Komparasi Performansi Algoritma Naive Bayes dan Logistic Regression pada Malware Android

Andreas Putra Wijaya[1*], Handri Santoso[2]
[1*,2] Magister Information of Technology; Pradita University; Scientia Business Park, Jl. Gading Serpong Boulevard No.1, Curug Sangereng, Kelapa Dua, Tangerang, Banten; Indonesia
andreas.putra@student.pradita.ac.id*, handri.santoso@pradita.ac.id

Abstrak

Saat ini masyarat Indonesia sudah memanfaatkan teknologi *Internet*, untuk berbagai kebutuhan. Mulai dari transportasi, berbelanja sampai dunia pendidikan memanfaatkan *Internet*. Peralatan dalam mengakses *Internet* beragam, mulai dari komputer, laptop sampai perangkat komunikasi seperti perangkat seluler. Perangkat seluler saat ini yang cukup banyak digunakan masyarakat adalah perangkat seluler berbasis system operasi Android. Dalam situasi ini mendorong pihak-pihak tertentu memanfaatkan celah untuk mencari keuntungan, salah satunya pembuatan *Malware*. Selain itu perkembangan dibidang kecerdasan buatan saat ini sangat maju dan mendorong banyak penelitian berbagai bidang memanfaatkannya. Keadaan inilah yang menjadikan peneliti berfokus pada analisa *Malware* dengan memanfaatkan teknologi kecerdasan buatan. Tujuan dari penelitian ini adalah menganalisa file-file APK Android dengan melakukan mengklasifikasi keluarga *Malware*. Pengukuran performansi dan akurasi juga akan disajikan perbandingan antara algoritma *Naïve bayes* dengan algoritma *Logistic Regression*. Metode yang digunakan adalah klasifikasi *Supervised Learning*, menggunakan algoritma *Naïve Bayes* dan *Logistic Regression*. Dimana kedua metode tersebut merupakan algoritma *Machine Learning dan* bagian dari kecerdasan buatan.
Kata kunci: Malware, Android, Supervised Learning, Naïve Bayes, Logistic Regression.

Abstract

Currently, Indonesian people have used Internet technology for various needs. Starting from transportation, shopping to the world of education using the Internet. Equipment in accessing the Internet varies, ranging from computers, laptops to communication devices such as mobile devices. Currently, mobile devices that are quite widely used by the public are mobile devices based on the Android operating system. In this situation it encourages certain parties to take advantage of loopholes to seek profit, one of which is the creation of Malware. In addition, developments in the field of artificial intelligence are currently very advanced and encourage many researches in various fields to use it. This situation makes researchers focus on malware analysis by utilizing artificial intelligence technology. The purpose of this study is to analyze Android APK files by classifying the Malware family. Performance and accuracy measurements will also be presented in a comparison between the Naïve Bayes algorithm and the Logistic Regression algorithm. The method used is Supervised Learning classification, using Naïve Bayes algorithm and Logistic Regression. Everywhere both methods are Machine Learning algorithms and part of artificial intelligence.
keywords: Malware, Android, Supervised Learning, Naïve Bayes, Logistic Regression.

1. PENDAHULUAN

Malware merupakan kepanjangan dari *Malicious Software*, yang berisikan script atau program jahat dengan maksud dan tujuan merugikan pihak lain. Saat ini perkembangan sangat cepat dan sangat merugikan bagi pengguna yang terinfeksi. Contoh *Malware* seperti *Virus, Trojan, Ransomware* dan lain-lain. Kerugian akibat adanya Malware yang menyerangkan sistem operasi Android sangat besar. Banyak dari pengguna perangkat Android tidak menyadari bahwa perangkatnya sudah terinfeksi oleh *Malware*.

Salah satu penyebabnya adalah mengunduh file Application Package Kit (APK), tanpa melakukan proses pengecekan atau scanning dan melakukan instalasi file APK ke dalam sistem Android. Ada beberapa situs pengecekan file APK seperti virustotal.com. Situs tersebut mampu melakukan scanning file APK, apakah

file tersebut *Malware* atau bukan *Malware.* Kelompok *Malware* juga banyak sekali seperti kelompok *Malware* yang menyerang industry *Banking*, kelompok *Malware* yang menyerang dengan menggunakan media *SMS*, kelompok *Malware* yang meminta tebusan yang dikenal dengan *Ransomware* dan masih banyak kelompok atau keluarga *Malware* yang lainnya.

Beberapa penelitian dibidang *Malware* sangat banyak sekali, seperti menganalisa *Malware*. Pada dasarnya analisa *Malware* terbagi menjadi analisa statis, analisa dinamis dan analisa gabungan atau *Hybrid.* Analisa statis *Malware* adalah analisa tanpa melakukan instalasi *Malware* ke sistem operasi, tetapi melakukan *Reverse Engineering* terhadap *file-file APK Malware*. Hasil *Reverse Engineering* berupa *file Source Code* dan siap untuk dilakukan analisa. Sedangkan analisa dinamis tanpa melakukan *Reverse Engineering* dan menjalankan *file Malware*, kemudian menganalisa *Behavior Malware* menggunakan *log* dari sistem Android. Analisa *Malware* menggunakan metode gabungan atau *Hybrid* adalah dengan melakukan analisa *Source Code* dan menganalisa log dari system Android.

