JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Penerapan Word2Vec dan SVM dengan Hyperparameter Tuning untuk Deteksi Phishing

Hilman Singgih Wicaksana*, Khairul Huda

Fakultas Hukum, Manajemen, dan Informatika, Program Studi Informatika, Universitas Karya Husada, Semarang, Indonesia
Email: [1,*]singgih.hilman@gmail.com, [2]khairulhuda@unkaha.ac.id
Email Penulis Korespondensi: singgih.hilman@gmail.com
Submitted **11-06-2025;** Accepted **30-06-2025;** Published **30-06-2025**

Abstrak

Kemajuan teknologi informasi pada zaman digital sekarang ini berlangsung dengan sangat pesat dari satu waktu ke waktu yang lain. Fenomena tersebut diiringi dengan meningkatkan ancaman keamanan siber berupa phishing. Tautan phishing seringkali dirancang dengan struktur uniform resource locator (URL) yang tampak meyakinkan dan sulit dibedakan dari tautan asli. Penelitian ini mengusulkan pendekatan word-to-vector (Word2Vec) dan Support Vector Machine (SVM) dengan hyperparameter tuning dimana Word2Vec sebagai teknik word embedding yang digunakan untuk membuat representasi vektor kata dari URL tertentu, SVM digunakan sebagai pendekatan machine learning (ML) yang digunakan dalam penelitian ini, dan hyperparameter tuning digunakan sebagai teknik untuk mencari kombinasi parameter terbaik agar mampu menghasilkan model SVM yang optimal dalam mendeteksi phishing. Tujuan dari penelitian ini yaitu melakukan komparasi performansi antara model SVM dan XGBoost yang sudah dioptimasi dan melakukan deployment model ML ke dalam sistem prediksi dengan menggunakan framework Streamlit untuk melakukan deteksi phishing berdasarkan input yang dilakukan oleh pengguna berupa URL tertentu. Temuan dari studi ini mengindikasikan bahwa model SVM menunjukkan kinerja yang sangat baik daripada model XGBoost, dengan nilai precision, recall, f1-score, dan accuracy sekitar 99,84% untuk SVM. Di sisi lain, model XGBoost mencatat nilai precision, recall, f1-score, dan accuracy masing-masing sekitar 99,70%. Dengan demikian, model SVM menjadi model yang optimal untuk mendeteksi phishing dengan tepat dan akurat.

Kata Kunci: Hyperparameter Tuning; Machine learning; Phishing; Support Vector Machine; Word2Vec

Abstract

The advancement of information technology in today's digital age takes place very rapidly from one time to another. This phenomenon is accompanied by increasing cybersecurity threats like phishing. Phishing links are often designed with uniform resource locator (URL) structures that appear convincing and are difficult to distinguish from genuine links. This research proposes a word-to-vector (Word2Vec) and Support Vector Machine (SVM) approach with hyperparameter tuning where Word2Vec is a word embedding technique used to create a word vector representation of a particular URL, SVM is used as a machine learning (ML) approach used in this research, and hyperparameter tuning is used as a technique to find the best combination of parameters to produce an optimal SVM model in detecting phishing. The purpose of this research is to compare the performance between SVM and XGBoost models that have been optimized and deploy ML models into a prediction system using the Streamlit framework to detect phishing based on input made by users in the form of certain URLs. The findings of this study indicated that the SVM model performed very well compared to the XGBoost model, with precision, recall, f1-score, and accuracy values of about 99.84% for SVM. On the other hand, the XGBoost model recorded precision, recall, f1-score, and accuracy values of about 99.70% each. Thus, the SVM model is the optimal model to detect phishing precisely and accurately.

Keywords: Hyperparameter Tuning; Machine Learning; Phishing; Support Vector Machine; Word2Vec

1. PENDAHULUAN

Di era digital saat ini, pengembangan teknologi informasi memiliki dampak besar pada cara orang berinteraksi dan berurusan dengan orang-orang di dunia maya. Namun, kemajuan ini juga memiliki peningkatan ancaman terhadap keamanan siber, salah satunya adalah serangan *phishing* [1]. *Phishing* dilakukan oleh pelaku yang tidak bertanggung jawab dalam bentuk serangan siber dengan tujuan agar pengguna memberikan informasi sensitif seperti kata sandi, data kartu kredit, dan informasi pribadi lainnya melalui media tautan palsu yang memiliki kemiripan dengan tautan asli [2], [3]. Serangan ini sulit dikenali oleh pengguna awam, sehingga memerlukan perhatian khusus dalam upaya penanggulangannya.

Tautan *phishing* sering dirancang sedemikian rupa sehingga tampak meyakinkan dan sulit dibedakan dari tautan asli. Salah satu elemen utama dalam serangan *phishing* adalah *Uniform Resource Locator* (URL). Dalam URL terdapat serangkaian karakter yang mengikuti format tertentu dan digunakan untuk menunjukkan lokasi suatu sumber seperti gambar dan dokumen di dunia maya agar pengguna dapat mengaksesnya saat terhubung ke internet [4]. Pelaku seringkali mengubah bentuk URL dengan menambahkan tanda khusus, menggunakan subdomain yang mirip dengan domain asli atau menyisipkan parameter tertentu [5]. Dengan demikian, URL menjadi sangat penting dalam mendeteksi *phishing*, karena pola dan karakteristik tertentu dalam URL dapat menjadi tanda kemungkinan bahaya.

Untuk mengatasi permasalahan ini, pendekatan berbasis *machine learning* menawarkan solusi yang fleksibel dengan kemampuan mendeteksi pola tersembunyi dalam struktur URL. Dalam hal ini, representasi dari URL menjadi sangat penting untuk mengidentifikasi konteks dan hubungan makna antara bagian-bagian tersebut dengan menggunakan teknik *word embedding* [6]. Teknik *word embedding* yang digunakan dalam penelitian ini adalah Word2Vec. Teknik tersebut mampu mencari makna semantik dari kata dalam sekumpulan teks dan menggambarkan setiap kata yang berbeda dengan serangkaian angka yang dapat disebut sebagai vektor [7], [8]. Oleh karena itu, Word2Vec digunakan dalam penelitian ini sebagai teknik *word embedding* untuk merepresentasikan *token* URL dalam bentuk vektor.

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Penelitian yang dilakukan oleh [9] menggunakan model *Support Vector Machine* (SVM) untuk mendeteksi tautan *phishing*. Proses pelatihan model dalam penelitian tersebut menggunakan teknik *K-fold cross-validation* dengan *fold* sebanyak 3, 5, dan 10. Selain itu, penelitian tersebut menggunakan *grid search* sebagai teknik *hyperparameter tuning*. Parameter terbaik yang dihasilkan yaitu clf__C sebesar 10, clf_kernel dengan menggunakan linear, vect__max_features sebesar 1000, dan vect__ngram_range dengan (1, 1). Model tersebut menghasilkan performansi yang sangat baik dengan *accuracy* sebesar 95%, *precision* sebesar 94%, *recall* sebesar 95%, dan *f1-score* sebesar 94%, sehingga mampu memprediksi *phishing* dengan akurat. Namun, dalam penelitian tersebut belum dilakukan komparasi performansi model. Oleh karena itu, perbandingan kinerja model dilakukan dalam penelitian ini untuk mengukur seberapa baik model SVM memprediksi prediksi *phishing* dibandingkan dengan model lain.

Di samping itu, penelitian yang dilakukan oleh [10], menggunakan *dataset* dari Kaggle yang dibagi menjadi 80% data *training* dan 20% data *testing*. Penelitian tersebut menjelaskan bahwa model SVM memiliki performansi yang kurang baik dibandingkan dengan Random Forest. Pada model SVM memiliki tingkat *accuracy* sebesar 77,43%, *precision* sebesar 78,94%, *recall* sebesar 70,88%, dan *f1-score* sebesar 70,88%. Sedangkan, pada model Random Forest memiliki tingkat *accuracy* sebesar 84,95%, *precision* sebesar 83,71%, *recall* sebesar 84,39%, dan *f1-score* sebesar 84,05%. Performasi model tersebut dicapai dengan penerapan *K-fold cross validation* dengan 5 *fold*, sehingga dalam hal penelitian ini perlu ditingkatkan dalam jumlah *fold* yang digunakan pada saat pelatihan model dilakukan.

