Optimasi Model Support Vector Machine dengan Particle Swarm Optimization untuk Mendeteksi Serangan Injeksi SQL (Studi Kasus : PT. Naisha Inspirasi Muslimah).

Adam Arnap*[1] dan Kusrini[2]

1,2 Program Magister Teknik Informatika Program Pascasarjana Universitas Amikom Yogyakarta
Jl. Padjajaran, Ring Road Utara, Kel. Condongcatur, Kec. Depok,Kab.Sleman, Prop. Daerah Istimewa Yogyakarta 55283
adam.arnap.s2@students.amikom.ac.id;kusrini@amikom.ac.id

Abstrak

PT. Naisha Inspirasi Muslimah merupakan perusahaan yang mengoperasikan toko online untuk penjualan produk tekstil dan pernah mengalami kehilangan data akibat serangan injeksi SQL. Untuk meningkatkan manajemen keamanan komputer, PT. Naisha melakukan tata kelola teknologi informasi dengan menggunakan ISACA Design Toolkit COBIT 2019. Penelitian ini melakukan studi literatur mengenai deteksi injeksi SQL dan menemukan bahwa beberapa penelitian sebelumnya menggunakan model *Support Vector Machine* (SVM) yang dioptimasi dengan *Particle Swarm Optimization* (PSO). Pada penelitian terdahulu, optimasi model SVM dengan PSO dapat meningkatkan nilai performa model. Penelitian ini menggunakan model SVM yang dioptimasi dengan PSO untuk menangani dataset berupa kueri-kueri SQL. Proses vektorisasi dengan *Term Frequency - Inverse Document Frequency* (TF-IDF) juga diterapkan untuk memberikan bobot pada setiap token. Hasil penelitian menunjukkan bahwa nilai optimal yang ditemukan adalah *Best C* sebesar 593,0497396215296 dan *Best gamma* sebesar 0,07795813722739078. Dengan parameter tersebut, model SVM+PSO berhasil mencapai akurasi sebesar 99,66%, precision sebesar 99,66%, recall sebesar 99,66% dan F1 Score sebesar 99,66%, yang secara signifikan lebih tinggi dibandingkan dengan model SVM biasa yang hanya mencapai akurasi sebesar 79,27%, precision sebesar 85,34%, recall sebesar 79,27% dan F1 Score 78,34%. Model SVM+PSO juga menunjukkan performa lebih baik dibandingkan model machine learning lainnya seperti *Decision Tree* dan *Logistic Regression*. Hasil ini menunjukkan bahwa optimasi SVM dengan PSO secara substansial meningkatkan kinerja model SVM dalam mendeteksi injeksi SQL, sehingga dapat menjadi solusi yang efektif untuk meningkatkan keamanan sistem informasi di PT. Naisha Inspirasi Muslimah.

Kata Kunci injeksi SQL, support vector machine, particle swarm optimization, keamanan komputer, machine learning

Digital Object Identifier 10.36802/jnanaloka.2024.v5-no2-75-86

1 Pendahuluan

PT. Naisha Inspirasi Muslimah (PT. Naisha) merupakan perusahaan yang bergerak di sektor penjualan produk tekstil. Penjualan dilakukan baik secara offline maupun *online*. Untuk menjangkau konsumen di luar daerah yang tidak dapat dijangkau oleh penjualan *offline*, PT.

* Corresponding author.

Jnanaloka
Jurnal Open Access
Yayasan Lentera Dua Indonesia

Naisha memiliki situs web toko online. Situs ini memungkinkan konsumen dari berbagai lokasi untuk membeli produk Naisha dengan mudah, sehingga memperluas jangkauan pasar dan meningkatkan aksesibilitas bagi pelanggan di seluruh wilayah.

Seiring dengan berkembangnya kebutuhan dan tantangan di pasar, PT. Naisha terus berinovasi dengan memanfaatkan teknologi informasi untuk meningkatkan operasional dan layanan. Dengan integrasi teknologi informasi, perusahaan dapat lebih efektif dalam mengelola data penjualan, mengoptimalkan rantai pasokan, serta memberikan layanan pelanggan yang lebih responsif. Teknologi ini juga memungkinkan PT. Naisha untuk mengumpulkan dan menganalisis data yang relevan, sehingga dapat mengambil keputusan yang lebih tepat dan strategis.

Tata kelola teknologi informasi yang tidak direncanakan secara sistematis akan menyebabkan perusahaan kehilangan skala prioritas, sehingga penerapan teknologi informasi tidak akan selaras dengan tujuan perusahaan [1]. Melalui penerapan teknologi informasi, diharapkan berbagai manfaat dapat dirasakan oleh para pemangku kepentingan, baik dalam proses pengambilan keputusan yang lebih akurat, pemenuhan kebutuhan sumber daya manusia yang lebih efisien, maupun dalam transformasi model bisnis yang lebih adaptif dan inovatif [2].

Oleh karena itu penting dilakukannya analisis tata kelola teknologi informasi pada PT. Naisha. Analisis tata kelola teknologi informasi yang dilakukan memanfaatkan kerangka kerja COBIT 2019. COBIT 2019 dikenal sebagai kerangka kerja yang sangat fleksibel dan dianggap sebagai pendekatan yang paling komprehensif dalam mengelola teknologi informasi [3]. COBIT 2019 telah menjadi standar kerangka kerja yang banyak digunakan untuk mengimplementasikan tata kelola teknologi informasi di berbagai perusahaan dan lembaga pemerintah [3].