Dari pengumpulan data dari berbagai jurnal dengan topik Android *Malware* kebanyakan berfokus pada penelitian *Machine Learning, Feature Selection, Mobile Malware Detection* (Su et al. 2020) (Garg and Baliyan 2019) (Abawajy, Darem, and Alhashmi 2021). Berikut topik penelitian Android Malware yang banyak dibahas:

1. File AndroidManifest diekstraksi, Algoritma TF-IDF digunakan untuk menghitung nilai permission dari setiap permission dan nilai sensitivitas apk (SVOA) dari setiap aplikasi. SVOA dan jumlah permission yang digunakan dan diuji dengan machin learning. Hasil eksperimen menunjukkan bahwa hanya menggunakan fitur permission Malware untuk membedakan malware dan bukan malware. Untuk deteksi malware, pendekatan yang diusulkan mencapai akurasi hingga 99,5% (Yuan et al. 2020).
2. Deteksi aplikasi yang terinfeksi malware dan kerentanan terindikasi pada fitur permission file APK Android. Melakukan percobaan menggunakan fitur permission pada aplikasi Android. Melakukan fitur seleksi yang terbaik dari Dataset, mengembangkan model deteksi malware dengan memanfaatkan pendekatan pembelajaran LSSVM (Least Square Support Vector Machine) yang terhubung melalui tiga fungsi kernel yang berbeda yaitu, linear, radial basis dan polinomial. Dengan metode tersebut mencapai akurasi 98,8% (Mahindru and Sangal 2021).
3. Metode analisis hibrid untuk mendeteksi malware Android dan mengklasifikasikan kelompok malware disajikan dalam makalah ini, dan sebagian dioptimalkan untuk data multi-fitur. Untuk analisis statis, penggunaan fitur *Permission* dan *Intent* sebagai fitur statis dan menggunakan tiga metode pemilihan fitur untuk membentuk subset dari tiga kandidat fitur. Dibandingkan dengan berbagai model, *termasuk k-Nearest Neighbor* dan *Random Forest*, hasil dengan algoritma *Random Forest* adalah yang terbaik, dengan tingkat deteksi 95,04%, sedangkan uji chi-square adalah metode seleksi fitur terbaik. Dalam analisis dinamis berdasarkan lalu lintas jaringan, tidak seperti yang berfokus pada arus lalu lintas satu arah dan bekerja pada protokol HTTP, protokol lapisan transport, lapisan protokol. Hasil eksperimen menunjukkan terjadi akurasi dari 74% menjadi 99% (Ding et al. 2021).
4. SmartMal menyajikan kerangka kerja deteksi malware perilaku berorientasi layanan baru untuk perangkat seluler. SmartMal memperkenalkan konsep service-oriented architecture (SOA) dan behavior analisis ke dalam paradigma deteksi malware. Framework yang diusulkan bergantung pada arsitektur client-server, client terus-menerus mengekstrak berbagai fitur dan mentransfernya ke server, dan tugas utama server adalah mendeteksi anomali menggunakan algoritme deteksi. Beberapa server terdistribusi secara bersamaan menganalisis vektor fitur menggunakan berbagai detektor dan fusi informasi digunakan untuk menggabungkan hasil detektor. Pendekatan statistik berbasis siklus untuk deteksi anomali perangkat seluler, menganalisis pola penggunaan reguler pengguna. Hasil empiris menunjukkan bahwa Framework yang diusulkan dengan algoritma deteksi anomali sangat efektif dalam mendeteksi malware pada perangkat Android (Wang et al. 2014).
5. Penelitian ini memanfaatkan tingkat positif palsu yang rendah dari deteksi penyalahgunaan dan kemampuan deteksi anomali untuk mendeteksi malware zero-day. Usulan sistem deteksi hibrid berdasarkan tools

CuckooDroid, yang memungkinkan penggunaan fitur Cuckoo Sandbox untuk menganalisis malware Android melalui analisis dinamis dan statis. Sistemnya terdiri dari dua bagian: mesin pendeteksi anomali yang melakukan deteksi aplikasi abnormal melalui analisis dinamis; mesin pendeteksi Signature yang melakukan deteksi dan klasifikasi malware dengan kombinasi analisis statis dan dinamis. Eksperimen menunjukkan bahwa mesin pendeteksi anomali mampu mendeteksi malware zero-day dengan tingkat negatif palsu yang rendah (1,16 %) dan tingkat positif palsu yang dapat diterima (1,30 %). Akurasi klasifikasi sampel malware dengan tingkat positif rata-rata 98,94% (Wang, Yang, and Zeng 2015).

6. Penelitian ini mengusulkan model deteksi Malware berdasarkan fitur ketidakpastian. Algoritma logistik regresi menggambarkan hubungan input (Permission) dan output (label). Algoritma Markov Chain Monte Carlo (MCMC) untuk menyelesaikan ketidakpastian fitur. Setelah bereksperimen dengan 2037 sampel, untuk deteksi malware, mencapai akurasi 95,5%, dan False Positive Rate sebesar 1,2%. Akurasi deteksi lebih tinggi dari akurasi langsung menggunakan 24 fitur Permission. Hasil akurasi deteksi sampel Malware baru adalah 92,7%. Dibandingkan dengan pendekatan mutakhir lainnya, metode ini efektif dalam mendeteksi malware (Yuan 2020).