Penelitian yang dilakukan oleh [11] menunjukkan bahwa model XGBoost mencapai performansi klasifikasi tertinggi dengan *accuracy* sebesar 96,79% dibandingkan dengan SVM dengan *accuracy* mencapai 86,03% pada *dataset* uji pertama. Kemudian, pada *dataset* uji yang kedua, model XGBoost kembali mengungguli model SVM dengan masing-masing performansi klasifikasi sebesar 90,83% untuk model XGBoost dan 86% untuk model SVM. *Dataset* yang diuji dalam percobaan pertama dan kedua bersumber dari UCI. Dengan teknik *K-fold cross validation* menggunakan 10 *fold*, model XGBoost memiliki keunggulan dibandingkan model-model lainnya dalam memprediksi *phishing*. Performansi model tersebut dicapai tanpa penerapan teknik *hyperparameter tuning*. Oleh karena itu, pada studi ini membutuhkan *hyperparameter tuning* sebagai teknik untuk mengoptimalkan model *machine learning* untuk meningkatkan kinerja model dalam tugas klasifikasi.

Berdasarkan beberapa penelitian terdahulu, model SVM dan XGBoost sama-sama memiliki performansi yang sangat baik dalam memprediksi *phishing*. SVM dapat menjadi algoritma yang dapat menghasilkan solusi yang optimal dalam melakukan tugas klasifikasi. Model berbasis *kernel* ini juga dapat diterapkan untuk tugas regresi [12]. Selain itu, *Support Vector Machine* (SVM) menunjukkan kemampuan generalisasi yang baik, bahkan ketika jumlah data pelatihan terbatas atau ketika data berada dalam ruang *input* berdimensi tinggi, khususnya untuk kasus yang memiliki karakteristik *linear* [13]. Jika data yang digunakan bersifat *non-linear*, maka SVM perlu menggunakan fungsi *kernel* yang dimiliki oleh model tersebut [14]. Namun demikian, meskipun XGBoost sering menampilkan performa lebih unggul pada *dataset* besar dan kompleks, kajian yang membandingkan kedua algoritma ini secara sistematis dalam konteks deteksi *phishing* masih terbatas. Oleh karena itu, penelitian lanjutan diperlukan untuk mengidentifikasi kondisi spesifik yang memungkinkan salah satu model memberikan performa lebih konsisten dibandingkan yang lain.

Namun demikian, walaupun sudah banyak penelitian sebelumnya dilakukan, masih terdapat beberapa kekurangan yang perlu dikaji lebih mendalam. Penelitian sebelumnya cenderung menggunakan pendekatan ekstraksi fitur URL berbasis karakter atau kemunculan kata tanpa secara mendalam menggunakan teknik *word embedding* seperti Word2Vec, yang mampu menangkap hubungan semantik dan konteks lebih jelas antar komponen URL. Selain itu, sebagian besar penelitian terdahulu juga terbatas pada evaluasi performansi dari satu algoritma saja tanpa melakukan analisis perbandingan secara komprehensif terhadap beberapa model klasifikasi seperti SVM dan XGBoost. Kemudian, metode *hyperparameter tuning* yang digunakan dalam studi sebelumnya umumnya masih terbatas pada pendekatan konvensional seperti *grid search*, yang kurang efisien ketika menangani parameter dalam jumlah besar. Oleh sebab itu, penelitian ini bertujuan mengatasi keterbatasan tersebut dengan secara sistematis membandingkan kinerja model SVM dan XGBoost dengan Word2Vec sebagai teknik *word embedding* serta menerapkan *Bayesian Optimization* sebagai teknik *hyperparameter tuning* yang lebih efisien.

Penelitian ini mengusulkan pendekatan deteksi *phishing* dengan mengombinasikan teknik Word2Vec sebagai metode ekstraksi fitur semantik dan model SVM sebagai algoritma klasifikasi utama. Untuk mengoptimalkan performa model, penelitian ini menerapkan teknik *Bayesian Optimization* dalam proses *hyperparameter tuning*. Berbeda dengan *grid search* yang melakukan pencarian menyeluruh terhadap semua kombinasi parameter, *Bayesian Optimization* membangun model probabilistik dari fungsi objektif dan memilih kombinasi parameter yang diharapkan memberikan performa terbaik [15]. Pendekatan ini dinilai lebih efisien dan efektif dalam pencarian ruang parameter yang kompleks.

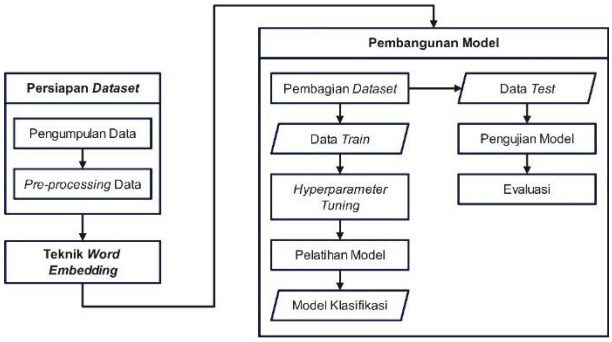
Dengan demikian, penelitian ini bertujuan untuk mengevaluasi efektivitas model SVM dan XGBoost dalam mendeteksi *phishing*, dengan memanfaatkan teknik *word embedding* Word2Vec serta proses *hyperparameter tuning* menggunakan metode *Bayesian Optimization*. Penelitian ini menggunakan *dataset* sebanyak 22.000 URL yang terdiri atas data *phishing* yang diperoleh dari situs PhishTank serta data *non-phishing* yang diambil dari Majestic. Dengan demikian, studi ini diharapkan mampu memberikan kontribusi yang signifikan dalam mendukung upaya pencegahan bagi pengguna dalam mengidentifikasi perbedaan antara tautan *phishing* dan *non-phishing*, sekaligus berperan sebagai acuan dalam penilaian kinerja model deteksi *phishing* di masa yang akan datang.

2. METODOLOGI PENELITIAN

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

2.1 Tahapan Penelitian

Tahapan yang dilakukan dalam penelitian ini meliputi persiapan *dataset*, teknik *word embedding*, dan pembangunan model. Pada saat tahapan persiapan *dataset* terdiri dari 2 bagian yaitu pengumpulan data yang dilakukan pada penelitian ini dan *pre-processing* data terhadap data yang telah dikumpulkan. Pada tahapan selanjutnya yaitu teknik *word embedding* yang digunakan dalam penelitian ini sebagai ekstraksi fitur yang menghasilkan representasi vektor kata. Setelah itu, dilanjutkan ke tahap pengembangan model yang terdiri dari pembagian *dataset* yang dibagi menjadi data *train* dan data *test*, *hyperparameter tuning*, pelatihan model, model klasifikasi, pengujian model, dan evaluasi. Alur penelitian secara detail dan keterkaitan antar tahapan tersebut dapat diilustrasikan secara jelas seperti yang terlihat pada Gambar 1.



Gambar 1. Tahapan Penelitian

2.1.1 Persiapan *Dataset*

Dalam studi ini, pengumpulan *dataset* dilakukan dengan menggunakan teknik *web scraping* menggunakan *library* dari BeautifulSoap. *Dataset* yang terdiri dari kumpulan *link phishing* dapat dilakukan *scraping* dari *website* PhishTank sebanyak 11.000 data dan kumpulan *link non-phishing* yang didapat melalui teknik *scraping* dari *website* Majestic sebanyak 11.000 data, sehingga total data yang digunakan dalam penelitian ini sebanyak 22.000 data. *Dataset* yang digunakan dalam penelitian ini dibagi menjadi dua bagian: 80% untuk data *training* dan 20% untuk data *testing*.

