsi data positif yang benar diklasifikasikan dari keseluruhan data positif yang ada.
3. F1-Skor (*F1-Score*): Ini adalah ukuran gabungan yang menggabungkan presisi dan *Recall*. F1-Skor memberikan keseimbangan antara presisi dan *Recall*, dan biasanya digunakan ketika ada ketidakseimbangan kelas dalam *Dataset*.
4. Matriks Confusion (*Confusion Matrix*): Ini adalah tabel yang digunakan untuk menggambarkan klasifikasi yang benar dan salah yang dilakukan oleh model. Matriks ini mencakup empat istilah: *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).
5. Evaluasi & Validasi Hasil

Evaluasi hasil dalam penelitian ini adalah proses pengujian model mesin learning untuk deteksi malware yang telah dibuat dengan menggunakan pola data baru yaitu data uji 1089 baris data 216 kolom. Dimana pada saat pembuatan proses model mesin learning menggunakan data latih sebanyak 7259 baris data 216 kolom.

Tabel 3. 5 Evaluasi & validasi hasil

|  |  |
| --- | --- |
| *Dataset* | Jumlah |
| Data Latih | 7259 baris data 216 kolom |
| Data Uji | 1089 baris data 216 kolom |

# IMPLEMENTASI DAN PENGUJIAN

## Hasil

Tujuan penelitian ini adalah untuk menerapkan metode algoritma *Neural Network* untuk mendeteksi serangan andorid *malware*. Penelitian ini dilakukan terhadapat *Dataset* drebin.

### **Persiapan Data**

Dalam penelitian menggunakan *Dataset* Drebin yang tersedia secara publik di situs kaggle.com tersedia secara publik . Seperti pada tabel di bawah ini :

Tabel 4. 1 Persiapan Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama | Record | Feature | *Class* |
| 1 | *Dataset* Drebin | 15036 | 215 | 1 |

*Dataset* ini banyak digunakan oleh para peneliti sebelumnya (Refhaldo, Eko Budiarto, Putri Anggun Sari & Sella Monica, 2022),

### **Implementasi Metode**

## Pembahasan

### **Pengolahan Data**

Pada tahap ini dilakukan pengolahan data dengan mengkonversi data non-numerik menjadi format yang dapat dipahami oleh model atau algoritma *Neural Network*.

### **Eksplorasi Data**

Pada tahap ini dilakukan proses analisa dari *dataset* dengan tujuan untuk mengenali dataset yang digunakan terkait dengan dimensi, tipe serta pola data yang ada pada dataset.

#### Analisa Data

Dalam analisa data pada dataset dilakukan pemisahan terkait dengan *class* atau label yang berfungsi sebagai target atau *output* dengan atribut atau *feature* yang berfungsi sebagai input

Tabel 4. 2 Analisa Data

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Atribut** | **Type Data** |
|  |  |  |
| 1 | *Transact* | Int |
| 2 | *Onserviceconnected* | Int |
| 3 | *Bindservice* | Int |
| 4 | *Attachinterface* | Int |
| 5 | *Serviceconnection* | Int |
| 6 | *Android.os.Binder* | Int |
| 7 | *Send\_sms* | Int |
| 8 | *Ljava.lang.Class.getcanonicalname* | Int |
| 9 | *Ljava.lang.Class.getmethods* | Int |
| 10 | *Ljava.lang.Class.cast* | Int |
| 11 | *Ljava.net.urldecoder* | Int |
| 12 | *Android.content.pm.Signature* | Int |
| 13 | *Android.telephony.smsmanager* | Int |
| 14 | *Read\_phone\_state* | Int |
| 15 | *Getbinder* | Int |
| 16 | *Classloader* | Int |
| 17 | *Landroid.content.Context.registerreceiver* | Int |
| 18 | *Ljava.lang.Class.getfield* | Int |
| 19 | *Landroid.content.Context.unregisterreceiver* | Int |
| 20 | *Get\_accounts* | Int |
| 21 | *Receive\_sms* | Int |
| 22 | *Ljava.lang.Class.getdeclaredfield* | Int |
| 23 | *Read\_sms* | Int |
| 24 | *Getcallinguid* | Int |
| 25 | *Ljavax.crypto.spec.secretkeyspec* | Int |
| 26 | *Android.intent.action.BOOT\_COMPLETED* | Int |
| 27 | *Use\_credentials* | Int |
| 28 | *Manage\_accounts* | Int |
| 29 | *Android.content.pm.packageinfo* | Int |
| 30 | *Keyspec* | Int |
| 31 | *Telephonymanager.getline1number* | Int |
| 32 | *Dexclassloader* | Int |
| 33 | *Httpget.init* | Int |
| 34 | *Secretkey* | Int |
| 35 | *Ljava.lang.Class.getmethod* | Int |
| 36 | *System.loadlibrary* | Int |
| 37 | *Android.intent.action.SEND* | Int |
| 38 | *Ljavax.crypto.Cipher* | Int |
| 39 | *Write\_sms* | Int |
| 40 | *Read\_sync\_settings* | Int |
| 41 | *Authenticate\_accounts* | Int |
| 42 | *Android.telephony.gsm.smsmanager* | Int |
| 43 | *Write\_history\_bookmarks* | Int |
| 44 | *Telephonymanager.getsubscriberid* | Int |
| 45 | *Mount* | Int |
| 46 | *Install\_packages* | Int |
| 47 | *Runtime.getruntime* | Int |
| 48 | *Camera* | Int |
| 49 | *Ljava.lang.Object.getclass* | Int |
| 50 | *Write\_sync\_settings* | Int |
| 51 | *Read\_history\_bookmarks* | Int |
| 52 | *Ljava.lang.Class.forname* | Int |
| 53 | *Internet* | Int |
| 54 | *Android.intent.action.PACKAGE\_REPLACED* | Int |
| 55 | *Binder* | Int |
| 56 | *Android.intent.action.SEND\_MULTIPLE* | Int |
| 57 | *Record\_audio* | Int |
| 58 | *Ibinder* | Int |
| 59 | *Android.os.ibinder* | Int |
| 60 | *Createsubprocess* | Int |
| 61 | *Nfc* | Int |
| 62 | *Access\_location\_extra\_commands* | Int |
| 63 | *Urlclassloader* | Int |
| 64 | *Write\_apn\_settings* | Int |
| 65 | *Abortbroadcast* | Int |
| 66 | *Bind\_remoteviews* | Int |
| 67 | *Android.intent.action.TIME\_SET* | Int |
| 68 | *Read\_profile* | Int |
| 69 | *Telephonymanager.getdeviceid* | Int |
| 70 | *Modify\_audio\_settings* | Int |
| 71 | *Getcallingpid* | Int |
| 72 | *Read\_sync\_stats* | Int |
| 73 | *Broadcast\_sticky* | Int |
| 74 | *Android.intent.action.PACKAGE\_REMOVED* | Int |
| 75 | *Android.intent.action.TIMEZONE\_CHANGED* | Int |
| 76 | *Wake\_lock* | Int |
| 77 | *Receive\_boot\_completed* | Int |
| 78 | *Restart\_packages* | Int |
| 79 | *Ljava.lang.Class.getpackage* | Int |
| 80 | *Chmod* | Int |
| 81 | *Ljava.lang.Class.getdeclaredclasses* | Int |
| 82 | *Android.intent.action.ACTION\_POWER\_DISCONNECTED* | Int |
| 83 | *Android.intent.action.PACKAGE\_ADDED* | Int |
| 84 | *Pathclassloader* | Int |
| 85 | *Telephonymanager.getsimserialnumber* | Int |
| 86 | *Runtime.load* | Int |
| 87 | *Telephonymanager.getcallstate* | Int |
| 88 | *Bluetooth* | Int |
| 89 | *Read\_calendar* | Int |
| 90 | *Read\_call\_log* | Int |
| 91 | *Subscribed\_feeds\_write* | Int |
| 92 | *Read\_external\_storage* | Int |
| 93 | *Telephonymanager.getsimcountryiso* | *Object* |
| 94 | *Sendmultiparttextmessage* | Int |
| 95 | *Packageinstaller* | Int |
| 96 | *Vibrate* | Int |
| 97 | *Remount* | Int |
| 98 | *Android.intent.action.ACTION\_SHUTDOWN* | Int |
| 99 | *Senddatamessage* | Int |
| 100 | *Access\_network\_state* | Int |
| 101 | *Chown* | Int |
| 102 | *Httppost.init* | Int |
| 103 | *Ljava.lang.Class.getclasses* | Int |
| 104 | *Subscribed\_feeds\_read* | Int |
| 105 | *Telephonymanager.isnetworkroaming* | Int |
| 106 | *Change\_wifi\_multicast\_state* | Int |
| 107 | *Write\_calendar* | Int |
| 108 | *Android.intent.action.PACKAGE\_DATA\_CLEARED* | Int |
| 109 | *Master\_clear* | Int |
| 110 | *Httpurirequest* | Int |
| 111 | *Update\_device\_stats* | Int |
| 112 | *Write\_call\_log* | Int |
| 113 | *Delete\_packages* | Int |
| 114 | *Get\_tasks* | Int |
| 115 | *Global\_search* | Int |
| 116 | *Delete\_cache\_files* | Int |
| 117 | *Write\_user\_dictionary* | Int |
| 118 | *Android.intent.action.PACKAGE\_CHANGED* | Int |
| 119 | *Android.intent.action.NEW\_OUTGOING\_CALL* | Int |
| 120 | *Reorder\_tasks* | Int |
| 121 | *Write\_profile* | Int |
| 122 | *Set\_wallpaper* | Int |
| 123 | *Bind\_input\_method* | Int |
| 124 | *Dividemessage* | Int |
| 125 | *Read\_social\_stream* | Int |
| 126 | *Read\_user\_dictionary* | Int |
| 127 | *Process\_outgoing\_calls* | Int |
| 128 | *Call\_privileged* | Int |
| 129 | *Runtime.exec* | Int |
| 130 | *Bind\_wallpaper* | Int |
| 131 | *Receive\_wap\_push* | Int |
| 132 | *Dump* | Int |
| 133 | *Battery\_stats* | Int |
| 134 | *Access\_coarse\_location* | Int |
| 135 | *Set\_time* | Int |
| 136 | *Android.intent.action.SENDTO* | Int |
| 137 | *Write\_social\_stream* | Int |
| 138 | *Write\_settings* | Int |
| 139 | *Reboot* | Int |
| 140 | *Bluetooth\_admin* | Int |
| 141 | *Telephonymanager.getnetworkoperator* | Int |
| 142 | */System/bin* | Int |
| 143 | *Messengerservice* | Int |
| 144 | *Bind\_device\_admin* | Int |
| 145 | *Write\_gservices* | Int |
| 146 | *Iremoteservice* | Int |
| 147 | *Kill\_background\_processes* | Int |
| 148 | *Set\_alarm* | Int |
| 149 | *Account\_manager* | Int |
| 150 | */System/app* | Int |
| 151 | *Android.intent.action.CALL* | Int |
| 152 | *Status\_bar* | Int |
| 153 | *Telephonymanager.getsimoperator* | Int |
| 154 | *Persistent\_activity* | Int |
| 155 | *Change\_network\_state* | Int |
| 156 | *Onbind* | Int |
| 157 | *Process.start* | Int |
| 158 | *Android.intent.action.SCREEN\_ON* | Int |
| 159 | *Context.bindservice* | Int |
| 160 | *Receive\_mms* | Int |
| 161 | *Set\_time\_zone* | Int |
| 162 | *Android.intent.action.BATTERY\_OKAY* | Int |
| 163 | *Control\_location\_updates* | Int |
| 164 | *Broadcast\_wap\_push* | Int |
| 165 | *Bind\_accessibility\_service* | Int |
| 166 | *Add\_voicemail* | Int |
| 167 | *Call\_phone* | Int |
| 168 | *Processbuilder* | Int |
| 169 | *Bind\_appwidget* | Int |
| 170 | *Flashlight* | Int |
| 171 | *Read\_logs* | Int |
| 172 | *Ljava.lang.Class.getresource* | Int |
| 173 | *Defineclass* | Int |
| 174 | *Set\_process\_limit* | Int |
| 175 | *Android.intent.action.PACKAGE\_RESTARTED* | Int |
| 176 | *Mount\_unmount\_filesystems* | Int |
| 177 | *Bind\_text\_service* | Int |
| 178 | *Install\_location\_provider* | Int |
| 179 | *Android.intent.action.CALL\_BUTTON* | Int |
| 180 | *Android.intent.action.SCREEN\_OFF* | Int |
| 181 | *Findclass* | Int |
| 182 | *System\_alert\_window* | Int |
| 183 | *Mount\_format\_filesystems* | Int |
| 184 | *Change\_configuration* | Int |
| 185 | *Clear\_app\_user\_data* | Int |
| 186 | *Intent.action.RUN* | Int |
| 187 | *Android.intent.action.SET\_WALLPAPER* | Int |
| 188 | *Change\_wifi\_state* | Int |
| 189 | *Read\_frame\_buffer* | Int |
| 190 | *Access\_surface\_flinger* | Int |
| 191 | *Runtime.loadlibrary* | Int |
| 192 | *Broadcast\_sms* | Int |
| 193 | *Expand\_status\_bar* | Int |
| 194 | *Internal\_system\_window* | Int |
| 195 | *Android.intent.action.BATTERY\_LOW* | Int |
| 196 | *Set\_activity\_watcher* | Int |
| 197 | *Write\_contacts* | Int |
| 198 | *Android.intent.action.ACTION\_POWER\_CONNECTED* | Int |
| 199 | *Bind\_vpn\_service* | Int |
| 200 | *Disable\_keyguard* | Int |
| 201 | *Access\_mock\_location* | Int |
| 202 | *Get\_package\_size* | Int |
| 203 | *Modify\_phone\_state* | Int |
| 204 | *Change\_component\_enabled\_state* | Int |
| 205 | *Clear\_app\_cache* | Int |
| 206 | *Set\_orientation* | Int |
| 207 | *Read\_contacts* | Int |
| 208 | *Device\_power* | Int |
| 209 | *Hardware\_test* | Int |
| 210 | *Access\_wifi\_state* | Int |
| 211 | *Write\_external\_storage* | Int |
| 212 | *Access\_fine\_location* | Int |
| 213 | *Set\_wallpaper\_hints* | Int |
| 214 | *Set\_preferred\_applications* | Int |
| 215 | *Write\_secure\_settings* | Int |
| 216 | *Class* | *Object* |