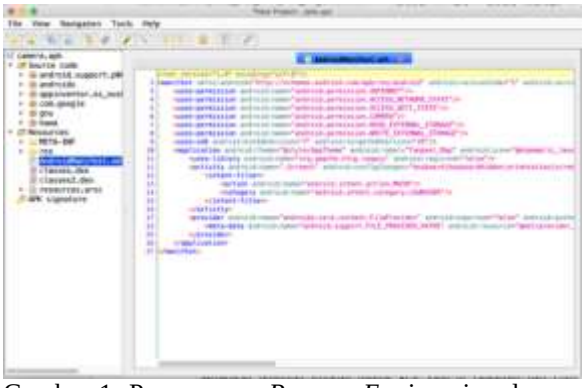
Sedangkan penelitian tentang Android *Malware Classification* (Rashed and Suarez-Tangil 2021), (Mat et al. 2021) masih tergolong sedikit, maka state-of-the-art pada penelitian ini, melakukan klasifikasi Malware dengan dengan analisa static yang sederhana dengan memperhitungan fitur Permission dan fitur Intent sebagai parameter klasifikasi dengan menggunakan algoritma *Naïve Bayes* dan *Logistic Regression.*

Tujuan dari penelitian ini adalah untuk mendapatkan model klasifikasi dari keluarga *Android Malware*, dimana nantinya dari model tersebut dapat diketahui performansi akurasinya. Pengukuran akurasi sangat penting, jika makin akurat performasi dari model yang dihasilkan maka, model tersebut dapat mendeteksi *Malware* dengan benar. Urgensi dari penelitian ini untuk mengetahui apakah file-file APK yang diunduh merupakan *Malware* atau bukan *Malware*, dimana dampaknya akan menjadi perhatian bagi para pengguna perangkat seluler yang berbasis Android untuk lebih berhati-hati dalam melakukan instalasi atau mengunduh aplikasi APK Android. Tidak melakukan instalasi ke perangkat tetapi melakukan pengecekan file APK Android yang telah diunduh.

2. METODE

2.1. REVERSE ENGINEERING

Reverse Engineering merupakan metode untuk melakukan decompiled file Execute menjadi file Source Code. Penggunaan metode ini sangat cocok jika diterapkan dalam menganalisa file yang terindikasi Malware. Salah satu tools yang digunakan dalam melakukan Android Reverse Engineering adalah Jadx. Tools tersebut mengubah file APK Android menjadi *Folder Source code*, *Folder Resources* dan *APK Signature*. Module Jadx ada dua macam, yaitu Jadx, dijalankan lewat Command Line (CLi) sedangkan Jadx-gui, berbasis Graphical User Interface (GUI).



Gambar 1. Penggunaan *Reverse Engineering* dengan Jadx-gui.

Di *Folder Resources*, terdapat file *AndroidManifest.xml*, file inilah yang berisikan informasi mengenai nama paket aplikasi, komponen aplikasi, *Permission* dari aplikasi serta fitur perangkat keras dan fitur perangkat lunak dari aplikasi. Fitur *Permission* dan fitur *Intent* adalah dua komponen yang dibutuhkan Android dalam menjalankan *file* APK Android.

Hasil dari *Reverse Engineering* yang dijadikan sebagai *Dataset* dalam proses klasifikasi adalah *file AndroidManifest.xml*. *File* tersebut berikan informasi mengenai permission, intent, dan lain-lain. Jadx merupakan module ekstraks file APK Android yang bersifat *Open-Source*.

2.2. DATASET MALWARE

Penggunaan klasifikasi menggunakan Machine Learning memerlukan Dataset untuk proses

probability menghasilkan keputusan keluarga Malware. Dataset diambil dari pembacaan isi file APK Android, yang terbaca adalah parameter atau fitur permission dan fitur intent. Setelah itu disimpan kedalam Dataset Malware.

Keluarga Malware yang dilakukan penelitian adalah: Banking APK, Ransomware APK, Riskware APK, SMS APK, Benign APK.

Tabel 1. Malware dan label (target)

No	Malware	Label
1	Adware APK	0
2	Banking APK	1
3	Ransomware APK	2
4	Riskware APK	3
5	SMS APK	4
6	Benign APK	5

Dataset di tabel 1, menggunakan fitur-fitur *Permission* yang digunakan untuk memberikan ijin dari perangkat seluler. Selain fitur Permission, fitur Intent juga dimasukan ke *Dataset Malware.* Fitur *Intent* ini yang melakukan *Action* terhadap perangkat yang sudah diberikan *Permission* melalu fitur *Permission*.

Didalam penelitian ini menggunakan Dataset 600 file APK Android. Setalah dilakukan proses ekstrak dari file tersebut didapatkan 1277 fitur. Malware Banking adalah kelompok Malware yang sering menyerang pada sektor perbankan, dimana target utama ada mengambil informasi dari nasabah bank. Ransomware adalah kelompok Malware yang bertujuan meminta tebusan dari korban atau pengguna yang terinfeksi Ransomware.

2.3. NAIVE BAYES

Naïve Bayes Classifier adalah metoda klasifikasi berdasarkan teori Bayes. Metode ini berdasarkan probabilitas yang dipresentasikan oleh ilmuwan Inggris Thomas Bayes. Peluang dari prediksi masa depan berdasarkan pengalaman di masa sebelumnya. Asumsi yang kuat (Naive) terhadap indepensi parameter merupakan ciri-ciri dari metode *Naive Bayes*.

Salah satu kelebihan dari *Naive Bayes*, tidak membutuhkan Dataset yang besar, mudah dipahami dan masih banyak kelebihan-kelebihan teori *Naive Bayes.*

Sebagai dasar teori *Naive Bayes*

P (A|B) = (P(B|A) * P(A)) / P(B) ..... (1)

Dimana,

P (A|B) = Likelihood.

P(B) = Evidence.

P(A) = Prior Probability.

P(A|B) = Probabilitas terjadinya A sebagai syarat B telah terjadi.

