TESIS

PREDIKSI SERANGAN *SQL INJECTION* PADA JARINGAN COMPUTER MENGGUNAKAN METODE *NAÏVE BAYES*

Disusun oleh:

Nama : **Pramono**
NIM : **19.52.1208**
Konsentrasi : **Informatics Technopreneurship**

PROGRAM STUDI S2 TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2022

TESIS

PREDIKSI SERANGAN *SQL INJECTION* PADA JARINGAN COMPUTER MENGGUNAKAN METODE *NAÏVE BAYES*

SQL INJECTION ATTACK PREDICTION IN COMPUTER NETWORK USING NAÏVE BAYES METHOD

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama : **Pramono**
NIM : **19.52.1208**
Konsentrasi : **Informatics Technopreneurship**

PROGRAM STUDI S2 TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2022

HALAMAN PENGESAHAN

PREDIKSI SERANGAN *SQL INJECTION* PADA JARINGAN COMPUTER MENGGUNAKAN METODE *NAÏVE BAYES*

SQL INJECTION ATTACK PREDICTION IN COMPUTER NETWORK USING NAÏVE BAYES METHOD

Dipersiapkan dan Disusun oleh

Pramono

19.52.1208

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Rabu, 3 Agustus 2022

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 3 Agustus 2022
Rektor

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

HALAMAN PERSETUJUAN

PREDIKSI SERANGAN *SQL INJECTION* PADA JARINGAN COMPUTER MENGGUNAKAN METODE *NAÏVE BAYES*

SQL INJECTION ATTACK PREDICTION IN COMPUTER NETWORK USING NAÏVE BAYES METHOD

Dipersiapkan dan Disusun oleh

Pramono

19.52.1208

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Rabu, 3 Agustus 2022

Pembimbing Utama	**Anggota Tim Penguji**
Prof. Dr. Kusrini, M.Kom	**Dr. Andi Sunyoto, M.Kom.**
NIK. 190302106	**NIK. 190302052**
	Dhani Ariatmanto, M.Kom., Ph.D.
Pembimbing Pendamping	**NIK. 190302197**
Eko Pramono, S.Si, M.T.	**Prof. Dr. Kusrini, M.Kom.**
NIK. 555006	**NIK. 190302106**

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 3 Agustus 2022
Direktur Program Pascasarjana

Prof. Dr. Kusrini, M.Kom.
NIK. 190302106

HALAMAN PERNYATAAN KEASLIAN TESIS

Yang bertandatangan di bawah ini,

Nama mahasiswa : Pramono
NIM : 19.52.1208
Konsentrasi : Informatics Technopreneurship

Menyatakan bahwa Tesis dengan judul berikut:
Prediksi Serangan *Sql Injcetion* pada Jaringan Computer Menggunakan Metode *Naïve Bayes*

Dosen Pembimbing Utama : Prof. Dr. Kusrini, M.Kom.Dosen
Pembimbing Pendamping : Eko Pramono, S.Si., M.T.

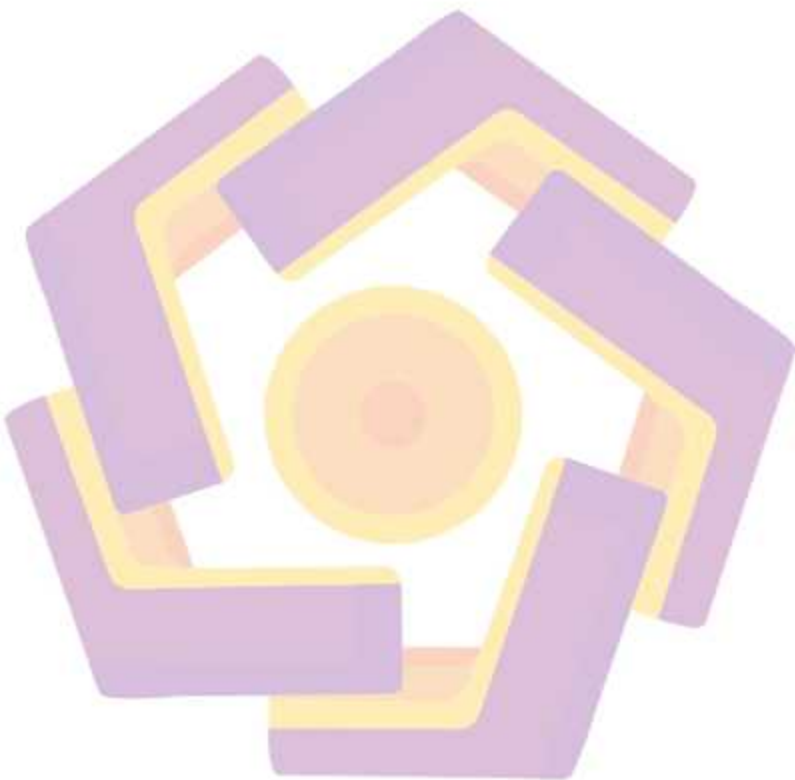
1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Tim Dosen Pembimbing
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini
4. Perangkat lunak yang digunakan dalam penelitian ini sepenuhnya menjadi tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi

Yogyakarta, 3 Agustus 2022
Yang Menyatakan,

Pramono

HALAMAN PERSEMBAHAN

Untuk istri tercinta dan Anaku Felisha, atas cinta dan dukungan abadi mereka



HALAMAN MOTTO

- Ilmu pengetahuan itu bukanlah yang dihafal, melainkan yang memberi manfaat. Imam Syafi'i
- Barang siapa yang keluar rumah untuk mencari ilmu, maka ia berada di jalan Allah hingga ia pulang. HR. Tirmidzi
- Amalan yang lebih dicintai Allah adalah amalan yang terus menerus dilakukan walaupun sedikit
- Sesungguhnya Allah tidak akan mengubah keadaan suatu kaum, kecuali mereka mengubah keadaan mereka sendiri. (QS Ar Ra'd 11)
- Maka sesungguhnya bersama kesulitas itu ada kemudahan. Sesungguhnya bersama kesulitan itu ada kemudahan (QS Al Insyirah 5-6)

KATA PENGANTAR

Puji syukur kehadirat Allah SWT, berkat rahmat, taufik serta hidayahnya sehingga dapat diselesaikannya laporan tesis ini di Universitas Amikom Yogyakarta. Shalawat serta salam tak lupa saya panjatkan kepada junjungan Nabi Muhammad SAW, beserta keluarga dan sahabat-sahabat nya, dan para pengikutnya sampai akhir zaman. Laporan Tesis yang berjudul **"Prediksi Serangan Sql Injcetion pada jaringan Computer menggunakan metode Naive Bayes"** diajukan untuk memenuhi salah satu syarat akademik dalam menempuh kelulusan di program magister (S2) teknik informatika dan sebagai syarat dalam memperoleh gelar Magister Komputer di Universitas Amikom Yogyakarta. Ucapan terima kasih yang sebesar-besarnya kepada semua pihak yang telah banyak membantu dalam menyelesaikan laporan ini utamanya kepada yang terhormat :

1. Bapak/ Ibu Pimpinan dan Pembantu Pimpinan di Universitas Amikom Yogyakarta, atas segala dukungan yang sangat menunjang keberhasilan penulis dalam menyelesaikan studi di Universitas Amikom Yogyakarta.
2. Prof. Dr. Kusrini, M.Kom selaku dosen pembimbing I dan Eko Pramono, S.Si, M.T. selaku dosen pembimbing II yang selalu membimbing, mengarahkan dan memberikan masukan dalam proses penyusunan tesis ini dengan penuh kesabaran dan memberikan ilmu yang terbaik kepada penulis.
3. Bapak/ Ibu dosen MTI S2 dan Karyawan di Universitas Amikom Yogyakarta, yang telah membantu penulis untuk menyelesaikan pendidikan di Universitas Amikom Yogyakarta.

4. Teman-teman kelas MTI angkatan 19 B yang memberikan dukungan, bantuan, dan kerjasama selama di perkuliahan.
5. Orang, Mertua, adik-adik, Istri dan anak-anak serta seluruh keluarga besar yang selalu memberikan doa, semangat serta motivasi hingga dapat menyelesaikan proses studi ini.
6. Serta pihak-pihak yang tidak dapat disebutkan satu persatu, yang telah membantu terselesainya tesis ini.

Yogyakarta, April 2022

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

SQL Injection adalah serangan yang mencoba untuk mendapatkan akses tidak sah ke database dengan menyuntikkan kode dan mengeksploitasi query SQL. Injeksi SQL adalah serangan yang mudah dieksekusi tetapi sulit dideteksi dan diklasifikasi karena banyak jenisnya. Kerentanan SQLI adalah hasil dari validasi input pengguna yang tidak benar, memungkinkan penyerang untuk memanipulasi kueri programmer dengan menambahkan operator SQL baru. Oleh karena itu, penelitian ini menggunakan Naïve Bayes untuk mengklasifikasikan. Dataset yang akan digunakan dalam penelitian ini berasal dari sebuah website bernama Kaggle. Penelitian ini menganalisis metode yang dihasilkan dari proses klasifikasi berdasarkan nilai akurasi confusion matrix, precision, recall. Naive Bayes, menghasilkan tingkat akurasi 95,594%.

Kata kunci: *Sql Injection, Klasifikasi, Naïve Bayes*

ABSTRACT

SQL Injection is an attack that attempts to gain unauthorized access to a database by injecting code and exploiting SQL queries. SQL injection is an attack that is easy to execute but difficult to detect and classify because of the many types. The SQLI vulnerability is the result of incorrect validation of user input, allowing attackers to manipulate programmer queries by adding new SQL operators. Therefore, this study uses Naïve Bayes to classify. The dataset that will be used in this research comes from a website called Kaggle. This study analyzes the method generated from the classification process based on the value of confusion matrix accuracy, precision, recall. Naive Bayes, resulting in an accuracy rate of 95.594%..

Keyword: Sql Injection, Classification, Naïve Bayes

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Perkembangan teknologi informasi, khususnya jaringan komputer memungkinkan terjadinya pertukaran informasi yang cepat dan semakin kompleks. Pengaturan jaringan komputer yang baik tentu akan memaksimalkan pemanfaatan informasi tersebut. Oleh karena itu jaringan komputer harus diatur dan diawasi sehingga kelancaran pengiriman informasi dapat berjalan dengan baik. Semakin besar dan luas sistem jaringan komputer, semakin sulit untuk mengatur dan mengawasinya(Lika et al., 2018). Keamanan teknologi informasi (IT) merupakan sebuah hal mendasar yang penting untuk diperhatikan dalam sebuah lingkaran organisasi maupun individu. Berbagai serangan terhadap server pada organisasi hingga pembajakan akun pada individu, apapun bentuknya tindakan ini hanya mendatangkan kerugian(Maraj et al., 2017). Baik *software* maupun *hardware* serta perangkat lainnya yang saling terhubung satu sama lain.

Dalam dunia sekarang ini yang lebih kompleks dan semakin banyak adalah serangan jaringan, mengikuti data proses bisnis utama organisasi, salah satu data terpenting mereka adalah data dari alat-alat keamanan jaringan (NS), dan memastikan operasi yang dilakukan seharusnya aman dan tangguh tanpa gangguan penyediaan layanan kepada penggunanya(Aliero et al., 2016). Oleh karena itu, perlindungan terhadap jaringan komputer adalah salah satu tugas paling mendasar bagi *user* dan organisasi, karena bahkan satu serangan pun dapat mengakibatkan

kerusakan serius pada data dan kerugian parah. Kehilangan besar dan serangan sering memaksakan kebutuhan untuk teknik deteksi yang dapat diandalkan dan akurat. Deteksi perangkat lunak dibagi menjadi analisis statis, yang berarti analisis file atau program yang dikompilasi dan analisis dinamis yang berarti menganalisis perilaku jangka waktu seperti konsumsi baterai, membaca dan menulis memori, dan pemanfaatan jaringan perangkat(Pham & Subburaj, 2020).

Strategi utama untuk deteksi dan untuk pergerakan serangan yaitu dengan pemanfaatan *Intrusion Detection System* (IDS). Setiap serangan yang teridentifikasi biasanya diungkapkan atau akumulasi dengan menggunakan kerangka kerja *Security Information and Event Management* (SIEM). Sistem kerangka kerja SIEM muncul karena berbagai sumber dan memanfaatkan prosedur pengayakan pencegahan untuk menentukan validitas serangan yang diidentifikasi. *Network Intrusion Detection Systems* (NIDS) diposisikan secara strategis dan mendemonstrasikan gerak layar kerangka kerja antara semua node pada kerangka kerja. Hal Ini bertujuan untuk mengawasi tindakan di seluruh jaringan dan kegiatan subnet yang tidak biasa terkait dengan serangan yang sudah diketahui. Setelah serangan diakui, atau perilaku tidak teratur terdeteksi, kehati-hatian dapat dikirim ke administrator. Sebuah kasus NIDS akan mengenal subnet di mana firewall berada, sehingga untuk memeriksa apakah seseorang berusaha masuk ke firewall. Pada kenyataannya sekarang, orang akan memeriksa semua aktivitas masuk dan keluar; Namun, melakukan hal tersebut, dapat membuat kemacetan yang akan melemahkan kecepatan umum sistem. OPNET dan NetSim adalah instrumen yang sering digunakan untuk mereproduksi kerangka kerja *Intrusion Detection System* (Chellam et al., 2018).

Aplikasi web dengan database yang menyimpan informasi penting merupakan salah satu target dari SQL Injection (SQLIA) dikarenakan database yang berisi informasi penting dapat diakses oleh penyerang dengan cara menyuntikan querry SQL yang kemudian informasi penting yang terdapat dalam database dapat hilang. Ditemukan bahwa salah satu ancaman inti untuk sebagian besar aplikasi web saat ini adalah Serangan SQL Injeksi (SQLIA). Kemudian, proyek keamanan aplikasi web terbuka (OWASP) telah mengidentifikasi ancaman utama (10 teratas) untuk keamanan aplikasi web yang ditemukan pada tahun 2017, di mana serangan SQL injection mendapat skor pertama atau berada di puncak daftar teratas(Maraj et al., 2017).