Pada dataset ini, terdapat kolom masih dalam tipe data objek yang dimana nantinya harus di *encode* agar dapat di proses.

Kemudian dilakukan pengecekan terhadap sampel data yaitu *Class* atau label.

1. *Class* atau Label

*Class* atau label pada dataset ini adalah *class* yang akan berfungsi sebagai output dari model mesin *learning* yang dibuat, hasil analisa pada kolom *class* dapat dilihat pada tabel dibawah ini.

**Tabel 4. 3 Class atau label**

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Keunikan setiap data | 2 Baris Data  B = *Benign*  S = *Malware* |
| Frequensi kemunculan | 9476 |
| Data yang sering muncul | B |

1. Atribut atau *feature*

Pengecekan sampel data dari sepuluh atribut atau *feature (**Transact, Android.os.binder, Onserviceconnected, Bindservice, Serviceconnection, Write\_History\_Bookmarks, Read\_History\_Bookmarks, Send\_sms, Read\_sms, Telephonymanager.getline1number*.

* 1. *Transact*

Pada kolom *transact* ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.426443, standar deviasi 0.494576, nilai minimal 0 dan maksimal 1.

**Tabel 4. 4 *Transact***

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.426443 |
| Standar Deviasi | 0.494576 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Android.os.binder*

Pada kolom *Android.os.binder* ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.486898, standar deviasi 0.499845, nilai minimal 0 dan maksimal 1.

**Tabel 4. 5 *Android.os.binder***

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.486898 |
| Standar Deviasi | 0.499845 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Onserviceconnected*

Pada kolom *Onserviceconnected*  ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.446595, standar deviasi 0.497156, nilai minimal 0 dan maksimal 1.

Tabel 4. 6 *Onserviceconnected*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.446595 |
| Standar Deviasi | 0.497156 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Bindservice*

Pada kolom *Bindservice* ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.446595, standar deviasi 0.497156, nilai minimal 0 dan maksimal 1.

Tabel 4. 7 *Bindservice*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.442671 |
| Standar Deviasi | 0.496719 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Serviceconnection*

Pada kolom *Serviceconnection* ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.444932, standar deviasi 0.496975, nilai minimal 0 dan maksimal 1.

Tabel 4. 8 *Serviceconnection*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.444932 |
| Standar Deviasi | 0.496975 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Write\_History\_Bookmarks*

Pada kolom date ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.079343, standar deviasi 0.270282, nilai minimal 0 dan maksimal 1.

Tabel 4. 9 *Write\_History\_Bookmarks*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.079343 |
| Standar Deviasi | 0.270282 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Read\_History\_Bookmarks*

Pada kolom date ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.092910, standar deviasi 0.290316, nilai minimal 0 dan maksimal 1.

Tabel 4. 10 *Read\_History\_Bookmarks*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.092910 |
| Standar Deviasi | 0.290316 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Send\_sms*

Pada kolom date ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.236632, standar deviasi 0.425029, nilai minimal 0 dan maksimal 1.

Tabel 4. 11 *Send\_sms*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.236632 |
| Standar Deviasi | 0.425029 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Read\_sms*

Pada kolom date ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.186752, standar deviasi 0.389725, nilai minimal 0 dan maksimal 1.

Tabel 4. 12 Read\_sms

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.186752 |
| Standar Deviasi | 0.389725 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

* 1. *Telephonymanager.getline1number*

Pada kolom date ini terdapat 15036 baris data yang dimana data tesebut memiliki nilai rata-rata 0.247739, standar deviasi 0.431714, nilai minimal 0 dan maksimal 1.

Tabel 4. 13 *Telephonymanager.getline1number*

|  |  |
| --- | --- |
| Keterangan | Nilai |
| Jumlah Baris | 15036 Baris Data |
| Rata-rata | 0.247739 |
| Standar Deviasi | 0.431714 |
| Nilai Minimal | 0 |
| Nilai Maksimal | 1 |

#### Statistik Data

Pada proses ini dilakukan proses penggalian informasi terkait nilai-nilai statistik pada dataset, khususnya pada kolok-kolom yg bertipe numerik.