Dalam penggunaan teori Bayes, dapat mencari peluang terjadinya A, sebagai syarat B telah terjadi. Penggunaan nilai B sebagai bukti dan nilai A sebagai hipotesis. Dengan asumsi bahwa nilai prediktor/fiturnya independen. Dapat diartikan bahwa kehadiran satu fitur tertentu tidak mempengaruhi fitur yang lain. Maka hal ini disebut sebagai Naive. Rumus ini berubah menjadi:

P (y | X) = (P (X | y) * P(y)) / P(X) ..... (2)

Sebagai contoh, sekolah mengadakan acara kegiatan penyelenggaraan perkemahan di alam. Sebelum melakukan kegiatan tersebut sudah melihat tentang ramalan cuaca. Hasil dari informasi cuaca sebagai berikut: Hari ini hujan dan mulai berawan = 50%, hari ini meulai berawan = 40%, hari ini pada bulan september kemungkinan hujan = 10%. Bagaimana menyelesaikan masalah tersebut berdasarkan teorim Bayes? Apakah acara perkemahan sekolah jadi dilaksanakan?

Berdasarkan teori Bayes:

P (hujan|berawan) = (P(hujan) * P(berawan|hujan)) / P(berawan)

P (hujan|berawan) = (0,1 * 0,5) / 0,4

P (hujan|berawan) = 0,125 x 100%

P (hujan|berawan) = 12,5%

Dari hasil perhitungan dengan menggunakan teori Bayes, maka didapatkan 12,5% kemungkinan hujan. Maka dapat diputuskan bahwa acara perkemahan sekolah di alam, tetep dapat dilaksanakan.

Masih banyak permasalahan untuk menyelesaikan dengan teori Bayes, seperti penyelesaian teks dalam dokumen, masalah spam didalam email, deteksi *Malware* APK Android dan lain-lain.

Naïve Bayes Classifier atau bisa disebut sebagai *Multinomial Naïve Bayes* merupakan model penyederhanaan dari Metoda Bayes yang cocok dalam pengklasifikasian teks atau dokumen. Persamaannya adalah:

$V_{MAP} = \arg\max P(V_j \mid a_1, a_2, ...a_n)$ ……..... (3)

Menurut persamaan (3), maka persamaan (1) dapat ditulis:

$$V_{MAP} = v_j \in V \quad \arg\max \frac{P(a_1, a_2, \ldots a_n \mid v_j)\, P(v_j)}{P(a_1, a_2, \ldots a_n)} \ldots\ldots\ldots (4)$$

$P(a_1, a_2, \ldots\ldots a_n)$ sebagai konstan, sehingga dapat dihilangkan dan persamaan menjadi:

$$V_{MAP} = \arg\max_{v_j \in V} P(a_1, a_2, \ldots a_n \mid v_j)\, P(v_j) \ldots\ldots\ldots (5)$$

Nilai $P(a_1,a_2,\ldots..a_n|v_j)$ sulit dilakukan perhitungan, maka diasumsikan bahwa setiap *Malware* pada file APK tidak mempunyai keterkaitan.

Jenis Pengklasifikasi Naive Bayes:

1. Multinomial Naive Bayes:

Frekwensi kata yang ada didalam dokumen menggunakan parameter fitur atau prediktor sebagai pengklasifikasi (Balakrishnan and Kaur 2019), (Jiang et al. 2016). Metode ini digunakan untuk klasifikasi dokumen, yaitu apakah suatu dokumen termasuk dalam kategori teknologi, berita, olahraga, dll.

2. Bernoulli Naive Bayes (Singh et al. 2020):

Metode ini hamper sama dengan Mulitnomial Naive Bayes multinomial, perbedaan di prediktornya adalah variabel boolean. Parameter yang digunakan untuk memprediksi variabel kelas dengan bernilai ya atau tidak (Artur 2021).

3. Gaussian Naive Bayes:

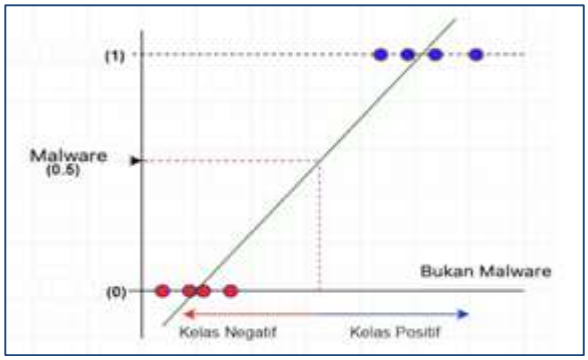
Ketika nilai-nilai kontinu dan tidak diskrit diambil oleh predictor yang diasumsikan sebagai nilai-nilai dari distribusi Gaussian (Petschke and Staab 2019), (Ontivero-Ortega et al. 2017).

Untuk lebih lanjut mengenai Naïve Bayes, penelitian ini membahas implementasi, menggunakan bahasa pemrograman *Python. Naive Bayes* juga merupakan salah satu *Machine Learning.*

Metode *Machine Learning* (Tsai et al. 2021), (Gundersen, Shamsaliei, and Isdahl 2022) adalah metode yang sangat sederhana dalam memproses setiap fitur dan menghasilkan keputusan yang lebih baik. Sehingga metode ini sangat cocok untuk berbagai penyelesaian.

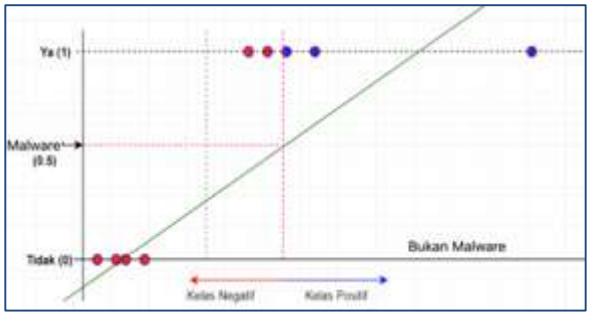
2.4. LOGISTIC REGRESSION

Sebelum membahas mengenai *Logistic Regression* (Ahmadini 2021), (Verma et al. 2021), (Książek, Gandor, and Pławiak 2021), maka disampaikan mengenai *Linear Regression*. *Linear Regression* adalah suatu cara permodelan masalah keterhubungan antara suatu variabel independen terhadap variabel dependen (Taraba 2021), (Brzeziński, Józefiak, and Zbiciak 2021). Contohnya adalah menentukan apakah file-file termasuk kedalam *Malware* atau bukan *Malware*.