Berdasarkan pengalaman yang dialami oleh PT. Naisha, analisis tata kelola teknologi dengan menggunakan kerangka kerja COBIT 2019 dapat dilakukan lebih efektif. Proses audit difokuskan pada beberapa poin kunci yang dipetik dari pengalaman langsung PT. Naisha. Salah satu pengalaman signifikan yang dialami adalah terkait keamanan data di toko *online* PT. Naisha, di mana terjadi serangan Injeksi SQL yang menyebabkan kehilangan data pada database. Kejadian ini memiliki dampak serius baik bagi PT. Naisha sendiri maupun bagi konsumen yang menggunakan *platform* toko *online* Naisha untuk bertransaksi produk. Kejadian tersebut menyoroti pentingnya implementasi standar keamanan informasi seperti yang diatur dalam COBIT 2019, untuk mengurangi risiko keamanan dan melindungi kepentingan perusahaan serta kepercayaan pelanggan.

Di antara berbagai ancaman keamanan terhadap aplikasi web, serangan injeksi SQL (SQLIA) telah menjadi ancaman utama selama 15 tahun terakhir [4]. Tidak hanya pada PT. Naisha saja, serangan injeksi SQL telah banyak terjadi pada kasus-kasus lain. Serangan siber mengakibatkan kerugian ekonomi rata-rata hampir $ 50 miliar per tahun, di mana lebih dari seperlima disebabkan oleh injeksi SQL [5]. Menurut sebuah studi terhadap 300.000 serangan di seluruh dunia dalam satu bulan tertentu, 24,6% adalah injeksi SQL [6]. Setelah serangan SQL skala kecil, biaya pemeliharaan sistem rata-rata mencapai lebih dari $ 196.000 (lebih dari 1,2 juta yuan). Fakta ini merujuk pada laporan *Global Threat Intelligence* 2014 oleh NTT Corporation [7].

Untuk menentukan tujuan dari proses yang akan dievaluasi sebagai proses unggulan demi keuntungan perusahaan, digunakan *toolkit* Desain (ISACA COBIT 2019) [8]. Dengan mengatur beberapa parameter dari design factor 1 hingga design factor 4, dapat diidentifikasi 3 domain utama dengan nilai tertinggi. Ketiga domain tersebut adalah APO04 *Managed Innovation*, APO12 *Managed Risk*, dan APO13 *Managed Security*. Berdasarkan

pengalaman PT. Naisha terhadap serangan Injeksi SQL, APO13 merupakan salah satu dari ketiga domain ini yang memiliki nilai tinggi dalam penelitian ini, karena beberapa parameter pada *design factor* difokuskan pada bagian keamanan komputer. Serangan injeksi SQL yang berhasil dapat menyebabkan konsekuensi serius bagi organisasi yang menjadi korban, seperti kerugian finansial, kehilangan reputasi, serta pelanggaran terhadap kepatuhan dan regulasi [9]. Oleh karena itu, penerapan strategi perlindungan terhadap serangan aplikasi web, termasuk injeksi SQL, adalah tugas keamanan yang esensial dan vital untuk menjaga privasi pengguna serta data perusahaan [5].

Beberapa model *machine learning* dan *deep learning* telah diterapkan pada penelitian terdahulu untuk mendeteksi serangan injeksi SQL. *Model Hidden Markov Model* (HMM) digunakan untuk deteksi serangan Injeksi SQL, yang menghasilkan nilai akurasi sebesar 99,53% [10]. M*odel Deep Learning* yakni *Multi Layer Perceptron* (MLP) digunakan sebagai model untuk mendeteksi serangan injeksi SQL yang menghasilkan nilai akurasi sebesar 99% [7]. Model *Recurrent Neural Network* (RNN) juga telah digunakan untuk mendeteksi serangan injeksi SQL, model RNN mendapatkan hasil akurasi sebesar 94% [11]. Model SQLNN Deep Neural Network model digunakan untuk mendeteksi serangan injeksi SQL [12].

Pelatihan model jaringan saraf dalam menggunakan fungsi ReLU dan metode Dropout model dilatih dengan data, dan hasilnya dibandingkan dengan metode pembelajaran mesin tradisional dan algoritma LSTM, dan model ini menghasilkan nilai akurasi diatas 96% [12]. Model *Probabilistic Neural Networks* (PNN) digunakan untuk deteksi serangan injeksi SQL [13]. Peningkatan kinerja pendeteksian serangan SQL injection melalui penggunaan algoritma optimasi BAT untuk mengoptimalkan parameter smoothing dalam PNN, model yang diusulkan menghasilkan nilai akurasi sebesar 99,19% [13]. Model Naïve Bayes dengan optimasi menggunakan Particle Swarm Optimization (PSO) untuk memproses data teks pada keperluan analisis sentimen menghasilkan akurasi 89,16% [14]. Model SVM dengan optimasi PSO digunakan untuk mengolah data teks pada data pendonor darah, model yang diusulkan menghasilkan nilai akurasi sebesar 90% [15]. Penggunaan model SVM juga digunakan untuk mendeteksi data opini film dengan optimasi PSO, model yang diusulkan menghasilkan nilai akurasi terbesar yakni 87,84% [16]. Model yang sama yakni SVM digunakan untuk analisis sentimen menghasilkan nilai akurasi sebesar 79,06%, setelah diterapkannya PSO nilai akurasi meningkat menjadi 81,15% [17].