Tabel 4. 14 Statistik Data

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Transact** | **Onserviceconnection** | **Bindservice** | **Atachinterface** | **…** | **Set\_preferred\_applications** | **Write\_secure\_settings** | **Class** |
| **Count** | 7259 | 7259 | 7259 | 7259 | … | 7259 | 7259 | 7259 |
| **Mean** | 0.542499 | 0.562199 | 0.557239 | 0.557239 | … | 0.009781 | 0.038986 | 0.236809 |
| **Std** | 0.498225 | 0.557239 | 0.496747 | 0.499527 | … | 0.009781 | 0.009781 | 0.425154 |
| **Min** | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 |
| **25%** | 0 | 0 | 0 | 0 | … | 0 | 0 | 0 |
| **50%** | 1 | 1 | 1 | 1 | … | 0 | 0 | 0 |
| **75%** | 1 | 1 | 1 | 1 | … | 0 | 0 | 0 |
| **Max** | 1 | 1 | 1 | 1 | … | 1 | 1 | 1 |

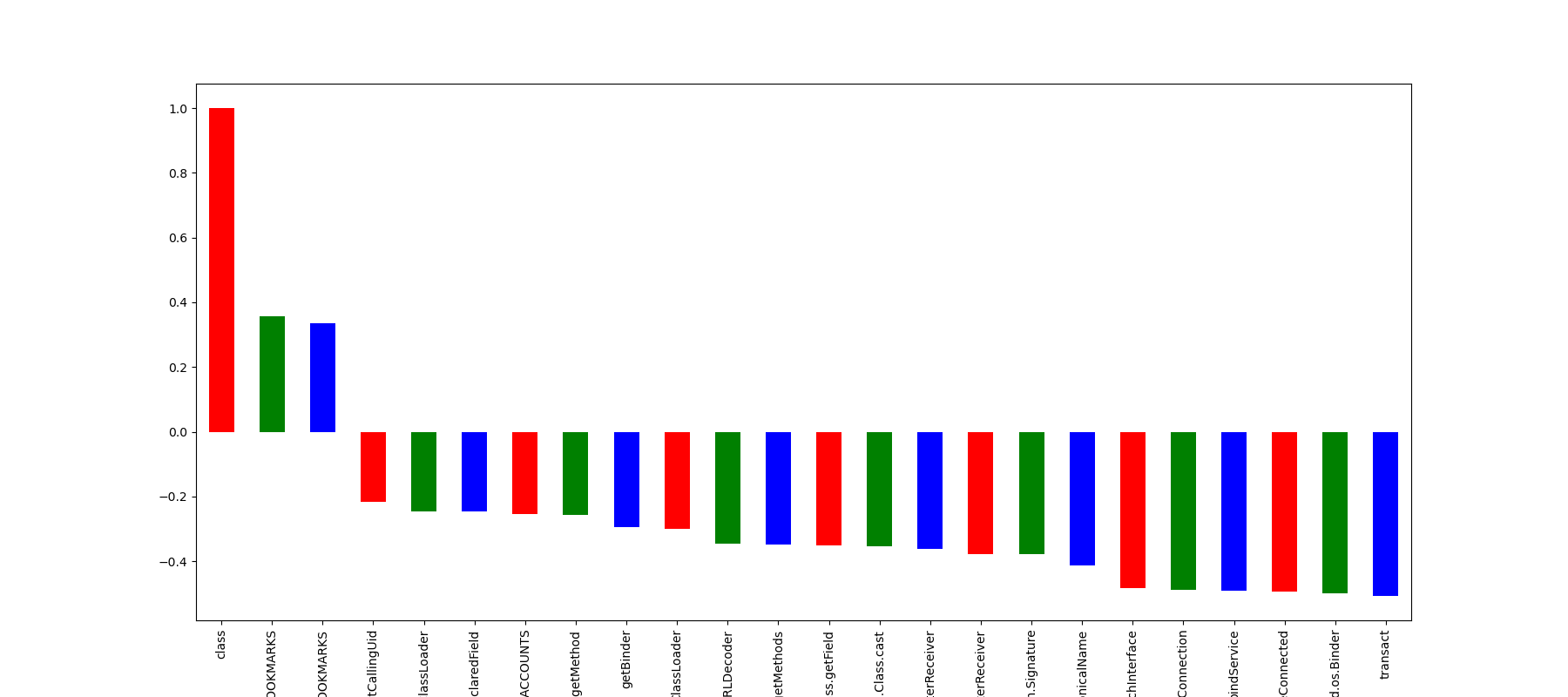
#### Korelasi Atribut (*Input*) terhadap *Label* (*Output*)

Pada tahap ini, melakukan analisis korelasi antara atribut-atribut yang digunakan sebagai *input*, dengan *label* atau *output*.

Tabel 4. 15 Korelasi Atribut

|  |  |
| --- | --- |
| Atribut | *C*lass |
| *Transact* | -0.507 |
| *Onserviceconnected* | -0.493 |
| *Bindservice* | -0.491 |
| *Attachinterface* | -0.483 |
| *Serviceconnection* | -0.489 |
| … | … |
| *Access\_fine\_location* | 0.127 |
| *Set\_wallpaper\_hints* | 0.006 |
| *Set\_preferred\_applications* | 0.060 |
| *Write\_secure\_settings* | -0.059 |
| *Class* | 1.000 |

Jika nilai korelasi mendekati angka 1, itu menunjukkan adanya hubungan yang kuat antara atribut input dan output. Semakin dekat nilai korelasi dengan 1, semakin kuat hubungan linier antara kedua variabel tersebut. Dalam hal ini, perubahan dalam atribut input akan secara signifikan mempengaruhi atribut output, dan sebaliknya.



Gambar 4. 1 Korelasi Atribut

Pada penelitain terdapat 2 jenis korelasi sebagai berikut :

1. korelasi positif

Atribut yg berpengaruh positif terhadap label atau output, hasil korelasi menunjukkan bahwa lima data sampel atribut atau fitur memiliki hubungan yang kuat dengan *class* atau label, seperti *Write\_History\_Bookmarks, Read\_History\_Bookmarks, Send\_sms, Read\_sms, Telephonymanager.getline1number*. Fitur-fitur tersebut dapat memberikan kontribusi besar dalam memprediksi label atau kelas yang dituju.

1. Korelasi negatif

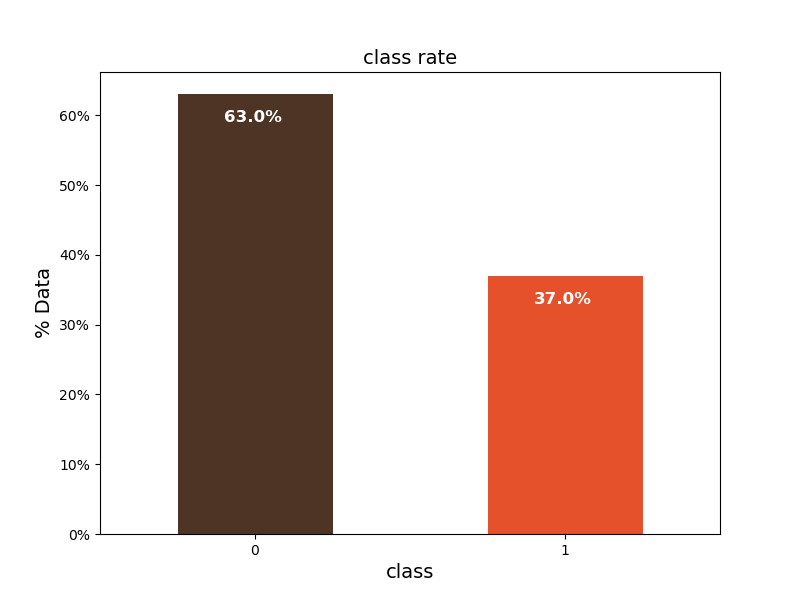
Dari 5 sampel data atribut yg memilki pengaruh negatif terhadap label atau *class*, fitur yang memiliki korelasi rendah dengan *class* ataulabel seperti *Transact, Android.os.binder, Onserviceconnected, Bindservice, Serviceconnection* tidak memberikan kontribusi besar dalam memprediksi label atau kelas yang dituju.

#### Visualisasi Data

Pada tahap ini akan dilakukan eksplorsi data menggunakan grafik atau diagram untuk memberikan gambaran terhadap sebuah data pada tiap kolom.

1. *Class* atau label

Dari total 7259 data *Class,* terdapat sebanyak 5540 (63%) data yang bukan merupakan mallware dan sebanyak 1719 (37%) data yang merupakan *malware*.



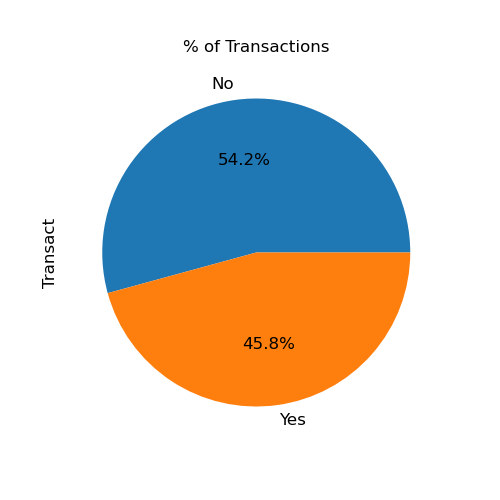
Gambar 4. 2 *Class*

1. Atribut atau *feature*

Sampel data dari sepuluh atribut atau *feature (Transact, Android.os.binder, Onserviceconnected, Bindservice, Serviceconnection, Write\_History\_Bookmarks, Read\_History\_Bookmarks, Send\_sms, Read\_sms, Telephonymanager.getline1number*.

1. *Transact*
2. Distribusi data *Transact*

Hasil dari distribusi data *transact* diperoleh nilai sebagai berikut.