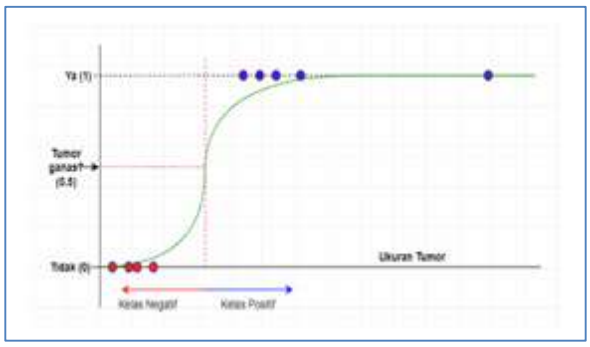
Gambar 2. Klasifikasi dengan Linear Regression

Gambar 2, garis dari *Linear Regression* dapat melakukan klasifikasi *Malware* dengan baik. Tetapi, bermasalah jika terdapat Outlier Data. Terlihat pada gambar 3.



Gambar 3. Data Outlier

Pada gambar 3, terdapat Outlier Data menyebabkan garis pada Linear Regression tidak dapat melakukan klasifikasi dengan baik. Terdapat 2 data sebagai bukan Malware, setelah terdapat outlier hanya ada 1 data bukan Malware.



Gambar 4. Logistic Regression

Pada gambar 4, terlihat sebuah metode yang dapat menangani permasalahan klasifikasi, sehingga menghasilkan klasifikasi yang lebih baik dan tidak terjadi kegagalan terhadap 2 data kelas positip pada Linear Regression, seperti pada gambar 3. Gambar 4 adalah metode Logistic Regression, dimana metode tersebut mampu menangani permasalahan data Outleir.

Logistic Regression merupakan algoritma untuk klasifikasi dengan mencari relasi antara fitur (input) *Discrete/Continue* dengan probabilitas output pada *Discrete/Continue* tertentu.

Beberapa tipe dari *Logistic Regression*:

Binary Logistic Regression: merupakan *Logistic Regression* yang mengklasifikasikan dua output saja. Misalkan: Ya dan Tidak, *Malware* dan bukan *Malware*.

Multinomial Logistic Regression merupakan *Logistic Regression* yang melakukan klasifikasi dua output atau lebih. Misalkan kelas famili *Malware (Ransomware, Banking Malware, SMS Malware, Trojan).*

Ordinal Logistic Regression merupakan *Logistic Regression* yang melakukan klasifikasi dua kelas atau lebih, dengan menghasilkan output yang berurutan. Misalkan membagi kelas siswa ke dalam range Index Prestasi Kumulatif (IPK) seperti 1.0, 2.0, 3.0 sampai 4.0.

Logistic Function merupakan fungsi yang dihasilkan dari persamaan Y didalam Linear Function dan Y didalam *Sigmoid Function.* Yang bertujuan untuk menggambarkan data-data yang ke dalam bentuk fungsi Sigmoid.

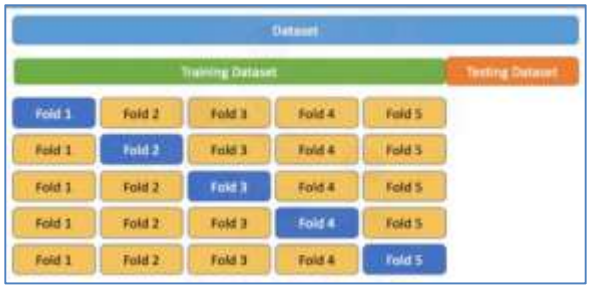
2.5. Cross Validation

Kebutuhan untuk memvalidasi stabilitas dari model Machine Learning. Penyesuaian model dengan data tarining dan mengharapkan hasil yang akurat untuk data yang riil. Karena itulah memerlukan jaminan bahwa model yang didapat sebagian besar menggunakan pola dari data yang benar. Disamping itu dataset tidak terlalu banyak noise atau bias.

Validation adalah proses mengambil keputusan hasil numerik untuk kuantitas hipotesis antara variabel, yang diterima sebagai deskripsi data disebut validasi. Estimasi error untuk model dilakukan setelah training disebut evaluasi residual. Estimasi numerik dari perbedaan respons yang nyata diprediksi dan respons yang dilakukan disebut kesalahan training. Untuk model yang kurang pas atau terlalu pas dengan data. Jadi, masalah dengan teknik evaluasi ini adalah tidak mengindikasi seberapa bagus learning yang menggeneralisasi ke dataset independen. Model ini disebut sebagai *Cross Validation*.

Cross Validation K-Fold adalah metode untuk mengatasi dataset yang tidak cukup untuk melakukan training. Adanya penghapusan sebagian dataset untuk validasi akan berdampak masalah kekurangan Dataset. Dengan melakukan pengurangan dataset training, akan menimbulkan resiko kehilangan pola penting dalam Dataset. Disamping itu dapat menimbulkan tingkat kesalahan yang disebabkan oleh bias. Oleh karena itu ada metode yang menyediakan banyak dataset untuk training model serta menyisakan banyak dataset untuk validasi. K Fold Cross Validation dapat mengatasi masalah tersebut.

