Klasifikasi Malicious Websites Menggunakan Algoritma K-NN Berdasarkan Application Layers dan Network Characteristics

Malicious Websites Classification Using K-NN Algorithm Based on Application Layers and Network Characteristics

[1]**Green Arther Sandag,** [2]**Jonathan Leopold,** [3]**Vinky Fransiscus Ong**
[1,2,3] Progam Studi Teknik Informatika, Universitas Klabat, Airmadidi
e-mail: [1]greensandag@unklab.ac.id

Abstrak

Dalam kehidupan di era teknologi sekarang ini semua aktivitas manusia telah dipengaruhi oleh internet. Berbagi informasi, komunikasi, sosialisasi, berbelanja, berbisnis, pendidikan dan banyak hal lainnya yang dapat dilakukan menggunakan internet. Seiring dengan berkembangnya internet berbagai macam ancaman keamanan menjadi lebih beragam. Virus adalah musuh nomor satu di internet. Virus memanfaatkan berbagai metode untuk dapat menghindari anti-virus, salah satunya adalah Malware. Malware adalah salah satu kode berbahaya yang dapat mengubah, merusak dan mencuri data pribadi yang dapat merugikan individual ataupun kelompok. Penelitian ini akan memprediksi malicious website berdasarkan application layer dan network characteristics menggunakan metode K-Nearest Neighbor. Penelitian ini menggunakan metode data cleaning dan data reduction untuk data preprocessing, dan feature selection untuk pemilihan attribut yang paling berpengaruh pada malicious website. Untuk memprediksi malicious website penulis menggunakan algoritma K-NN dengan hasil 2,42% precision lebih tinggi dibandingkan dengan penelitian sebelumnya yang menggunakan algoritma Naïve Bayes.

Keywords : Klasifikasi, Network Characteristics, Malicious Websites, Application Layers, K-NN, Naïve Bayes

Abstract

In this era of technology all human activities have been influenced by the internet. Information sharing, communication, socialization, shopping, business, education and many other things that can be done using the internet or the Web. One of the development of the internet a variety of security threats are becoming more diverse. Virus is the number one enemy on the internet. Virus utilize various methods to be able to avoid anti-virus, one of which is malware. Malware is one of the malicious codes that can alter, destroy and steal personal data that could harm a person or a group. In this research will predict malicious website based on application layer and network characteristics using K-Nearest Neighbor method. This research uses data cleaning and data reduction methods for preprocessing data, and feature selection to select most influential attributes in malicious website, to predict malicious website writers use K-NN algorithm with a result of 2,42% precision higher than previous research using Naïve Bayes algorithm.

Keywords : Classification, Network Characteristics, Malicious Websites, Application Layers, K-NN, Naïve Bayes

1. PENDAHULUAN

Dengan berkembangnya teknologi informasi, internet telah menjadi alat yang semakin populer dalam kehidupan sehari-hari. Namun, bersama dengan itu juga menyebabkan banyak ancaman terhadap keamanan *Web*. Saat ini, keamanan *Web* menjadi faktor kunci dalam pengembangan *web*. Untuk meningkatkan keamanan situs *Web*, maka digunakan firewall pada suatu situs untuk keamanan, dan mencoba untuk mendeteksi adanya celah keamanan yang dapat dimanfaatkan oleh penyerang atau orang yang tidak bertanggung jawab [1].

Modus kejahatan di dunia *cyber* saat ini sangat beragam. Teknik yang digunakan oleh penyeranganpun semakin beragam dan kompleks. Berbagai serangan tersebut melibatkan *malicious software* atau yang bisa disebut *malware* yang merupakan suatu program jahat. Ancaman *malware* dan penyebarannya bisa melalui berbagai cara, yaitu cara yang sering menyisipkan disebuah aplikasi ataupun *file* tertentu [2].