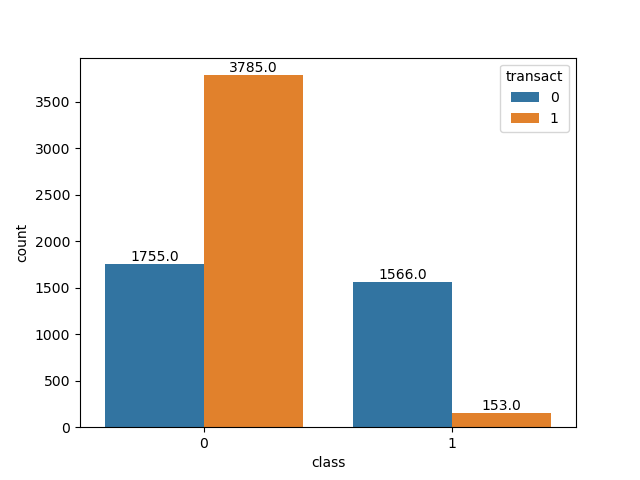


Gambar 4. 3 Transact

Dari total 7259 data *transact* terdapat sebanyak (54.2%) data yang bukan merupakan *transact* dan sebanyak (45.8 %) data yang merupakan *transact.*

1. *Transact* v*s Class*

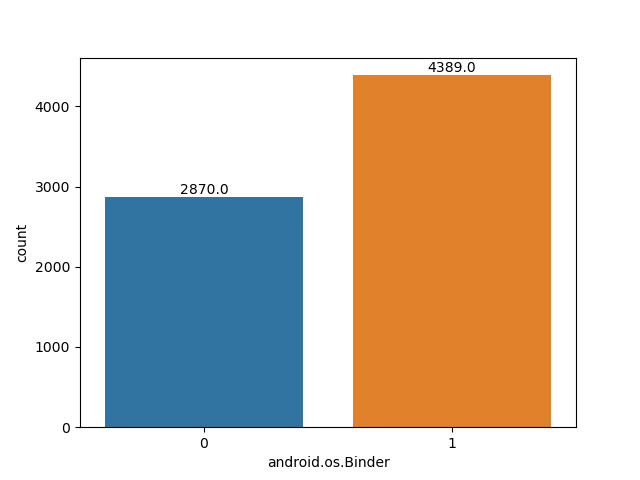
Hasil dari visualisasi dari *Transact* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 4 *Transact vs Class*

1. Sebanyak 3938 data yang merupakan *transact*, 153 data termasuk kedalam kategori *malware*, dan 3785 data yang tidak termasuk kategori *malware*.
2. Sebanyak 3321 data bukan merupakan transact, 1566 data termasuk kedalam kategori malware, dan 1755 data yang
3. *Android.os.binder*
   1. Distribusi data *Android.os.binder*

Hasil dari distribusi data *Android.os.binder* diperoleh nilai sebagai berikut.

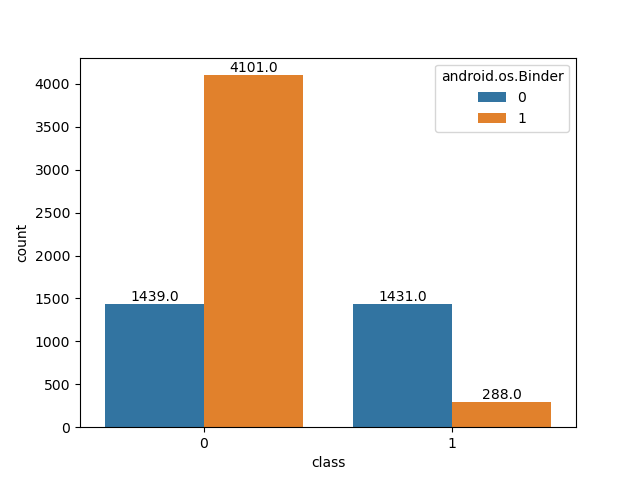


Gambar 4. 5 *Android.os.binder*

Dari total 7259 data *Android.os.binder* terdapat sebanyak 2870 data yang bukan merupakan *Android.os.binder* dan sebanyak 4389 data yang merupakan *Android.os.binder.*

* 1. *Android.os.binder vs class*

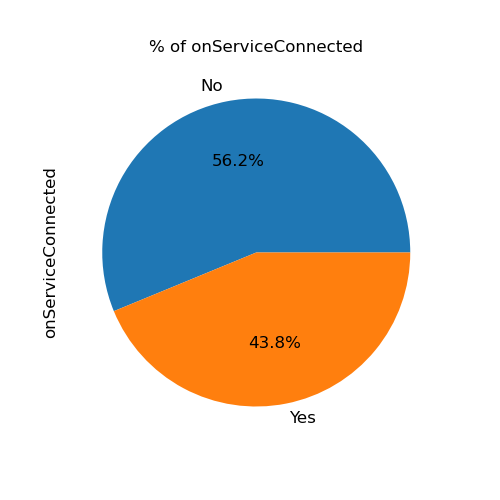
Hasil dari visualisasi dari *Andorid.os.binder* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 6 Android.os.binder vs class

1. Sebanyak 4389 data yang merupakan *Android.OS.Binder*, 288 data termasuk kedalam kategori *malware*, dan 4101 data yang tidak termasuk kategori *malware*.
2. Sebanyak 2870 data bukan merupakan *Android.OS.Binder*, 1431 data termasuk kedalam kategori *malware*, dan 1439 data yang tidak termasuk kategori *malware*.
3. *Onserviceconnected*
4. Distribusi data *Onserviceconnected*

Hasil dari distribusi data *Onserviceconnected* diperoleh nilai sebagai berikut.

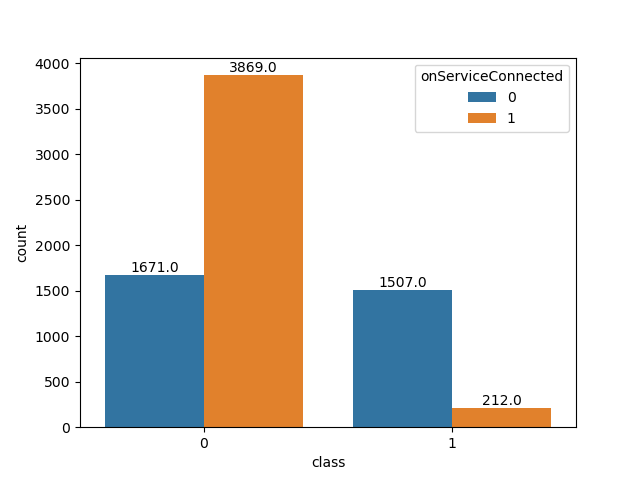


Gambar 4. 7 *Onserviceconnected*

Dari total 7259 data *Onserviceconnected*  terdapat sebanyak (56.2%) data yang bukan merupakan *Onserviceconnected* dan sebanyak (43.8 %) data yang merupakan *Onserviceconnected.*

1. *Onserviceconnected vs class*

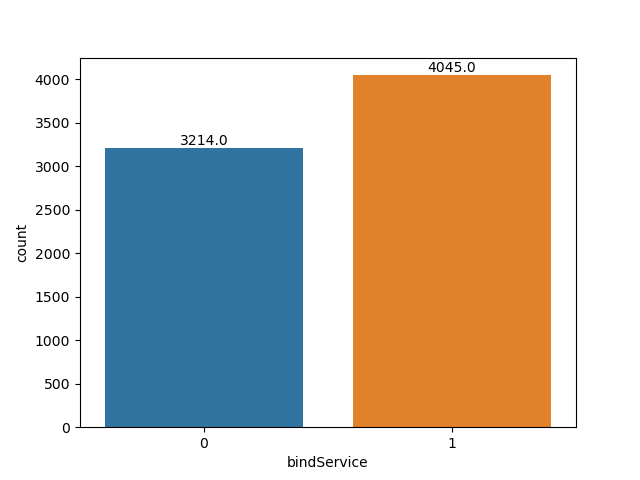
Hasil dari visualisasi dari *Onserviceconnected* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 8 *Onserviceconnected vs Class*

1. Sebanyak 4081 data yang merupakan *onServiceConnected*, 212 data termasuk kedalam kategori *malware*, dan 3869 data yang tidak termasuk kategori *malware*.
2. Sebanyak 3178 data bukan merupakan *onServiceConnected*, 1507 data termasuk kedalam kategori *malware*, dan 1671 data yang tidak termasuk kategori *malware*.
3. *Bindservice*
4. Distribusi data *Bindservice*

Hasil dari distribusi data *Bindservice* diperoleh nilai sebagai berikut.