SQL injection merupakan salah satu serangan yang mudah dilakukan tetapi sulit untuk dideteksi dan diklasifikasi karena keberagaman jenisnya. Kerentanan SQLI dihasilkan dari validasi input yang tidak tepat dari pengguna, yang memungkinkan penyerang untuk memanipulasi permintaan pemrogram dengan menambahkan operator SQL baru(Dinata, 2019). Pada penelitian ini telah merancang sebuah model pengklasifikasi untuk mendeteksi serangan SQL injection. Pengklasifikasian ini bertujuan untuk membantu developer jaringan/atau website mengetahui bahwa jaringan/website yang dibuat sering terinjection oleh serangan SQL jenis apa, dengan mengetahui serangan SQL Injection secara spesifik sehingga developer bisa mengetahui celah keamanan dari jaringan/website yang dibuat dan bisa memberikan perhatian khusus terhadap celah tersebut agar nantinya jaringan/website yang dibuat tidak terus terserang oleh SQL Injection.

Penelitian yang dilakukan (McWhirter et al., 2018)pada penelitian ini menyajikan solusi baru untuk mengklasifikasikan kueri SQL murni pada fitur dari string kueri awal. *Gap-Weighted String Subsequence Kernel* diimplementasikan untuk mengidentifikasi *subsequence* dari karakter bersama antara *string kueri* untuk *output* sebuah *similarity metric*. SVM dilatih *similarity metric* antara *kueri string* yang diketahui kemudian digunakan untuk mengklasifikasikan kueri pengujian yang tidak diketahui. Dengan mengumpulkan dan menampilkan data dari *kueri string*, informasi tambahan dari aplikasi sumber yang dibutuhkan. Sifat probabilistik dari model yang dipelajari memungkinkan solusi untuk beradaptasi dengan ancaman baru selama beroperasi. Solusi yang diusulkan dievaluasi menggunakan jumlah kumpulan data uji yang berasal dari *testbed Amnesia* kumpulan data Perangkat lunak demonstrasi mencapai akurasi 97,07% *Select type queries* dan akurasi 92,48% *Insert type queries*.

Berbeda dengan Penelitian yang dilakukan oleh (Fibrianda & Bhawiyuga, 2018) penelitian ini menggunakan metode *behavior based* dimana dalam proses kerjanya membutuhkan sebuah dataset dan metode. Tetapi tidak semua algoritma data mining memiliki kinerja yang baik dalam mengklasifikasi jenis serangan. Oleh karena itu, penelitian ini melakukan perbandingan beberapa algoritma yaitu *Naïve Bayes*, *SVM Linear*, *SVM Polynomial*, dan *SVM Sigmoid*. Dataset yang digunakan dalam penelitian ini adalah dataset dari ISCX2012 *testbed* tanggal 14 Juni 2012. Penelitian ini menganalisis perbandingan metode yang dihasilkan dari proses klasifikasi berdasarkan nilai akurasi *confusion matrix*, *precision*, *recall*, dan *f1 score*. *Naïve bayes*, SVM *Linear*, SVM *Polynomial* dan SVM *Sigmoid*

menghasilkan persentase akurasi berturut-turut sebesar 85,055%, 99,995%, 99,999%, dan 99,995%. Persentase akurasi tertinggi diperoleh SVM *Polynomial*, sedangkan *Naïve Bayes* menghasilkan persentase akurasi terendah.

Penelitian yang dilakukan (Olalere, 2018)mengusulkan model pengenalan pola *Naïve Bayes* untuk deteksi dan kategorisasi serangan injeksi SQL. 16.050 halaman web/URL dihasilkan dari dmoz. 7.000 diasumsikan rentan terhadap tujuh jenis serangan injeksi SQL paling populer dan 9.050 diberi label sebagai akuntansi yang tidak rentan terhadap 16.050 contoh berdasarkan pola. Validasi model dilakukan dengan validasi silang bertingkat 10 lipatan dengan 1-10 *random seeds*. Hasil eksperimennya menunjukkan akurasi deteksi dan kategorisasi masing-masing 98% dan 99%

1.2. Rumusan Masalah

Rumusan masalah yang diangkat pada Penelitian ini dapat dipaparkan sebagai berikut:

1. Berapa nilai *accuracy, precision dan recall* dari algoritma *Naïve bayes* dalam mengklasifikasikan Serangan dan Non serangan?
2. Apakah *Naïve Bayes* mampu di terapkan sebagai model dalam memprediksi serangan pada Sql Injection?
3. Apakah metode *Naïve Bayes* dapat digunakan pada proses klasifikasi untuk identifikasi?

1.3. Batasan Masalah

Permasalahan yang dibahas dalam penelitian ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Algoritma yang digunakan untuk klasifikasi serangan dan non serangan adalah *Naïve Bayes*
2. Penelitian ini hanya berfokus dalam mengklasifikasikan jenis serangan dan non serangan Sql *Injection*
3. Dataset yang digunakan untuk proses *training* dan *testing* diperoleh dari *kaggle*
4. Jenis serangan yang diklasifikasikan adalah *Sql Injection*

1.4. Tujuan Penelitian

Tujuan dari Penelitian ini adalah

1. Untuk mengetahui nilai *accuracy, precision dan recall* dari algoritma *Naïve bayes* dalam mengklasifikasikan Serangan dan Non serangan
2. Untuk menganalisis akurasi deteksi serangan pada jaringan komputer dengan menggunakan metode *Naïve Bayes*
3. Mengimplementasikan algoritma *Naïve Bayes* menjadi suatu metode dalam mengklasifikasikan terhadap *query* serangan SQL *Injection*.

1.5. Manfaat Penelitian

Dalam pelaksanaan penelitian ini diharapkan dapat bermanfaat yaitu:

a. Bagi peneliti, mengetahui tingkat akurasi tertinggi model klasifikasi *Naïve Bayes*

2. Memberikan pengetahuan baru mengenai model yang dapat melakukan klasifikasi Serangan dan Non serangan pada sql injection.
3. Diharapkan penelitian ini dapat menambah referensi penelitian di bidang jaringan, khususnya mengenai *Sql Injection, Naive Bayes*,

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Nuniek Fahriani, Putri Aisyiyah R. Devi, Darmawan Aditama. (Tahun 2017) melakukan penelitian tentang Alternatif Penanganan Jenis Serangan Pencurian Data Pada Jaringan Komputer yang bertujuan melindungi, mengamankan informasi yang berada di dalamnya (data dan fisik). Dengan syarat dari keamanan adalah *prevention* (pencegahan), yaitu memperkecil peluang penembusan oleh pemakai yang tak diotorisasi. *Observation* (observasi) yaitu identifikasi dan otentifikasi. Dengan penelitian ini diharapakan oleh penulis menjadi alternative, penanganan keamanan data dari serangan pencurian data yaitu contoh yang dilakukan menggunakan teknik enkrip dan dekrip, dimana pada implementasi yang dilakukan mengenkrip file audio menggunakan algoritma blowfish, bahwa aplikasi yang dibuat bisa diimplementasikan dengan baik karena file yang sudah dienkripsi menjadi tidak bisa dijalankan dan isinya tidak dapat dipahami lagi.

Penelitian Selanjutnya tentang Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode *Naïve Bayes* Dan *Support Vector Machine* (SVM) yang dilakukan oleh Mercury Fluorida Fibrianda, Adhitya Bhawiyuga, (2018) penilitian ini bertujuan untuk Melihat tingkat akurasi dari metode *Naive Bayes*, SVM *Linear*, SVM *Polynomial*,dan SVM *Sigmoid* dengan menggunakan dataset ISCX 2012. Dataset ini dipilih karena merupakan dataset baru yang dikembangkan dari tahun 2009 sampai 2012 oleh Fakultas Ilmu Komputer,

Universitas New Brunswick. Dimana pada penelitian sebelumnya dataset yang digunakan yaitu KDD Cup 1999 dan DARPA 1998 yang merupakan dataset lama sehingga kurang akurat jika dilakukan pengujian deteksi serangan saat ini. Penelitian ini kemudian menghasilkan ahapan klasifikasi serangan menggunakan metode *behavior based* membutuhkan sebuah dataset dan metode, memerlukan Fitur yang digunakan dalam proses klasifikasi, performa yang dihasilkan ersentase akurasi bertunut-turut sebesar 85,055%,99,995%, 99,999% dan 99,995%.

Penelitian berikutnya Adalah Intrusion Detection in Computer Networks using Lazy Learning Algorithm yang dilakukan oleh Aditya Chellam, Ramanathan L, Ramani S,(2018). Dalam penelitian tersebut menjelaskan bahwa Penggunaan lazy learning methodologies untuk meningkatkan kinerja IDS secara keseluruhan. Teknik pengindeksan berdasarkan berat heuristik baru telah digunakan untuk mengatasi kelemahan kompleksitas pencarian yang tinggi yang melekat dalam lazy learning dan pada Makalah ini menjelaskan keuntungan belajar di IDS. belajar IDS meningkatkan efisiensi NIDS dengan menghilangkan pre-fetching overhead yang terdapat dalam algoritma pembelajaran yang populer digunakan saat ini.

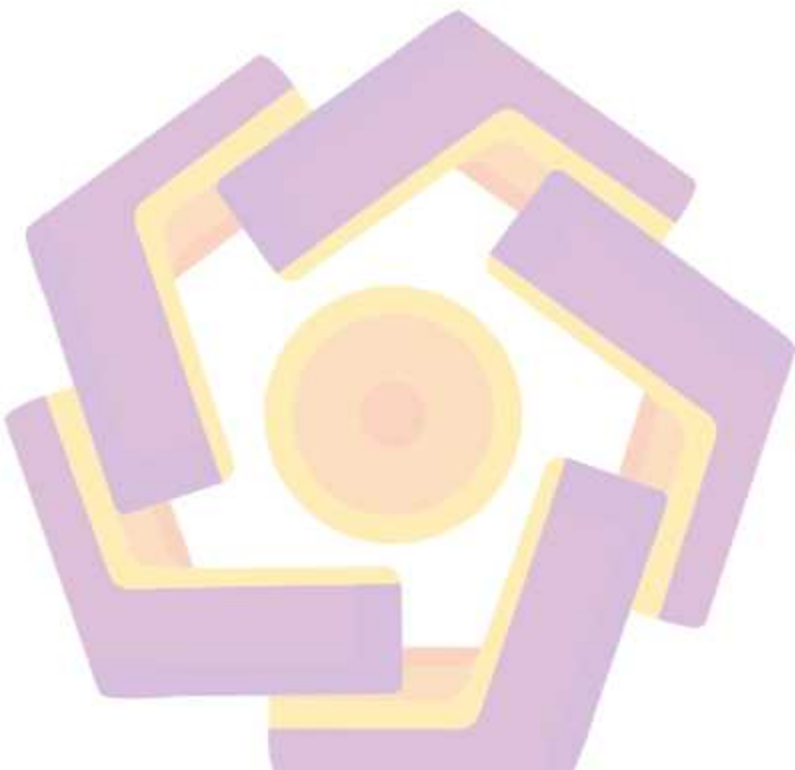
Penelitian berikutnya Natalia Miloslavskaya, (2019) meneliti tentang Stream Data Analytics for Network Attacks' Prediction. Penelitian ini menjelaskan bagaimana Menyederhanakan, arsitektur stream data terkait NS, dan baik untuk memprediksi serangan terhadap aset dan layanan jaringan yang disediakan. Kurangnya sumber daya dan kompetensi staf adalah hambatan paling signifikan dalam mulai menggunakan teknologi analitik yang dapat melakukan pemrosesan aliran data terkait NS secara efisien dan efektif. Namun, penyebaran alat streaming

data tidak akan mengubah pendekatan organisasi untuk memastikan NS dari reaktif menjadi proaktif dalam semalam.

Penelitian lain dilakukan oleh Hanane KALKHA, Hassan SATORI, Khalid SATORI (2018) memuat tentang Preventing Black Hole Attack in Wireless Sensor Network Using HMM dengan tujuan Memberikan solusi dengan HMM untuk mengidentifikasi node berbahaya di nirkabel jaringan sensor melalui pencegahan serangan lubang hitam. Prose pengukuran berdasarkan metrik seperti penundaan end-to-end dan paket memberikan rasio. Algoritma berbasis HMM telah digunakan untuk memodelkan urutan keputusan jalur terpendek yang dipilih oleh node sumber untuk berkomunikasi dengan node tujuan. Pendekatan ini memungkinkan mencegah jalur dan simpul jahat, dan hasil percobaan ini menunjukkan efisiensi yang diusulkan untuk mencegah simpul jahat.

Penelitian terakhir yang dijadikan acuan datang dari studi literatur milik Eddy Prasetyo Nugroho, Eki Nugraha, Mokhamad Nizar Zulfikar, (2019) yang membahas tentang Sistem *Reporting* Keamanan pada Jaringan *Cloud Computing* Melalui *bot* Telegram dengan Menggunakan Teknik *Intrussion Detection and Prevention System*, tujuan penelitian ini adalah Untuk menanggulangi serangan yang terjadi, dengan menggunakan aplikasi Fail2Ban dan Port Scan Attack Detector (PSAD) untuk menutup akses dari IP penyerang. Pemanfaatan sebuah *Intrusion Detection System* (IDS) sangat berpengaruh untuk memantau aktifitas lalu lintas pada jaringan. Berdasarkan hasil penelitian pada paper ini, sistem IDS berhasil mendeteksi serangan, yang mana dilakukan terlebih dahulu proses konfigurasi dan penambahan *rules* agar snort bisa mendeteksi serangan

berdasarkan pencocokan dengan *signature* yang terdapat pada *rules* tersebut. Cepat lambatnya sistem mendeteksi sebuah serangan ditentukan dari jenis serangan tersebut serta pola *signature* dari *rules* snort, yang mana tiap serangan memiliki pengaturan konfigurasi *rules* yang berbeda



2.2. Keaslian Penelitian

Tabel 2.1 Keaslian Penelitian

Prediksi Serangan *Sql Injection* Pada Jaringan Computer Menggunakan Metode *Naïve Bayes*