Malware dapat menyebar dengan cepat di jaringan tanpa campur tangan dari pengguna. Sistem pendeteksi *malware* masih menjadi masalah, karena *malware* baru yang selalu berveolusi dengan menggunakan teknik yang berbeda untuk menghindari metode pendeteksian. Untuk itu diperlukan pengembangan teknik deteksi *malware* yang dapat mendeteksi *malware* secara akurat. Sasaran utama dari *malware* adalah untuk memata-matai seseorang, mencuri informasi atau data pribadi orang lain seperti *m-banking*, membobol *security program* dan lain-lain. Pada umumnya, sebuah *malware* diciptakan untuk merusak atau membobol suatu *software* atau sistem operasi melalui *script* yang dirahasiakan, dalam arti lain disisipkan secara tersembunyi oleh penyerang.

Perkembangan *malware* semakin pesat mengharuskan pengguna komputer semakin waspada agar informasi pribadi ataupun *file* yang penting tidak diambil oleh orang yang tidak berhak [2]. Demikian juga bagi para pelaku bisnis baik perusahaan maupun perorangan yang bergantung dengan sistem komputer untuk menjaga agar datanya tetap aman, dan *malware* tidak dapat mencuri atau merusak data yang dimiliki.

Hari ini internet menjadi populer sehingga mengubah cara berpikir dan standar hidup orang banyak. Salah satu factor pendorog dari banyaknya malware berevolusi yaitu pesatnya pertumbuhan e-commerce, persoalan computer yang tidak aman dan penetrasi internet yang terus meningka [2]. Sehingga membutuhkan suatu teknik baru untuk mendeteksi malware seperti machine learning. Teknologi machine learning dapat mendeteksi *malware* dengan mempelajari perilaku malware dan executable yang berbahaya [3].

Salah satu cara untuk deteksi malicious website yaitu dengan "*exploit*" application layer. *Application layer* adalah *layer* tertinggi dalam *Open System Interconnection* (OSI) *layer*. *Layer* ini berfokus pada *process-to-process communication* melewati *IP network* dan menyediakan layanan *interface* komunikasi dan *user services*. *Application layer* menyediakan banyak layanan diantaranya: *file transfer, network data sharing, web surfing, web chat*, dan *email clients* [4]. Untuk membedakan *malicious website* dan yang bukan maka digunakan metode klasifikasi. Klasifikasi merupakan proses untuk mengambil *input* dan akan dimasukkan ke dalam kelas yang sudah ada. Metode ini dapat digunakan untuk memprediksi data baru.

Penelitian sebelumnya oleh Altaher (2017) yaitu mengklasifikasi *phishing website* menggunakan *hybrid K-NN* dan *SVM* dengan menggabungkan kedua algoritma tersebut sehingga mendapatkan hasil akurasi sebesar 90.04% [5]. Kemudian pada penelitian yang lain tentang klasifikasi *malicious website* berdasarkan *url features*, diketahui bahwa penelitian tersebut menggunakan algoritma *Naïve Bayes* dengan *precision* sebesar 87% [6]. Pada

penelitian ini penulis mencoba untuk menggunakan algoritma *K-Nearest Neighbour* untuk memprediksi *malicious website*.

Tujuan dari penelitian ini adalah untuk dapat memprediksi *malicious website* berdasarkan *application layers* dan *network characteristics*. Manfaat dari penelitian ini adalah kedepannya mampu memberikan informasi yang berguna bagi masyarakat untuk membedakan website or *malicious website*.

2. METODE PENELITIAN

2.1 Malicious and benign websites data

Data yang digunakan pada penelitian ini adalah *Malicious and Benign Websites dataset* yang dapat di akses di Kaggle [7]. *Dataset* ini memiliki 1781 *rows*, dan 20 *attributes* yang dijelaskan di Tabel 1.