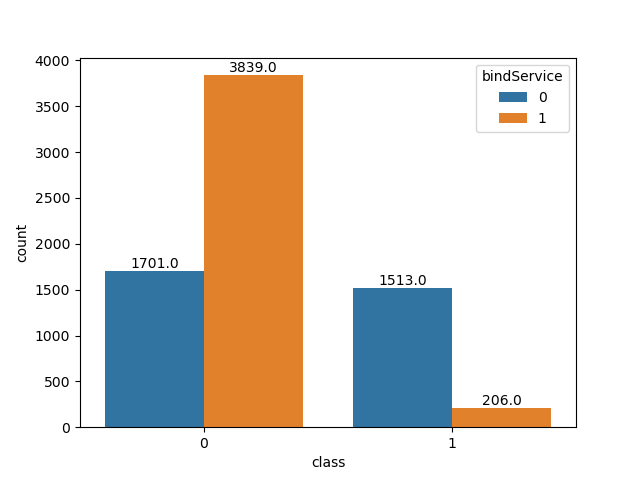


Gambar 4. 9 *Bindservice*

Dari total 7259 data *Bindservice* terdapat sebanyak 3214 data yang bukan merupakan *Bindservice* dan sebanyak 4045 data yang merupakan *Bindservice.*

1. *Bindservice vs class*

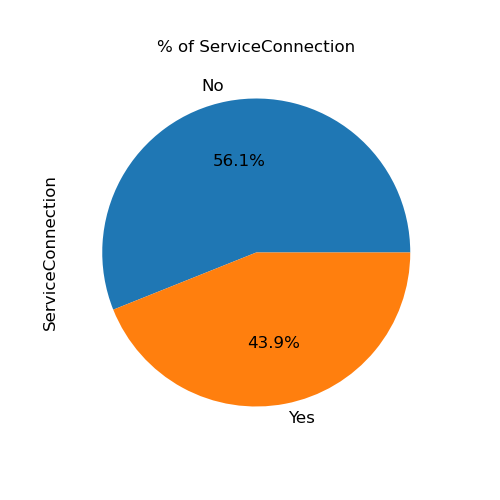
Hasil dari visualisasi dari *Bindservice* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 10 *Bindservice vs Class*

1. Sebanyak 4045 data yang merupakan *Bindservice*, 206 data termasuk kedalam kategori *malware*, dan 3839 data yang tidak termasuk kategori *malware*.
2. Sebanyak 3214 data bukan merupakan *Bindservice*, 1513 data termasuk kedalam kategori *malware*, dan 1701 data yang tidak termasuk kategori *malware*.
3. *Serviceconnection*
4. Distribusi data *Serviceconnection*

Hasil dari distribusi data *Bindservice* diperoleh nilai sebagai berikut.

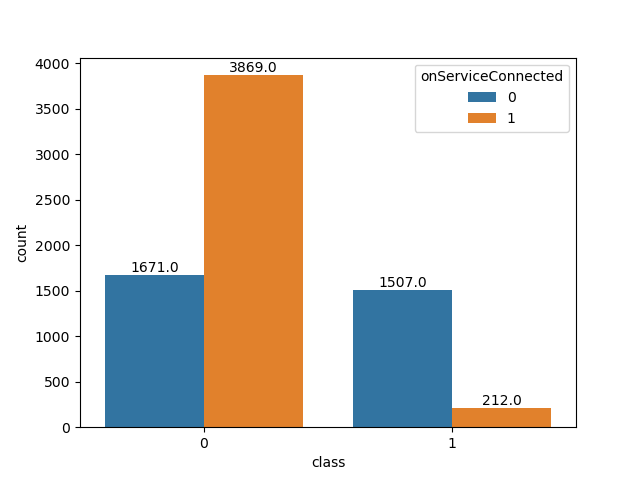
**

Gambar 4. 11 *Serviceconnection*

Dari total 7259 data *Serviceconnection* terdapat sebanyak (56.1%) data yang bukan merupakan *Serviceconnection* dan sebanyak (43.9%) data yang merupakan *Serviceconnection.*

1. *Serviceconnection vs class*

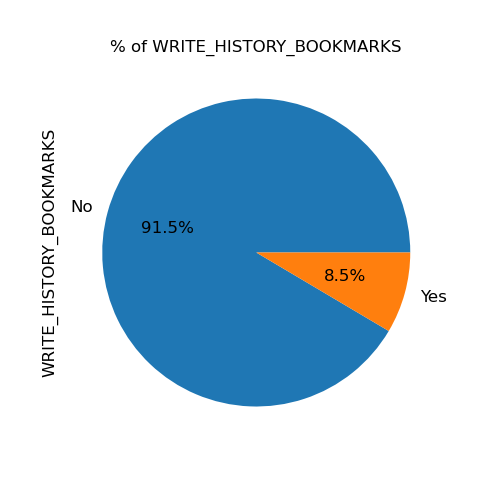
Hasil dari visualisasi dari *Serviceconnection* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 12 *Serviceconnection vs class*

1. Sebanyak 4081 data yang merupakan *Serviceconnection*, 212 data termasuk kedalam kategori *malware*, dan 3869 data yang tidak termasuk kategori *malware*.
2. Sebanyak 3178 data bukan merupakan *Serviceconnection*, 1507 data termasuk kedalam kategori *malware*, dan 16711 data yang tidak termasuk kategori *malware.*
3. *Write\_History\_Bookmarks*
4. Distribusi data *Write\_History\_Bookmarks*

Hasil dari distribusi data *Write\_History\_Bookmarks* diperoleh nilai sebagai berikut.

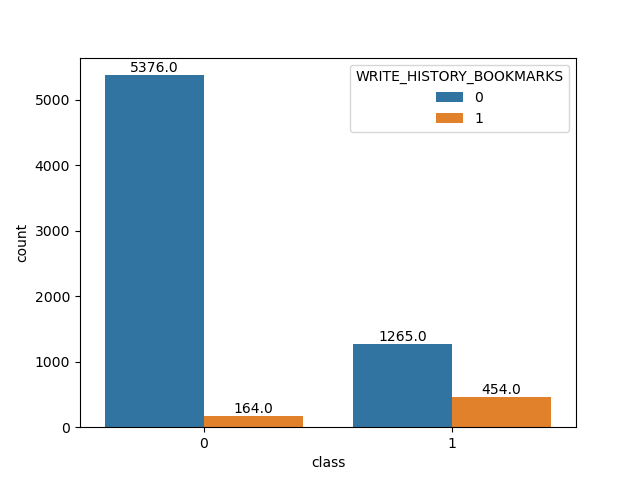
**

Gambar 4. 13 *Write\_History\_Bookmark*

Dari total 7259 data *Write\_History\_Bookmarks* terdapat sebanyak (91.5%) data yang bukan merupakan *Write\_History\_Bookmarks* dan sebanyak (8.5%) data yang merupakan *Write\_History\_Bookmarks.*

1. *Write\_History\_Bookmarks vs class*

Hasil dari visualisasi dari *Write\_History\_Bookmarks* v*s Class* dengan total 7259 data sebagai berikut :



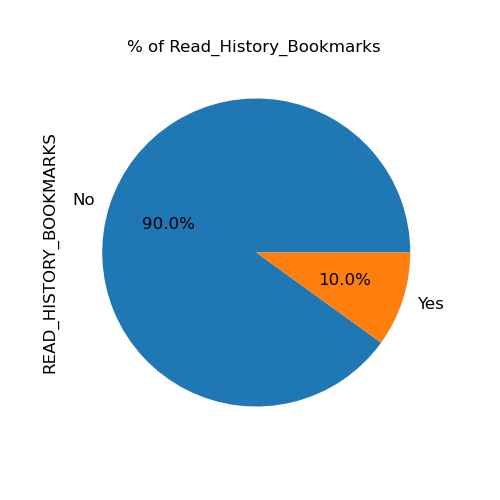
Gambar 4. 14 *Write\_History\_Bookmarks vs Class*

1. Sebanyak 618 data yang merupakan Write\_History\_Bookmarks, 454 data termasuk kedalam kategori *malware*, dan 164 data yang tidak termasuk kategori *malware*.
2. Sebanyak 6641 data bukan merupakan *Write\_History\_Bookmarks*, 1265 data termasuk kedalam kategori *malware*, dan 5376 data yang tidak termasuk kategori *malware*.

­­

1. *Read\_History\_Bookmarks*
2. Distribusi data *Read\_History\_Bookmarks*

Hasil dari distribusi data *Read\_History\_Bookmarks* diperoleh nilai sebagai berikut.

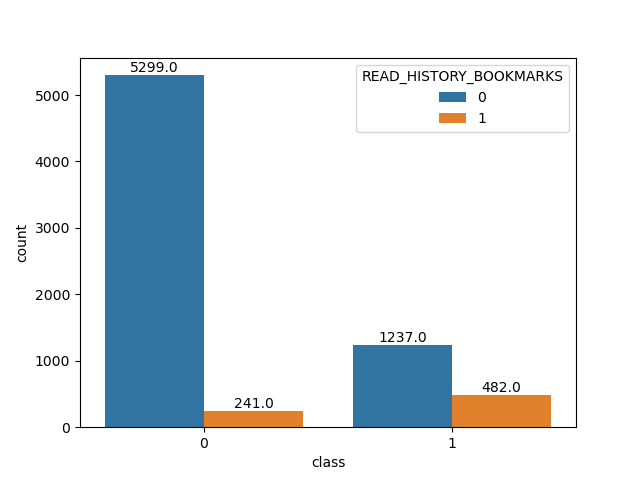
**

Gambar 4. 15 *Read\_History\_Bookmarks*

Dari total 7259 data *Read\_History\_Bookmarks* terdapat sebanyak (90.0%) data yang bukan merupakan *Read\_History\_Bookmarks* dan sebanyak (10%) data yang merupakan *Read\_History\_Bookmarks*

1. *Read\_History\_Bookmarks vs class*

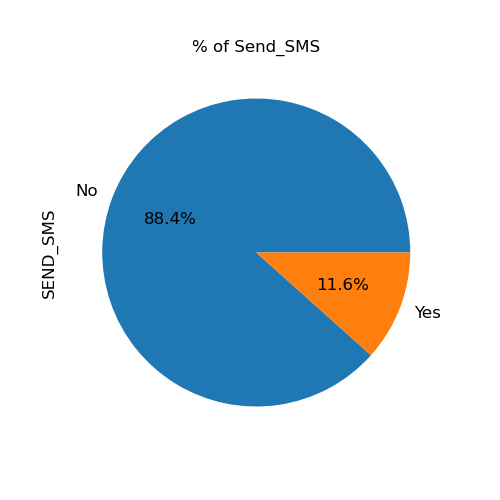
Hasil dari visualisasi dari *Read\_History\_Bookmarks* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 16 *Read\_History\_Bookmarks vs Class*

1. Sebanyak 723 data yang merupakan *Read\_History\_Bookmarks*, 482 data termasuk kedalam kategori *malware*, dan 241 data yang tidak termasuk kategori *malware*.
2. Sebanyak 6536 data bukan merupakan *Read\_History\_Bookmarks,* 1237 data termasuk kedalam kategori *malware*, dan 5299 data yang tidak termasuk kategori *malware*.
3. *Send\_sms*
4. Distribusi data *Send\_sms*

Hasil dari distribusi data *Send\_sms* diperoleh nilai sebagai berikut.