No	Judul	Pencliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1	Alternatif Penanganan Jenis Serangan Pencurian DataPada Jaringan Komputer	Nuniek Fahriani, Putri Aisyiyah R. Devi, Darmawan Aditama. Tahun 2017	Bagaimana melindungi, mengamankan informasi yang berada di dalamnya (data dan fisik). Dengan yarat dari keamanan adalah *prevention* (pencegahan), yaitu memperkecil peluang penembusan oleh pemakai yang tak diotorisasi. *Observation* (observasi) yaitu identifikasi dan otentifikasi	Menjadi alternative pertama, penanganan keamanan data dari serangan pencurian data yaitu contoh yang dilakukan menggunakan teknik enkrip dan dekrip, dimana pada implementasi yang dilakukan mengenkrip file audio menggunakan algoritma blowfish, bahwa aplikasi yang dibuat bisa diimplementasikan dengan baik karena file yang sudah dienkripsi menjadi tidak bisa dijalankan dan isinya tidak dapat dipahami lagi.	Untuk menghindari kehilangan dan perubahan data, untuk keamanan data dapat dilakukan sistem *backup* yaitu dengan konsep replikasi. Saran yang dapat diberikan adalah pada konsep desain manajemen jaringan untuk proses implementasi. Alternatif penggunaan algoritma keamanan data selain blowfish, teknik yang digunakan lebih bervariatif selain enkripsi dan dekripsi.	Penelitian yang dilakukan hanya mencontohkan pada pengamanan data berupa pengamanan data bentuk *file* adalah *file* audio dan algoritma blowfish yang diimplementasikan dengan perangkat lunak java netbeans 8.0.2 Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes* (Lanjutan)

Tabel 2.1 Keaslian Penelitian (lanjutan)

2	Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode *Naïve Bayes* Dan *Support Vector Machine* (SVM)	Mercury Fluorida Fibrianda, Adhitya Bhawiyuga, Tahun 2018	Melihat tingkat akurasi dari metode *Naive Bayes*, SVM *Linear*, SVM *Polynomial*, dan SVM *Sigmoid* dengan menggunakan dataset ISCX 2012. Dataset ini dipilih karena merupakan dataset baru yang dikembangkan dari tahun 2009 sampai 2012 oleh Fakultas Ilmu Komputer, *Universitas New Brunswick*. Dimana pada penelitian sebelumnya dataset yang digunakan yaitu KDD Cup 1999 dan DARPA 1998 yang merupakan dataset lama sehingga kurang akurat jika dilakukan pengujian deteksi serangan saat ini	1. Tahapan klasifikasi serangan menggunakan metode *behavior based* membutuhkan sebuah dataset dan metode. 2. Memerlukan Fitur yang digunakan dalam proses klasifikasi. 3. Performa yang dihasilkan ersentase akurasi berturut-turut sebesar 85,055%, 99,995%, 99,999% dan 99,995%	Untuk meningkatkan nilai akurasi dari sebuah metode dapat dilakukan dengan beberapa teknik diantaranya teknik *bagging* dan *boosting*. Dalam penelitian ini menggunakan teknik *random sampling* dan belum menggunakan kedua teknik tersebut, karena penelitian ini hanya terbatas pada perbandingan metode *Naive Bayes*, SVM *Linear*, SVM *Polynomial* dan SVM *Sigmoid*	Penelitian yang dilakukan penelitian ini hanya terbatas pada perbandingan metode *Naive Bayes*, SVM *Linear*, SVM *Polynomial* dan SVM *Sigmoid*. Penelitian ini juga menngunakan data dari ISCX *testbed*. Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes* (Lanjutan)

Tabel 2.1 Keaslian Penelitian (lanjutan)

3	Intrusion Detection in Computer Networks using Lazy Learning Algorithm	Aditya Chellam, Ramanathan L., Ramani S, Tahun 2018	Penggunaan lazy learning methodologies untuk meningkatkan kinerja IDS secara keseluruhan. Teknik pengindeksan berdasarkan berat heuristik baru telah digunakan untuk mengatasi kelemahan kompleksitas pencarian yang tinggi yang melekat dalam lazy learning	Makalah ini menjelaskan keuntungan belajar di IDS. belajar IDS meningkatkan efisiensi NIDS dengan menghilangkan pre-fetching overhead yang terdapat dalam algoritma pembelajaran yang populer digunakan saat ini	Peningkatan k-nearest neighbour algorithm diusulkan untuk mengurangi kompleksitas pencarian menggunakan sistem pengindeksan berdasarkan berat heuristic.	Penelitian yang dilakukan Dalam proses deteksi serangan menggunakan Intrusion Detection in Metode yang digunakan Lazy Learning Algorithm Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes*
4	Stream Data Analytics for Network Attacks' Prediction	Natalia Miloslavskaya Tahun 2019	Menyederhanakan, arsitektur stream data terkait NS, dan baik untuk memprediksi serangan terhadap aset dan layanan jaringan yang disediakan.	Kurangnya sumber daya dan kompetensi staf adalah hambatan paling signifikan dalam mulai menggunakan teknologi analitik yang dapat melakukan pemrosesan aliran data terkait NS secara efisien dan efektif. Namun, penyebaran alat streaming data tidak akan mengubah pendekatan organisasi untuk memastikan NS dari reaktif menjadi proaktif dalam semalam	langkah selanjutnya adalah mengikat semua aliran data yang dikumpulkan bersama. Dari informasi ini, mereka akan menghasilkan pengetahuan baru, yang selanjutnya digunakan untuk beradaptasi dengan kondisi baru operasi jaringan mereka. Karenanya, penelitian lebih lanjut kami rencanakan di bidang ini	Penelitian yang dilakukan dalam memprediksi serangan menggunakan Stream Data Analytics yang kemudian menyederhanakan stream data terkait NS. Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes* (Lanjutan)

Tabel 2.1 Keaslian Penelitian (lanjutan)

5	Preventing Black Hole Attack in Wireless Sensor Network Using HMM	Hanane KALKHA, Hassan SATORI, Khalid SATORI Tahun 2018	Memberikan solusi dengan HMM untuk mengidentifikasi node berbahaya di nirkabel jaringan sensor melalui pencegahan serangan lubang hitam	Pengukuran berdasarkan metrik seperti penundaan end-to-end dan paket memberikan rasio.	Algoritma berbasis HMM telah digunakan untuk memodelkan urutan keputusan jalur terpendek yang dipilih oleh node sumber untuk berkomunikasi dengan node tujuan. Pendekatan ini memungkinkan mencegah jalur dan simpul jahat, dan hasil percobaan ini menunjukkan efisiensi yang diusulkan untuk mencegah simpul jahat	Penelitian yang dilakukan adalah proses preventing dalam serangan Black hole di wireless dan menggunakan Metode HMM Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes*
6	Sistem *Reporting* Keamanan pada Jaringan *Cloud Computing* Melalui *bot* Telegram dengan Menggunakan Teknik *Intrussion Detection and Prevention System*	Eddy Prasetyo Nugroho, Eki Nugraha, Mokhamad Nizar Zulfikar, Tahun 2019	Untuk menanggulangi serangan yang terjadi, dengan menggunakan aplikasi Fail2Ban dan Port Scan Attack Detector (PSAD) untuk menutup akses dari IP penyerang.	Pemanfaatan sebuah *Intrusion Detection System* (IDS) sangat berpengaruh untuk memantau aktifitas lalu lintas pada jaringan. Berdasarkan hasil penelitian pada paper ini, sistem IDS berhasil mendeteksi serangan, yang mana dilakukan terlebih dahulu proses konfigurasi dan penambahan *rules* agar snort bisa mendeteksi serangan berdasarkan pencocokan dengan *signature* yang terdapat pada *rules* tersebut	Cepat lambatnya sistem mendeteksi sebuah serangan ditentukan dari jenis serangan tersebut serta pola *signature* dari *rules* snort, yang mana tiap serangan memiliki pengaturan konfigurasi *rules* yang berbeda	Penelitian yang dilakukan Fokus kepada membangun sebuah sistem deteksi intrusi dengan meenghasilkan *output* tidak hanya berupa catatan aktifitas intrusi pada *database* tetapi juga notifikasi melalui *instant messaging* Telegram Penelitian yang akan dilakukan adalah memprediksi serangan Sql dan hanya melakukan clasifikasi serangan Sql Injection dengan metode *Naïve Bayes* (Lanjutan)

2.3. Landasan Teori

2.3.1 *SQL Injection*

SQL *Injection* merupakan salah satu risiko aplikasi/situs web yang paling sering ditemukan. Berdasarkan laporan keamanan aplikasi yang dikeluarkan oleh *Veracode*, 32% dari aplikasi web paling tidak memiliki satu kerentanan SQL *Injection*. Berdasarkan *Open Web Application Security Project* (OWASP), *injection* juga merupakan merupakan ancaman nomor satu terhadap keamanan aplikasi web. Sebagai tambahan, berdasarkan laporan kerentanan dari komunitas yang diperoleh Badan Siber dan Sandi Negara (BSSN) melalui *Voluntary Vulnerability Disclosure Program* (VVDP), dari bulan Januari s.d. April 2019, 73% dari laporan kerentanan yang diterima merupakan kerentanan SQL *Injection*. Berdasarkan statistik di atas, meskipun termasuk ke dalam kerentanan yang mudah untuk dicegah, saat ini SQL *Injection* tetap menjadi risiko pada aplikasi web yang paling sering ditemukan dan banyak organisasi memiliki kerentanan terhadap potensi kebocoran data akibat serangan *SQL Injection*.terduga.

SQL merupakan singkatan dari *Structured Query Language* yang merupakan bahasa pemrograman untuk mengakses dan memanipulasi database. Database atau basis data sendiri merupakan sekumpulan data yang dikelola sedemikian rupa dengan ketentuan-ketentuan tertentu sehingga menjadi data yang terorganisir. sedangkan. *SQL Injection* adalah sebuah teknik penyerangan terhadap kelemahan atau celah yang dimiliki oleh SQL. Penyerangan tersebut memanfaatkan celah dimana user dapat menginputkan karakter apapun yang berarti user dapat memasukkan command atau query SQL dengan tujuan untuk mendapatkan

informasi penting yang berada di dalam database, seperti data personal dari user lain, login sebagai user lain, atau bahkan sebagai admin.

Gambar 2.1 Data Rekapitulasi VVD BSSSN

2.3.1.1 Jenis Serangan

Secara umum, terdapat beberapa jenis serangan SQL *Injection*, yaitu:

1. *Tautology Based Attack*

a. Tujuan utama dari serangan jenis Tautology-based attack adalah untuk menginjeksi kode pada satu atau lebih baris perintah SQL dengan kondisi bersyarat sehingga baris perintah tersebut akan selalu dieksekusi dengan nilai "*True*".

b. Penyerang mengeksploitasi parameter yang dapat diinjeksi yang digunakan di dalam baris perintah *query* dengan sintaks kondisional bersyarat *WHERE*.

c. Tujuan Serangan:

Melakukan bypass terhadap mekanisme otentikasi, mengidentifikasi parameter yang dapat diinjeksi, melakukan ekstrasi data.

2. ***Query Ilegal* / *Query* yang Secara Logik Salah (*Illegal/Logically Incorrect Queries*)**

   a. Serangan ini memungkinkan penyerang untuk memperoleh informasi mengenai tipe dan struktur dari back-end basis data dari sebuah aplikasi berbasis web.

   b. Tujuan Serangan:

      Mengidentifikasi parameter yang dapat diinjeksi, melakukan database finger-printing, dan melakukan ekstraksi data.

3. ***Union Query***

   a. Serangan jenis ini, penyerang melakukan eksploitasi terhadap parameter yang rentan untuk mengubah data set yang dikembalikan dari perintah query yang dijalankan.

   b. Tujuan Serangan:

      Melakukan bypass otentikasi, melakukan ekstraksi data.

4. ***Piggy-Backed Query***

   a. Pada serangan jenis ini, penyerangan berupaya untuk menginjeksi tambahan perintah *query* ke dalam perintah query yang asli. Kerentanan terhadap serangan jenis ini umumnya bergantung pada konfigurasi basis data yang mengizinkan beberapa perintah *query* SQL di dalam satu string tunggal.

   b. Tujuan Serangan:

      Melakukan ekstraksi data, menambahkan atau memodifikasi data, melakukan serangan DoS (*Denial of Services*), mengeksekusi perintah jarak jauh (*remote commands execution*).

5. Stored-Procedures

a. Serangan jenis ini penyerang berusaha untuk mengeksekusi "*stored procedures*".

b. Tujuan Serangan:

Melakukan *privilege escalation*, melakukan serangan *Denial of Service* (DoS), melakukan eksekusi perintah jarak jauh (*remote command execution*).

6. Inference

a. Pada teknik serangan ini, penyerang melakukan modifikasi terhadap *query* untuk menyusun kembali ke dalam bentuk aksi yang dieksekusi berdasarkan jawaban atas pertanyaan benar/salah mengenai nilai data yang tersimpan di dalam database

b. Penyerang umumnya menggunakan teknik ini untuk menyerang situs yang memiliki tingkat keamanan yang sudah cukup baik sehingga pada saat injeksi berhasil dilakukan, tidak ada informasi berguna yang diperoleh penyerang dari pesan error melalui basis data.

c. Tujuan Serangan:

Mengidentifikasi parameter yang dapat diinjeksi, melakukan ekstraksi data serta menentukan skema basis data dari sistem target.

7. Alternate Encodings

a. Teknik serangan ini dilakukan dengan memodifikasi teks yang diinjeksi dengan tujuan untuk menghindari deteksi oleh mekanisme defensif dari

b. teknik pemrograman dan juga beberapa mekanisme penegahan otomatis terhadap serangan SQL *Injection*.

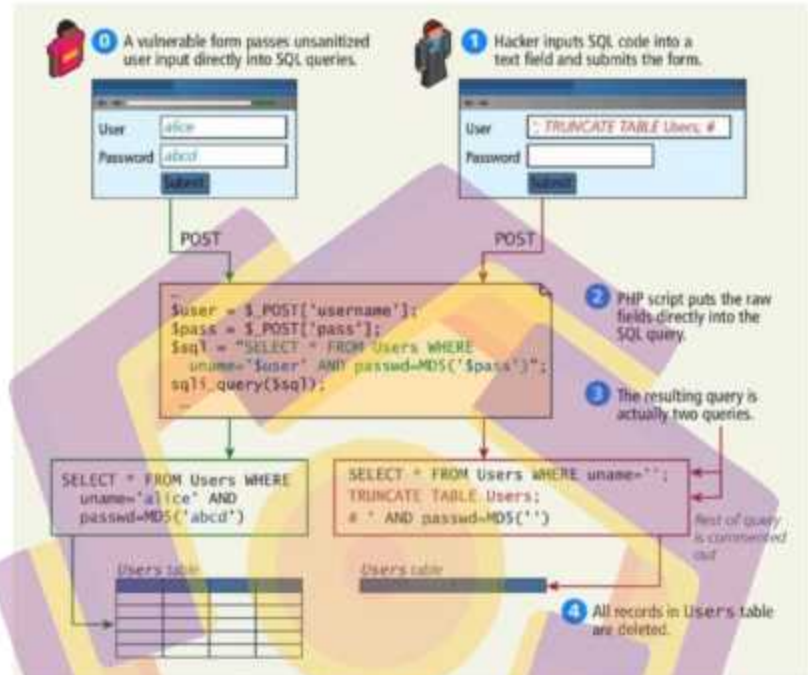
c. Tujuan Serangan: Menghindari/mengelabui deteksi perimeter keamanan.