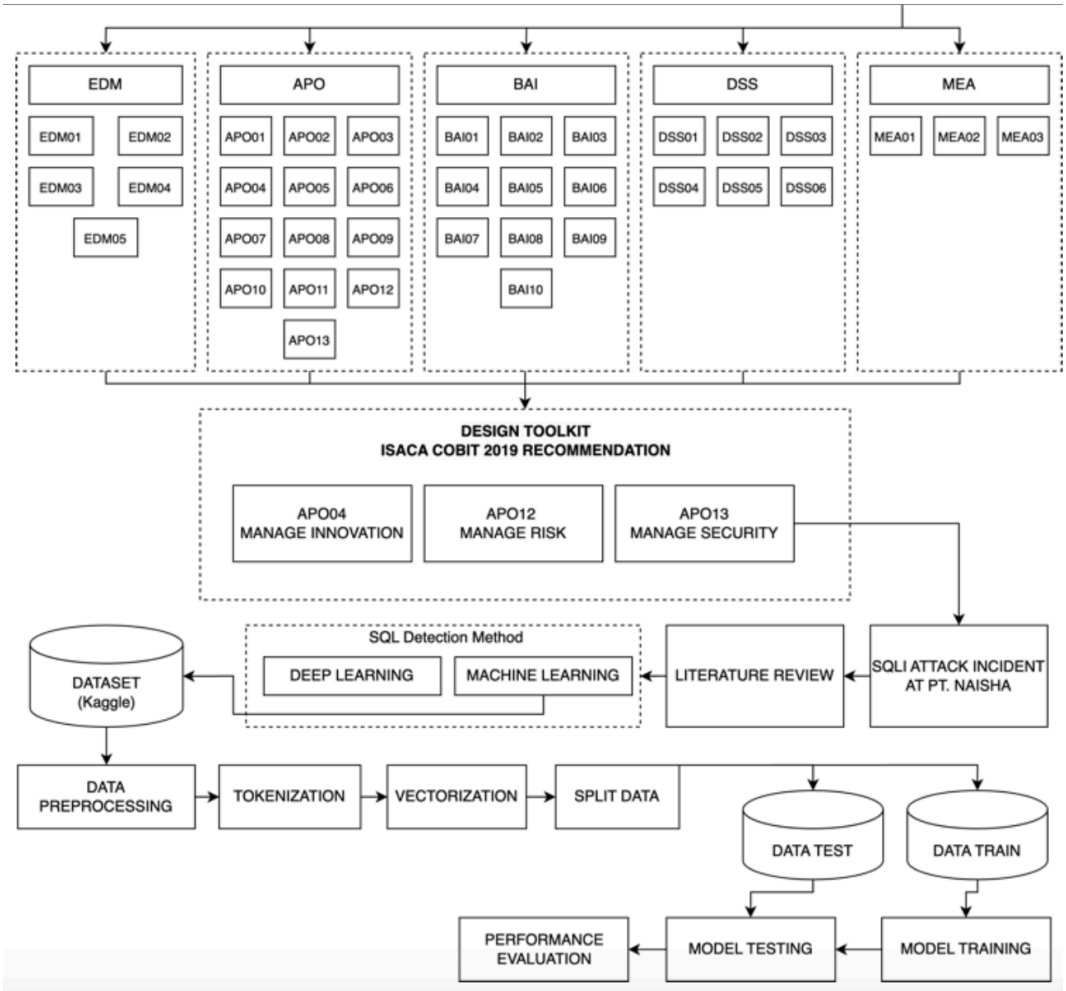
Dari beberapa penelitian terdahulu, penggunaan model SVM yang dioptimasi dengan PSO menunjukkan perubahan nilai akurasi yang semakin membaik. Oleh karena itu peneliti dalam penelitian ini ingin menerapkan model SVM dengan optimasi PSO pada dataset kueri-kueri injeksi SQL publik yang didapatkan dari platform penyedia atau penampungan dataset yakni Kaggle. Tujuan utama dari penelitian ini adalah mengembangkan model deteksi SQL *injection* menggunakan optimasi SVM dan mengimplementasikannya untuk keamanan komputer di PT. Naisha.

Penelitian ini juga bertujuan untuk mengevaluasi efektivitas model dalam mendeteksi serangan SQL *injection* serta memberikan rekomendasi lebih lanjut berdasarkan hasil implementasi dan evaluasi model untuk kebutuhan penelitian di masa yang akan datang atau untuk keperluan peneliti lain yang meneliti dalam bidang yang sama. Manfaat dari penelitian ini diharapkan dapat dirasakan oleh berbagai pihak, antara lain bagi PT. Naisha dalam meningkatkan keamanan sistem informasi dan mengurangi risiko serangan siber; bagi peneliti untuk memperdalam pengetahuan dan keterampilan dalam penerapan machine learning untuk keamanan siber; bagi dunia penelitian diharapkan penelitian ini dapat menyumbangkan penelitian yang memperkaya literatur tentang penggunaan SVM dalam deteksi serangan SQL *injection*; serta bagi industri secara umum dengan menyediakan studi kasus yang da-

pat menjadi referensi bagi perusahaan lain dalam meningkatkan sistem keamanan informasi mereka. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi yang signifikan baik dalam praktik maupun teori di bidang keamanan siber.