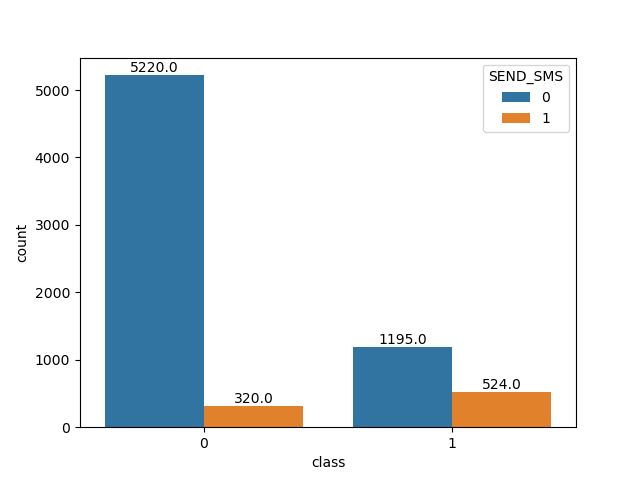
**

Gambar 4. 17 Send\_sms

Dari total 7259 data *Send\_sms* terdapat sebanyak (88.4%) data yang bukan merupakan *Send\_sms* dan sebanyak (11.6%) data yang merupakan *Send\_sms.*

1. *Send\_sms vs class*

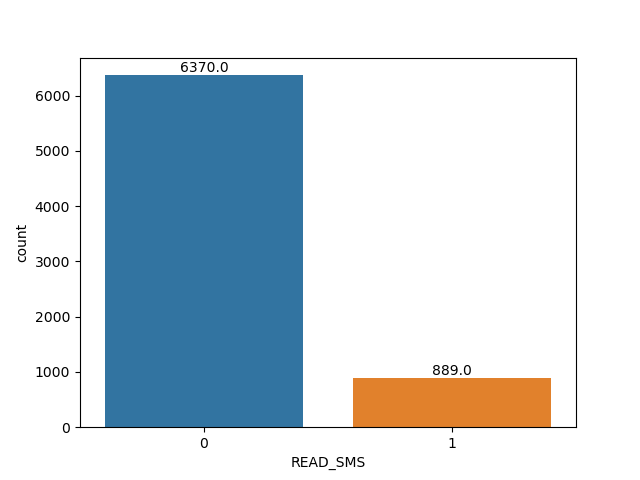
Hasil dari visualisasi dari *Send\_sms*  v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 18 *Send\_sms vs Class*

1. Sebanyak 844 data yang merupakan *Send\_sms*, 524 data termasuk kedalam kategori *malware*, dan 320 data yang tidak termasuk kategori *malware*.
2. Sebanyak 6415 data bukan merupakan *Send\_sms*, 1195 data termasuk kedalam kategori *malware*, dan 5220 data yang tidak termasuk kategori *malware*.
3. *Read\_sms*
4. Distribusi data *Read\_sms*

Hasil dari distribusi data *Read\_sms* diperoleh nilai sebagai berikut.

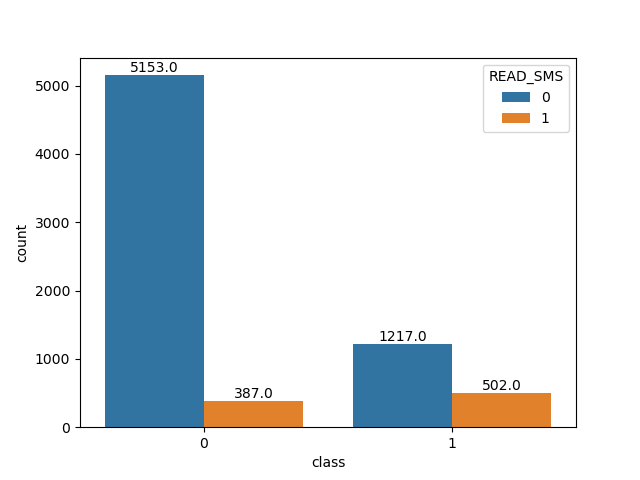


Gambar 4. 19 *Read\_sms*

Dari total 7259 data *Read\_sms* terdapat sebanyak 6370 data yang bukan merupakan *Read\_sms* dan sebanyak 889 data yang merupakan *Read\_sms.*

1. *Read\_sms vs class*

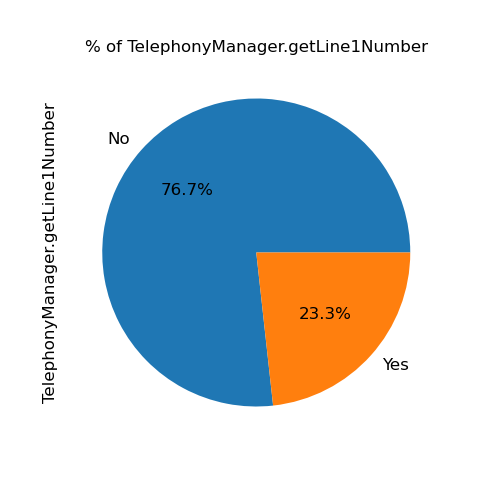
Hasil dari visualisasi dari *Read\_sms*  v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 20 *Read\_sms vs Class*

1. Sebanyak 889 data yang merupakan *Read\_sms*, 502 data termasuk kedalam kategori *malware*, dan 387 data yang tidak termasuk kategori *malware*.
2. Sebanyak 6370 data bukan merupakan *Read\_sms*, 1217 data termasuk kedalam kategori *malware*, dan 5153 data yang tidak termasuk kategori *malware*.
3. Telephonymanager.getline1number
4. Distribusi data *Telephonymanager.getline1number*

Hasil dari distribusi data *Telephonymanager.getline1number* diperoleh nilai sebagai berikut.

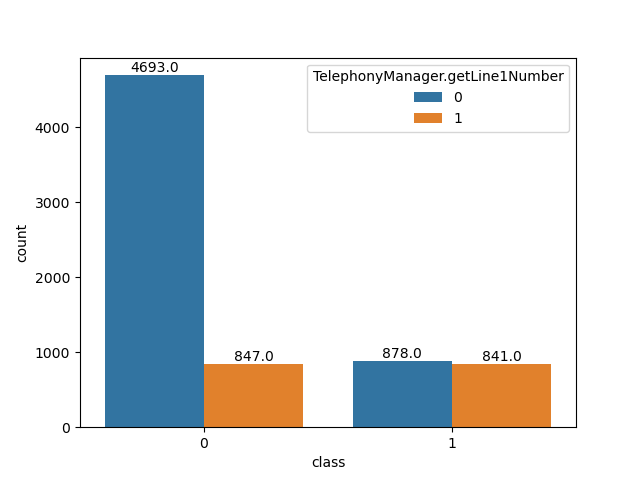


Gambar 4. 21 *Telephonymanager.getline1number*

Dari total 7259 data *Telephonymanager.getline1number* terdapat sebanyak data yang bukan merupakan *Telephonymanager.getline1number* dan sebanyak data yang merupakan *Telephonymanager.getline1number.*

1. *Telephonymanager.getline1number vs class*

Hasil dari visualisasi dari *Telephonymanager.getline1number* v*s Class* dengan total 7259 data sebagai berikut :



Gambar 4. 22 Telephonymanager.getline1number vs Class

1. Sebanyak 1688 data yang merupakan Telephonymanager.getline1number, 841 data termasuk kedalam kategori malware, dan 847 data yang tidak termasuk kategori malware.
2. 2Sebanyak 5571 data bukan merupakan Telephonymanager.getline1number, 878 data termasuk kedalam kategori malware, dan 4693 data yang tidak termasuk kategori malware.

### **Pembersihan data**

Pada tahap ini, dilakukan pembersihan data (data *cleaning*) pada *dataset* Drebit dengan mengidentifikasi, mengevaluasi, dan memperbaiki atau menghapus kesalahan, inkonsistensi, dan anomali dalam *dataset*.

1. *Missing Value*

Setelah dilakukan pengecekan terdapat adanya data *Missing Value* dalam *dataset* ini pada atribut *TelephonyManager.getSimCountryIso* seperti pada tabel berikut:

Tabel 4. 16 *Missing Value*

|  |  |
| --- | --- |
| Keterangan | Jumlah |
| *Missing Value* | 5 |

1. Data Duplikat

Pada tahap pengecekan ini terdapat adanya data yang duplikat atau diulang, seperti pada tabel berikut

Tabel 4. 17 Data Duplikat

|  |  |
| --- | --- |
| Keterangan | Jumlah |
| Data Duplikat | 6865 |

1. *Outlier* Data

Pada tahap ini dilakukan pengecekan nilai yang diluar dari ketentuan atribut atau disebut *outlier.*

Tabel 4. 18 Data *Outlier*

|  |  |
| --- | --- |
| Keterangan | Jumlah |
| Data *Outlier* | 0 |

### ***Encoding* (*Feature* *Engineering*)**

*Encoding* adalah proses konversi data mentah ke dalam bentuk yang dapat diproses oleh mesin, sehingga memungkinkan analisis lebih lanjut dan pengambilan keputusan yang akurat, dalam tahap ini ditemukan dua atribut bertipe *object* yaitu *Telephonymanager.getsimcountryiso* dan *class,* sehingga perlu di konversi menjadi tipe data *int,* seperti table dibawah ini.