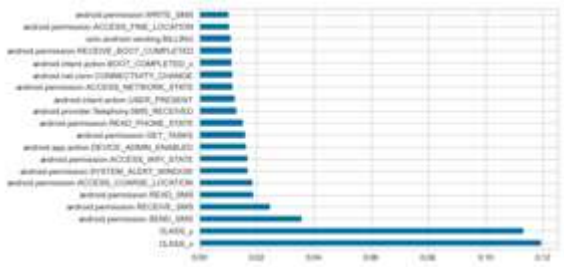
Proses K Fold Cross Validation, Dataset dibagi menjadi K subset. Selanjutnya metode Holdout diulang K kali, sehingga setiap kali, salah satu dari k subset digunakan sebagai test set/validation set dan k-1 subset lainnya disatukan untuk membentuk training set. Estimasi kesalahan dirata-ratakan pada semua K eksperimen untuk mendapatkan efektivitas total model. Seperti yang dapat dilihat, setiap titik data berada dalam set validasi tepat satu kali, dan berada dalam dataset training k-1 kali. Secara signifikan mengurangi bias karena menggunakan sebagian besar dataset untuk pemasangan. Disamping itu signifikan mengurangi varians karena sebagian besar dataset digunakan di dalam set validasi. Pertukaran dataset training dan dataset test, menambah efektivitas metode ini. Sebagai aturan dan bukti empiris, nilai K = 5 atau K = 10 umumnya lebih disukai, tetapi tidak ada ketetapan mengenai nilai K.



Gambar 5. K-Flod Cross Validation

3. HASIL DAN PEMBAHASAN

Didalam eksperiment ini menggunakan bahasa pemrograman python 3.8, library pandas, numpy, sklearn. Sebelum menggunakan algoritma *Naive Bayes* dan *Logistic Regression*, *Dataset Malware* perlu dilakukan analisa. Salah satunya ada mencari fitur yang paling penting didalam Dataset seperti pada gambar 5.



Gambar 6. Feature Importance

1. Penggunaan algoritma *Naive Bayes*.

Algoritma Naive Bayes sangat sederhana jika ingin menggunakan Library Sklearn. Kesederhanaan ini bukan berarti kemampuannya rendah, tetapi kemampuan akurasi dalam menghasilkan model sangat bagus. Salah satu kelebihannya tidak membutuhkan dataset yang besar jika ingin melakukan klasifikasi.

```
# ------------------------ Naive Bayes Python ----------------------------
Naive = GaussianNB()
Naive.fit(X_train, y_train)
y_pred = Naive.predict(X_test)
y_pred

# ---------------------------------- Output ------------------------------------
GaussianNB()
Array   ([2, 0, 4, 1, 4, 3, 1, 0, 4, 1, 4, 3, 5, 4, 2, 3, 4, 2, 1, 1, 5, 2,
          0, 4, 2, 5, 1, 2, 3, 2, 3, 5, 5, 0, 1, 2, 3, 0, 5, 3, 5, 3, 2, 2,
          2, 5, 4, 4, 5, 2, 4, 1, 1, 1, 5, 5, 1, 5, 0, 4, 3, 4, 4, 2, 5, 5,
          0, 2, 2, 5, 2, 0, 5, 0, 2, 2, 5, 3, 5, 5, 1, 3, 0, 4, 4, 2, 4, 4,
          5, 0, 5, 5, 0, 5, 5, 4, 0, 5, 3, 0, 1, 1, 2, 0, 3, 3, 3, 3, 3, 3,
          2, 0, 3, 4, 2, 0])
```

Model Naïve Bayes sama seperti Model *Logistic Regression* dalam melakukan Split untuk Dataset Training dan Testing. Tujuan Dataset Training dilakukan Split Dataset untuk proses belajar pada Machine Learning.

```
# ------------------- Akurasi Naive Bayes Python ---------------------
model1 = metrics.accuracy_score(y_test, y_pred)
Akurasi = model1 * 100
Tampil = str(Akurasi) + ' %'
print('Akurasi Naive Bayes = ', Tampil)

# ------------------------------ Output -------------------------------------
Akurasi Naive Bayes = 96,55 %
```

Gambar 7. Confusion Matrix using Seaborn

2. Penggunaan algoritma *Logistic Regression*

Eksperimen dengan menggunakan python, langsung memanggil fungsi LogisticRegressin(), yang merupakan module dari sklearn.linear_model. Jika dihitung secara manual, menggunakan excel hasil perhitungannya akan sama.

```
# ------------------------ Logistic Regression ----------------------
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
y_pred

# ------------------------------ Output -----------------------------
LogisticRegression()
array([3, 1, 2, 0, 1, 5, 1, 4, 1, 4, 0, 1, 5, 0, 0, 1, 0, 0, 4, 0, 3, 4,
       3, 3, 4, 3, 3, 0, 4, 2, 1, 4, 3, 3, 1, 4, 3, 3, 1, 1, 5, 3, 0, 3,
       0, 2, 1, 2, 0, 5, 5, 5, 4, 2, 4, 0, 3, 4, 5, 5, 1, 0, 4, 0, 2, 5,
       3, 3, 0, 3, 3, 0, 2, 4, 3, 5, 5, 3, 2, 2, 3, 0, 0, 3, 2, 3, 0, 5,
       4, 0, 2, 1, 0, 5, 1, 3, 1, 0, 5, 4, 4, 3, 5, 5, 2, 4, 3, 2, 0, 1,
       1, 3, 1, 4, 0, 5])
```