Selanjutnya, *dataset* tersebut dilakukan tahapan *pre-processing* yang meliputi melakukan penghapusan protokol seperti HTTP atau HTTPS, penghapusan terhadap *trailing slash* dan karakter "/" di *path*, *lowecasing*, hapus subdomain WWW, hapus angka atau IP *address*, dan hapus tanda baca seperti titik (.) dan *dash* (-) pada data URL. Tahapan-tahapan tersebut sangat penting untuk dilakukan dalam menganalisis data teks sehingga data dapat diproses oleh mesin yang kemudian diekstraksi ke dalam bentuk vektor [16]. Hasil *pre-processing* data yang dilakukan dalam penelitian ini dapat ditunjukkan pada Tabel 1.

Tabel 1. Hasil *Pre-processing* Data

Tahapan	Sebelum *Pre-processing*	Setelah *Pre-processing*
Penghapusan protokol HTTP/HTTPS	https://plannedparenthood.org/home	plannedparenthood.org/home
Penghapusan *trailing slash* dan karakter "/"	plannedparenthood.org/home	plannedparenthood.org home
Lowercasing	plannedparenthood.org home	plannedparenthood.org home
Hapus subdomain WWW	plannedparenthood.org home	plannedparenthood.org home
Hapus angka/IP Address	plannedparenthood.org home	plannedparenthood.org home
Hapus tanda baca tiitk (.) dan *dash* (-)	plannedparenthood org home	plannedparenthood org home

Berdasarkan Tabel 1 terdapat domain "https://plannedparenthood.org/home" yang dilakukan tahapan *pre-processing*. Penghapusan protokol HTTP/HTTPS dilakukan untuk membersihkan URL dari protokol HTTP/HTTPS agar dapat diproses pada tahapan selanjutnya. Tahapan penghapusan *traling slash* dan karakter "/" merupakan tahapan yang dilakukan untuk menghapus tanda garis miring "/" pada URL. Pada tahapan *lowercasing* digunakan jika masih ada huruf kapital, maka akan dikonversi menjadi karakter yang lebih kecil di URL. Setelah itu, tahapan-tahapan lainnya yaitu menghapus *subdomain* WWW, menghapus angka atau IP *address*, dan mnghapus tanda baca titik (.) dan *dash* (-) jika kondisi tersebut terdapat pada URL agar lebih mudah dalam teknik *word embedding* dan diproses oleh model *machine learning*. Jika URL sudah bersih, maka pada setiap tahapan tersebut akan menghasilkan *output* akhir dari seluruh tahapan *pre-processing* data menjadi "plannedparenthood org home". Dari hasil final *pre-processing* tersebut, maka selanjutnya dilakukan teknik *word embedding* untuk di-*train* agar dapat membentuk korpus teks dengan *unique tokens* pada setiap kata yang ada.

2.1.2 Teknik *Word Embedding*

Dalam penelitian ini diperlukan vektorisasi yang dapat mengubah kata per kata ke dalam bentuk vektor. Teknik vektorisasi kata yang digunakan pada penelitian ini sebagai *word embedding* adalah Word2Vec. Word2Vec digunakan sebagai representasi vektor kata dengan ukuran *embedding* yang sudah ditentukan [17]. Dalam model Word2Vec terdapat dua jenis arsitektur yaitu CBOW dan Skip-gram. Dalam penelitian ini, jenis arsitektur yang digunakan sebagai teknik

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

word embedding adalah Skip-gram. Model Word2Vec tersebut kemudian dilatih dengan *dataset* penelitian ini sebanyak 22.000 data.

Dalam implementasi pada model Word2Vec, terdapat beberapa variabel yang digunakan untuk membangun arsitektur Word2Vec antara lain *sentences*, *vector_size*, *window*, *min_count*, *workers*, dan *sg*. Dalam tahap pelatihan, model tersebut menggunakan data berupa *token* dari kalimat yang telah diproses sebelumnya dan disimpan dalam variabel *sentences*. Parameter *vector_size* ditetapkan sebesar 300 yang menunjukkan bahwa setiap kata akan diwakili dalam ruang vektor berdimensi 300, sehingga dapat menangkap berbagai hubungan semantik dan sintaksis antara kata dengan lebih mendalam. Parameter *window* diatur pada nilai 5 yang mengartikan bahwa konteks dari satu kata akan diambil berdasarkan 5 kata di kiri dan kanan kata tersebut sehingga model dapat memahami kata dalam konteks lokal yang lebih luas. Selanjutnya, parameter *min_count* diatur ke nilai 1 yang menunjukkan bahwa semua kata termasuk yang jarang muncul akan dimasukkan dalam proses pelatihan model. Hal tersebut penting apabila korpus pelatihan cukup kecil dan kata-kata yang jarang tetap dianggap penting untuk penerapan tertentu. Parameter *workers* ditetapkan dengan nilai 4 yang mengartikan bahwa pelatihan model dilakukan secara paralel dengan menggunakan 4 inti prosesor, mempercepat proses pelatihan, terutama jika korpus yang digunakan cukup besar. Parameter *sg* yang disetel dengan nilai 1 menunjukkan bahwa model akan menerapkan algoritma Skip-grapm dalam pelatihan, sebuah metode yang lebih efektif dalam menangkap hubungan semantik untuk kata-kata yang jarang muncul dibandingkan algoritma *continuous bag of words* (CBOW).

Arsitektur Skip-gram pada model Word2Vec telah menghasilkan ukuran dimensi vektor yaitu (10779, 300) dimana jumlah total *unique tokens* yang terdapat dalam korpus pelatihan dan memenuhi syarat *min_count* yang digunakan dalam penelitian ini dengan nilai 1. Sedangkan, dimensi dari setiap vektor kata yang digunakan saat membangun model dengan nilai 300. Dengan demikian, model Word2Vec telah membuat representasi vektor dengan 300 dimensi untuk masing-masing dari 10.779 *unique tokens*. Setiap baris dalam matriks ini merepresentasikan satu kata, sementara setiap kolom menunjukkan salah satu dimensi dari vektor kata tersebut. Matriks tersebut dapat disebut sebagai *embedding matrix* yang berfungsi untuk merepresentasikan kata-kata dalam bentuk numerik yang dapat diolah oleh algoritma *machine learning* sehingga memungkinkan untuk analisis semantik dan sintaksi yang lebih mendalam.

2.1.3 Pengembangan Model

Dalam pengembangan model yang dilakukan dalam penelitian ini dimulai dari tahapan pembagian *dataset*. Pembagian *dataset* dibagi menjadi data *train* dan data *testing* masing-masing sebesar 80% dan 20%. Pada saat pelatihan model menggunakan teknik *hyperparameter tuning* dengan menggunakan *Bayesian Optimization* dan dilakukan teknik *cross-validation* menggunakan teknik *K-fold cross-validation* dengan 10 *folds*. Teknik *Bayesian Optimization* mampu mencari kombinasi parameter terbaik agar model dapat menjadi lebih optimal. Sedangkan, teknik K-fold *cross-validation* digunakan dalam penelitian ini untuk memanfaatkan *dataset* secara maksimal, mengurangi variansi evaluasi, dan mencegah *overfitting* pada saat pelatihan model.

Penelitian ini mengembangkan 2 model yang berbeda yaitu SVM dan XGBoost. Masing-masing model menggunakan teknik *word embedding* dengan Word2Vec. Word2Vec tersebut menciptakan representasi vektor dari setiap kata dimana setiap kata memiliki *unique tokens* masing-masing. Hasil dari *word embedding* tersebut menjadi *input* dari masing-masing model. Selanjutnya, model akan memproses *input* berupa *word embedding* dengan menggunakan *hyperparameter tuning* pada saat pelatihan model berlangsung menggunakan teknik *Bayesian Optimization*. Parameter-parameter yang didaftarkan pada saat dilakukan *hyperparameter tuning* pada model SVM antara lain C, gamma, dan kernel, sedangkan model XGBoost dengan n_estimators, max_depth, dan learning_rate. Tabel 1 menjelaskan tentang nilai-nilai pada parameter yang digunakan pada setiap model.