Tabel 4. 19 Rekayasa Fitur

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Atribut** | **Tipe Data Sebelum Konversi** | **Tipe Data Sesudah Konversi** |
| 1 | *Telephonymanager.getsimcountryiso* | *Object* | *Int* |
| 2 | *Class* | *Object* | *Int* |

### **Membangun Model**

Membangun model merupakan langkah krusial dalam analisis data, di mana kita menciptakan suatu representasi matematis yang dapat menggambarkan hubungan antara variabel dalam *Dataset*. Pada penelitian ini saya menggunakan metode Neural Network untuk membangun model *machine learning* ini.

Tabel 4. 20 Dataset yang digunakan

|  |  |
| --- | --- |
| Dataset yang Digunakan | Dimensi |
| *Drebin* | 7259 baris data dan 216 kolom |

* 1. Split data

Pada tahap ini, data akan dibagi menjadi data latih dan data uji menggunakan *train\_test\_split*. Adapun besaran dari data latih yaitu 80% dan data uji yaitu 20%. Seperti pada tabel berikut:

Tabel 4. 21 Split data *training*

|  |  |
| --- | --- |
| *Dataset* | Jumlah |
| Data Latih | 5807 baris dan 216 kolom |
| Data Tes | 1452 baris dan 216 kolom |

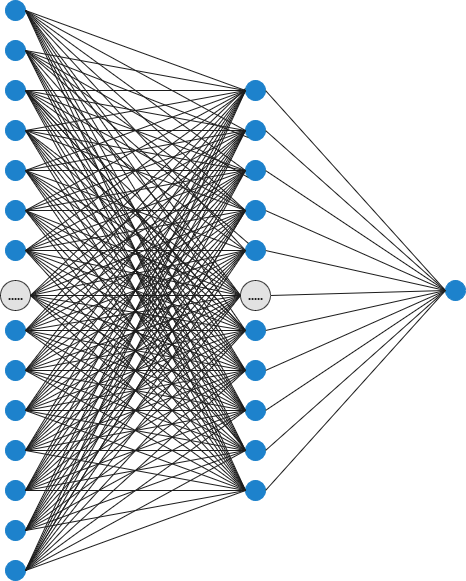
* + 1. *Training* Model

Model yang digunakan yaitu klasifikasi, yang bertujuan untuk mendektesi malware berdasarkan berbagai fitur atau atribut yang ada dalam *dataset*.

Tabel 4. 22 *Training* Model

|  |  |
| --- | --- |
| Keterangan | Nilai |
| *Epoch* | 23 |
| *Learning Rate* | 0.001 |
| *Layer* | 1 *layer* input dengan 215 unit neuron  1 *hidden* *layer* dengan 100 unit neuron  1 *layer* *output* dengan 1 unit neuron |

Berdasarkan gambar dibawah ini, terdapat 215 *neuron* pada *layer* input, 100 *neuron* pada *hidden* *layer*, dan 1 *neuron* pada *layer* *output*. Cara kerja ketiga layer tersebut adalah input layer menerima masukan nilai awal dari data yang akan diproses, nantinya dilewatkan ke hidden layer lalu akan keluar melalui output layer dalam bentuk bobot pula.



Gambar 4. 23 Model *Neural Network*

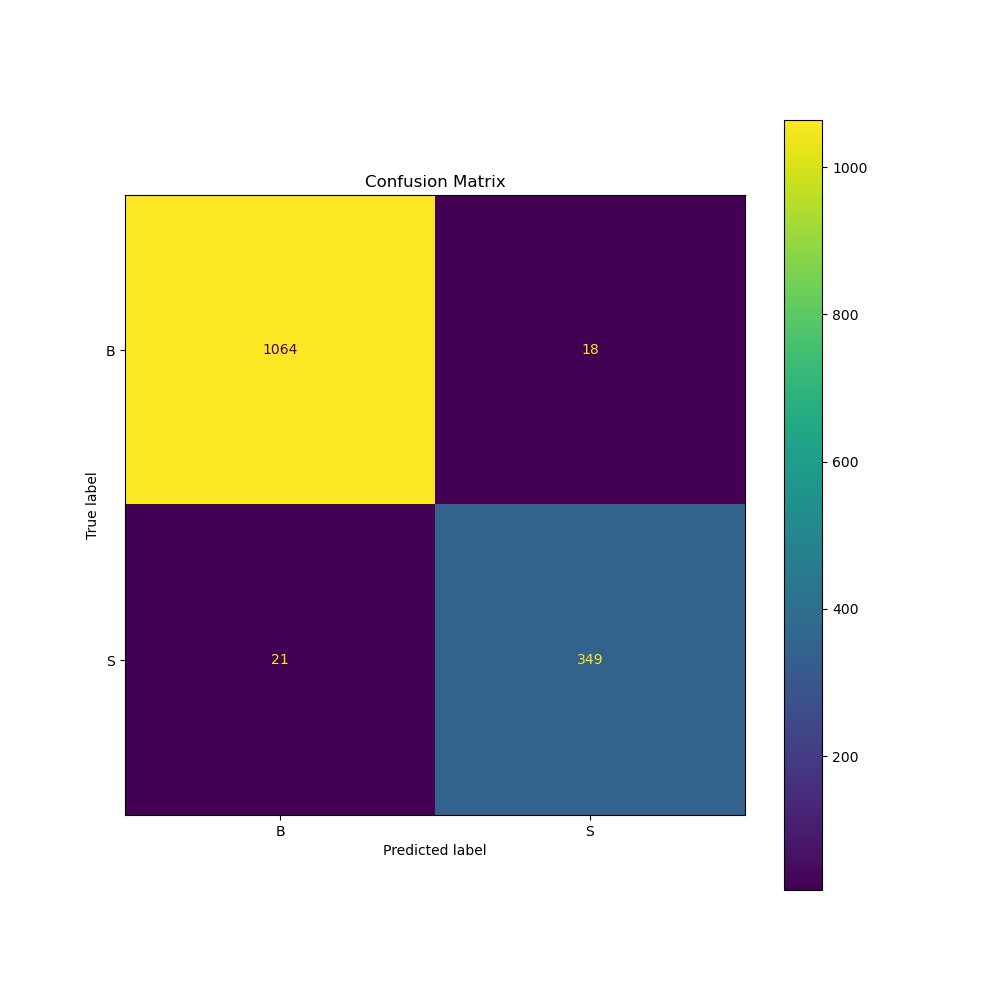
### **Pengujian Model**

Pada tahap ini dilakukan pengujian performa model dengan menggunakan metrik evaluasi yang digunakan dalam metode *Neural Network* (NN). Metrik evaluasi ini akan memberikan pemahaman tentang sejauh mana model dapat mendeteksi *malware* pada sistem andorid dengan akurat. Beberapa metrik evaluasi yang digunakan dalam pengujian model ini meliputi akurasi, *recall*, *F1-Score*, dan *Confusion matrix*.

Tabel 4. 23 Pengujian Model Data *Training*

|  |  |
| --- | --- |
| Model | Hasil |
| Akurasi | 97.3 % |
| *Recall* | 95.1 |
| *F1-Score* | 94.7 |

Pada gambar matrix confusion di bawah ini, didapatkan hasil:



Gambar 4. 24 *Confusion* *Matrix*

True Positive (TP) yaitu kelas yang diprediksi positif dan pada kenyataannya positif, sebanyak 349 data.

True Negative (TN) yaitu kelas yang diprediksi negatif dan pada kenyataannya negatif., sebanyak 1064 data.

False Positive (FP) yaitu kelas yang diprediksi positif tetapi pada kenyataannya negatif, sebanyak 21 data.

False Negative (FN) yaitu kelas yang diprediksi negatif tetapi pada kenyataannya positif, sebanyak 18 data

## Evaluasi dan Validasi hasil

Pada tahap ini dilakukan proses evaluasi model mesin *learning* yang telah dibuat dengan menggunakan pola data atau *dataset* baru yang telah disiapkan sebelumnya, adapun *dataset* yang digunakan dapat dilihat pada tabel dibawah ini:

Tabel 4. 24 Dimensi Data Latih dan Data Uji

|  |  |
| --- | --- |
| *Dataset* | Jumlah |
| Data Uji | 1089 baris dan 215 kolom |

Dari tabel diatas, dapat disimpulkan bahwa dataset yang akan digunakan untuk pengujian evaluasi dan validasi hasil merupakan hasil split data dari dataset sebelumnya, dengan total data sebanyak 1089 baris data dengan 215 kolom. Dari dataset tersebut mendapatkan hasil nilai parameter sebagai berikut:

Tabel 4. 25 Pengujian Model Data *Testing*

|  |  |
| --- | --- |
| Model | Hasil |
| Akurasi | 99.8 % |
| *Recall* | 99.2 |
| *F1-Score* | 99.6 |

# KESIMPULAN DAN SARAN

## Kesimpulan

Berdasarkan implementasi dan pengujian yang dilakukan sebelumnya didapatkan hasil akurasi f1score, recall , dengan hasil yang telah disebutkan terbukti bahwa

1. Menerapkan algoritma neural network pada sistem deteksi malware android sehingga dapat meminimalisir ancaman serangan malware
2. Menerapkan algoritma neural network pada sistem deteksi malware android sehingga mempermudah mengidentifikasi jenis serangan malware

## Saran

Adapun saran untuk peneltian berikutnya antara lain

1. Penelitian ini dapat dikembangkan lagi yaitu dengan melakukan penelitain dengan menggunakan data rel atau yang sesungguhnya.
2. Penelitian ini dapat dikembangkan lagi yaitu dengan mengiimplementasikan model ke dalam sebuah aplikasi yang berbasis android.