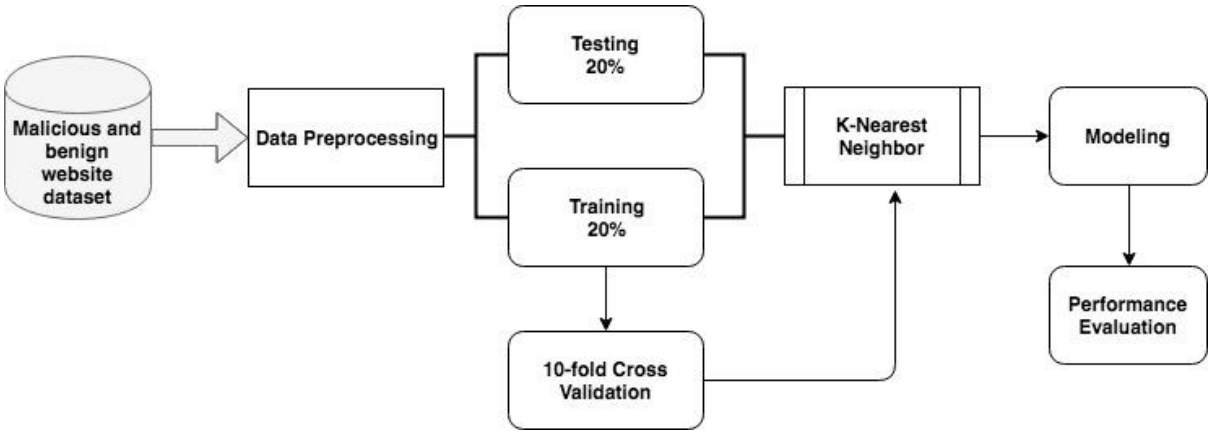
Tabel 1. Parameter Dataset

Parameters	Details	Value
URL	Identifikasi dari URL yang dianalisis dalam penelitian ini	*Polynominal*
URL_LENGTH	Jumlah karakter dari URL	*Integer*
NUMBER_SPECIAL_CHARACTERS	Jumlah special karakter dari URL seperti: "/", "%", "#", "&", ". ", "="	*Integer*
CHARSET	Standar dari encoding karakter	*Polynominal*
SERVER	Jenis server yang digunakan	*Polynominal*
CONTENT_LENGTH	Besarnya konten dari HTTP	*Integer*
WHOIS_COUNTRY	Negara asal dari server	*Polynominal*
WHOIS_STATEPRO	Kota / Provinsi dari server	*Polynominal*
WHOIS_REGDATE	Tanggal registrasi awal server	*Polynominal*
WHOIS_UPDATED_DATE	Tanggal terakhir kali server diupdate	*Polynominal*
TCP_CONVERSATION_EXCHANGE	Jumlah pertukaran TCP packet antara server dan client	*Integer*
DIST_REMOTE_TCP_PORT	Jumlah dari port yang diketahui berbeda dari TCP	*Integer*
REMOTE_IPS	Jumlah dari IP yang terhubung	*Integer*
APP_BYTES	Jumlah byte yang ditransfer	*Integer*
SOURCE_APP_PACKETS	Paket yang dikirim ke server	*Integer*
REMOTE_APP_PACKETS	Paket yang diterima dari server	*Integer*
SOURCE_APP_BYTES	Jumlah byte yang ditransfer ke server	*Integer*
REMOTE_APP_BYTES	Jumlah byte yang diterima dari server	*Integer*
APP_PACKETS	Jumlah IP yang di generate selama komunikasi antara honeypot dan sever	*Integer*
DNS_QUERY_TIMES	Jumlah DNS yang di generate selama komunikasi antara honeypot dan server	*Integer*

2.2 Desain Penelitian

Gambar 1 memperlihatkan proses klasifikasi *malicious website*. Proses pertama adalah mengambil *Malicious and Benign websites dataset* yang diambil dari Kaggle, dilanjutkan *data preprocessing* untuk mengolah data. Dataset dibagi menjadi 80% *training data* yang terdiri atas 1425 data yang terbagi atas 173 *malicious* dan 1252 *benign data*, dan 20% *testing data* yang terdiri atas 356 data yang terbagi atas 43 *malicious* dan 313 *benign data*. Penelitian ini