Tabel 2. Nilai pada Hyperparameter Tuning

Model	Parameter	Nilai/Tipe
SVM	C	1, 10, 100
	gamma	0,0001, 0,001, 0,01
	kernel	linear, rbf
XGBoost	n_estimators	100, 200, 300
	max_depth	3, 6, 10
	learning_rate	0,001, 0,01, 0,1

Berdasarkan Tabel 2, masing-masing model memiliki parameter-parameter yang berbeda-beda. Parameter-parameter yang terdapat pada SVM yang digunakan dalam penelitian ini meliputi C, gamma, dan kernel dimana C terdiri dari nilai 1, 10, dan 100, gamma terdiri dari nilai 0,0001, 0,001, dan 0,01, dan kernel yang terdiri dari linear dan rbf. Sedangkan, parameter-parameter yang terdapat pada model XGBoost yang digunakan dalam penelitian ini meliputi n_estimators, max_depth, dan learning_rate dimana n_estimators terdiri dari nilai 100, 200, dan 300, max_depth terdiri dari nilai 3, 6, dan 10, dan learning_rate terdiri dari nilai 0,001, 0,01, dan 0,1.

Selama proses *training* dengan menggunakan teknik *Bayesian Optimization*, maka model akan mendapatkan kombinasi parameter terbaik untuk setiap model baik model SVM maupun XGBoost. Pada penelitian yang dilakukan oleh [15], teknik *hyperpamater tuning* tersebut mampu menghasilkan performansi model yang optimal dan menghasilkan prediksi yang akurat dari kombinasi parameter terbaik yang diperolehnya selama proses pelatihan model. Kombinasi

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

parameter terbaik yang diperoleh selama proses *hyperparameter tuning* dengan teknik *Bayesian Optimization* pada model SVM yaitu C dengan nilai 100, gamma dengan nilai 0,001, dan kernel dengan tipe linear. Sedangkan, pada model XGBoost telah mendapatkan kombinasi parameter terbaiknya yang meliputi learning_rate dengan nilai 0,1, max_depth dengan nilai 6, dan n_estimators dengan nilai 200. Dengan kombinasi parameter terbaik tersebut, maka dapat dilanjutkan ke tahap selanjutnya untuk mengetahui seberapa baik performansi model dalam mendeteksi *phishing* berdasarkan tautan berupa URL.

Setelah proses pelatihan selesai, maka model akan memberikan hasil prediksi yang terdiri dari dua kelas yaitu *phishing* dan *non-phishing* dimana jika terdapat URL yang menyerupai struktur dari situs aslinya tetapi memiliki struktur yang berbeda, maka dapat dipastikan bahwa model akan mendeteksi sebagai *phishing* dan sebaliknya. Model *machine learning* baik SVM dan XGBoost mampu memprediksi *phishing* karena model tersebut telah mempelajari *dataset* yang sudah dilatih dengan memperhatikan *hyperparameter tuning* dengan *Bayesian Optimization* sebagai teknik optimasi kinerja model. Dengan demikian, model-model tersebut mampu memprediksi tautan *phishing* berdasarkan URL tertentu.

Di samping itu, tahap selanjutnya yaitu evaluasi model dengan menggunakan *confusion matrix*. *Confusion matrix* merupakan suatu metode yang digunakan untuk mengukur performansi model klasifikasi dalam memprediksi *phishing* [18]. Metrik evaluasi model yang diterapkan dalam penelitian ini antara lain *precision*, *recall*, *f1-score*, dan *accuracy*. *Precision* digunakan untuk mengukur seberapa akurat sebuah model dalam memberikan prediksi yang tepat untuk kategori positif dari seluruh prediksi positif yang dilakukan [19]. *Recall* digunakan untuk menunjukkan seberapa efektif sebuah model dalam mengenali kelas positif dengan tepat [20]. Kemudian, metrik *f1-score* digunakan untuk mengukur keseimbangan antara *precision* dan *recall* pada model. Sedangkan, *accuracy* digunakan untuk mengukur seberapa akurat model dalam menghasilkan prediksi yang benar dibandingkan dengan semua prediksi yang dilakukan [21]. Untuk bisa mengukur performansi dari setiap metrik tersebut, maka diperlukan tabel yang selanjutnya dapat disebut sebagai *confusion matrix*, seperti yang dapat ditunjukkan pada Gambar 2.

Confusion matrix digunakan dalam studi ini sebagai metode untuk mengevaluasi performansi model pada tugas klasifikasi yang memberikan gambaran secara menyeluruh terkait dengan hasil prediksi model [22]. Pada setiap metrik evaluasi terdapat 4 bagian yang meliputi *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN) dimana bagian-bagian tersebut terdapat pada *confusion matrix*. Beberapa formula yang digunakan oleh setiap metrik evaluasi dapat dilihat pada Persamaan (1) sampai dengan Persamaan (4).

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \quad (3)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

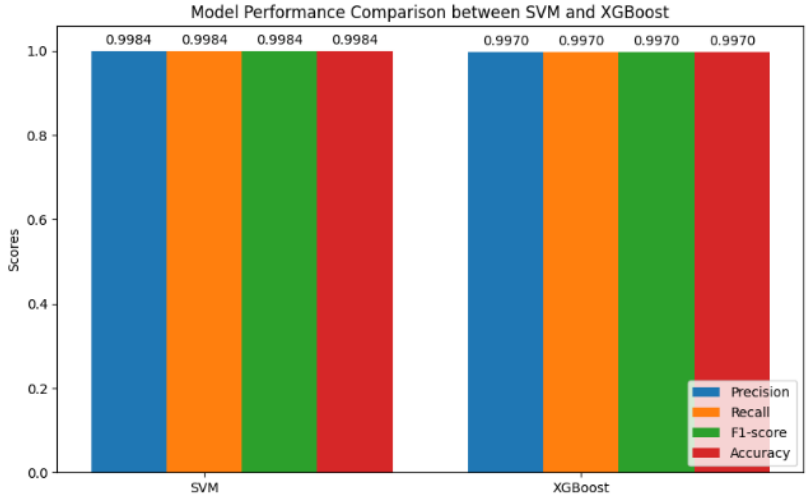
3. HASIL DAN PEMBAHASAN

Pada bagian ini akan dilakukan pembahasan menjadi 3 bagian yaitu evaluasi performansi model, pengujian model, dan perbandingan dengan penelitian sebelumnya. *Hyperparamer tuning* dilakukan untuk mencari kombinasi parameter terbaik secara otomatis selama proses pelatihan model. Evaluasi performansi model dilakukan untuk menunjukkan performansi dari masing-masing model yang dapat diukur baik dari tingkat *precision*, *recall*, *f1-score*, maupun *accuracy*. Pengujian model dilakukan untuk menguji seberapa baik model dalam memprediksi data dari suatu *link* yang diprediksi sebagai *phishing* atau *non-phishing*.

3.1 Evaluasi Performansi Model

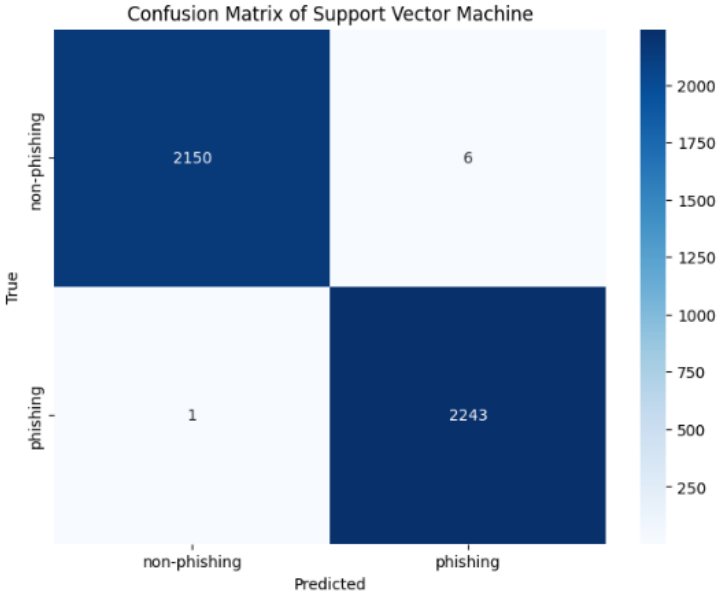
Pada tahapan ini dilakukan evaluasi model berdasarkan metrik evaluasi yang terdiri dari *precision*, *recall*, *f1-score*, dan *accuracy* pada setiap model. *Precision* mengukur seberapa banyak prediksi positif yang tepat dibandingkan dengan seluruh hasil yang diperkirakan sebagai positif. *Recall* menunjukkan proporsi kasus positif yang berhasil teridentifikasi dengan benar dari semua kasus positif yang ada. *F1-score* merupakan nilai rata-rata harmonis dari precision dan recall, dengan mempertimbangkan keseimbangan antara keduanya. Sementara itu, *accuracy* menunjukkan proporsi total prediksi yang benar, baik positif maupun negatif, terhadap seluruh jumlah data yang dianalisis. Gambar 2 menunjukkan perbandingan antara performansi model SVM dan XGBoost dari setiap metrik evaluasi.