2.3.1.2 Bagaimana cara serangan SQL Injection bekerja

Secara umum untuk melakukan serangan dengan teknik SQL Injection umumnya penyerang menjalankan dua tahapan proses yaitu [1]:

1. Tahapan Riset. Penyerang umumnya melakukan riset dengan mengirimkan berbagai macam nilai yang tidak diharapkan sebagai argumen input, kemudian dia mempelajari respon dari aplikasi, kemudian dia menentukan apakah target URL rentan atau tidak terhadap SQL Injection.
2. Tahap Serangan. Setelah melakukan riset mengenai kerentanan pada aplikasi, maka pada tahapan ini penyerang membuat payload tertentu sebagai nilai input dari argumen sehingga payload tersebut akan diinterpretasikan sebagai bagian dari perintah SQL ketimbang hanya data. Kemudian database melakukan eksekusi terhadap perintah yang telah dibuat oleh penyerang.

Berikut ini merupakan ilustrasi singkat mengenai bagaimana serangan SQL *Injection* bekerja.



Gambar 2.2 Ilustrasi Serangan SQLi

Serangan SQL Injection terjadi karena aplikasi web yang tidak melakukan validasi terhadap nilai yang diterima dari formulir isian pada web, cookie, parameter input, dan lainnya sebelum melewatkan nilai tersebut ke query SQL yang akan dieksekusi pada server database. Dengan demikian, hal ini memungkinkan penyerang untuk memanipulasi input sehingga data yang dikirimkan dapat diinterpretasikan sebagai kode/perintah SQL ketimbang sebagai data.

Berikut merupakan contoh sederhana dari teknik serangan SQL Injection. SQL statement dinamis umumnya dikonstruksi pada saat eksekusi dilakukan, misalkan contoh berikut ditulis dalam bahasa pemrograman .NET dimana parameter input harus diisikan oleh pengguna

```
Query = "SELECT * FROM users WHERE username = ' "
+request.getParameter("input")+ " ' " ;
```

Maka baris kode tersebut akan menjalankan perintah *query* SQL sebagai berikut:

*SELECT * FROM users WHERE username = 'input'*

Salah satu teknik yang umum dilakukan oleh penyerang adalah dengan melakukan manipulasi terhadap SQL statement, dimana penyerang berupaya memodifikasi SQL query statements dan menyisipkan "*exploited statements*" ke dalam database.

*SELECT * FROM users WHERE loginName = '$user' AND loginPassword =* 'Spassword'

Lalu apa yang akan terjadi jika pengguna memasukan nilai berikut:

Suser = ' OR '1' = '1
Spassword = ' OR '1'= '1

Karena nilai 1=1 memiliki arti selalu menghasilkan nilai benar atau *TRUE*, maka query tersebut akan berhasil dieksekusi meskipun tidak ada *username* dan *password* yang valid yang dimasukan, dengan demikian penyerang berhasil mem-bypass mekanisme otentikasi. Versi lainnya dari serangan ini adalan dengan menyisipkan dua karakter dash secara berurutan (--) atau karakter # yang oleh database MySQL dianggap sebagai penanda bagian komentar sehingga baris perintah setelahnya akan

diabaikan.

Dengan demikian penyerang dapat mem-bypass mekanisme otentikasi, karena perintah setelah (--) atau # yakni AND Pin = anything akan dianggap sebagai baris komentar oleh database

2.3.2 *Naïve Bayes*.

Naive Bayes menurut Xhemali et al (2009) adalah metode atau algoritma klasifikasi sederhana yang dapat berkontribusi pada keputusan akhir dan setiap atribut memiliki atribut yang independen. Menurut Hang et al (2006), Naive Bayes adalah proses klasifikasi statistik yang dapat digunakan untuk memprediksi probabilitas pada suatu kelas. Sedangkan menurut Bustami (2013) Naive Bayes adalah metode klasifikasi probabilitas dan statistik yang diperoleh Thomas Bayes, seorang ilmuwan Inggris, dengan memprediksi peluang masa depan berdasarkan pengalaman sebelumnya di sana. Teori probabilitas Bayesian adalah teori statistik matematika yang memberikan kemampuan untuk memodelkan ketidakpastian suatu peristiwa dengan menggabungkan pengetahuan umum dengan fakta yang dapat diamati. Beberapa keunggulan yang ada dalam teori Bayes menurut Grainner (1998) adalah mampu dipahami dengan mudah, mampu dilakukan hanya dengan kode sederhana, mampu melakukan perhitungan dengan cepat. Namun, teori Bayesian juga memiliki kelemahan, yaitu dalam teori tersebut, secara probabilitas, belum mampu mengukur secara presisi seberapa dalam. Dapat dikatakan bahwa ia tidak dapat memberikan bukti keaslian jawaban yang dihasilkan oleh teori Bayesian

Teorema Bayes yang biasanya dipakai dalam pemrograman memiliki

bentuk umum sebagai berikut (Han, Kamber, & Pei, 2011):

$$P(H|X) = \frac{P(X|H)P(H)}{P(H)} \quad (1)$$

Dari persamaan rumus di atas, dapat disimpulkan bahwa H merepresentasikan suatu kelas, sedangkan X merepresentasikan suatu atribut. P(H) dalam persamaan rumus diatas merupakan prior probability H sedangkan P(X) merupakan prior probability X

Tabel 2.2 Atribut Naive Nayes

Atribut	Keterangan
X	Sebuah data yang belum diketahui class-nya
H	Suatu *class* yang spesifik pada hipotesis data X
P(H|X)	Probabilitas hipotesis H berdasarkan kondisi X *(posterior probability)*
P(H)	*Prior probability* H, dapat disebut probabilitas dari hipotesis H
P(X|H)	Probabilitas hipotesis X berdasarkan kondisi H *(posterior probability)*
P(X)	*Prior probability* X, dapat disebut probabilitas dari hipotesis X

Pada *Teorema Bayes*, klasifikasi *naive bayes* mempunyai persamaan berikut:

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)} \quad (2)$$

Alur kerja Klasifikasi Naïve Bayes

Dalam melakukan klasifikasi, langkah pertama adalah memahami masalah dan mengidentifikasi fitur dan label potensial. Fitur adalah karakteristik atau atribut yang mempengaruhi hasil label. Kemudian Klasifikasi memiliki dua fase, fase pembelajaran, dan fase evaluasi. Pada fase pembelajaran, classifier melatih modelnya pada dataset yang diberikan dan

pada fase evaluasi, ia menguji kinerja classifier. Kinerja dievaluasi berdasarkan berbagai parameter seperti akurasi, kesalahan, presisi, dan penarikan/recall.

Gambar 2.3 Arsitetur *Naive Baye*

Sumber : (Bekti Maryuni Susanto, 2013)

2.3.3 *Confusion Matrix*

Confusion matrix merupakan sebuah tabel yang terdiri dari atas banyaknya baris data uji yang diprediksi benar dan tidak benar oleh model klasifikasi, tabel ini diperlukan untuk menentukan kinerja suatu model klasifikasi (Wijayanto 2015). Berikut merupakan tabel dari sebuah confusion matrix

Tabel 2.3 Confusion Matrix

		Kelas Prediksi	
		Positive	*Negative*
Hasil Prediksi	*Positive*	*TP (True Positive)*	*FP (False Positive)*
	Negative	*FN (False Negative)*	*TN (True Negative)*

Dari *Confusion Matrix*, tersebut dapat di peroleh:

1. *Accuracy*

2. *Recall (Sensitivity or True positive rate)*

3. *Precision (Positive predictive value)*

4. *Specificity (True negative rate)*

Dari *Confusion Matix*, keakuratan model dapat dihitung sebagai berikut:

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

Accuracy juga disebut sebagai tingkat perhitungan, dan merupakan ukuran bagaimana contoh yang berbeda diklasifikasikan dengan benar (Staudemeyer,2015).

Recall, juga disebut sebagai sensitivitas atau *True Positif rate* mengacu pada porsi contoh positif yang diklasifikasikan dengan benar sebagai positif. Hal ini dihitung sebagai:

$$Recall \frac{TP}{TP + FN} \quad (5)$$

Precision mengacu pada kemampuan contoh untuk diklasifikasikan dengan benar. Skor yang benar dihitung sebagai:

$$Precision \frac{TP}{TP + FP} \quad (6)$$

Specificity atau *True Negatif rate* mengacu pada bagian dari contoh negatif yang benar diprediksi sebagai negatif. Ini dihitung sebagai:

$$Specificity \frac{TN}{TN + FP} \quad (7)$$

2.3.4 *Term Frequency – Inverse document Frequency (TF-IDF)*

Term Frequency - Inverse Document Frequency atau TF-IDF adalah suatu metode algoritma yang berguna untuk menghitung bobot setiap kata yang umum digunakan. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen.

Bagaimana cara kerja atau mempraktekan metode algoritma TF — IDF dalam JavaScript yang kemudian akan digabungkan kedalam metode Cosine Similarity. Jadi hasil akhir adalah; membuat sebuah program berbasis JavaScript yang bisa menghitung tingkat kemiripan dua buah dokumen.

Langkah Pertama, membahas soal TF atau Term Frequency. Untuk menentukan berapa seringnya kata tsb muncul dalam sebuah dokumen. Jadi, semakin banyak frekuensi kemunculan dari kata tsb, semakin besar pula nanti nilainya.

Pada *Term Frequency* (TF), terdapat beberapa jenis formula yang dapat digunakan:

1. TF *biner* (*binary* TF), hanya memperhatikan apakah suatu kata atau term ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak diberi nilai nol (0).

2. TF murni (*raw* TF), nilai TF diberikan berdasarkan jumlah kemunculan suatu term di dokumen. Contohnya, jika muncul lima (5) kali maka kata tersebut akan bernilai lima (5).
3. TF normalisasi, menggunakan perbandingan antara frekuensi sebuah term dengan nilai maksimum dari keseluruhan atau kumpulan frekuensi term yang ada pada suatu dokumen.
4. TF *logaritmik*, hal ini untuk menghindari dominansi dokumen yang mengandung sedikit term dalam *query*, namun mempunyai frekuensi yang tinggi.

$$\mathbf{TF} = \begin{cases} \mathbf{1 + log_{10}(f_{t,d})}, & f_{t,d} > 0 \\ \mathbf{0}, & f_{t,d} = 0 \end{cases} \quad (8)$$

Dimana nilai $f_{t,d}$ adalah frekuensi term (t) pada document (d). Jadi jika suatu kata atau term terdapat dalam suatu dokumen sebanyak 5 kali maka diperoleh bobot = $1 + log_5 = 1.699$ tetapi jika term tidak terdapat dalam dokumen tersebut, bobotnya adalah nol (0).

Langkah kedua adalah IDF (Inverse Document Frequency). Metode IDF merupakan sebuah perhitungan dari bagaimana term didistribusikan secara luas pada koleksi dokumen yang bersangkutan. Berbeda dengan TF yang semakin sering frekuensi kata muncul maka nilai semakin besar, dalam IDF, semakin sedikit frekuensi kata muncul dalam dokumen, maka makin besar nilainya. Untuk menentukan besaran nilai IDF, dengan menggunakan rumus :

$$\mathbf{IDF_i} = \boldsymbol{log(D/df_j)} \quad (9)$$

Dimana D adalah jumlah semua dokumen dalam koleksi sedangkan df_j adalah jumlah dokumen yang mengandung term (tj). Jenis formula TF yang biasa digunakan untuk perhitungan adalah TF murni (raw TF). Dengan demikian rumus umum untuk Term Weighting TF-IDF adalah penggabungan dari formula perhitungan raw TF dengan formula IDF dengan cara mengalikan nilai TF dengan nilai IDF:

$$w_{ij} = tf_{ij} \, x \, idf_j \tag{10}$$

$$w_{ij} = tf_{ij} \, x \, log(D/df_j) \tag{11}$$

Dimana w_{ij} adalah bobot term (t_j) terhadap dokumen(d_i). Sedangkantf_{ij} adalah jumlah kemunculan term(t_j) dalam dokumen(d_i). D adalah adalah jumlah semua dokumen yang ada dalam database dand_{fj} adalah jumlah dokumen yang mengandung term(t_j) (minimal ada satu kata yaitu term (t_j)). Berapapun besarnya nilai tf_{ij}, apabila D = df_{ij}, maka akan didapatkan hasil 0 (nol), dikarenakan hasil dari log 1, untuk perhitungan IDF. Untuk itu dapat ditambahkan nilai 1 pada sisi IDF, sehingga perhitungan bobotnya menjadi sebagai berikut:

$$w_{ij} = tf_{ij} \, x \, log(D/df_j) + 1 \tag{12}$$

2.3.5 Flowchart

Pada dasarnya, flowchart adalah sebuah diagram yang menggambarkan suatu proses, sistem, atau algoritma dari sebuah jaringan dan komputer. Flowchart banyak digunakan di berbagai bidang untuk mendokumentasikan, mempelajari, merencanakan, dan memperbaiki sebuah proses kerja suatu sistem.

Bagan flowchart umumnya terdiri simbol-simbol flowchart seperti bentuk bangunan dua dimensi. Simbol nantinya akan dimasukan ke dalam sebuah diagram yang detail dan ringkas(Santoso & Nurmalina, 2017).

Salah satu bentuk simbol diagram alur paling umum adalah persegi panjang, oval, dan panah. Simbol ini menentukan langkah-langkah bersamaan dengan panah yang menentukan alurnya. Flowchart dapat berbentuk grafik diagram sederhana yang digambar dengan tangan hingga diagram complex komputer yang terdiri dari banyak proses, langkah, dan alur kerja. Diagram flowchart ini bisa digunakan dalam berbagai bidang. Contohnya untuk menyajikan prosedur alur kerja atau SOP (Standard Operating Procedure) dalam sebuah perusahaan

Salah satu fungsi flowchart adalah untuk mengelola workflow atau alur kerja yang ada pada sebuah sistem dan proyek, dengan flowchart, kita mampu mengelola integritas dan validitas dari proses dan sistem tersebut. Ini akan memberikan hasil yang diharapkan sesuai dengan prosedur,

Fungsi lain dari flowchart adalah untuk mendokumentasikan dan menyimpan semua proses yang terjadi pada sebuah sistem dan proyek, flowchart menjadi alat yang baik untuk memenuhi segala dokumentasi dan penyimpanan tersebut karena diagram alur ini lebih efisien dan tertata.