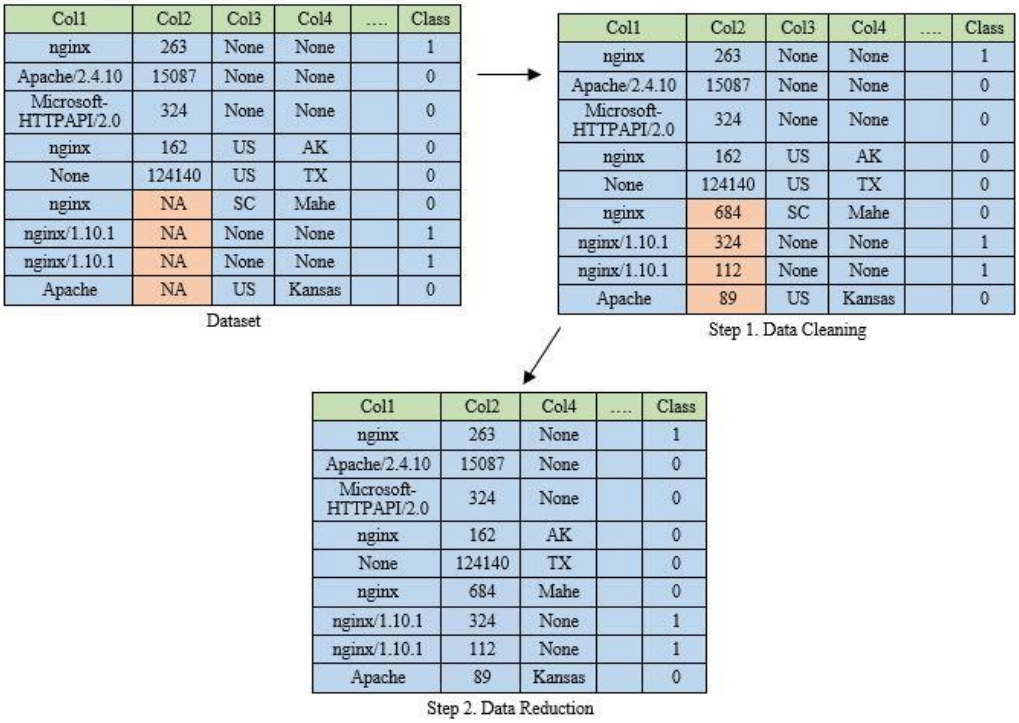
menggunakan *10-fold cross validation* untuk membagi data menjadi 10 bagian dan diuji sebanyak 10 kali sebelum melakukan *modelling*. Selanjutanya data di proses menggunakan algoritma *K-nearest neighbor*, yang kemudian akan dibuat modelnya dan dievaluasi.



Gambar 1. Arsitektur Untuk Klasifikasi *Malicious Website*

2.2.1. Data Preprocessing

Data *preprocessing* dibagi menjadi 2 bagian, yaitu: *data cleaning* dan *data reduction*. *Data cleaning* adalah proses pembersihan data *incomplete* pada *attribute* di *dataset* untuk membuat data menjadi lebih konsisten. Sedangkan, *data reduction* adalah proses untuk menghapus data pada *attribute* yang kurang dominan sehingga data bisa dikurangi, namun tetap menghasilkan data yang akurat. Gambar 2 mendeskripsikan proses *data cleaning* yang dilakukan yaitu, mengganti nilai kosong dengan nilai yang baru.