Logistic Regression pendekatan hampir sama dengan *Naive Bayes*, membagi Dataset Training dan Dataset Testing.

```
model1 = metrics.accuracy_score(y_test, y_pred)
Akurasi = model1 * 100
Tampil = str(Akurasi) + ' %'
print('Akurasi Logistic Regression = ', Tampil)

# ----------------------Output ----------------------
Akurasi Logistic Regression = 91,38 %
```

Selanjutnya dengan melakukan cross validation = 5, maka didapat hasil sebagai berikut:

```
logmodel = LogisticRegression()
score =  cross_val_score(logmodel, features, label, cv=5)
print("CROSS VALIDATION SCORE : ",statistics.mean(score))
logmodel.fit(X_train, y_train)
print("SCORE : ",logmodel.score(X_test, y_test))

# ---------------------------- Output -------------------------------
CROSS VALIDATION SCORE:  0.9428
SCORE:  0.9137
```

Terjadi peningkatan akurasi setelah dilakukan Cross Validation (Valente et al. 2021), dari akurasi 91,37% menjadi 94,28%.

Evaluasi dari model *Logistic Regression*:

```
classes = ['0','1','2','3','4','5']
y_pred = logmodel.predict(X_test)
print(classification_report(y_test, y_pred, target_names=classes))
```

	precision	recall	f1-score	support
0	0.91	0.95	0.93	22
1	1.00	0.86	0.92	21
2	0.85	0.92	0.88	12
3	0.88	0.85	0.86	26
4	1.00	1.00	1.00	18
5	0.84	0.94	0.89	17
accuracy			0.91	116
macro avg	0.91	0.92	0.91	116
weighted avg	0.92	0.91	0.91	116

Gambar 8. Evaluasi model

Pada gambar 12, *Class* 0 mewakili dari *Malware* dari *Adware* APK. *Class* 1 mewakili dari *Malware Banking* APK. *Class* 2 mewakili dari *Malware* Ransomware APK. Class 3 mewakili dari Malware Riskware APK. Class 4 mewakili dari

Malware SMS APK. Class 5 mewakili dari *Benign* APK atau normal APK.

Gambar 9. Klasifikasi Logistic Regression

Untuk melakukan pengukuran performa pada permasalahan klasifikasi *Logistic Regression*, dimana outputnya bisa terdiri atas dua kelas atau lebih. *Confusion Matrix* juga menyajikan tabel matriks dengan empat kelompok yang berbeda pada nilai riil dan nilai prediksi. Terdapat empat definisi yang mewakili klasifikasi dari *Confusion Matrix*. Kelompok *True Positif* (TP), *True Negatif* (TN), *False Positif* (FP), dan *False Negatif* (FN).

Gambar 10. Confusion Matrix using Seaborn

4. KESIMPULAN

Penelitian ini sangat berbeda dengan beberapa penelitian yang lainnya, dimana ekstraksi fitur *Permission* dan fitur *Intent* dengan menggunakan statis analisis, mampu mendeteksi keluarga *Malware*. Penggunaan algoritma *Naive Bayes* mampu mengklasifikasi keluarga *Malware* dengan tingkat akurasi 96,55%, sedangkan algoritma *Logistic Regression* akurasinya 91,37%. Meskipun akurasi tidak setinggi penelitian sebelumnya, teknik analisa statis dengan fitur *Permission* dan fitur *Intent* cukup sederhana untuk mendeteksi file APK Android adalah *Malware* atau bukan *Malware*. Penyelesaian dengan teknik yang sederhana merupakan keunggulan dalam menyelesaikan permasalahan pada deteksi *Malware* Android.

Terjadi peningkatan akurasi bila memakai *Cross Validation*, untuk algoritma *Logistic Regression* dari akurasi 91,375 menjadi 94,278.

5. SARAN

Saat ini masih meneliti dengan menggunakan algoritma *Naive Bayes* dan *Logistic Regression*. Penelitian dapat dilanjutkan dengan metode statis dan algortima *Supervised*, maka dapat menggunakan *Support Vector Machine*, *Decision Tree.* Jika label pada dataset dihilangkan dapat menggunakan algoritma *Unsupervised* seperti *K-Mean* untuk mendapatkan klas dari *Malware*.

DAFTAR PUSTAKA

Abawajy, Jemal, Abdulbasit Darem, and Asma A. Alhashmi. 2021. "Feature Subset Selection for Malware Detection in Smart Iot Platforms." *Sensors (Switzerland)* 21(4):1–19. doi: 10.3390/s21041374.

Ahmadini, Abdullah Ali H. 2021. "A Novel Technique for Parameter Estimation in Intuitionistic Fuzzy Logistic Regression Model." *Ain Shams Engineering Journal* (xxxx). doi: 10.1016/j.asej.2021.06.004.

Artur, Mechetin. 2021. "Review the Performance of the Bernoulli Naïve Bayes Classifier in Intrusion Detection Systems Using Recursive Feature Elimination with Cross-Validated Selection of the Best Number of Features." *Procedia Computer Science* 190(2019):564–70. doi: 10.1016/j.procs.2021.06.066.

Balakrishnan, Vimala, and Wandeep Kaur. 2019. "String-Based Multinomial Naïve Bayes for Emotion Detection among Facebook Diabetes Community." *Procedia Computer Science* 159:30–37. doi: 10.1016/j.procs.2019.09.157.

Brzeziński, Karol, Kazimierz Józefiak, and Artur Zbiciak. 2021. "On the Interpretation of Shear Parameters Uncertainty with a Linear Regression Approach." *Measurement: Journal of the International Measurement Confederation* 174(May 2020). doi: 10.1016/j.measurement.2020.108949.