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom



Gambar 2. Perbandingan Performansi Model

Berdasarkan Gambar 2, performansi model pada SVM dan XGBoost sama-sama memiliki performansi model yang baik dan memiliki selisih yang sangat sedikit pada tingkat *precision*, *recall*, *f1-score*, dan *accuracy*. Model SVM memiliki tingkat *precision* sebesar 0,9984336237873375 atau sekitar 99,84%, *recall* sebesar 0,998385717923533 atau sekitar 99,84%, *f1-score* sebesar 0,99840837985937 atau sekitar 99,84%, dan *accuracy* sebesar 0,9984090909090909 atau sekitar 99,84%. Berdasarkan metrik evaluasi tersebut, pada Gambar 3 menjelaskan penggunaan *confusion matrix* pada model SVM.

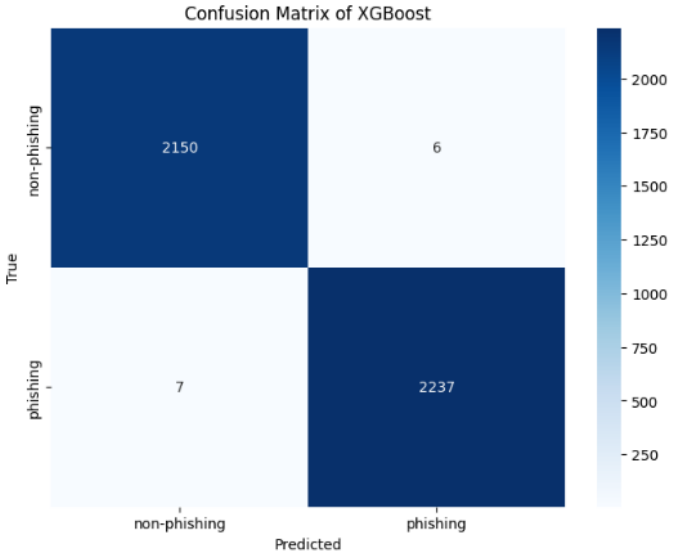


Gambar 3. *Confusion Matrix* pada Model SVM

Pada Gambar 3 menunjukkan *confusion matrix* untuk mengevaluasi performansi model SVM dalam mendeteksi *phishing*. Sampel data sebanyak 2.150 data telah diidentifikasi dengan tepat sebagai *non-phishing*, sementara hanya 6 sampel data yang terklasifikasi sebagai *phishing*. Selain itu, model SVM juga telah berhasil mendeteksi dengan tepat pada 2.243 sampel data sebagai *phishing* dan hanya 1 sampel data yang terklasifikasi salah sebagai *non-phishing*. Hal ini menunjukkan bahwa model SVM memiliki performansi yang sangat baik dalam mendeteksi tautan berupa URL antara *phishing* dan *non-phishing*. Dengan tingkat kesalahan klasifikasi yang sangat minim, model SVM tergolong sangat efisien untuk dapat digunakan dalam mendeteksi *phishing*.

Setelah melakukan evaluasi performansi model SVM sebagai model utama dalam penelitian ini, diperlukan juga evaluasi terhadap model XGBoost sebagai model pembanding. Pada model XGBoost memiliki performansi pada tingkat *precision* sebesar 0,997039881558058 atau sekitar 99,70%, *recall* sebesar 0,9970488195278111 atau sekitar 99,70%, *f1-score* sebesar 0,9970442989823862 atau sekitar 99,70%, dan *accuracy* sebesar 0.9970454545454546 atau sekitar 99,70%. Model XGBoost memiliki selisih 0,14% dibandingkan dengan model SVM pada tingkat *precision*, *recall*, *f1-score*, dan *accuracy*. Berdasarkan metrik evaluasi tersebut dapat disajikan *confusion matrix* untuk model XGBoost dapat ditunjukkan pada Gambar 4.

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom



Gambar 4. *Confusion Matrix* pada Model XGBoost

Pada Gambar 4 menunjukkan *confusion matrix* untuk model XGBoost dalam mendeteksi *phishing*. *Confusion matrix* tersebut menunjukkan bahwa jumlah data yang terklasifikasi benar untuk kelas *non-phishing* tetap sebanyak 6 dengan jumlah data yang terklasifikasi salah sebagai kelas *non-phishing* sebanyak 6, sama seperti pada model SVM. Namun, XGBoost memiliki kelemahan dalam mengklasifikasikan data *phishing* sebanyak 7 sampel data daripada hanya 1 kesalahan klasifikasi pada model SVM. Jumlah klasifikasi benar terhadap data *phishing* pada model XGBoost sedikit lebih rendah sebanyak 2.237 sampel data dibandingkan dengan sampel data sebnyak 2.243 pada model SVM. Oleh karena itu, walaupun kedua model menunjukkan performansi yang baik, model SVM memiliki tingkat ketepatan yang sedikit mengungguli model XGBoost dalam mendeteksi *phishing* dalam data tautan berupa URL.

Berdasarkan metrik evaluasi yang digunakan untuk mengukur performansi model baik SVM maupun XGBoost, model SVM memiliki nilai yang tinggi pada seluruh metrik yang digunakan baik *precision*, *recall*, *f1-score*, dan *accuracy*. Model SVM memiliki performansi yang lebih baik dibandingkan dengan model XGBoost. Hal tersebut dapat menunjukkan bahwa model SVM memiliki kemampuan dalam memprediksi dengan akurat untuk mengklasifikasikan data *phishing* dan *non-phishing* pada saat pengujian model dilakukan.

Dalam penelitian ini, penggunaan teknik *hyperparameter tuning* dengan pendekatan *Bayesian Optimization* berfungsi penting dalam meningkatkan kinerja model SVM. *Bayesian Optimization* memungkinkan pencarian angka terbaik untuk parameter-parameter penting seperti jenis kernel, C, dan gamma, dengan cara yang efisien dan teratur. Metode ini tidak hanya menghindari eksplorasi parameter yang tidak relevan, tetapi juga mempercepat proses menuju pengaturan terbaik yang memberikan hasil klasifikasi tertinggi. Oleh karena itu, ketepatan model dalam mendeteksi *phishing* meningkat dengan signifikan, yang terlihat dari rendahnya tingkat kesalahan pada matriks kebingungan yang dihasilkan.