Dataset

Col1	Col2	Col3	Col4	....	Class
nginx	263	None	None		1
Apache/2.4.10	15087	None	None		0
Microsoft-HTTPAPI/2.0	324	None	None		0
nginx	162	US	AK		0
None	124140	US	TX		0
nginx	NA	SC	Mahe		0
nginx/1.10.1	NA	None	None		1
nginx/1.10.1	NA	None	None		1
Apache	NA	US	Kansas		0

Step 1. Data Cleaning

Col1	Col2	Col3	Col4	....	Class
nginx	263	None	None		1
Apache/2.4.10	15087	None	None		0
Microsoft-HTTPAPI/2.0	324	None	None		0
nginx	162	US	AK		0
None	124140	US	TX		0
nginx	684	SC	Mahe		0
nginx/1.10.1	324	None	None		1
nginx/1.10.1	112	None	None		1
Apache	89	US	Kansas		0

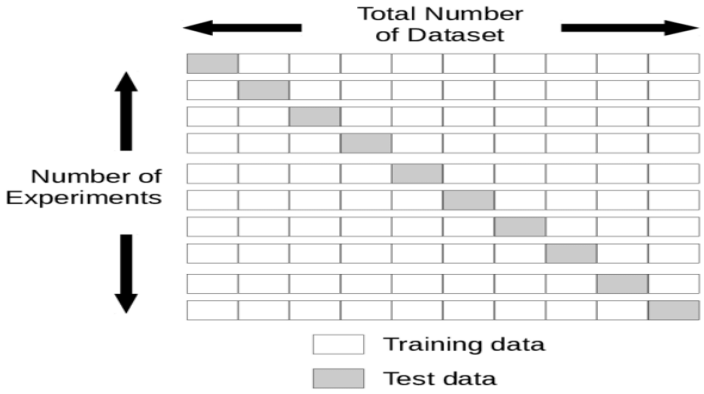
Step 2. Data Reduction

Col1	Col2	Col4	....	Class
nginx	263	None		1
Apache/2.4.10	15087	None		0
Microsoft-HTTPAPI/2.0	324	None		0
nginx	162	AK		0
None	124140	TX		0
nginx	684	Mahe		0
nginx/1.10.1	324	None		1
nginx/1.10.1	112	None		1
Apache	89	Kansas		0

Gambar 2. Proses *Data Cleaning* dan *Data Reduction*

2.2.2. 10-Fold Cross Validation

Setelah data telah dibagi menjadi 80% *data training* dan 20% *data testing*, maka akan dilakukan 10-*fold cross validation* pada *data training*. *Cross Validation* adalah teknik untuk mengevaluasi model dengan cara mempartisi sampel asli ke dalam *training set* untuk melatih model, dan *test set* untuk mengevaluasi model. Dalam *k-fold cross validation*, sampel asli secara acak dipartisi dalam *k equal size subsample*. Dari *subsample* k, satu *subsample* akan digunakan sebagai *testing data* dan sisanya akan menjadi *training data*. Proses *cross validation* akan diulang sebanyak k kali (kelipatan), dengan masing – masing dari *subsample* k digunakan sekali sebagai *validation data* [8]. Pada Gambar 3 menunjukkan proses 10-*fold cross validation*, data dibagi menjadi 10 partisi dan akan diuji sebanyak 10 kali sebelum dibuat modelnya.



Gambar 3. 10 Fold Cross Validation

2.2.3. K-Nearest Neighbor

Setelah proses 10-*fold cross validation* maka data akan diproses menggunakan algoritma *K-NN*. Algoritma *K-NN* adalah metode untuk klasifikasi objek berdasarkan *training examples* yang terdekat. *K-NN* adalah *instance-based learning* atau disebut juga *lazy learning*, karena fungsinya hanya didekati secara lokal dan semua perhitungan ditunda hingga proses klasifikasi. *K-NN* adalah teknik klasifikasi yang paling sederhana ketika tidak ada pengetahuan tentang distribusi data [9]. Jarak dalam *K-NN* dihitung menggunakan *Euclidean Distance* dengan rumus:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(xi - yi)}^{2}, \quad \text{..... (1)}$$