Selain mengelola workflow dan mendokumentasikan proses, flowchart juga umumnya digunakan untuk mendeteksi kelemahan dan kekurangan dalam suatu proyek Cara flowchart membantu proses ini adalah dengan membagi setiap langkah yang ada pada sistem ke dalam bagian-bagian yang lebih kecil dan detail. Dengan

begitu, akan bisa lebih mudah memeriksa dan melakukan perbaikan dan pembaruan pada bagian yang sekiranya tidak berfungsi dengan baik.

Flowchart sendiri juga mempunyai jenis-jenis berbeda yang lebih khusus sesuai dengan bidangnya, Berikut adalah jenis-jenis flowchart

1. **System Flowchart**

   Jenis flowchart yang satu ini dapat didefinisikan sebagai diagram yang menunjukan flow pekerjaan secara keseluruhan dari sebuah sistem. Selain itu flowchart sistem juga akan mendeskripsikan urutan dan detail dari setiap prosedur dan proses yang ada di dalam sistem tersebut.

2. **Document Flowchart**

   Flowchart dokumen yang juga sering disebut dengan paperwork flowchart adalah diagram yang berfungsi untuk menelusuri alur form dari satu bagian ke bagian yang lainnya. Diagram flowchart menunjukan arus dari satu bagian ke bagian lainnya, termasuk bagaimana cara laporan diproses, dicatat, dan disimpan.

3. **Schematic Flowchart**

   Alur flowchart yang satu ini sebenarnya hampir mirip dengan system flowchart. Tugasnya untuk menggambarkan prosedur di dalam sistem dan jaringan tersebut.

   Yang membedakan kedua flowchart tersebut adalah simbol-simbol yang digunakan. Pada schematic flowchart, ada gambar komputer dan peralatan digital lainnya. Tujuan dari simbol-simbol tersebut adalah untuk memudahkan

komunikasi dengan orang-orang yang tidak paham dengan simbol-simbol flowchart tersebut.

4. Program Flowchart

Program flowchart menunjukkan aliran data saat menulis program atau algoritma Fungsinya adalah untuk menjelaskan proses dengan cepat saat mereka berkolaborasi dengan orang lain. Diagram alur pemrograman ini juga menganalisis logika di balik program untuk memproses kode pemrograman. Diagram alur pemrograman dapat berfungsi dengan cara yang berbeda. Misalnya, menganalisis kode, memvisualisasikan,mengerjakan, dan membantu mengetahui struktur aplikasi untuk menyadari bagaimana pengguna menavigasi dalam suatu alat.

5. Process Flowchart

Process flowchart adalah diagram yang menunjukan langkah-langkah berurutan dari suatu proses dan keputusan yang diperlukan untuk membuat proses bekerja dengan baik. Dalam bagan process flowchart, setiap langkah ditunjukkan sebagai sebuah bentuk. Bentuk ini dihubungkan oleh garis dan panah untuk menunjukkan gerakan dan arah proses. Bagian proses flowchart distandarisasi sedemikian rupa sehingga siapapun yang memiliki pemahaman tentang bagan alur dapat melihatnya dan mengetahui proses tersebut.

Tabel 2.4 Simbol Flowchart

Sumber : (Santoso & Nurmalina, 2017)

No	Simbol Flowchart	Nama	Arti Simbol Flowchart
1		*Process*	Simbol flowchart persegi panjang yang menunjukkan langkah aliran proses normal yang ada di dalam sebuah sistem
2		*Connector*	Simbol ini fungsinya adalah untuk menyederhanakan hubungan antar simbol yang letaknya berjauhan atau rumit bila dihubungkan dengan garis dalam satu halaman
3		*Terminator*	Menunjukkan permulaan (start) atau akhir (stop) dari suatu proses
4		*Decision*	Merupakan simbol yang digunakan untuk memilih proses atau keputusan berdasarkan kondisi yang ada. Simbol ini biasanya ditemui pada flowchart program
5		*Data*	Menunjukkan proses input-output yang terjadi tanpa bergantung dari jenis peralatannya.

Tabel 2.4 Simbol Flowchart (lanjutan)

6		*Document*	Digunakan untuk menunjukkan sebuah dokumen atau laporan dari suatu proses
7		*Flow*	Simbol yang digunakan untuk menghubungkan antara simbol yang satu dengan simbol yang lain (connecting line). Simbol ini juga berfungsi untuk menunjukkan garis alir dari proses.
8		*Off Page Connector*	Simbol ini digunakan untuk menghubungkan simbol dalam halaman berbeda. label dari simbol ini dapat menggunakan huruf atau angka
9		*On Page Connector*	Simbol penghubung alur dalam halaman yang sama.(lanjutan)

2.3.6. Algoritma Machine Learning

Algoritma machine learning adalah algoritma yang digunakan dalam proses machine learning, di mana sistem melakukan pembelajaran berdasarkan data. Algoritma machine learning diterapkan dalam membuat model, berdasarkan kumpulan data. Semakin banyak data, algoritma akan menyesuaikan diri agar model dapat bekerja lebih baik.

Ada banyak pengelompokan algoritma dalam machine learning. Berikut 3 jenis atau macam-macam kelompok algoritma machine learning

1. Supervised learning

Supervised learning atau pembelajaran yang diawasi adalah metode pembelajaran mesin (machine learning) yang ditentukan berdasarkan penggunaan kumpulan data berlabel. Pada dataset ini terdapat "label", yaitu kolom yang menjadi target keluaran model. Dalam pembelajaran terawasi, model dilatih menggunakan satu set data training dan dilatih dengan pengawasan (supervise) untuk mengklasifikasikan atau memprediksi output berdasarkan data berlabel yang telah ditentukan sebelumnya. Supervised learning dapat dibagi menjadi dua kategori, yaitu:

a. Klasifikasi
b. Regresi

2. Unsupervised learning

Unsupervised learning merupakan metode pembelajaran dengan menggunakan algoritme machine learning untuk menganalisis dan mengelompokkan kumpulan data yang tidak berlabel (unlabelled data). Algoritme ini menemukan pola tersembunyi dalam data tanpa perlu campur tangan manusia, sehingga disebut dengan unsupervised (tanpa pengawasan). Unsupervised learning digunakan untuk melakukan:

a. clustering
b. asosiasi

c. dimensionality reduction

3. Reinforcement learning

Reinforcement learning adalah model machine learning yang mirip dengan supervised learning, tetapi algoritme tidak dilatih menggunakan data sampel atau data training. Pada reinforcement learning, model ini belajar sambil berjalan dengan menggunakan trial and error.

BAB III

METODE PENELITIAN

3.1 Jenis, Sifat, dan Pendekatan Penelitian

Pendekatan yang dilakukan pada penelitian ini bersifat kuantitatif dengan jenis penelitian eksperimental, dimana penelitian ini melakukan skenario pengujian dengan mengguakan algoritma *Naïve Bayes* dalam mendeteksi Serangan Sql Injection dari dalam bentuk serangan dan non serangan. Skenario percobaan kemudian akan dievaluasi hasil untuk menilai keakuratan dalam mendeteksi dan mengklasifikasikan Serangan dan Non serangan dari setiap skenario untuk diketahui fakta-fakta dari hasil penelitian. Kemudian Penelitian yang dilakukan penulis bersifat deskriptif, dimana data dijelaskan pada deskriptif angka dan tabel atau

3.2 Metode Pengumpulan Data

Penelitian dilakukan menggunakan dataset yang terdiri dari plan SQL dan SQLI. Kemudian dataset di berikan tiga label yaitu label 0 menandakan Non serangan SQL, kemudian label 1 menandakan serangan SQLI Union, lalu label 2 menandakan serangan SQLI Tautology, dan yang terakhir label 3 yang menandakan serangan SQLI Blind. Memberi label pada data membantu model untuk mempelajari jenis serangan dan jenis non serangan. Contoh dataset dapat dilihat pada tabel 3.1.

Tabel 3.1 Contoh Dataset

Sentence	Label
To believe thought, believe true private heart true men, — genius	0
1))) union all select null --	1
or 1 = 1 or "" =	2
1" Waitfor delay '0:0:5'	3

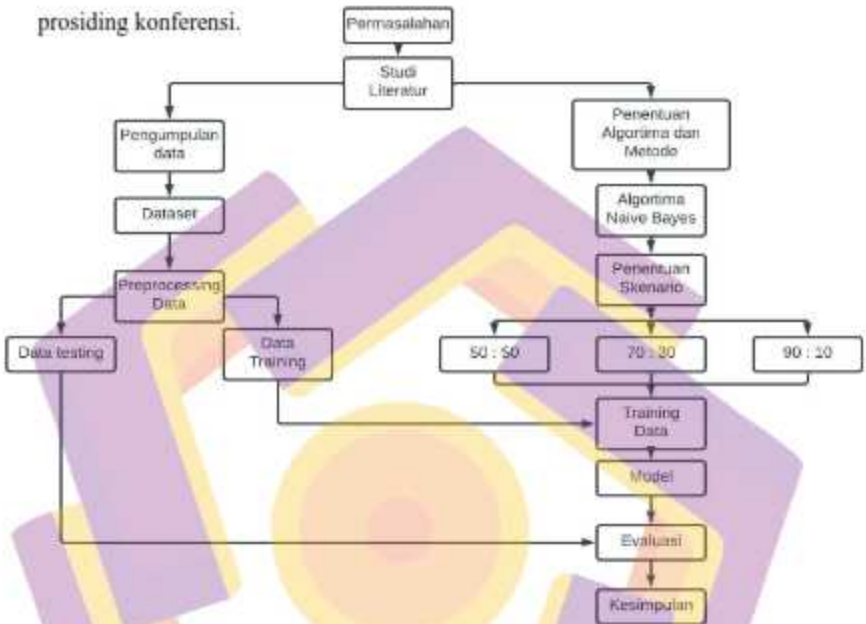
3.3 Metode Analisis Data

Metode analisis yang digunakan dalam penelitian ini adalah analisis kuantitatif menggunakan algoritma *Naïve Bayes*. Dataset yang diperoleh akan dilakukan preproccessing Tahap ini memiliki tujuan untuk mengolah data yang belum siap menjadi data yang sudah siap untuk masuk kedalam pembentukan model. Kemudian proses *preprocessing* pada penelitian ini memiliki tiga proses diantaranya *Case Folding*, *Stopwords Removal*, dan *Tokenisasi*. Percobaan pada penilitian ini melalui 3 skenario utama yang menggunakan *Naïve Bayes*. Dalam melakukan percobaan ini dilakukan dengan bahasa pemrograman *python* dengan bantuan libray *sklearn* dalam membangun arsitektur *Naïve Bayes*

3.4 Alur Penilitian

Dalam Alur penelitian ini menggunakan tahapan diagram alur dari flowchart. Sedangkan pengertian dari Flowchart adalah sebuah diagram yang menggambarkan suatu proses, sistem, atau algoritma dari sebuah jaringan dan komputer. Flowchart banyak digunakan di berbagai bidang untuk mendokumentasikan, mempelajari, merencanakan, dan memperbaiki sebuah proses kerja suatu system selanjutnya dil

akukan pencarian dan pengumpulan literatur yang berkaitan dengan penelitian ini. Sumber yang dicari berupa dokumen primer seperti artikel ilmiah dan prosiding konferensi.



Gambar 3.1 Alur Penelitian

Sumber : (Dinata, 2019)

Alur penelitian secara sistematis dapat dilihat pada Gambar 3.1 dan akan dijelaskan sebagai berikut:

1. Permasalahan

Tahap ini merupakan tahap awal untuk memulai sebuah penelitan dengan melihat permasalahan yang ada saat ini, penjelasan untuk permasalahan ini pada Bab I pendahuluan pada latar belakang masalah

2. Studi Literatur

Setelah menentukan permasalah, proses selanjutnya adalah mencari informasi mengenai hal yang berhubungan dengan masalah tersebut melalui studi literatur. Studi literartur dilakukan dengan membaca jurnal penelitian dan buku yang diaggap relevan dengan permasalahan yang akan diangkat. Proses ini juga sebagai bahan rujukan penelitian untuk memilih metode atau algoritma yang dianggap sesuai dengan permasalahan. Pada proses ini penjelasan pada BAB II tinjauan Pustaka

3. **Penentukan algoritma dan metode**

   Dari studi literatur, penulis telah menentukan algoritma dan metode yang digunakan untuk memprediksi Serangan Sql Injcetion menggunakan Algoritma Naive Bayes pada penelitian ini. Penjelesana pada algoritma ini pada BAB II Landasan Teori sub bab 2.3.2

4. **Skenario**

   Setelah mentukan algoritma dan metode, diperlukan skenario untuk penelitian yang akan dilakukan. Dari proses ini ditentukan tiga skenario yang akan dilakukan. Skenario pertama dengan menggunakan data training dan dan data test 50%:50%, skenario kedua data training dan dan data test 70%:30%selanjutnya scenario ketiga data training dan dan data test 90%:10%

5. **Pengumpulan dataset**

   Machine Learning membutuhkan data untuk memperoleh kecerdasan, sehingg proses pengumpulan data sangat penting pada penelitian ini. Data diperoleh dari media dari website bernama kaggle. Akan tetapi, data tersebut masih berupa text

sehingga dilakukan preprocessing terlebi dahulu. Penjelasan dapat dilihat pada BAB IV sub bab 4.1.1 Pengumpulan dataset

6. **Dataset**

Data yang telah diproses dari tahap pengumpulan data, selanjutnya akan diolah dengan cara melabelkan sesuai dengan serangan. Label yang akan digunakan pada penelitian ini adalah Serangan dan bukan serangan. Data selanjutnya dikelompokkan kedalam kedalam folder sesuai dengan kelasnya. Penjelasan pada BAB IV sub bab 4.1.2 *Preprocessing data*

7. **Training Data dan Model**

Setelah menentukan skenario yang ada yaitu dengan menggunakan perbandingan Antara data uji dan data traning dengan rasio masing-masing (50%:50%), (70%:30%) dan (90%:10%), kita mulai melakukan percoban menggunakan dataset yang telah disediakan sebelumnya. Training data ini juga bertujuan untuk mencari yang mana dari ketiga skenario ini yang mempunyai akurasi atau hasil yang terbaik dan nanti nya arsitektur tersebut menjadi sebuah model untuk direkomendasikan untuk melakukan evaluasi atau pengujian data.