Jika ditinjau dari studi-studi sebelumnya, penelitian yang dilakukan oleh [9] hanya menerapkan satu model dengan teknik *hyperparameter tuning* dengan menggunakan *grid search* untuk mendeteksi *phishing* yaitu model SVM. Di sisi lain, penelitian yang dilakukan oleh [10] menggunakan model Random Forest dan SVM, sedangkan penelitian yang dilakukan oleh [11] menggunakan model XGBoost dan Catboost. Penelitian [10] dan [11] umumnya menjelaskan tentang perbandingan performansi kedua model untuk mengidentifikasi *phishing* dan menggunakan pengaturan parameter standar tanpa adanya penggunaan teknik *hyperparameter tuning*. Dalam penelitian ini, terdapat pembaruan yang dilakukan melalui penerapan teknik *hyperparameter tuning* dengan memanfaatkan *Bayesian Optimization* pada dua model, yaitu model SVM dan XGBoost. Selain itu, dilakukan evaluasi kinerja model untuk mengidentifikasi model yang paling efektif dalam mendeteksi *phishing* dan mengimplementasikannya menggunakan *framework* Streamlit. Model yang paling optimal dalam penelitian ini dievaluasi berdasarkan metrik seperti *precision*, *recall*, *f1-score*, dan *accuracy* dimana model SVM terbukti sebagai model yang lebih unggul dibandingkan XGBoost.

3.2 Pengujian Model

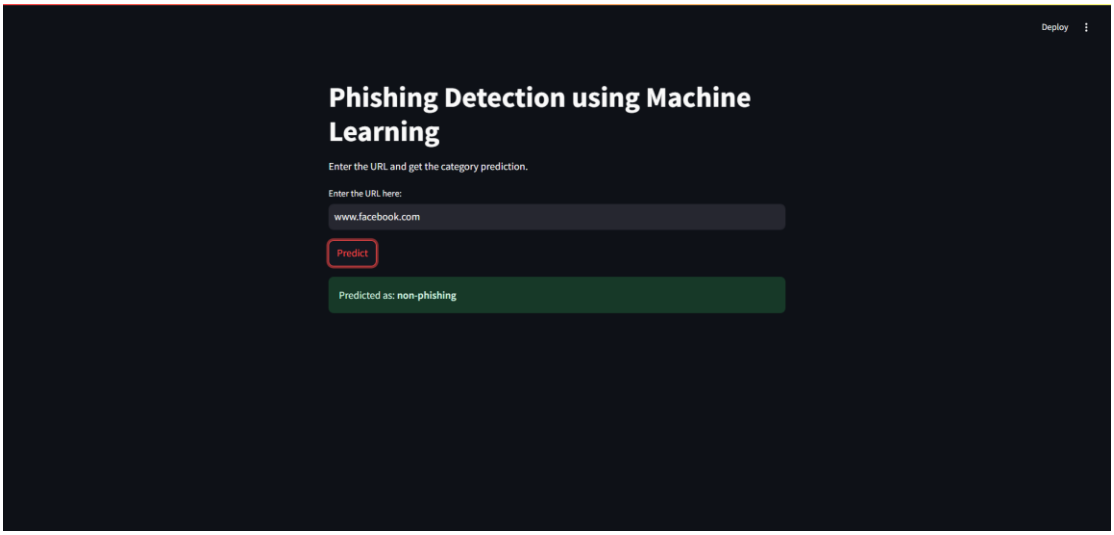
Pada tahapan ini dilakukan pengujian model menggunakan model terbaik yaitu SVM. Pengujian model ini dilakukan dengan menggunakan *framework* Streamlit dengan melakukan *deployment* model tersebut. Streamlit merupakan suatu *framework* dalam bahasa pemrograman Python yang dapat digunakan untuk melakukan *deployment* model *machine learning* dalam bentuk *website* [23]. Dalam penggunaan Streamlit perlu dilakukan instalasi beberapa *library* yang diperlukan terlebih dahulu seperti *library* untuk Streamlit itu sendiri, NLTK, Gensim, dan Sklearn. Setelah itu, lakukan *deployment* model terbaik dengan format file ".pkl", *word embedding* dengan format file ".model", dan *label encoder* dengan format ".pkl".

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Pada sistem prediksi berbasis Streamlit yang dibangun dalam penelitian ini terdiri dari *form input* dimana pengguna dapat memasukkan URL tertentu dan tombol *predict* yang dapat digunakan oleh pengguna sebagai aksi untuk memulai prediksi URL. Cara kerja sistem prediksinya yaitu ketika pengguna melakukan *input* URL pada *form* yang tersedia, lalu menekan tombol *predict*, maka sistem akan memproses *input* tersebut terlebih dahulu. Kemudian, di dalam sistem prediksi terdapat model SVM sebagai model terbaik yang mempelajari *input* dari pengguna dan mencocokkan dengan data URL yang pernah dilatih sebelumnya. Setelah sistem memproses *input* pengguna, maka sistem akan menampilkan *output* berupa prediksi *phishing* dan *non-phishing*. Sistem akan memberikan *output* prediksi berupa *phishing* ditandai dengan kotak panjang bertema *success*, sedangkan *output* prediksi berupa *non-phishing* ditandai dengan kotak panjang bertema *warning*.

Dalam sistem prediksi yang dibangun ke dalam penelitian ini dalam penelitian ini, model SVM bekerja dengan mengubah *input* URL yang ditentukan pengguna menjadi representasi numerik yang dapat dipahami dengan algoritma. Proses ini dimulai dengan melakukan pemecahan URL menjadi beberapa bagian yang lebih kecil yang dapat disebut sebagai *token* berdasarkan karakter pemisah seperti titik, garis miring, atau tanda hubung, yang menggambarkan struktur dasar dari sebuah URL pada umumnya [24]. *Token-token* ini kemudian dilakukan konversi sebagai vektor dalam ruang vektor yang telah dilatih sebelumnya. Selanjutnya, semua vektor *token* dalam satu URL dirata-rata untuk menghasilkan satu vektor yang mewakili seluruh URL. Vektor inilah yang berfungsi sebagai *input* untuk model SVM. Pada tahap *classifier*, SVM akan menempatkan vektor tersebut dalam ruang yang mewakili lebih banyak dimensi dimana model sudah membuat suatu *hyperplane* sebagai sebuah bidang pemisah yang idealnya memisahkan antara kategori *phishing* dan *non-phishing* berdasarkan data pelatihan yang ada [25]. Dengan menggunakan jarak maksimum antara data dari kedua kategori, model SVM dapat memprediksi dengan tepat terhadap URL yang baru. Prediksi dilakukan dengan menentukan dimana posisi vektor URL itu terhadap *hyperplane*. Jika vektor itu berada di sisi yang terkait dengan kategori *phishing*, maka sistem akan menunjukkan *output* berupa *phishing* dan sebaliknya jika berada di sisi *non-phishing*. Model ini membuat sistem prediksi mampu mengidentifikasi pola-pola tersembunyi yang terdapat pada struktur URL yang sering digunakan dalam serangan *phishing* maupun URL yang *non-phishing*, sehingga meningkatkan ketepatan prediksi dan keandalan sistem dalam membantu pengguna mendeteksi potensi ancaman keamanan siber.

Model SVM dapat dengan tepat mendeteksi *phishing* karena cara kerjanya yang meningkatkan jarak antara berbagai kategori di ruang fitur yang memiliki banyak dimensi [26]. Pada saat mendeteksi *phishing*, SVM sangat handal dalam mengenali pola-pola yang membedakan antara URL yang valid dan yang mencurigakan berdasarkan karakteristik teks atau struktur dari URL tersebut. Misalnya, model dapat mengetahui bahwa URL www.facebook.com merupakan domain resmi dan sah dengan struktur domain yang biasa, tanpa adanya subdomain yang mencurigakan atau ekstensi yang tidak familiar. Selama proses pelatihan, SVM belajar dari banyak contoh URL yang terkategori sebagai *phishing* dan yang *non-phishing*, dan mampu membuat *hyperplane* yang memisahkan keduanya dengan optimal berdasarkan fitur-fitur seperti panjang domain, penggunaan simbol, jenis ekstensi domain, dan adanya kata kunci tertentu. Hal ini dapat ditunjukkan seperti halnya pada Gambar 5.