Ding, Chao, Nurbol Luktarhan, Bei Lu, and Wenhui Zhang. 2021. "A Hybrid Analysis-Based Approach to Android Malware Family Classification." *Entropy* 23(8). doi:

10.3390/e23081009.

Garg, Shivi, and Niyati Baliyan. 2019. "Data on Vulnerability Detection in Android." *Data in Brief* 22:1081–87. doi: 10.1016/j.dib.2018.12.038.

Gundersen, Odd Erik, Saeid Shamsaliei, and Richard Juul Isdahl. 2022. "Do Machine Learning Platforms Provide Out-of-the-Box Reproducibility?" *Future Generation Computer Systems* 126:34–47. doi: 10.1016/j.future.2021.06.014.

Jiang, Liangxiao, Shasha Wang, Chaoqun Li, and Lungan Zhang. 2016. "Structure Extended Multinomial Naive Bayes." *Information Sciences* 329:346–56. doi: 10.1016/j.ins.2015.09.037.

Książek, Wojciech, Michał Gandor, and Paweł Pławiak. 2021. "Comparison of Various Approaches to Combine Logistic Regression with Genetic Algorithms in Survival Prediction of Hepatocellular Carcinoma." *Computers in Biology and Medicine* 134. doi: 10.1016/j.compbiomed.2021.104431.

Mahindru, Arvind, and A. L. Sangal. 2021. "FSDroid:- A Feature Selection Technique to Detect Malware from Android Using Machine Learning Techniques: FSDroid." *Multimedia Tools and Applications*. doi: 10.1007/s11042-020-10367-w.

Mat, Sharfah Ratibah Tuan, Mohd Faizal Ab Razak, Mohd Nizam Mohmad Kahar, Juliza Mohamad Arif, Salwana Mohamad, and Ahmad Firdaus. 2021. *Towards a Systematic Description of the Field Using Bibliometric Analysis: Malware Evolution*. Vol. 126. Springer International Publishing.

Ontivero-Ortega, Marlis, Agustin Lage-Castellanos, Giancarlo Valente, Rainer Goebel, and Mitchell Valdes-Sosa. 2017. "Fast Gaussian Naïve Bayes for Searchlight Classification Analysis." *NeuroImage* 163:471–79. doi: 10.1016/j.neuroimage.2017.09.001.

Petschke, Danny, and Torsten E. M. Staab. 2019. "A Supervised Machine Learning Approach Using Naive Gaussian Bayes Classification for Shape-Sensitive Detector Pulse Discrimination in Positron Annihilation Lifetime Spectroscopy (PALS)." *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 947(September):162742. doi: 10.1016/j.nima.2019.162742.

Rashed, Mohammed, and Guillermo Suarez-Tangil. 2021. "An Analysis of Android Malware Classification Services." *Sensors* 21(16):5671. doi: 10.3390/s21165671.

Singh, Mandeep, Mohammed Wasim Bhatt, Harpreet Singh Bedi, and Umang Mishra. 2020. "Performance of Bernoulli's Naive Bayes Classifier in the Detection of Fake News." *Materials Today: Proceedings* (xxxx). doi: 10.1016/j.matpr.2020.10.896.

Su, Xin, Qingbo Gong, Yi Zheng, Xuchong Liu, and Kuan Ching Li. 2020. "An Informative and Comprehensive Behavioral Characteristics Analysis Methodology of Android Application for Data Security in Brain-Machine Interfacing." *Computational and Mathematical Methods in Medicine* 2020. doi: 10.1155/2020/3658795.

Taraba, Peter. 2021. "Linear Regression on a Set of Selected Templates from a Pool of Randomly Generated Templates." *Machine Learning with Applications* 6(May):100126. doi: 10.1016/j.mlwa.2021.100126.

Tsai, Chun-Wei, Yi-Ping Chen, Tzu-Chieh Tang, and Yu-Chen Luo. 2021. "An Efficient Parallel Machine Learning-Based Blockchain Framework." *ICT Express* (xxxx):0–7. doi: 10.1016/j.icte.2021.08.014.

Valente, Giancarlo, Agustin Lage Castellanos, Lars Hausfeld, Federico De Martino, and Elia Formisano. 2021. "Cross-Validation and Permutations in MVPA: Validity of Permutation Strategies and Power of Cross-Validation Schemes." *NeuroImage* 238(April):118145. doi: 10.1016/j.neuroimage.2021.118145.

Verma, Rajesh, Navdha Bhardwaj, Pushap Deep Singh, Arnav Bhavsar, and Vishal Sharma. 2021. "Estimation of Sex through Morphometric Landmark Indices in Facial Images with Strength of Evidence in Logistic Regression Analysis." *Forensic Science International: Reports* 4:100226. doi: 10.1016/j.fsir.2021.100226.

Wang, Chao, Zhizhong Wu, Xi Li, Xuehai Zhou, Aili Wang, and Patrick C. K. Hung. 2014. "SmartMal: A Service-Oriented Behavioral Malware Detection Framework for Mobile Devices." *Scientific World Journal* 2014. doi: 10.1155/2014/101986.

Wang, Xiaolei, Yuexiang Yang, and Yingzhi Zeng. 2015. "Accurate Mobile Malware Detection and Classification in the Cloud."

SpringerPlus 4(1):1–23. doi: 10.1186/s40064-015-1356-1.

Yuan, Hongli. 2020. "MADFU : An Improved Malicious Application." *Entropy*.

Yuan, Hongli, Yongchuan Tang, Wenjuan Sun, and Li Liu. 2020. "A Detection Method for Android Application Security Based on TF-IDF and Machine Learning." *PLoS ONE* 15(9 September):1–19. doi: 10.1371/journal.pone.0238694.