8. **Evaluasi**

Model yang telah terbentuk dari proses training, akan di evaluasi dengan menggunakan data testing untuk mendapatkan nilai akurasi menggunakan Confusion Matrix. Penjelesan pada confusion matrix pada BAB II sub bab 2.3.3

9. **Kesimpulan**

Setelah melakukan proses evaluasi dari skenario yang telah ditentukan sebelumnya, proses ini akan menyajikan hasil dari penelitian. Hasil penelitian

berupa data fakta yang dihasilkan oleh akurasi dari confusion matrix terkait dengan Arsitektur dari algoritma Naïve Bayes. Penjelasan pada evaluasi dapat di lihat pada BAB V Penutup sub bab 5.1 Kesimpulan

3.5 Modeling Proses Analisis data

Proses Analisis Data dilakukan untuk memberikan suatu gambaran bagaimana penelitian ini akan dilakukan Model ini menggambarkan alur bagaimana dataset yang diambil pada webset kaggel harus melalui beberapa tahapan yang dilakukan, yaitu pengelompokan data melalui n*preprocessing*, *tokenization*, *feature extraction*, kemudian dilakukan proses klasifikasi pada masing-masing logaritama dalam hal ini adalah *Naïve Bayes* serta pengujian model klasifikasi hingga menghasilkan *output confusion matrix*. Dari metode klasifikasi tersebut selanjutnya dilakukan perbandingan nilai *accuracy*, *precision*, *recall*. Hasil pengujian dari proses klasifikasi dataset tersebut menghasilkan output *confusion matrix*. Gambar perancangan model dapat dilihat pada gambar 3.2

Gambar 3.2 Proses Analisis Data

Sumber : (Dinata, 2019)

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

Pada bab ini akan dijelaskan tentang analisa dan implementasi dari rancangan tahapan alur penelitian. Terdapat beberapa tahapan proses mulai dari pengambilan data, skenario penelitian, *preprocessing*, *Case Folding*, *feature extraction*, Tokenisasi, kemudian klasifikasi dengan menggunakan metode *Naïve Bayes).*

4.1 Dataset

4.1.1 Pengumpulan data

Pada penilitian ini dataset yang akan gunakan berasal dari website bernama *kaggle* yang merupakan situs dan platform untuk menganalisa dan memprediksi suatu dataset dan dilakukan pengunduhan secara manual melalui link (https://www.kaggle.com/datasets/syedsaqlainhussain/sql-injection-dataset) kenapa penelitian ini mengambil dataset dari website tersebut. Pada dataset plain ini merupakan kumpulan data yang terdiri dari kombinasi URL, karakter khusus, data tekstual dan data numerik. Dataset ini terdiri dari kalimat teks biasa yang merupakan Non serangan atau disebut SQL yang berisi 3060 baris. Kemudian yang kedua yaitu dataset SQLI atau dataset yang merupakan serangan dataset ini berisi sekitar 1127 baris. Dataset disimpan di dalam *google drive*, karena proses klasifikasinya menggunakan *google colab.* Dibawah ini pada gambar 4.1 memperlihatkan hasil screnshoot contoh isi dataset yang diambil dari web *Kaggle*.

Gambar 4.1 Dataset dari kaggle

4.1.2 Preprocessing Data

Pada tahap ini, proses diawali dengan melakukan load data dari *google drive* ke dalam *google colab.* Tahap ini memiliki tujuan untuk mengolah data yang belum siap menjadi data yang suda siap untuk masuk kedalam pembentukan model. Kemudian proses *preprocessing* pada penelitian ini memiliki tiga proses diantaranya *Case Folding, Stopwords Removal*, dan *Tokenisasi*

4.1.2.1 *Case Folding*

Tahap *Preprocessing* yang pertama adalah *Case Folding*. *Case folding* adalah proses untuk menyetarakan seluruh kata menjadi sama atau setiap huruf besar atau *uppercase* menjadi huruf kecil atau *lowercase.* Berikut perintah untuk prose dari *case folding*. Dan Hasil dapat di lihat pada tabel

```
df['Sentence'] = df['Sentence'].str.lower()
print('Case Folding Result : \n')
print(df['Sentence'].head(5))
print('\n\n\n')
```

Tabel 4.1 Hasil Proses *Case folding*

Sebelum			Sesudah		
0	a	1	0	a	1
1	a'	1	1	a'	1
2	a' --	1	2	a' --	1
3	a' or 1 = 1; --	1	3	a' or 1 = 1; --	1
4	@	1	4	@	1
...	...	...	...	...	...
4195	org/?option = com_k2 <a href = "http://corfopym	0	4195	org/?option = com_k2 <a href = "http://corfopym	0
4196	com/?option = com_k2 <act> <![CDATA[procMemb...	0	4196	com/?option = com_k2 <act> <![cdata[procmemb...	0
4197	picsearch	0	4197	picsearch	0
4198	com/is?-WIk-uhyLerKNiYLvAbWl3W4oh5f749n r2KUmFF...	0	4198	com/is?-wik-uhylerkniylvabkl3w4oh5f749n r2kumff...	0
4199	de]]> </email_address> <find_account answer...	0	4199	de]]> </email_address> <find_account answer...	0
4200 rows × 2 columns			4187 rows × 2 columns		

Pada tabel 4.1 diperlihatkan hasil *case folding* sebelum dan sesudah proses penyetaraan setiap huruf Kapital di ubah menjadi huruf kecil. Dalam tabel sebelum proses pada data ke 4196 terdata kata "CDATA" kemudian setelah melalui proses penyetaraan pada tabel sesudah menjadi "cdata".

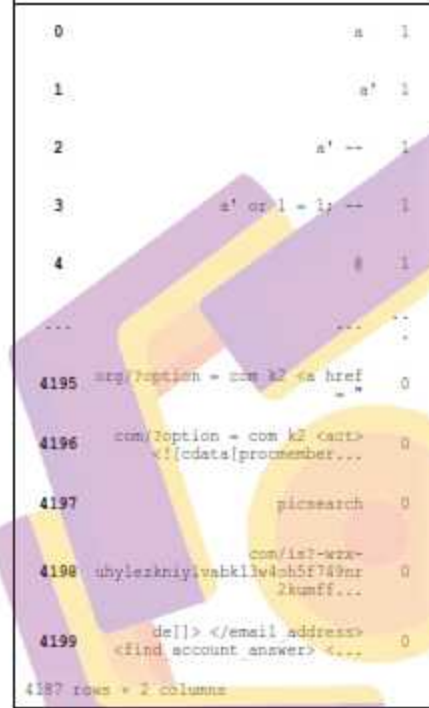
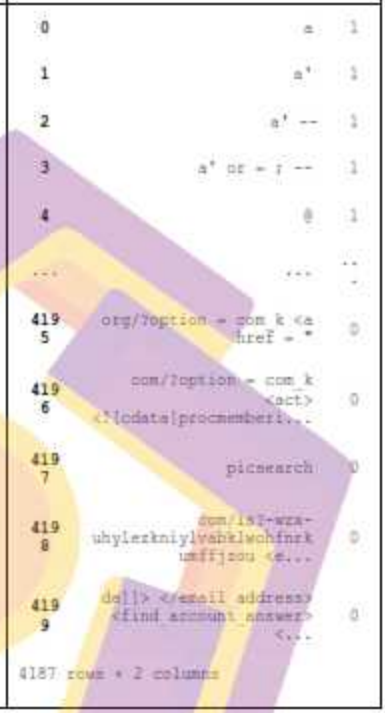
4.1.2.2 *Stopword Removal*

Tahap Selanjutnya yaitu *Stopword Removal*. *Stopword removal* adalah proses menyingkirkan sebuah kata dan yang dianggap tidak memiliki keterkaitan pada suatu kalimat atau kurang penting. Kata-kata yang dihilangkan yaitu: - penghubung antar kata, seperti: dan, atau, serta - preposisi, seperti: di, ke, pada - kata-kata yang tidak diinginkan. Berikut beberapa perintah untuk melakukan proses *stopword removal*. Dibawah ini perintah dan contoh menghapus angka :

```
def remove_number(text):
    return  re.sub(r"\d+", "", text)

df['Sentence'] = df['Sentence'].apply(remove_number)
```

Tabel 4.2 Hasil Pengapusan Angka

Sebelum	Sesudah
	

Pada tabel diperlihatkan proses penghapusan angka. Pada gambar pertama terdapat pada data ke 3 terdapat text " a' or 1=1;" kemudian setelah proses penghapusan angka dapat di lihat pada kolom sesudah data ke 3 berubah menjadi "a'or =;" angka yang sebelumnya sudah ada menjadi terhapus.

Perintah menghapus tanda baca

```
def remove_punctuation(text):
    return text.translate(str.maketrans("","",string.punctuation))
df['Sentence'] = df['Sentence'].apply(remove_punctuation)
```

Tabel 4.3 Hasil Penghapusan Tanda Baca

Sebelum			Sesudah		
0	a	1	0	a	1
1	a'	1	1	a	1
2	a' --	1	2	a	1
3	a' or 1=1 --	1	3	a or	1
4	#	1	4		1
...	...	...	...	...	...
4195	otg/?option = com k <a href = "	0	4195	orgoption comk a href	0
4196	com/?option = com k <act> <![cdata[procmemberi...	0	4196	comoption comk act cdataprocmemberinsert act ...	0
4197	picsearch	0	4197	picsearch	0
4198	com/is?-wzx-uhylezkniylvabklwohfnrk umffjzou <e...	0	4198	comiswzxuhylezkniylvabklwohfn zkumffjzou emaila...	0
4199	de]]> </email_address> <find_account_answer> <...	0	4199	de emailaddress findaccountanswer cdatafqaepqd0...	0
4187 rows × 2 columns			4187 rows × 2 columns		

Pada Tabel memperlihatkan proses penghapusan tanda baca yang tidak digunakan. Terlihat pada tabel sebelum banyak tanda baca yang masih terdapat pada setiap text kemudian setelah proses penghapusan tanda baca dapat di lihat pada tabel sesudah semua tanda baca sudah terhapus dan tidak terlihat tanda baca lagi.

Perintah menghapus 1 huruf yang tidak dalam bentuk kata

```
def remove singl char(text):
    return re.sub(r"\b[a-zA-Z]\b", "", text)
df['Sentence'] = df['Sentence'].apply(remove singl char)
```

Tabel 4.4 Hasil Penghapusan Single Char

	Sebelum		Sesudah	
0	a	1	0	1
1	a	1	1	1
2	a	1	2	1
3	a or	1	3	or 1
4		1	4	1
...	...	...	...	...
4195	orgoption comk a href	0	4195	orgoption comk href 0
4196	comoption comk act cdataprocmemberinsert act a...	0	4196	comoption comk act cdataprocmemberinsert act a... 0
4197	picsearch	0	4197	picsearch 0
4198	comiswrxuhylerkniylvabklwoh fnrkumffjzou emaila...	0	4198	comiswrxuhylerkniylvabklwoh fnrkumffjzou emaila... 0
4199	de emailaddress findaccountanswer cdatafqaepgd...	0	4199	de emailaddress findaccountanswer cdatafqaepgf... 0
4187 rows × 2 columns			4187 rows × 2 columns	

Tabel memperlihatkan pada kolom sebelum masih terdapat 1 huruf yang tidak ada makna kata data o sampai dengan data ke 4 terlihat masih ada huruf "a" yang tidak dalam bentuk kata. Kemudian setelah dilakukan proses penghapusan *single char* pada data ke 0 sampai dengan data ke 3 sudah terhapus.

4.1.2.3 *Tokenizing*

Pada tahap ini yaitu *Tokenizing* atau bisa juga disebut *parsing*. *Tokenizing* adalah proses pemotongan dokumen menjadi bagian-bagian kata yang disebut *token*. Spasi

digunakan untuk memisahkan antar kata tersebut. Sedangkan kata-kata yang tidak dibutuhkan akan dihilangkan melalui proses *filtering* dari hasil *tokenizing*. Berikut perintah untuk proses *tokenizing* dan hasil dapat di lihat pada gambar 4.2

```
def word_tokenize_wrapper(text):
    return word_tokenize(text)
df['Sentence_tokens'] = df['Sentence'].apply(word_tokenize_wrapper)
```

Tabel 4.5 Hasil proses Tokenizing

	Sentence		Sentence_tokens
3	or	1	[or]
6	and union all	1	[and, union, all]
7	or	1	[or]
8	and userid is null	1	[and, userid, is, null]
9	and email is null	1	[and, email, is, null]
...	...	...	...
4195	orgoption comk href	0	[orgoption, comk, href]
4196	comoption comk act cdataprocmemberinsert act a...	0	[comoption, comk, act, cdataprocmemberinsert, ...
4197	picsearch	0	[picsearch]
4198	comiswrxuhylezkniylvabklwohfnrk umffjzou emaila...	0	[comiswrxuhylezkniylvabklwohfnrku mffjzou, emai...
4199	de emailaddress findaccountanswer cdatafqsepqd...	0	[de, emailaddress, findaccountanswer, cdatafqa...

4007 rows × 3 columns

pada tabel 4.2 memperlihatkan prose *filtering* dimulai dengan pemotongan dokumen menjadi bentuk kata dengan di tandai tanda baca koma (,) mulai data ke 3 sampai dengan data ke 4199 dokumen yang ada pisahkan dalam bentuk kata dengan menggunakan koma, contoh pada data ke 8 tertulis " and userid is null"

kemudia dilakukan proses filtering menjadi " [and, userid, is, null]. Sedangkan kata yang tidak dibutuhkan juga akan terhapus seperti huruf "a" pada data ke 0 sampai dengan data ke 2.

4.1.3 *Feature Extraction*

Setelah proses *Preprocessing* selanjutnya adalah *Feature Extraction*. Dalam hal ini *Feature Extraction* berguna dalam mengenali karekteristik dalam query pada dataset dan menjadikan sebagai fitur. Dalam penggunaan machine learning fitur-fitur digunakan untuk menghasilkan performasi yang baik sehingga feature extraction dianggap penting. Pada penelitian ini dilakukan *feature extraction* menggunakan TF-IDF kemudian data dan label disimpan kedalam variabel dalam bentuk *array*.