Keterangan:

d : Jarak *i* : Jumlah data *n* : Banyaknya data

x : Titik awal *y*: Titik akhir

2.2.4. Performance Evaluation

Setelah pembuatan model maka langkah selanjutnya adalah melakukan evaluasi dengan *performance evaluation*. *Performance evaluation* berguna untuk menguji performa dari *classifier*. *Recall*, *precision*, dan *accuracy*. *Recall* adalah kumpulan data positif yang diklasifikasikan dengan benar sebagai data positif. *Precision* adalah kumpulan data yang diklasifikasikan sebagai positif yang benar – benar positif. *Accuracy* adalah ketepatan klasifikasi

data [10]. Berikut ini adalah rumus *recall*, *precision*, dan *accuracy* dalam *performance evaluation*:

Rumus *Recall*:

$$Recall = \frac{TP}{TP+FN} \text{, ... (2)}$$

Rumus *Precision*:

$$Precision = \frac{TP}{TP+FN} \text{, ... (3)}$$

Rumus *Accuracy*:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \text{, ... (4)}$$

Keterangan:

TP	: Nilai *true positive*	TN	: Nilai *true negative*
P	: Jumlah data positif	FP	: Nilai *false positive*
N	: Jumlah data *negative*	FN	: Nilai *false negative*

2.3. *Feature Selection*

Metode *feature selection* memegang peran penting dalam memilih *attribute* yang siginifikan, melalui penghapusan *attribute* yang tidak relevan, dan oleh karena itu dapat digunakan untuk identifikasi *attribute* yang berpengaruh [11]. Penulis menggunakan metode *information gain ratio* untuk menentukan berapa besar pengaruh suatu *attribute* dalam *dataset*. Machine *learning information gain* dapat digunakan untuk membuat peringkat dari *attributes*. *Attributes* yang memiliki *information gain* yang tinggi harus diberi peringkat lebih tinggi daripada *attributes* yang lain karena lebih berpengaruh dalam mengklasifikasikan data [12]. Berikut ini adalah rumus dalam *information gain*:

$$IG(A) = H(S) - \sum \frac{Si}{S} H(Si), \text{ .... (5)}$$

Keterangan:

H(S) : Entropi dari *dataset*

H(Si) : Entropi dari i *subset* yang dihasilkan oleh partisi S

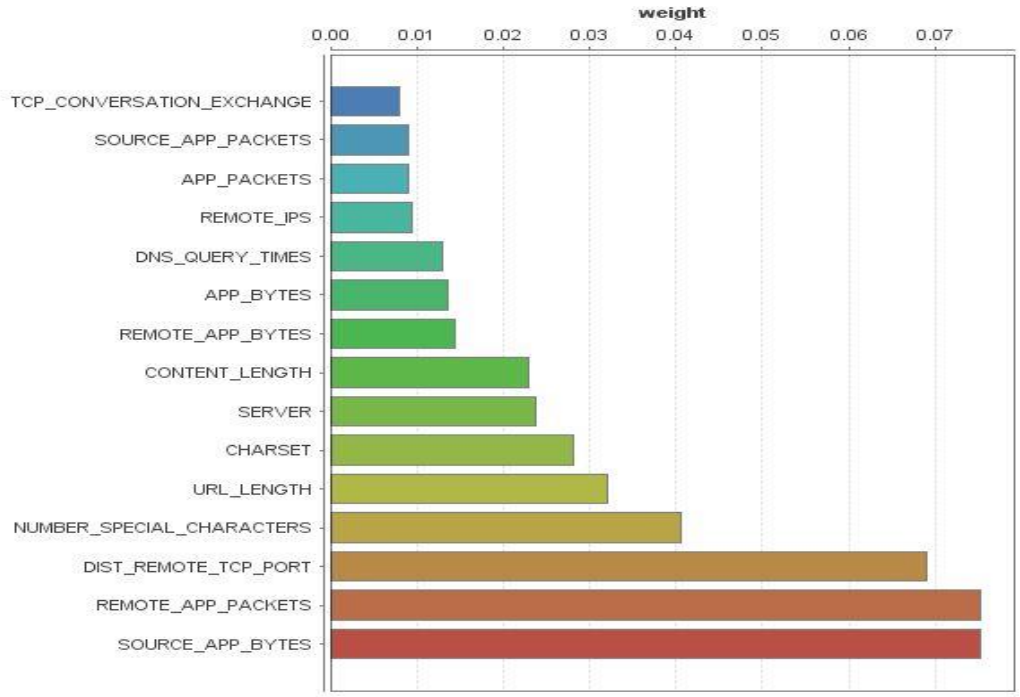
A : Atribut dalam dataset

3. HASIL DAN PEMBAHASAN

Pada bagian ini penulis melakukan analisa terhadap *malicious and benign website dataset* menggunakan algoritma K-NN

3.1 Hasil Feature Selection

Berdasarkan Gambar 4 dapat disimpulkan bahwa *attribute REMOTE_APP_PACKETS* dan *SOURCE_APP_BYTES* paling berpengaruh dalam memprediksi *malicious* dan *benign website* dan memiliki *weight* sebesar 0.075. Untuk attribute lain, *DIST_REMOTE_TCP_PORT* 0.069, *NUMBER_SPECIAL_CHARACTERS* 0.040, *URL_LENGTH* 0.032, *CHARSET* 0.028, *SERVER* 0.024, *CONTENT_LENGTH* 0.023, *REMOTE_APP_BYTES* 0.014, *APP_BYTES* 0.014, *DNS_QUERY_TIMES* 0.013, *REMOTE_IPS* 0.009, *SOURCE_APP_PACKETS* 0.009, *APP_PACKETS* 0.009, dan *TCP_CONVERSATION_EXCHANGE* 0.008.



Gambar 4. Hasil *Feature Selection*

3.2 Performance Evaluation Independent Dataset Terhadap Malicious Websites

Tabel 2. Hasil Performance Evaluation Independent

Independent Dataset				
Algorithm	**Accuracy (%)**	**Recall (%)**	**Precision (%)**	**RMSE**
K-Nearest Neighbor	95.51	89.42	89.42	0.212
Decision Tree	89.33	57.82	83.7	0.309
Logistic Regression	92.42	76.63	84.57	0.24
Random Forest	87.92	50	43.96	0.325

Pada Tabel 2 memperlihatkan hasil *performance evaluation* dari algoritma klasifikasi tanpa menggunakan *cross validation* yaitu dengan *independent dataset*. Dari hasil tersebut dapat dilihat bahwa algoritma K-NN memiliki *accuracy*, *recall*, *precision*, dan RMSE tertinggi dibandingkan dengan algoritma lain dengan nilai *accuracy* sebesar 95.51%, *recall* sebesar 89.42%, *precision* sebesar 89.42% dan nilai *RMSE* 0.212. Berdasarkan Tabel 2 algoritma K-NN dibandingkan dengan algoritma *Random Forest* memiliki selisih sebesar 7.59% *accuracy*, 39.42% *recall*, 45.46% *precision*, dan 0.113 RMSE.

3.3 Performance Evaluation of 10-Fold Cross Validation Dataset Terhadap Malicious Websites

Tabel 3 menunjukkan hasil *performance evaluation* dari algoritma klasifikasi yang menggunakan 10-*fold cross validation*. Dari hasil tersebut diketahui bahwa algoritma K-NN memiliki *accuracy*, dan *recall* tertinggi dibandingkan dengan algoritma lain dengan nilai *accuracy* sebesar 93.61%, *recall* sebesar 85.05%, *precision* sebesar 85.25% dan nilai *RMSE* 0.251. Algoritma K-NN memiliki tingkat *precision* dan RMSE tertinggi ke dua dibandingkan dengan *Logistic Regression* yang memiliki selisih sebesar 0.71% *precision* dan 0.004 RMSE. Perbandingan algoritma *Random Forest* dengan K-NN setelah melakukan 10-*fold cross validation* memiliki selisih sebesar 5.75% *accuracy*, 35.05% *recall*, 41.31% *precision*, dan 0.069 RMSE.