2 Metodologi

Alur penelitian yang dilakukan diilustrasikan dalam Gambar 1.



Gambar 1 Alur penelitian.

Penjelasan mengenai alur metodologi penelitian pada Gambar 1 adalah sebagai berikut :

1. Analisis Strategi SWOT
   Analisis SWOT (Strengths, Weaknesses, Opportunities, Threats) dilakukan untuk mengidentifikasi kekuatan, kelemahan, peluang, dan ancaman yang dihadapi PT. Naisha. Berdasarkan hasil analisis ini, strategi SO (Strength-Opportunities), ST (Strength-Threats), WO (Weakness-Opportunities), dan WT (Weakness-Threats) dibuat supaya peneliti dapat menganalisis apa yang dapat dilakukan dari strategi yang ditemukan.

2. Analisis COBIT 2019 (Design Toolkit ISACA COBIT 2019
   Dengan menggunakan COBIT 2019 Design Toolkit, analisis dilakukan berdasarkan strategi yang telah dibentuk dari hasil SWOT. Parameter pada design factor 1 hingga design factor 4 dimasukkan ke dalam toolkit ini. Proses ini bertujuan untuk menilai sejauh mana strategi yang telah ditetapkan dapat diimplementasikan melalui kerangka kerja COBIT 2019.
3. Hasil Design Toolkit ISACA COBIT 2019
   Hasil dari COBIT 2019 Design Toolkit menunjukkan tiga subdomain dengan nilai tertinggi: APO04 (Managed Innovation), APO12 (Managed Risk), dan APO13 (Managed Security). Urgensi dari ketiga subdomain ini dianalisis untuk menentukan prioritas bagi PT. Naisha. Mengingat pengalaman PT. Naisha yang pernah terkena serangan Injeksi SQL, subdomain APO13 Managed Security dipilih sebagai fokus utama untuk pengembangan sistem deteksi serangan SQL Injection.
4. Tinjauan Literatur
   Peneliti melakukan tinjauan literatur untuk mencari referensi model deteksi serangan Injeksi SQL. Literatur yang dicari mencakup studi terdahulu yang relevan dan metode yang telah terbukti efektif dalam mendeteksi serangan jenis ini. Tinjauan literatur difokuskan pada model deep learning dan machine learning, karena kedua pendekatan ini telah menunjukkan hasil yang signifikan dalam deteksi serangan siber. Peneliti mencari model-model yang dapat diterapkan untuk meningkatkan keamanan sistem PT. Naisha.
5. Pengumpulan Dataset
   Dataset yang didapatkan merupakan dataset bersifat publik. Dataset diperoleh dari platform penyedia layanan untuk menampung dataset. Dataset ini mencakup data historis serangan SQL Injection yang telah terverifikasi. Dataset yang digunakan terdiri dari 4.200 baris data, dengan 1.128 data berlabel 1 (terindikasi SQL injection) dan 3.072 data berlabel 0 (tidak terindikasi SQL injection).
6. Prapemrosesan Data
   Data yang dikumpulkan kemudian diproses melalui tahap preprocessing untuk membersihkan dan mempersiapkan data agar siap untuk pelatihan. Langkah-langkah preprocessing meliputi case folding (mengubah teks menjadi huruf kecil), penghapusan stopwords (kata-kata umum yang tidak memiliki makna signifikan), penghapusan data null (kosong), dan penghapusan data duplikat. Pada tahap preprocessing, dilakukan beberapa langkah penting untuk membersihkan dan mempersiapkan data sebelum diterapkan pada model. Langkah-langkah yang dilakukan adalah sebagai berikut:

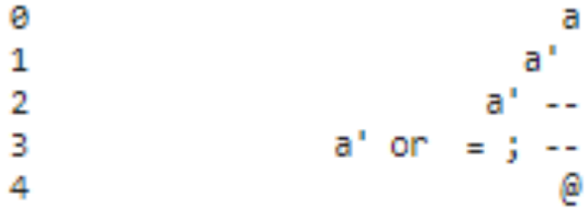
   a. *Case folding*
      Langkah ini melibatkan tahap penggantian elemen huruf besar dalam suatu dokumen dengan elemen standar, yaitu huruf kecil untuk memastikan konsistensi dalam analisis data [17]. Ini membantu dalam menghindari perbedaan yang disebabkan oleh format huruf yang berbeda, seperti memperlakukan "Data" dan "data" sebagai kata yang sama.
   b. *Stopword removal* (penghapusan angka)
      Langkah ini menghilangkan kata-kata umum yang umum digunakan tetapi tidak memiliki arti tersendiri [18]. Proses ini melibatkan penghapusan angka dari teks. Angka sering kali tidak memberikan informasi penting dalam analisis teks dan dapat mempengaruhi hasil model jika dibiarkan.
   c. Penghapusan data duplikat
      Sebelum data diolah oleh model, dilakukan upaya pembersihan data pada tahap pembersihan atau penghapusan data untuk menghasilkan hasil keluaran terbaik [19].

Langkah ini penting untuk menghapus baris data yang identik secara keseluruhan. Proses pengolahan yang terduplikat dilakukan dengan tujuan untuk menyeleksi sentimen-sentimen yang tumpang tindih agar tidak membebani proses penghitungan model yang diusulkan [17]. Ini dilakukan untuk mengurangi bias yang mungkin timbul dalam model akibat pengulangan data yang sama.

d. Penghapusan *data error*
Melibatkan identifikasi dan penghapusan baris data yang tidak valid atau mengandung kesalahan. Tujuannya adalah memastikan bahwa model hanya diberi data yang valid dan dapat diandalkan.