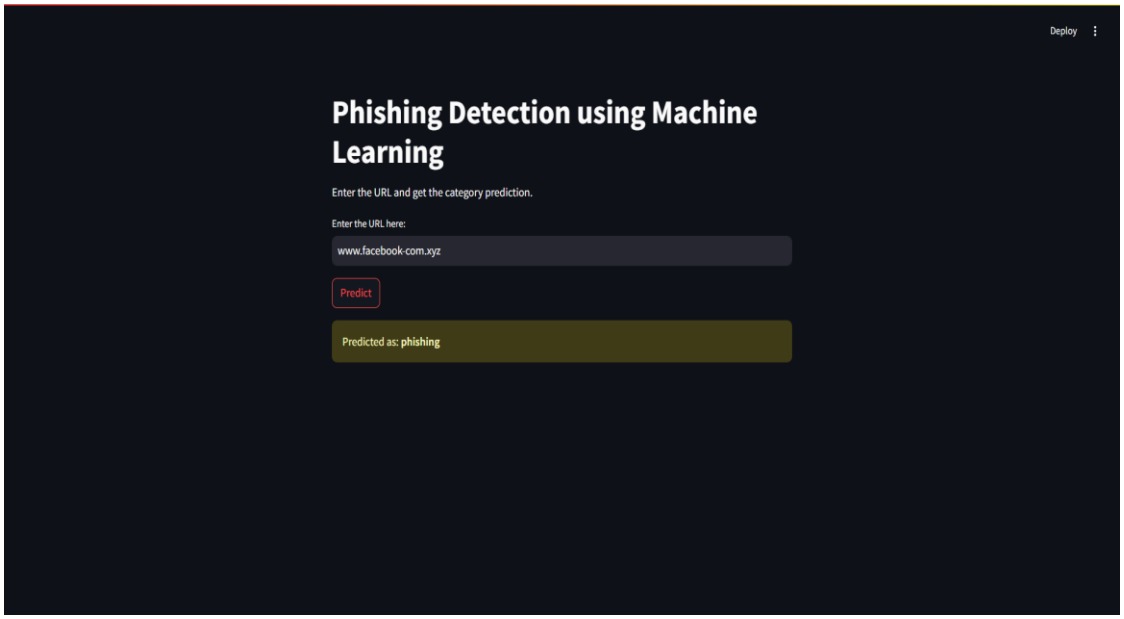


Gambar 5. Hasil Prediksi Kelas *Non-phishing*

Sementara itu, URL seperti www.facebook-com.xyz dianggap sebagai *phishing* karena menunjukkan beberapa tanda mencurigakan yang telah dipelajari oleh model sebagai indikator ancaman bagi pengguna internet. Nama domain tersebut memiliki struktur yang sama dengan domain resmi dengan menambahkan kata "facebook", tetapi menggunakan struktur yang aneh dan ekstensi .xyz, yang sering dipakai dalam domain *phishing* karena biayanya rendah dan tidak terverifikasi. Ciri-ciri seperti adanya tanda penghubung, penggunaan kata kunci terkenal yang tidak sesuai dengan domain resmi, dan *top-level domain* (TLD) yang langka, semuanya merupakan indikator yang kuat bagi model SVM untuk mengenali URL tersebut sebagai *phishing*. Dengan pemisahan kelas yang jelas dan kemampuannya untuk mengolah data

JURIKOM (Jurnal Riset Komputer), Vol. 12 No. 3, Juni 2025
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v12i3.8729
Hal 361-371
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

non-linear melalui kernel, SVM dapat dengan tepat membedakan antara URL yang asli dan yang berbahaya seperti dalam contoh ini. Hal ini dapat ditunjukkan seperti halnya pada Gambar 6.



Gambar 6. Hasil Prediksi Kelas *Phishing*

3.3 Perbandingan dengan Penelitian Terdahulu

Pada studi ini memiliki beberapa perbedaan dari penelitian sebelumnya. Studi yang dilakukan oleh [9] hanya menggunakan satu model yaitu model SVM, tidak melakukan komparasi performansi model antara model satu dengan model yang lain, dan *hyperparameter tuning* yang digunakan masih menggunakan teknik *grid search*, sedangkan dalam penelitian ini dilakukan pembuatan dua model sekaligus untuk dilakukan komparasi performansi model antara SVM dan XGBoost, serta *hyperparameter tuning* yang digunakan yaitu teknik *Bayesian Optimization* sebagai teknik optimasi model. Teknik optimasi memiliki keunggulan dalam mencari kombinasi parameter terbaik yang optimal dengan evaluasi yang lebih sedikit dibandingkan dengan *grid search*.

Kemudian, penelitian yang dilakukan oleh [10], model SVM memiliki kelemahan dalam memprediksi *phishing* daripada model pembandingnya, karena pada saat pelatihan model hanya menggunakan 5 *folds* dengan teknik *K-fold cross-validation*, sedangkan dalam penelitian ini SVM mampu menggungguli model pembandingnya dengan menggunakan 10 *folds*. Dengan demikian, penggunaan 10 *folds* sangat direkomendasikan untuk bisa memberikan agar model mampu mendapatkan estimasi performansi model yang lebih akurat, variansi hasil evaluasi yang lebih rendah dibandingkan hanya melakukan pembagian dataset antara data *train* dan data *test*, dan mampu menggeneralisasi pada data baru.

Selain itu, penelitian yang dilakukan oleh [11], model XGBoost menjadi model yang paling unggul dibandingkan dengan model lain tanpa *hyperparameter tuning* pada *dataset* uji yang berbeda. Dalam penelitian ini, *hyperparameter tuning* digunakan sebagai teknik optimasi model untuk meningkatkan performansi dan kinerja model dalam memprediksi *phishing*. Pentingnya *hyperparameter tuning* ini dapat menentukan kombinasi parameter yang tepat, sehingga tidak perlu dilakukan penyetelan parameter secara manual secara satu per satu. Dengan demikian, teknik *hyperparameter tuning* ini diterapkan dalam penelitian ini untuk model-model yang dibangun dalam penelitian ini.

Penelitian ini memberikan beberapa temuan bahwa teknik *cross-validation* yang diterapkan selama proses pelatihan model dengan 10 *folds* disertai dengan *hyperparameter tuning* dengan teknik *Bayesian Optimization* membuat model memiliki performansi yang sangat baik pada masing-masing model sehingga mampu menghasilkan tingkat tingkat *precision, recall, f1-score*, dan *accuracy* yang memiliki nilai persentase yang optimal. Meskipun demikian, model SVM memiliki performansi model dengan selisih yang sedikit dibandingkan dengan model XGBoost sekitar 0,14% pada tingkat *precision, recall, f1-score*, dan *accuracy*. Hal tersebut memberikan penemuan bahwa model SVM memiliki keunggulan dibandingkan dengan model XGBoost jika diterapkan teknik *hyperparameter tuning* dengan *Bayesian Optimization* dan memperhatikan penggunaan *K-fold cross-validation* dengan 10 *folds* pada saat proses pelatihan model agar performansi model menjadi lebih optimal dan memiliki kemampuan dalam menggeneralisasi data yang lebih baik.

4. KESIMPULAN

Penelitian ini memberikan kontribusi dalam penerapan teknik *hyperparameter tuning* menggunakan *Bayesian Optimization* dan penggunaan *K-fold cross-validation* dengan 10 *folds* sehingga mampu menghasilkan model yang memiliki performansi yang optimal dan mampu menggeneralisasi data dengan sangat baik. Selain itu, penelitian ini juga

menujukkan bahwa model SVM telah mengungguli model XGBoost dengan tingkat *precision*, *recall*, *f1-score*, dan *accuracy* masing-masing sebesar 99,84%. Kombinasi parameter terbaik yang digunakan oleh model SVM selama proses pelatihan dengan menggunakan teknik *Bayesian Optimization* yaitu parameter C dengan nilai sebesar 100, gamma sebesar 0,001, dan kernel dengan tipe *linear*. Dengan demikian, model SVM dapat menjadi model terbaik dalam penelitian ini dan dapat digunakan untuk mendeteksi *phishing* berdasarkan URL tertentu sebagai upaya pencegahan dini untuk dapat lebih memperhatikan keamanan siber di masa mendatang.