4.1.3.1 TF-IDF

Setelah melalui tahapan *Tokenizing* pada *preprocessing*, selanjutnya diberikan pembobotan pada token yang ada pada dokumen dengan menggunakan metode TF-IDF. TF-IDF terdiri (TF) *Term Frequency* dan (IDF) *Inverse Dokumen Frequency*. Nilai *term* yang tinggi pada sebuah dokumen akan berpengaruh terhadap nilai yang didapat. Sedangkan *Inverse Documen Frequency* (IDF) merupakan perhitungan dari *term* yang kemunculanya didistirbusikan pada dokumen yang terkait. Untuk memperoleh nilai IDF yang besar maka jumlah dokument yang mengandung term harus sedikit. Beikut perintah untuk melakukan proses TF-IDF.

Perintah Pembototan TF

```
def calc_TF(document):
    # Counts the number of times the word appears in review
    TF_dict = {}
    for term in document:
        if term in TF_dict:
            TF_dict[term] += 1
        else:
            TF_dict[term] = 1
    # Computes tf for each word
    for term in TF_dict:
        TF_dict[term] = TF_dict[term] / len(document)
    return TF_dict

df["TF_dict"] = df['Sentence_tokens'].apply(calc_TF)

df["TF_dict"].head()
```

Berikut Hasil Proses TF

Tabel 4.6 Hasil Proses TF

```
term    TF

                and     0.2857142857142857
utlinaddrgethostaddress         0.14285714285714285
             select     0.14285714285714285
     sysdatabasename    0.14285714285714285
               from     0.14285714285714285
               dual     0.14285714285714285
```

Pada tabel 4.6 memperlihatkan hasil pencarian term dokumen pada data training karena dokument yang sebelumnya masih random kemudian diperlihatkan frekuensi jumlah kemunculan kata pada sebuah dataset yang sudah dilakukan proses preprocessing. Karena panjang dari setiap dokumen bisa berbeda-beda, maka umumnya nilai TF ini dibagi dengan panjang dokumen (jumlah seluruh kata

pada dokumen). pada proses TF juga, dokumen sudah tersusun dari dokumen pertama hingga dokumen terakhir

Perintah pembobotan DF

```
def calc DF(tfDict):
    count DF = {}
    # Run through each document's tf dictionary and increment countDict's (term,
 doc) pair
    for document in tfDict:
        for term in document:
            if term in count DF:
                count DF[term] += 1
            else:
                count DF[term] = 1
    return count DF
DF = calc DF(df["TF dict"])
```

Berikut Hasil DF

```
DF

{'or': 587,
 'and': 111,
 'union': 279,
 'all': 21,
 'userid': 8,
 'is': 18,
 'null': 36,
 'email': 9,
 'anything': 5,
 'select': 566,
 'count': 11,
 'from': 521,
 'tabname': 3,
 'membersemail': 3,
 'fullname': 3,
 'like': 52,
 'bob': 3,
 'exec': 28,
 'masterxpcmdshell': 11,
 'ping': 5,
 'username': 17,
 'dbo': 2,
 'sp': 3,
```

Gambar 4.2 Hasil Proses DF

Pada gambar 4.4 adalah hasil selanjutnya dalam mencari DF, dimana DF merupakan dokumen yang mengandung term, dimana pada total merupakan jumlah dokument yang mengandung term tersebut.

Perintah pembobotan IDF

```
n_document = len(df)

def calc_IDF(__n_document, __DF):
    IDF_Dict = {}
    for term in __DF:
        IDF_Dict[term] = np.log(__n_document / (__DF[term] + 1))
    return IDF_Dict

#Stores the idf dictionary
IDF = calc_IDF(n_document, DF)
```

Berikut Hasil IDF

```
IDF

{'or': 1.9100711627375177,
 'and': 3.57729923434105,
 'union': 2.681888587466895,
 'all': 5.204755057277829,
 'userid': 6.098573533209925,
 'is': 3.351359133469784,
 'null': 4.032230404506498,
 'email': 5.993213017542899,
 'anything': 6.584038641488880,
 'select': 1.955438806908393,
 'count': 5.818891460848144,
 'from': 2.0381385227535856,
 'tabname': 6.909503749516254,
 'membersemail': 6.909503749516254,
 'fullname': 6.909503749516254,
 'like': 4.325586197884023,
 'bob': 6.909503749516254,
 'exec': 4.9285822886406706,
 'masterxpcmdshell': 5.818891460848144,
 'ping': 6.584038641488880,
 'username': 5.48542635273998,
```

Gambar 4.3 Hasil proses IDF

Pada gambar 4.5 memperlihatkan hasil prose perhitungan IDF namun perlu di ketahui untuk menghitung nilai IDF ada perhitungan sebelumnya yaitu D/DF dimana notasi D menyatakan banyaknya dokumen kemudian dari banyaknya

dokumen dibagi dengan nilai DF per term yang dihasilkan. Setelah mendapakan nilai D/DF kemudian nilai tersebut di Log kan untuk mencari nilai dari IDF.

Perintah pembobotan TF-IDF

```
#calc TF-IDF
def calc_TF_IDF(TF):
    TF_IDF_Dict = {}
    #For each word in the review, we multiply its tf and its idf.
    for key in TF:
        TF_IDF_Dict[key] = TF[key] * IDF[key]
    return TF_IDF_Dict
#Stores the TF-IDF Series
df["TF-IDF_dict"] = df["TF_dict"].apply(calc_TF_IDF)
```

Berikut Hasil TF-IDF

Tabel 4.7 Hasil Proses TF-IDF

term	TF	TF-IDF
and	0.11764705882352941	0.4208587340401235
utlinaddrgethostaddress	0.058823529411764705	0.256704263406577
select	0.11764705882352941	0.2300516243421638
distinct	0.11764705882352941	0.5205408832621474
tablename	0.11764705882352941	0.6938708220985616
from	0.11764705882352941	0.2397800615004242
rownum	0.058823529411764705	0.2356081570286913
as	0.058823529411764705	0.2302218515271919
limit	0.11764705882352941	0.5307774678491628
sysalltables	0.058823529411764705	0.3525419422142411
where	0.058823529411764705	0.1227769461493852

Pada gambar adalah hasil selanjutnta dari nilai TF-IDF. Proses hasil Nilai TF-IDF pada gambar diatas didapat dari nilai TF per term per dokument kemudian di kalikan nilai dari IDF per term per dokumen dan jadilah nilai TF-IDF yang terlihat pada gambar tersebut

4.1.3.2 *Array*

Setelah semua data memiliki ukuran yang seragam, selanjutnya data dan label data disimpan kedalam variabel dalam bentuk *array*. Dengan kata lain hasil akhir dari proses ini adalah matriks dalam bentuk array multidimensi, agar dapat menjadi masukan pada algoritma *Naïve bayes* yang tersimpan dalam variabel x dan y. proses tersebut di lakukan dengan perintah sebagai berikut perintah dan hasil dapat dilihat pada gambar

```
x = np.array(df["TF_IDF_Vec"].tolist())
```

Berikut adalah hasil variabel dalam bentuk *array*

Tabel 4.8 Variabel *x*

```
array([[1.91907116, 0. , 0. , ..., 0. , 0. , 0. ],
       [0. , 0. , 0. , ..., 0. , 0. , 0. ],
       [1.91907116, 0. , 0. , ..., 0. , 0. , 0. ],
       ...,
       [0. , 0. , 0. , ..., 0. , 0. , 0. ],
       [0. , 0. , 0. , ..., 0. , 0. , 0. ],
       [0. , 0. , 0. , ..., 0. , 0. , 0. ]])
```

Tabel 4.9 Variabel *y*

```
188  1
634  1
3990 0
2040 0
439  1
..
3798 0
1347 0
482  1
1248 0
4180 0
Name: Label, Length: 2004, dtype: int64
```

4.2 Pengelompokan data

Setelah melakukan preprocessing dan *feature extraction* terhadap dataset selanjutnya dilakukan proses pengelompokan data dengan membagi dataset menjadi dua bagian yaitu data training: data test: dengan rasio 50% data training: 50% data tets, 70% data training:30% data test, 90% data training:10% data test.