# DAFTAR PUSTAKA

Abdurrohman, H., Dini, R., & Muharram, A. P. (2018). Evaluasi Performa metode Deep Learning untuk Klasifikasi Citra Lesi Kulit The HAM10000. *Seminar Nasional Instrumentasi, Kontrol dan Otomasi (SNIKO)*.

Admojo, F. T., & Ahsanawati. (2020). Klasifikasi Aroma Alkohol Menggunakan Metode KNN. *Indonesian Journal of Data and Science*, 34-38.

Alfa Syahputra, M., & Eva Maulina, S. (2016). *Let's Build Your Andorid Apps With Android Studio.* Jakarta: PT Elex Media Komputindo.

Asry, D. W., Siswanto, E., & Huda, D. K. (2023). Deteksi Malware Statis Menggunakan Deep Neural Networks Pada Portable Executable. *Jurnal Ilmu Teknik dan Informatika (TEKNIK)*, 19-34.

Aulia, A., Aprianti, B., Supriyanto, Y., & Rozikin, C. (2022). prediksi harga emas dengan menggunakan algoritma support vector regression (Svr) dan linear regression (lr). *jurnal ilmiah wahana pendidikan*, 84-88.

Awad, M., & Khanna, R. (2015). *Efficient Learning Machines.* New York: Apress Media.

Bhakti, H. D. (2019). Aplikasi Artificial Neural Network (ANN) untuk Memprediksi Masa Studi Mahasiswa Program Studi Teknik Informatika Universitas Muhammadiyah Gresik. *Jurnal Eksplora Informatika*, 88-95.

Bustami. (2014). Penerapan Algoritma Naive Bayes Untuk Mengklasifikasi Data Nasabah Asuransi. *Jurnal Informatika* , 884-898.

Cholissodin, I., Sutrisno, Soebroto, A. A., Hasanah, U., & Febiola, Y. I. (2020). AI, MACHINE LEARNING & DEEP LEARNING (Teori & Implementasi). *FILKOM: Fakultas Ilmu Komputer*, 465.

Collonoval, F., Dafna, I., Ivanov, P., & Eric, C. (2023, July 4). *Project Jupyter’s origins and governance*. Retrieved from Jupyter: https://jupyter.org/

Efanntyo, & Mitra, A. R. (2021, November). Perancangan Aplikasi Sistem Pengenalan Wajah Dengan Metode Convolutional Neural Network(CNN) Untuk Pencatatan Kehadiran Karyawan. *Jurnal Instrumentasi dan Teknologi Informatika (JITI), vol.3*, 1-11.

Eska, J. (2016). Penerapan Data Mining Untuk Prediksi Penjualan Wallpaper Menggunakan Algoritma C4.5. *JURTEKSI (Jurnal Teknologi dan Sistem Informasi)*, 9-13.

Gulli, A., & Pal, S. (2017). *Deep Learning With Keras.* Brimingham: Packt Publishing.

Hadianto, N., Novitasari, H. B., & Rahmawati, A. (2019). Klasifikasi Peminjaman Nasabah Bank Menggunakan Metode Neural Network. *PILAR Nusa Mandiri*, 163-170.

Hadiprakoso, R. B., Aditya, W. R., & Pramitha, F. N. (2022). Analisis Statis Deteksi Malware Android Menggunakan Algoritma Supervised Machine Learning. *CyberSecurity dan Forensik Digital*, 1-5.

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining Concepts and Techniques* (3rd ed.).

Handayani, M., Riandini, M., & Situmorang, Z. (2022). Perbandingan Fungsi Optimasi Neural Network Dalam Klasifikasi Kelayakan Calon Suami. *Jurnal Informatika*, 78-84.

Hilpisch, Y. (2015). *Python for Finance.*

huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: theory and. *Neurocomputing*, 489–501.

Hunter, J. D. (2023, Juli). *Matplotlib*. Retrieved from pypi.org: https://pypi.org/project/matplotlib/

Husain, A. (2018). Analisis Data Lifting Migas Menggunakan Metode C4.5 Pada Asosiasi Daerah Penghasil Migas. *Jurnal String*, 325-335.

Istighfarizky, F., ER, N. A., Widiartha, I. M., Astutia, L. G., Putra, I. G., & Suhartana, I. K. (2022). Klasifikasi Jurnal menggunakan Metode KNN dengan Mengimplementasikan Perbandingan Seleksi Fitur. *Jurnal Elektronik Ilmu Komputer Udayana*, 167-176.

Mashabi, S. (2021, 09 14). *BSSN: Hingga Agustus 2021 tercatat 888 Juta Serangan Siber*. Retrieved from Kompas.com.

Munawarah, R., Soesanto, O., & Faisal, M. R. (2016). Penerapan Metode Support Vector Machine Pada Diagnosa Hepatitis. *Kumpulan Jurnal Ilmu Komputer (KLIK)*, 103-113.

Muslim, M. A., Prasetiyo, B., Mawarni, E. L., Herowati, A. J., Misqotussa'adah, Rukmana, S. H., & Nurzahputra, A. (2019). *Data Mining Algoritma C4.5 disertai cntoh kasus dan penerapannya.* Semarang: unnes.

Nasution, M. R., & Hayaty, M. (2019). Perbandingan Akurasi dan Waktu Proses Algoritma K-NN dan SVM dalam Analisis Sentimen Twitter. *JURNAL INFORMATIKA, 6*(2), 212-218.

Nelli, F. (2015). *Python Data Analytics.* New York: Apress Media.

Noviyanto. (2020). Penerapan Data Mining dalam Mengelompokkan Jumlah Kematian Penderita COVID-19 Berdasarkan Negara di Benua Asia. *Paradigma- Journal Informatika dan Komputer*, 183-188.

Novrianda, R., Kunang, Y. N., & Shaksono, P. (2014). Analisis Forensik Malware Pada Android . *Konferensi Nasional Ilmu Komputer (KONIK)*, 377-385.

Prasetyo, B., Suryani, V., & Anbiya, D. R. (2021). Analisis Deteksi Malware pada Aplikasi Android Fintech berdasarkan Permissions dengan menggunakan Naive Bayes dan Random Forest. *e-Proceeding of Engineering*, 9885.

Putra, I. T., Kartini, K. S., Suyitno, Y. K., Sugiarta, I., & Puspita, N. E. (2023). Penerapan Library Tensorflow, Cvzone, dan Numpypada Sistem Deteksi Bahasa Isyarat Secara Real Time. *Jurnal Krisnadana*, 412-423.

Ramdhan, N. A. (2019). Penerapan Metode Neural Network Untuk Prediksi Nilai Ujian Nasional (Study Kasus Di Smk Muhammadiyah Slawi). *Jurnal Ilmiah Indonesia*, 118-130.

Rawat, N., Amrita, & Singh, A. (2023). Permission-Based Malware Detection in Android Using. *YMER*, 1505-1517.

Refhaldo, M., Budiarto, E., Sari, P. A., & Monica, S. (2022). Klasifikasi Aplikasi Malware Android Menggunakan Algoritma C5.0. *Prosiding SAINTEK*, 854-859.

Rohman, y. A. (2019, 12 8). *Pengenalan NumPy, Pandas, Matplotlib*. Retrieved from medium.com: https://medium.com/@yasirabd/pengenalan-numpy-pandas-matplotlib-b90bafd36c0

Romadhon, M. S., & Kodar, A. (2020). IMPLEMENTASI METODE MARKET BASKET ANALYSIS (MBA) MENGGUNAKAN ALGORITMA APRIORI DALAM TRANSAKSI PENJUALAN (STUDI KASUS: KAFE RUANG TEMU). *Jurnal Saintekom, vol.2*, 138-147.

Russell, R. (2018). *Step-by-Step Guide To Implement Machine Learning Algorithms with Python.*

Sanjaya, J., Renata, V., Budiman, V. E., Anderson, F., & Ayub, M. (2020). Prediksi Kelalaian Pinjamanan Bank Menggunakan Random Forest And Adaptive Boostong. *J. Tek Inform*, 50-60.

Sitorus, Y. W., Sukarno, P., & Mandala, S. (2021). Analisis Deteksi Malware Android menggunakan metode Support Vector Machine & Random Forest. *e-Proceeding of Engineering*, 12500.

Syarovy, M., & Sutiarso, A. P. (2023). Pemanfaatan Model Neural Network Dalam Generasi Baru Pertanian Presisi di Perkebunan Kelapa Sawit. *Warta PPKS*, 39-54.

University, B. (2023, 7 2). *Confusion Matrix*. Retrieved from Retrieved from Binus University: https://socs.binus.ac.id/2020/11/01/confusion-matrix/

VanderPlas, J. (2017). *Python Data Science Handbook.* United States of America: O’Reilly Media, Inc.,.

Wadi, H. (2015). *Pemrograman Python : untuk pelajar dan mahasiswa.* Bandung: TR Publisher.

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software, 6*, 3021. Retrieved from https://doi.org/10.21105/joss.03021

# DAFTAR RIWAYAT HIDUP