Tabel 3. Hasil Performance Evaluation Cross Validation

Hasil 10 Folds Cross Validation				
Algorithm	**Accuracy (%)**	**Recall (%)**	**Precision (%)**	**RMSE**
K-Nearest Neighbor	93.61	85.05	85.25	0.251
Decision Tree	89.05	55.94	77.62	0.312
Logistic Regression	92.77	77.93	85.96	0.247
Random Forest	87.86	50	43.93	0.32

4. KESIMPULAN

Dari 4 algoritma yang telah di evaluasi yaitu *K-Nearest Neighbor*, *Decision Tree*, *Logistic Regression*, dan *Random Forest* dapat disimpulkan bahwa algoritma K-NN memiliki performa yang paling baik di antara algoritma lainnya dengan hasil 95.51% *accuracy*, 89.42% *recall*, 89.42% *precision*, dan 0.212 RMSE untuk hasil *independent* sedangkan untuk hasil 10 *fold cross validation* memiliki hasil 93.61% *accuracy*, 85.05% *recall*, 85.25% *precision*, dan 0.251 RMSE dalam mendeteksi *malicious* dan *benign website*.

5. SARAN

Untuk kedepannya diharapkan model ini dapat berguna untuk pembuatan aplikasi klasifikasi *malicious website*, dan untuk penelitian selanjutnya diharapkan peneliti dapat menggunakan metode dan algoritma lain agar dapat memaksimalkan *performance* dan mengurangi nilai error dalam pemodeling.

DAFTAR PUSTAKA

[1] J. Milan and P. Bajaj, "Techniques in Detection and Analyzing Malware Executebles: A Review," *International Journal of Computer Science and Mobile Computing*, vol. 13, no. 5, p. 930, 2014.

[2] A. Retno and L. A. Novarina, "Malware Dynamic," *Jurnal of Education and Information Communication Tecnology*, vol. 1, no. 1, p. 37, 2017.

[3] D. A. K. Dutta, "Detection of Malware and Malicious Executables Using E-Birch Algorithm,"

International Journal of Advanced Computer Science and Applications, vol. 7, no. 1, p. 124, 2016 .

[4] "Technopedia," [Online]. Available: https://www.techopedia.com/definition/6006/application-layer. [Accessed 7 May 2018].

[5] A. Altaher, "Phising Website Classification using Hybrid SVM and KNN Approach" *International Journal of Advanced Computer Science and Applications,* vol. 8, no.6, 2017.

[6] M. Aldwairi and R. Alsalman, "Malurls: A Lightweight Malicious Website," *Journal Of Emerging Technologies In Web Intelligence,* vol. 4, no. 2, 2012.

[7] "Kaggle," [Online]. Available: https://www.kaggle.com/xwolf12/malicious-and-benign-websites. [Accessed 12 April 2018].

[8] "OpenML," [Online]. Available: https://www.openml.org/a/estimation-procedures/1. [Accessed 20 April 2018].

[9] S. B. Imandoust and M. Bolandraftar, "Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background," *S B Imandoust et al. Int. Journal of Engineering Research and Applications,* vol. 3, no. 5, 2013.

[10] M. Bramer, "Principles of data mining," *Springer,* 2007.

[11] A. T. Liem, G. A. Sandag, I.-S. Hwang and A. Nikoukar, "Delay analysis of dynamic bandwidth allocation for triple-play-services in EPON," 2017.

[12] B. Sui, "Information Gain Feature Selection Based On Feature Interactions," 2013.

[13] C. Urcuqui, A. Navarro, J. Osorio and M. Garcia, "Machine Learning Classifiers to Detect Malicious Websites," *CEUR Workshop Proceedings,* vol. 1950, pp. 14-17, 2017.