7. Tokenisasi Data Data yang telah melalui preprocessing selanjutnya dipecah menjadi token-token melalui proses tokenisasi. Dalam penelitian terdahulu untuk kasus deteksi injeksi SQL peneliti terdahulu melakukan tokenisasi pada kueri SQL di mana kueri dipecah menjadi string, seperti kata kunci, simbol, frasa, dan tanda baca, yang juga disebut token [13]. Dalam pendekatan ini untuk mendeteksi injeksi SQL, setiap karakter dipertahankan, dan token dihasilkan melalui ekspresi reguler [13]. Token-token ini merupakan unit dasar yang digunakan dalam vektorisasi untuk representasi data yang lebih terstruktur. Tujuan dari proses tokenisasi adalah untuk memisahkan kalimat-kalimat dalam sebuah dokumen dan memecahnya menjadi kata-kata [17].

Peneliti dalam melakukan proses tokenisasi data menggunakan *library* Python yakni *Natural Language Toolkit* (NLTK). NLTK menyediakan alat-alat dan sumber daya yang diperlukan untuk pengolahan bahasa alami (*Natural Language Processing*/NLP). Langkah tokenisasi menggunakan pustaka nltk dan fungsi WordPunctTokenizer(). Fungsi ini digunakan untuk memisahkan setiap kata dengan spasi dan menghilangkan karakter tanda baca yang melekat pada kata yang belum melalui tahap pembersihan data [18]. Gambar 2 merupakan bentuk data kueri SQL yang belum dilakukan proses tokenisasi, dan Gambar 3 merupakan bentuk data kueri SQL setelah dilakukan proses tokenisasi.



Gambar 2 Bentuk data *query* sebelum dilakukan tokenisasi.

Gambar 3 Bentuk data *query* setelah dilakukan tokenisasi.

8. Vektorisasi Data Vektorisasi dilakukan menggunakan metode TF-IDF (Term Frequency-Inverse Document Frequency) untuk mengubah token-token menjadi representasi numerik. Sampel injeksi SQL, seperti bahasa alami, merupakan teks dengan aturan tata bahasa tertentu. Ada beberapa metode vektorisasi teks dalam pemrosesan bahasa alami, seperti model himpunan kata, BoW, TF-IDF, dan vektor kata terdistribusi [12]. Algoritma TF-IDF dapat mengukur pentingnya kata-kata berdasarkan frekuensi kata dan rasio dokumen terbalik [12]. Vektorisasi ini memungkinkan model machine learning dan untuk dapat membaca data dan memproses data guna untuk pelatihan dan pengujian data. TF, atau frekuensi kata, menunjukkan seberapa sering sebuah kata muncul dalam sebuah teks, seperti dalam rumus 1 [12].

$$TF = \frac{jumlah\ kemunculan\ kata\ dalam\ teks}{jumlah\ total\ kata\ dalam\ teks} \tag{1}$$

Frekuensi kata tunggal tidak dapat secara tepat menunjukkan pentingnya sebuah kata dalam teks, karena beberapa kata mungkin sering muncul dalam berbagai teks, seperti "the" dan "an" dalam bahasa Inggris atau "I" dan "you" dalam bahasa Mandarin. IDF, atau frekuensi teks terbalik, menggambarkan seberapa umum sebuah kata di seluruh teks. Nilai IDF cenderung lebih rendah jika kata tersebut sering muncul di banyak teks, seperti yang dijelaskan dalam rumus 2 [12].

$$IDF_x = log\frac{N+1}{N_x+1} + 1 \tag{2}$$

Dengan N merujuk pada jumlah total teks dalam set pelatihan, dan N_x adalah jumlah teks yang memuat kata x. Hasil akhir dari algoritma TF-IDF diperoleh dengan mengalikan dua nilai, yaitu frekuensi kata dan frekuensi teks terbalik, yang menggambarkan pentingnya sebuah kata dalam teks, seperti ditunjukkan dalam rumus 3 [12].

$$TF - IDF_x = TF \times IDF_x \tag{3}$$

9. Penanganan Dataset Tidak Seimbang
Dataset yang tidak seimbang dapat mempengaruhi kinerja model. Untuk mengatasi ini, digunakan teknik Synthetic Minority Over-sampling Technique (SMOTE) untuk menyeimbangkan jumlah data berlabel 1 dan 0. SMOTE bertujuan untuk menyeimbangkan distribusi kelas dengan cara meningkatkan dan menggandakan sampel dari kelas minoritas secara acak [20]. Metode ini menciptakan instance baru dari kelas minoritas dengan mencampurkan sampel yang sudah ada. SMOTE menggunakan interpolasi linier untuk menghasilkan catatan pelatihan virtual bagi kelas minoritas. Catatan pelatihan sintetis ini dibentuk dengan secara acak memilih satu atau lebih dari k-tetangga terdekat untuk setiap contoh dalam kelas minoritas [21]. Setelah SMOTE, jumlah data meningkat menjadi 5956 baris dengan distribusi label yang seimbang, yaitu masing-masing 2978 untuk label 1 dan 0.
10. Pembagian Dataset (Data Latih dan Data Uji)
Dataset dibagi menjadi dua bagian yakni data latih (training) dan data uji (testing). Pembagian ini bertujuan untuk melatih model dengan data training dan menguji kinerjanya dengan data testing, sehingga dapat dievaluasi performanya. Dataset dibagi dengan proporsi perbandingan 80% untuk data training, dan 20% untuk data testing.
11. Pelatihan Model
Model deteksi serangan Injeksi SQL dilatih menggunakan data training yang ekstensif.