Train_test_split yang disediakan oleh *Sklearn* merupakan *library Python* yang digunakan untuk membagi data secara acak dengan perintah sebagai berikut:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.5)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.1)
```

Dengan proses yang sudah dilakukan maka di hasilkan data pada rasio 50%: Jumlah data training: 2003 dan data test: 2004. Pada rasio 30%: jumlah data training: 2804 dan data test: 1203. Pada rasio 10%: jumlah data training: 3606 dan data test: 401

4.3 Analisis Data

Dataset yang digunakan pada penelitian ini merupakan plan query dan query SQL Injection yang berjumlah 4187 data yang terdiri dari 3060 plan query dan 1127 query SQL Injection, kemudian data tersebut diolah melalui proses *preprocesing* yang bertujuan untuk menyiapkan data agar bisa melakukan klasifikasi, yang kemudian dilanjutkan pada proses feature extraction yang bertujuan untuk melakukan pembobotan pada ciri atau karakteristik tertentu pada dokument, setelah itu data dan label data disimpan kedalam variabel dalam bentuk array multidimensi supaya menjadi masukan pada algoritma *Naïve bayes*. Setelah mendapatkan hasil dari setiap data pada proses klasifikasi kemudian hitung hasil nilai akurasi dengan

menggunakan data test yang jumlahnya berbeda-beda dalam hal ini dilakukan dengan menggunakan confusion matrix.

4.3.1 Analisi Hasil berdasarkan Algoritma *Naïve Bayes*

Pembahasan ini peneliti memberikan contoh perhitungan metode *Naïve Bayes* dengan menggunakan dataset sederhana, Kemudian dibuat sebuah tabel yang digunakan sebagai dataset. Berikut Tabel dataset yang sudah ditentukan.

Tabel 4.10 Sampel Dataset

Data Test	Data Training					
	K1	K2	K3	K4	K5	K6
G1	1	0	1	0	0	0
G2	0	1	0	0	0	1
G3	0	0	1	0	0	0
G4	0	1	0	1	1	1

Keterangan:

K = Data Training

G = Data Test

1 = Serangan

0 = Bukan Serangan

Contoh Kasus:

Misalnya serangan yang ada pada dataset ada dua kejadian pada data ke G1 dan Data G3: Berdasarkan kejadian serangan tersebut maka langkah perhitungannya adalah sebagai berikut

Langkah 1: menentukan serangan yang muncul berdasarkan tabel dataset

Berdasarkan data yang muncul G1 dan G3, maka bisa dilihat dari tabel dataset indikasi serangan yang akan di prediksi yaitu K1 dan K3. karena pada K1 terdapat G1 dan G3 yang bernilai 1 dan pada K3 terdapat G3 yang bernilai 1. Maka untuk tahap selanjutnya yang di hitung menggunakan algoritma naive bayes adalah menghitung nilai probabilitas data dari K1 dan K3.

Langkah 2: menghitung nilai probabilitas serangan dan non serangan.

Pada langkah 1 sudah di dapatkan indikasi serangan yang di prediksi berdasarkan data yang muncul, sesuai tabel datase. Langkah selanjutnya yaitu menghitung nilai probabilias dari masing-masing serangan dan non serangan

Rumus menghitung probailitas nilai **K1**

$$\text{Rumus Probabilitas K1} = \frac{\text{jumlah kemungkinan serangan yang muncul}}{\text{jumlah semua serangan}} = \frac{1}{6} = 0{,}16$$

Keterangan:

Angka 1 di dapatkan dari prediksi minimal serangan yang muncul

Angka 6 di dapatkan dari jumlah semua serangan yang ada pada dataset

Rumus menghitung probabilitas serangan yang muncul

$$\text{G1} = \frac{\text{jumlah kemungkinan serangan}}{\text{jumlah kemungkinan seranga yang sering muncul}} = \frac{1}{2} = 0{,}5$$

$$G3 = \frac{\text{jumlah kemungkinan serangan}}{\text{jumlah kemungkinan serangan yang sering muncul}} = \frac{0}{2} = 0$$

Keterangan:

Jumlah kemungkinan serangan = jumlah nilai G1/G3 yang muncul pada K1 di tabel dataset

Jumlah kemungkinan serangan yang sering muncul = serangan yang muncul dari dataset kemudian dalam perhitungan kali ini didapatkan 2 serangan yang muncul yaitu K1 dan K3.

Rumus menghitung probabilitas nilai K3

$$\text{Rumus Probabilitas K3} = \frac{\text{jumlah kemungkinan kerusakan yang muncul}}{\text{jumlah semua kerusakan}} = \frac{1}{6} = 0{,}16$$

Keterangan:

Angka 1 di dapatkan dari prediksi minimal kerusakan yang muncul

Angka 6 di dapatkan dari jumlah semua serangan yang ada pada tabel dataset

Rumus menghitung probabilitas non serangan yang muncul

$$G1 = \frac{\text{jumlah kemungkinan serangan}}{\text{jumlah kemungkinan serangan akbita data sering muncul}} = \frac{1}{2} = 0{,}5$$

$$G3 = \frac{\text{jumlah kemungkinan}}{\text{jumlah kemungkinan serangan akibat data sering mucul}} = \frac{1}{2} = 0{,}5$$

Keterangan:

Jumlah kemungkinan = jumlah non serangan G1/G3 yang muncul pada K3 di tabel keputusan.

Jumlah kemungkinan serangan akibat data sering muncul = serangan yang muncul yang di akibatkan sering munculnya serangan dalam perhitungan kali ini didapatkan 2 serangan yang muncul yaitu K1 dan K3.

Langkah 3: Menghitung nilai bayes berdasarkan probabilitas serangan dan Non Serangan

Dari nilai probabilitas diatas selanjutnya tahap perhitungan nilai bayes dengan rumus sebagai berikut,

menghitung Nilai **K1**

$$K(K1|G1) = \frac{[K(G1|K1) * K(K1)]}{[K(G1|K1) * K(K1) + K(G1|K3) * K(K3)]} = \frac{0,5 \times 0,16}{(0,5 \times 0,16) + (0,5 \times 0,16)} = \frac{0,08}{0,16} = 0,5$$

$$K(K1|G3) = \frac{[K(G3|K1) * K(K1)]}{[K(G3|K1) * K(G3|K3) * K(K3)]} = \frac{0 \times 0,16}{0 \times 0,16 + 0,5 \times 0,16} = \frac{0}{0,08} = 0$$

Total nilai dari K1 yaitu:

Total K1 $= K(K1|G1) + K(K1|G3)$

Total K1 $= 0.5 + 0 = 0.5$

Menghitung Nilai **K3**

$$K(K3|G1) = \frac{[K(G1|K3) * K(K3)]}{[K(G1|K1) * K(K1) + K(G1|K3) * K(K3)]} = \frac{0,5 \times 0,16}{0,5 \times 0,16 + 0,5 \times 0,16} = \frac{0,08}{0,16} = 0,5$$

$$K(K3|G3) = \frac{[K(G3|K3) * K(K3)]}{[K(G3|K1) * K(K1) + K(G3|K3) * K(K3)]} = \frac{0,5 \times 0,16}{0,5 \times 0,16 + 0,5 \times 0,16} = \frac{0,08}{0,16} = 0,5$$

Total nilai bayes dari K3 yaitu:

Total K3 = K (K3 | G1) + K(K3 | G3)

Total K3 = 0.5 + 0.5 = 1

Menjumlahkan hasil nilai bayes dari K1 dan K3

Hasil Total = Total Bayes K1 + Total Bayes K3

= 0.5 + 1

= 1.5

Langkah 4 : Menghitung presentase nilai prediksi serangan dan non serangan

Dari perhitungan hasil total didapatkan nilai 1.5 . Angka tersebut nantinya di gunakan sebagai pembagi masing-masing nilai bayes dari K1 dan K3 untuk di ketehaui presentasenya. Berikut ini adalah hasil yang didapatkan dari perhitungan tersebut.

$$\text{Kerusakan pada IC charger (K1)} = \frac{\text{Total Nilai K1}}{\text{Total hasil}} \text{ x } 100\% = \frac{0{,}5}{1{,}5} \text{ x } 100\% = 33{,}3\%$$

$$\text{Kerusakan pada resistor (K3)} = \frac{\text{Total Nilai K3}}{\text{Total hasil}} \text{ x } 100\% = \frac{1}{1{,}5} \text{x} 100\% = 66{,}6\%$$

Dari hasil presentase diatas maka didapatkan nilai presentase tertinggi terutama pada Nilai K3. Dengan demikian jika ada data yang sering muncul G1 dan G3. Maka data tersebut mengalami serangan K3

4.3.2 Skenario Percobaan

Untuk mendapatkan model klasifikasi yang diharapkan, maka scenario percobaan yang dibuat adalah dengan membandingkan data training dan dan data test perbandingan yang digunakan adalah 50% data training: 50% data test, 70% data training: 30% data tets, 90% data training: 10% data test, mengapa demikian karena tujuannya untuk mengetahui akurasi, jika data test lebih besar dibanding dengan data training maka yang terjadi nilai akurasi justru semakin kecil. Sebalik semakin

besar data training dibanding dengan data test maka prosentase hasil yang didapat pada akurasi semakin tinggi.

Tabel 4.11 Skenario Percobaan

NO	skenario	Jumlah Data training	Jumlah Data test	Akurasi	Precision	Recall
1.	S1	2003	2004	0,9575	0,97130	0,91811
2.	S2	2804	1203	0,9576	0,97139	0,92057
3.	S3	3606	401	0,9625	0,97572	0,92990

Dari tabel di atas memperlihatkan S1 dengan data training lebih kecil dengan data test nilai akurasi hanya 0,9575, sedangkan pada S2 dengan data test yang lebih kecil nilai akurasinya jauh lebih besar yaitu dengan nilai 0,9576. Maka selanjutnya dilakukan proses scenario ke tiga untuk lebih melihat sejauh mana jika data test semakin kecil seperti yang terlihat pada S3 dengan data test lebih kecil hasil yang di dapat adalah 0,9625 hanya terpaut beberapa angka di belakang koma namun tidak tidak mengurangi akurasi yang jauh lebih tingi jika menggunakan data test lebih kecil

4.3.2.1 Skenario Satu

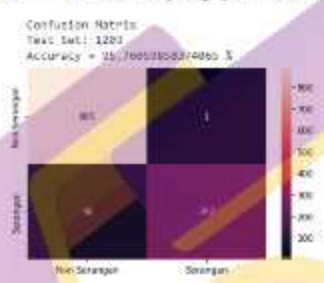
Berikut adalah hasil dari pengujian 2004 data uji terhadap 2003 data training

Gambar 4.4 Hasil Skenario 1

Berdasarkan hasil pada Gambar diatas tingkat akurasi pada pengujian dengan menggunakan 2004 Data uji sebesar 95, 7584%

4.3.2.2 Skenario Kedua

Berikut adalah hasil dari pengujian 1203 data uji terhadap 2804 data latih

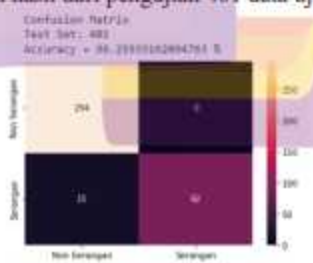


Gambar 4.5 Hasil Skenario 2

Berdasarkan hasil pada Gambar diatas tingkat akurasi pada pengujian dengan menggunakan 1203 Data uji sebesar 95, 7605%

4.3.2.3 Skenario Ketiga

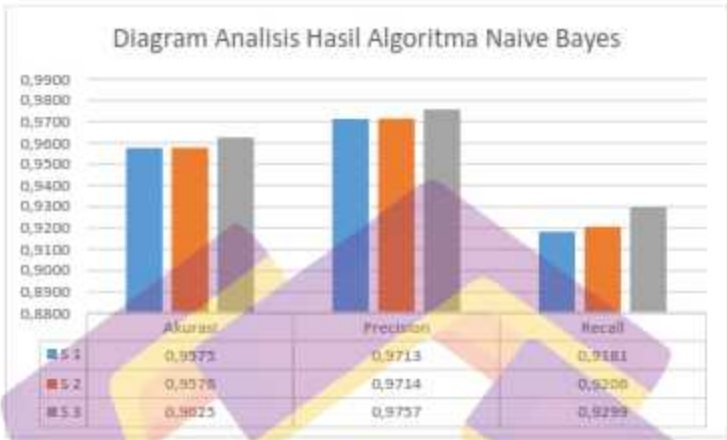
Berikut adalah hasil dari pengujian 401 data uji terhadap 3606 data latih



Gambar 4.5 Hasil Skenario 3

Berdasarkan hasil pada Gambar diatas tingkat akurasi pada pengujian dengan menggunakan 401 Data uji sebesar 96, 2593%

4.3.3 Evaluasi Hasil penelitian



Gambar 4.6 Grafik Evaluasi Hasil

Pada grafik di atas memperlihatkan nilai persentasi dari *accuracy,preccission* dan *recall* antara ketiga scenario. diketahui nilai *accuracy* dari algoritma *Naïve Bayes* cukup besar dengan nilai 0,9667 pada skenario 3. *Precision* pun nilai tertinggi terjadi pada scenario 3 dengan nilai 0,9757. Nilai *Recall* pun menunjukan angka yang tertinggi pada scenario ke 3 dengan nilai 0,9299.

4.3.4 Hasil Perbandingan dan Kontribusi Penelitian

Prediksi serangan *SQL Injection* dengan menggunakan metode *Naïve Bayes* sudah ada dalam beberapa literatur sejauh ini namun melalui penelitian ini penulis mencoba membandingan hasil akurasi dari masing-masing penelitian, dengan kata lain dalam hal ini peneliti mampu menaikan akurasi yang cukup tinggi jika dibanding dengan penelitian yang dilakukan oleh peniliti dimana dalam penelitian

ini menghasilkan akurasi sebesar 96, 2593%, berbeda dengan nilai akurasi yang dihasilkan pada penelitian(Hosam et al., 2021) sebesar 92,8% dan juga pada penelitian(Dinata, 2019) yang mendapatkan nilai akurasi sebesar 60%. Hal ini disebabkan perbedaan pada dataset yang digunakan pada penelitian ini yang menyebabkan kurangnya variasi terhadap pembentukan model.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan analisa hasil pada percobaan yang telah dilakukan terhadap 3 skenario percobaan dapat disimpulkan bahwa:

1. Dari percobaa yang dilakukan, performa yang dihasilkan dari 3 skenario menggunakan metode *Naïve Bayes* menghasilkan nilai Accurasi pada scenario 1 dengan jumlah data test 2004 adalah Akurasi 95.758, Precision 97.130%, Recall 91.811%. Pada scenario 2 dengan jumlah data test 1203 adalah Akurasi 95,760%, Precision 97.139%, Recall 92.057%. Pada scenario 3 dengan jumlah data test 401 adalah Akurasi 96, 2593%, Precision 97.572%, recall 92.990%.
2. Berdasarkan hasil pengujian yang sudah dilakukan dalam penelitian ini, maka bisa disimpulkan bahwa motode *Naïve Bayes* mampu diterapkan sebagai model dalam memprediksi serangan pada Sql Injection dengan scenario ke 3 dari jumlah data test 401 dan menghasilkan nilai akurasi 96,2593%
3. Dari hasil pengujian diperoleh nilai akurasi sebesar 96,2593%. Kemudian dapat di simpulkan bahwa metode *machine learning* dengan menggunakan model Naive Bayes dapat digunakan pada proses klasifikasi untuk terhadap SQL Injection

5.2. Saran

Berikut ini adalah beberapa saran yang dapat dijadikan pedoman untuk melakukan pengembangan penelitian ini, diantaranya adalah:

1. Penelitian ini hanya terbatas pada dataset yang di unduh di web *kaggel* sebaiknya pada penelitian selanjutnya dapat menggunakan dataset lain untuk mendapatkan hasil yang berbeda
2. Untuk meningkatkan nilai akurasi sebuah metode dapat dilakukan dengan Teknik diantaranya Teknik *bagging* dan *boosting*, karena penelitian ini hanya teknik *random sampling* dan belum menggunakan kedua teknik tersebut. selanjutnya dapat digunakan teknik *bagging* maupun *boosting* untuk peningkatan akurasi.
3. Sebaiknya proses training dilakukan menggunakan *library* dari *python* yang dijalankan secara lokal pada komputer. Karena jika menggunakan google colab waktu komputasi kurang tepat, karena hal tersebut sangat berpengaruh terhadap kecepatan internet pada setiap percobaan yang dilakukan.
4. Teknik mengubah jumlah rasio data training dan testing untuk menaikkan akurasi tidak bisa dilakukan sebagai bukti kenaikan akurasi. Sebaiknya yang harus dilakukan adalah merubah proses pada algoritmanya agar supaya akurasi yang dihasilkan pun bisa lebih maksimal.

DAFTAR PUSTAKA

Aliero, M. S., Ardo, A. A., Ghani, I., & Atiku, M. (2016). Classification of Sql Injection Detection And Prevention Measure. *IOSR Journal of Engineering (IOSRJEN) Www.Iosrjen.Org ISSN*, *06*, 1–06. www.iosrjen.org

Bekti Maryuni Susanto. (2013). Naïve Bayes Untuk Mendeteksi Gangguan Jaringan Komputer Dengan Seleksi Atribut Berbasis Korelasi. *Jurnal Bianglala Informatika*, *1*(2), 1–11. https://ejournal.bsi.ac.id/ejurnal/index.php/Bianglala/article/view/535

Chellam, A., Ramanathan, L., & Ramani, S. (2018). ScienceDirect ScienceDirect Intrusion Intrusion Detection Detection in in Computer Computer Networks Networks using using Lazy Lazy Learning Learning Algorithm Algorithm. *Procedia Computer Science*, *132*, 928–936. https://doi.org/10.1016/j.procs.2018.05.108

Dinata, R. (2019). *Implementasi Sistem Pendeteksi Serangan SQL Injection dengan Menggunakan Algoritme K-Nearest Neighbor*. *3*(12), 10984–10992. http://j-ptiik.ub.ac.id

Fibrianda, M. F., & Bhawiyuga, A. (2018). Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode Naïve Bayes Dan Support Vector Machine (SVM). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *II*(9), 3112–3123.

Hosam, E., Hosny, H., Ashraf, W., & Kaseb, A. S. (2021). SQL Injection Detection Using Machine Learning Techniques. *2021 8th International Conference on*

Soft Computing and Machine Intelligence, ISCMI 2021, *2*, 15–20. https://doi.org/10.1109/ISCMI53840.2021.9654820

Lika, S., Halim, R. D. P., & Verdian, I. (2018). Analisa Serangan Sql Injeksi Menggunakan Sqlmap. *POSITIF : Jurnal Sistem Dan Teknologi Informasi*, *4*(2), 88.

Maraj, A., Rogova, E., Jakupi, G., & Grajqevci, X. (2017). Testing techniques and analysis of SQL injection attacks. *2017 2nd International Conference on Knowledge Engineering and Applications, ICKEA 2017*, *2017-Janua*(January 2018), 55–59. https://doi.org/10.1109/ICKEA.2017.8169902

McWhirter, P. R., Kifayat, K., Shi, Q., & Askwith, B. (2018). SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel. *Journal of Information Security and Applications*, *40*(August), 199–216. https://doi.org/10.1016/j.jisa.2018.04.001

Olalere, M. (2018). A Naïve Bayes Based Pattern Recognition Model for Detection and Categorization of Structured Query Language Injection Attack. *International Journal of Cyber-Security and Digital Forensics*, *7*(2), 189–199. https://doi.org/10.17781/p002396

Pham, B. A., & Subburaj, V. H. (2020). An Experimental setup for Detecting SQLi Attacks using Machine Learning Algorithms. *Journal of The Colloquium for Information Systems Security Education*, *8*(1), 1–13. https://cisse.info/journal/index.php/cisse/article/view/124

Santoso, S., & Nurmalina, R. (2017). Perencanaan dan Pengembangan Aplikasi Absensi Mahasiswa Menggunakan Smart Card Guna Pengembangan Kampus

Cerdas (Studi Kasus Politeknik Negeri Tanah Laut). *Jurnal Integrasi*, *9*(1), 84–91.

https://dltsierra.medium.com/algoritma-tf-idf-633e17d10a80 diakses 12 april 2022

https://www.goldenfast.net/blog/flowchart-adalah/#Pengertian_Flowchart diakses 10 Juli 2022

https://geospasialis.com/algoritma-machine-learning diakses 12 Juli 2022

https://tambahpinter.com/prosedur-penelitian/ diakses 12 Juli 2022

http://doditsuprianto.blogspot.com/2020/04/klasifikasi-naive-bayes.html diakses 10 Juli 2022