REFERENCES

[1] I. Yurita, M. K. Ramadhan, and M. Candra, "Pengaruh Kemajuan Teknologi Terhadap Perkembangan Tindak Pidana Cybercrime," Jurnal Hukum, Legalita, vol. 5, no. 2, pp. 143–155, 2023.
[2] V. Aprelia Windarni, A. Ferdita Nugraha, S. Tri Atmaja Ramadhani, D. Anisa Istiqomah, F. Mahananing Puri, and A. Setiawan, "Deteksi Website Phishing Menggunakan Teknik Filter pada Model Machine Learning," Information System Journal (INFOS), vol. 6, no. 1, pp. 39–43, 2023.
[3] J. Pande, Introduction to Cyber Security. Uttarakhand Open University, 2017. [Online]. Available: http://uou.ac.in
[4] D. Prayama, Yuhefizar, and Amelia Yolanda, "Protokol HTTPS, Apakah Benar-benar Aman?," Journal of Applied Computer Science and Technology, vol. 2, no. 1, pp. 7–11, Jun. 2021, doi: 10.52158/jacost.v2i1.118.
[5] A. D. Harahap, D. Juardi, and A. S. Y. Irawan, "Rancang Bangun Sistem Pendeteksi Link Phishing Menggunakan Algoritma Random Forest Berbasis Web," Jurnal Informatika dan Teknik Elektro Terapan (JITET), vol. 12, no. 3, pp. 2677–2686, Aug. 2024, doi: 10.23960/jitet.v12i3.4858.
[6] N. Stevanovi´c, "Character and Word Embeddings for Phishing Email Detection," Computing and Informatics, vol. 41, no. 5, pp. 1337–1357, 2022, doi: 10.31577/cai.
[7] S. Khomsah, "Sentiment Analysis On YouTube Comments Using Word2Vec and Random Forest Sentimen Analisis pada Opini YouTube Menggunakan Word2Vec dan Random Forest," Jurnal Informatika dan Teknologi Informasi, vol. 18, no. 1, pp. 61–72, 2021, doi: 10.31515/telematika.v18i1.4493.
[8] M. T. Pilehvar and J. Camacho-Collados, Embeddings in Natural Language Processing Theory and Advances in Vector Representation of Meaning. 2021. doi: https://doi.org/10.1007/978-3-031-02177-0.
[9] M. Vebriani and W. Yustanti, "Klasifikasi Deteksi Link Phising DANA Kaget Menggunakan Metode Support Vector Machine Berbasis Website," Journal of Informatics and Computer Science, vol. 06, 2024, [Online]. Available: https://danakagetvezridd.
[10] F. F. Tampinongkol, A. R. Kamila, A. C. Wardhana, A. W. C. Kusuma, and D. Revaldo, "Implementation of Random Forest Classification and Support Vector Machine Algorithms for Phishing Link Detection," Journal of Informatics Information System Software Engineering and Applications (INISTA), vol. 7, no. 1, pp. 127–137, Dec. 2024, doi: 10.20895/inista.v7i1.1588.
[11] K. Sadaf, "Phishing Website Detection using XGBoost and Catboost Classifiers," in 2023 International Conference on Smart Computing and Application (ICSCA), IEEE, Feb. 2023, pp. 1–6. doi: 10.1109/ICSCA57840.2023.10087829.
[12] F. R. Lumbanraja et al., "Implementasi Support Vector Machine dalam Memprediksi Harga Rumah pada Perumahan di Kota Bandar Lampung," Jurnal Pepadun, vol. 2, no. 3, pp. 327–335, 2021.
[13] B. Filemon, V. C. Mawardi, and N. J. Perdana, "Penggunaan Metode Support Vector Machine untuk Klasifikasi Sentimen E-Wallet," Jurnal Ilmu Komputer dan Sistem Informasi, vol. 10, no. 1, pp. 1–6, Mar. 2022, doi: 10.24912/jiksi.v10i1.17824.
[14] T. Meisya Permata Aulia, N. Arifin, and R. Mayasari, "Perbandingan Kernel Support Vector Machine (SVM) dalam Penerapan Analisis Sentimen Vaksinisasi Covid-19," Science and Information Technology (SINTECH), vol. 4, no. 2, pp. 139–145, 2021, [Online]. Available: https://doi.org/10.31598
[15] H. S. Wicaksana, R. Kusumaningrum, and R. Gernowo, "Determining community happiness index with transformers and attention-based deep learning," IAES International Journal of Artificial Intelligence (IJ-AI), vol. 13, no. 2, p. 1753, Jun. 2024, doi: 10.11591/ijai.v13.i2.pp1753-1761.
[16] B. Hakim, "Analisa Sentimen Data Text Preprocessing Pada Data Mining Dengan Menggunakan Machine Learning," JBASE - Journal of Business and Audit Information Systems, vol. 4, no. 2, Aug. 2021, doi: 10.30813/jbase.v4i2.3000.
[17] P. Ayuningtyas and H. Tantyoko, "Perbandingan Metode Word2vec Model Skipgram pada Ulasan Aplikasi Linkaja menggunakan Algoritma Bidirectional LSTM dan Support Vector Machine," Jurnal Sistem dan Teknologi Informasi (JustIN), vol. 12, no. 1, p. 189, Jan. 2024, doi: 10.26418/justin.v12i1.72530.
[18] E. Septiana Pane and C. Caroline, "Optimalisasi Evaluasi Pelaksanaan Pelatihan Melalui Analisis Sentimen Otomatis Dengan Model Text Classification," in Prosiding PITNAS Widyaiswara , 2024, pp. 141–154.
[19] E. Andreas and W. Widhiarso, "Klasifikasi Penyakit Mata Katarak Menggunakan Convolutional Neural Network dengan Arsitektur Inception V3," in The 2nd MDP Student Conference 2023, 2023, pp. 107–113. [Online]. Available: https://www.kaggle.com/jr2ngb/cataractdataset
[20] Y. A. Singgalen, "Analisis Performa Algoritma NBC, DT, SVM dalam Klasifikasi Data Ulasan Pengunjung Candi Borobudur Berbasis CRISP-DM," Building of Informatics, Technology and Science (BITS), vol. 4, no. 3, Dec. 2022, doi: 10.47065/bits.v4i3.2766.
[21] N. B. Putri and A. W. Wijayanto, "Analisis Komparasi Algoritma Klasifikasi Data Mining Dalam Klasifikasi Website Phishing," Komputika : Jurnal Sistem Komputer, vol. 11, no. 1, pp. 59–66, Jan. 2022, doi: 10.34010/komputika.v11i1.4350.
[22] L. Palupi, E. Ihsanto, and F. Nugroho, "Analisis Validasi dan Evaluasi Model Deteksi Objek Varian Jahe Menggunakan Algoritma Yolov5," Journal of Information System Research (JOSH), vol. 5, no. 1, pp. 234–241, Oct. 2023, doi: 10.47065/josh.v5i1.4380.
[23] A. Putranto, N. L. Azizah, and I. R. I. Astutik, "Sistem Prediksi Penyakit Jantung Berbasis Web Menggunakan Metode SVM dan Framework Streamlit," KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen), vol. 4, no. 2, pp. 442–452, 2023, [Online]. Available: https://archive.ics.uci.edu/ml/datasets/heart+disease
[24] A. F. Mahmud and S. Wirawan, "Deteksi Phishing Website menggunakan Machine Learning Metode Klasifikasi," Sistemasi: Jurnal Sistem Informasi, vol. 13, no. 4, pp. 1368–1380, 2024, [Online]. Available: http://sistemasi.ftik.unisi.ac.id

[25] E. S. Shombot, G. Dusserre, R. Bestak, and N. B. Ahmed, "An application for predicting phishing attacks: A case of implementing a support vector machine learning model," Cyber Security and Applications, vol. 2, Jan. 2024, doi: 10.1016/j.csa.2024.100036.
[26] H. S. Wafa, A. I. Hadiana, and F. R. Umbara, "Prediksi Penyakit Diabetes Menggunakan Algoritma Support Vector Machine (SVM)," Informatics and Digital Expert (INDEX), vol. 4, no. 1, pp. 40–45, 2022, [Online]. Available: https://e-journal.unper.ac.id/index.php/informatics