Proses pelatihan ini melibatkan penyesuaian parameter model untuk memaksimalkan kemampuannya dalam mendeteksi berbagai jenis serangan. Dalam penelitian ini, peneliti menggunakan model *Support Vector Machine* sebagai metode dasar. Selanjutnya, dibandingkan kinerja daro model SVM ini dengan versi yang dioptimalkan menggunakan *Particle Swarm Optimization*. Optimalisasi ini diharapkan dapat meningkatkan akurasi dan efisiensi deteksi serangan Injeksi SQL. Dengan membandingkan kedua pendekatan ini, peneliti bertujuan untuk mengevaluasi sejauh mana PSO dapat meningkatkan performa SVM dalam konteks keamanan siber.Pengujian Model Setelah pelatihan selesai, model diuji menggunakan data testing untuk menilai kinerjanya. Pengujian ini penting untuk memastikan bahwa model dapat mendeteksi serangan Injeksi SQL dengan akurasi tinggi pada data yang belum pernah dilihat sebelumnya.

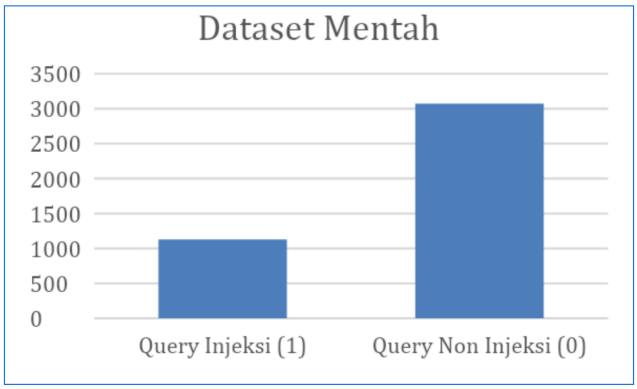
12. Evaluasi Performa Model
Langkah terakhir dalam penelitian ini adalah evaluasi performa model yang digunakan. Evaluasi dilakukan dengan menggunakan matriks kebingungan (confusion matrix) dan membandingkan hasil model SVM tanpa optimalisasi dengan PSO dan hasil model SVM yang telah dilakukan optimalisasi dengan PSO [17]. Evaluasi performa dilihat dengan menggunakan metrik evaluasi utama seperti akurasi dan F1-score. Akurasi mengukur seberapa tepat model dalam mengklasifikasikan data secara keseluruhan, sementara F1-score memberikan gambaran tentang keseimbangan antara presisi (proporsi prediksi positif yang benar) dan recall (proporsi insiden yang berhasil diidentifikasi). Evaluasi ini bertujuan untuk menilai seberapa baik model dalam mendeteksi serangan Injeksi SQL, serta untuk memastikan kelayakan model sebelum diterapkan di lingkungan produksi PT. Naisha.

3 Hasil dan pembahasan

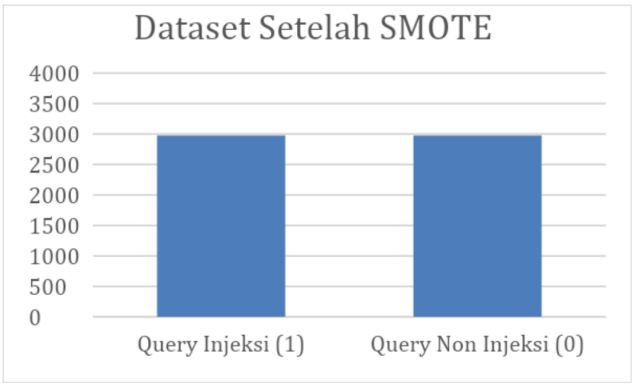
Hasil dari penelitian ini menunjukkan bahwa optimasi model SVM menggunakan PSO secara signifikan meningkatkan kinerja model dalam mendeteksi SQL injection. Langkah-langkah preprocessing seperti case folding, stopword removal, penghapusan data duplikat, dan penghapusan data error terbukti sangat penting untuk memastikan kualitas data yang digunakan dalam pelatihan model. Data yang bersih dan konsisten membantu meningkatkan akurasi model secara keseluruhan. Penggunaan teknik TF-IDF dalam vektorisasi memberikan bobot yang tepat pada kata-kata penting, sehingga membantu model lebih efektif dalam mengenali pola yang menunjukkan SQL injection.

Selain itu, penanganan dataset yang tidak seimbang melalui SMOTE juga sangat efektif dalam meningkatkan kinerja model. Dataset yang seimbang memastikan model dapat mendeteksi kedua kelas (SQL injection dan non-SQL injection) dengan baik. Penggunaan SMOTE meningkatkan jumlah data hingga menjadi 5.956 baris dengan distribusi label yang seimbang, yaitu masing-masing 2.978 untuk label 1 dan 0. Gambar 4 merupakan gambaran jumlah data pada dataset sebelum dilakukannya SMOTE dan gambar 5 merupakan gambaran jumlah data pada dataset setelah dilakukan proses SMOTE.

Hasil dari pelatihan dan pengujian model terhadap dataset yang telah dilakukan penanganan data yang tidak seimbang dan telah dilakukan proses pembagian data untuk proses pelatihan model dan pengujian model, optimasi parameter menggunakan PSO (Particle Swarm Optimization) menunjukkan hasil yang sangat baik. PSO adalah metode optimasi yang meniru perilaku kawanan burung atau ikan dalam mencari makanan, dan pada penelitian ini, PSO digunakan untuk menemukan parameter terbaik (C dan gamma) untuk SVM. Dengan menggunakan PSO, parameter optimal ditemukan, yaitu Best C sebesar



Gambar 4 Dataset sebelum diterapkan SMOTE.



Gambar 5 Dataset setelah diterapkan SMOTE.

593,0497396215296 dan Best gamma sebesar 0,07795813722739078. Gambar 7 merupakan detail dari hasil evaluasi performa model SVM tanpa dilakukan optimalisasi dengan PSO.

```
Accuracy of SVM on test set: 0.7927852348993288
F1 Score of SVM on test set: 0.7834886807784655
Precision of SVM on test set: 0.8534994068801898
Recall of SVM on test set: 0.7927852348993288
Classification Report:
              precision    recall  f1-score   support

           0       0.71      1.00      0.83       596
           1       1.00      0.59      0.74       596

    accuracy                           0.79      1192
   macro avg       0.85      0.79      0.78      1192
weighted avg       0.85      0.79      0.78      1192
```

Gambar 6 Detail evaluasi performa SVM tanpa PSO.

Gambar **??** merupakan detail dari hasil evaluasi performa model SVM yang dilakukan dengan optimalisasi dengan PSO. Hasil ini memungkinkan model SVM+PSO mencapai akurasi sebesar 0,9966 dan F1 Score sebesar 0,9966, jauh lebih tinggi dibandingkan dengan

model SVM tanpa optimasi yang hanya mencapai akurasi 0,7927 dan F1 Score 0,7834.

```
Accuracy of SVM on test set: 0.9966442953020134
F1 Score of SVM on test set: 0.9966386554621849
Precision of SVM on test set: 0.9983164983164983
Recall of SVM on test set: 0.9949664429530202
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       596
           1       1.00      0.99      1.00       596

    accuracy                           1.00      1192
   macro avg       1.00      1.00      1.00      1192
weighted avg       1.00      1.00      1.00      1192
```

Gambar 7 Detail evaluasi performa SVM + PSO.

Selain membandingkan model SVM dengan SVM+PSO, penelitian ini juga membandingkan dengan hasil performa dari model machine learning lainnya seperti Decision Tree dan Logistic Regression, dan hasil dari perbandingan model tersebut menunjukkan hasil evaluasi performa tertinggi didapatkan dari model SVM + PSO. Perbandingan dari hasil evaluasi dari model yang digunakan dalam penelitian ini dapat dilihat pada tabel 1.

Model	SVM	SVM+PSO	Decision Tree	Logistic Regression
Akurasi	79,27%	99,66%	98,90%	99,32%
Precision	85,34%	99,83%	98,91%	99,33%
Recall	79,27%	99,49%	98.90%	99,32%
F1-Score	78,34%	99,66%	98.90%	99,33%

Tabel 1 Hasil model yang digunakan

4 Kesimpulan dan saran

Dari hasil rancangan, penerapan dan pengujian pada basis data sistem informasi silsilah keluarga diperoleh hasil bahwa penerapan schema dapat meningkatkan keamanan dibandingkan tidak menerapkan schema dikarenakan ketika peran ingin mengakses tabel di dalam schema memiliki izin yang terbatas berdasarkan kriteria pemetaan hak akses sesuai kebutuhannya. Hal ini berbeda dengan pemberian hak akses tanpa schema yang mana schema public memiliki sifat publik sehingga dapat diakses oleh peran siapapun, hal ini dapat mempengaruhi berkurangnya keamanan terhadap data yang disimpan.

Penerapan hak akses peran terhadap schema, tabel dan kolom tertentu yang tepat selain diterapkan pada basis data juga dapat meningkatkan keamanan di sisi Back-End dengan melakukan login terhadap pengguna tertentu berdasarkan service atau transaksi yang diperlukan, sehingga kecil kemungkinan terjadinya klien menggunakan akun superuser seperti root atau postgres dalam melakukan transaksi. Hal ini dapat menghindari resiko terjadinya serangan sql injection yang mana hacker tidak dapat mengeksploitasi transaksi basis data secara bebas karena terkendala hak akses.

Di sisi lain, hasil dari penerapan keamanan hak akses dan schema selain menjaga keamanan tabel dan kolom tertentu, juga dapat ditingkatkan dengan melakukan keamanan

pada tingkat baris (Row-level security) sehingga peran hanya dapat melakukan transaksi pada baris tertentu untuk menjamin keabsahan dan integritas informasi yang disajikan.

Berdasarkan kesimpulan di atas, diperlukan adanya penelitian berikutnya untuk menguji praktik penerapan keamanan hak akses penggunaan schema pada sisi Back-end dalam menghadapi serangan sql injection. Pengamanan basis data selain menerapkan pada schema, tabel dan kolom, dapat memungkinkan adanya penelitian lebih lanjut mengenai kemanan pada Row-level security sehingga data yang tersimpan telah terkuci dengan aman dari tingkat schema, tabel, kolom dan barisnya berdasarkan hak aksesnya.

Pustaka

1 D. Darmawan dan A. F. Wijaya, "Analisis dan desain tata kelola teknologi informasi menggunakan framework cobit 2019 pada pt. xyz," *Journal of Computer and Information Systems Ampera*, vol. 3, no. 1, pp. 1–17, 2022.

2 M. A. Saputra dan M. R. Redo, "Penerapan framework cobit 2019 untuk perancangan tata kelola teknologi informasi pada perguruan tinggi," *Journal of science and social research*, vol. 4, no. 3, pp. 352–364, 2021.

3 A. Intan, A. Setiawan, dan M. R. Maengkom, "Studi literatur terhadap peran dan manfaat cobit 2019 dalam tata kelola teknologi informasi di indonesia," *Innovative: Journal Of Social Science Research*, vol. 3, no. 5, pp. 1681–1692, 2023.

4 T. O. T. . . team, "introduction: Welcome to the owasp top 10 - 2021," 2021, accessed: Jul. 18, 2024. [Online]. Available: https://owasp.org/Top10/A00_2021_Introduction/

5 D. P. Purbawa, A. J. Ulhaq, G. Ikhsan, A. M. Shiddiqi, D. Ary, dan M. Shiddiqi, "An enhanced sql injection detection using ensemble method," *JUTI: Jurnal Ilmiah Teknologi Informasi*, vol. 21, no. 1, pp. 1–9, 2023.

6 M. Hasan, Z. Balbahaith, dan M. Tarique, "Detection of sql injection attacks: a machine learning approach," in *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*. IEEE, 2019, pp. 1–6.

7 P. Tang, W. Qiu, Z. Huang, H. Lian, dan G. Liu, "Detection of sql injection based on artificial neural network," *Knowledge-Based Systems*, vol. 190, p. 105528, 2020.

8 N. Sakron, G. Firmansyah, H. Akbar, dan B. Tjahjono, "Audit of information technology governance on school operational cost flow in smkn west jakarta using cobit 2019," *Jurnal Indonesia Sosial Sains*, vol. 4, no. 09, pp. 763–772, 2023.

9 W. Rankothge, M. Randeniya, dan V. Samaranayaka, "Identification and mitigation tool for sql injection attacks (sqlia)," in *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*. IEEE, 2020, pp. 591–595.

10 A. S. Pramono dan E. Pramono, "Deteksi serangan sql injection menggunakan hidden markov model," *J. Tecnoscienza*, vol. 5, no. 2, p. 243, 2021.

11 M. Alghawazi, D. Alghazzawi, dan S. Alarifi, "Deep learning architecture for detecting sql injection attacks based on rnn autoencoder model," *Mathematics*, vol. 11, no. 15, p. 3286, 2023.

12 W. Zhang, Y. Li, X. Li, M. Shao, Y. Mi, H. Zhang, dan G. Zhi, "Deep neural network-based sql injection detection method," *Security and Communication Networks*, vol. 2022, no. 1, p. 4836289, 2022.

13 F. K. Alarfaj dan N. A. Khan, "Enhancing the performance of sql injection attack detection through probabilistic neural networks," *Applied Sciences*, vol. 13, no. 7, p. 4365, 2023.

14 H. B. Jatmiko, N. T. Kurniadi, dan D. Maulana, "Optimasi naïve bayes dengan particle swarm optimization untuk analisis sentimen formula e-jakarta," *Journal Automation Computer Information System*, vol. 2, no. 1, pp. 22–30, 2022.

15 I. G. N. E. Susena, M. T. Furqon, dan R. C. Wihandika, "Optimasi parameter support vector machine (svm) dengan particle swarm optimization (pso) untuk klasifikasi pendonor darah dengan dataset rfmtc," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 12, pp. 7278–7284, 2018.

16 A. Nurkholis, Z. Abidin, H. Sulistiani *et al.*, "Optimasi parameter support vector machine berbasis algoritma firefly pada data opini film," *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, vol. 5, no. 5, pp. 904–910, 2021.

17 P. Arsi, R. Wahyudi, R. Waluyo *et al.*, "Optimasi svm berbasis pso pada analisis sentimen wacana pindah ibu kota indonesia," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 2, pp. 231–237, 2021.

18 B. W. Rauf, "Sentimen analisis pertambangan di konawe utara dengan metode naïve bayes," in *Prosiding Seminar Nasional Pemanfaatan Sains dan Teknologi Informasi*, vol. 1, no. 1, 2023, pp. 97–102.

19 P. br Sihotang, F. D. br Sitanggang, N. Azriansyah, dan E. Indra, "Penerapan natural language processing untuk analisis sentimen terhadap aplikasi streaming," *Jurnal Ilmiah Betrik*, vol. 14, no. 02 AGUSTUS, pp. 273–282, 2023.

20 W. Rahayu, D. Jollyta, A. Hajjah, Y. N. Marlim, Y. Desnelita *et al.*, "Synthetic minority oversampling technique (smote) for boosting the accuracy of c4. 5 algorithm model," *Journal of Artificial Intelligence and Engineering Applications (JAIEA)*, vol. 3, no. 3, pp. 624–630, 2024.

21 A. Muneer, R. F. Ali, A. Alghamdi, S. M. Taib, A. Almaghthawi, dan E. A. Ghaleb, "Predicting customers churning in banking industry: A machine learning approach," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 26, no. 1, p. 539, 2022.