Klasifikasi SQL Injection Menggunakan Algoritma Naïve Bayes

Erik Prasetyo1[1], Parman Sukarno2[2], Erwid Musthofa Jadied3[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]erikprasetyo@students.telkomuniversity.ac.id, [2]parmansukarno@telkomuniversity.ac.id,
[3]jadied@telkomuniversity.ac.id

Abstrak

SQL Injection adalah serangan yang melakukan percobaan untuk mendapatkan akses secara tidak sah ke database dengan cara menyuntikkan kode dan mengeksploitasi query SQL. 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. Pada penelitian ini telah merancang sebuah model pengklasifikasi untuk mendeteksi serangan SQL injection. Serangan yang diklasifikasikan diantaranya, union, tautology, dan blind. Metode yang digunakan dalam pengklasifikasian adalah machine learning dengan menggunakan model algoritmaNaive Bayes. Dari hasil pengujian di peroleh nilai akurasi sebesar 79,82%. Kemudian dapat di simpulkan bahwa metode machine learning dengan menggunakan model Naive Bayes berhasil melakukan mengklasifikasi terhadap SQL Injection.

Kata kunci :Klasifikasi naïve bayes, SQL injection

Abstract

SQL Injection is an assault that tries to benefit unauthorized get entry to a database via way of means of injecting code and exploiting SQL queries. SQL injection is an assault that is straightforward to perform however tough to locate and classify due to the sort of types. The SQLI vulnerability from incorrect person enter validation, which allowed attackers to govern programmer requests via way of means of including new SQL operators. In this study, a classifier version has been designed to locate SQL injection attacks. Attacks are categorized as union, tautology, and blind. The approach used withinside the class is device gaining knowledge of the usage of the Naive Bayes set of rules version. From the check effects acquired an accuracy cost of 79.82%. Then it may be concluded that the device gaining knowledge of approach the usage of the Naive Bayes version has succeeded in classifying SQL Injection.

Keywords: Naïve bayees classifier, SQL injection

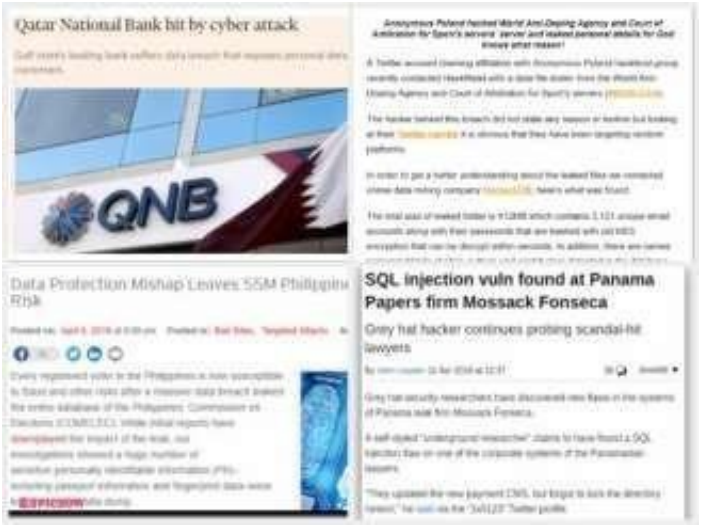
1. Pendahuluan

Latar Belakang

Teknologi Internet merupakan Infrastruktur informasi yang telah banyak digunakan pada kehidupan. Pemanfaatan dan kemajuan cepat dari infrastruktur Internet telah mendorong peningkatan jumlah data yang disimpan dalam database akhir-akhir ini [1].

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 [2]. 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 [2].

Terdapat contoh kasus besar yang diakibatkan oleh serangan *SQL Injection.* Contoh kasus besar tersebut dapat dilihat pada gambar 1.



Gambar 1. Contoh kasus SQL Injection (Sumber: Badan Siber Sandi Negara)

Contoh Kasus ketiga terjadi di bulan maret 2016, sekelompok *hacker* yang bernama *Mysterious Philippines* meretas situs web Komisi Pemilihan Umum Philippines (COMELEC) dan menghancurkan website tersebut, di hari yang sama kelompok *hacker LulzSec Philippines*, menyebarkan dokument yang berisikan informasi pribadi milik pemilih yang terdaftar di Filipina, informasi yang tersebar berupa informasi biometrik dan nomor paspor pemilih yang terdaftar di Filipina [3].

Kemudian yang terakhir contoh kasus keempat terjadi di bulan Juli 2015, sekelompok *hacker* menyerang *Qatar Public Bank* (QNB) yang menyebabkan terjadinya kebocoran data, data yang bocor berupa informasi pribadi milik nasabah *Qatar Public Bank*, selain itu terdapat informasi riwayat transaksi perbankan yang dimiliki oleh nasabah [3].

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 [4]. Pada penelitian ini telah merancang sebuah model pengklasifikasi untuk mendeteksi serangan SQL injection. Pengklasifikasian ini bertujuan untuk membantu developer website mengetahui bahwa website yang dibuat sering terinjection oleh serangan *SQL* jenis apa, dengan mengetahui serangan SQL Injection secara spesifik sehingga developer bisa mengetahui celah keamanan dari website yang dibuat dan bisa memberikan perhatian khusus terhadap celah tersebut agar nantinya website yang dibuat tidak terus terserang oleh SQL Injection.

Topik dan Batasannya

Permasalah yang akan di angkat dalam penelitian adalah melakukan penelitian berupa klasifikasi *SQL Injection.* Sistem yang dibangun akan membantu klasifikasi yang nantinya akan membantu developer mengetahui *query SQL* merupakan jenis serangan atau bukan.

Batasan dari penelitian ini hanya berfokus hanya serangan *Union*, *Blind*, dan juga *Tautology*. Kemudian dataset yang akan di gunakan (*SQL Injection* dataset) berasal dari Kaggle dikarenakan pada penelitian [4] menyebutkan bahwa sulitnya untuk mendapatkan dan mengumpulkan dataset menjadi sesuatu yang menantang pada penelitian ini di karenakan tidak adanya kumpulan data dengan akses ke publik untuk serangan *SQL Injection* yang tersedia. Pada paper [5] dengan judul *SQL Injection Detection Using Machine Learning* menggunakan data set yang berasal dari kaggle.

Tujuan

Tujuan dari penelitian ini yaitu mengimplementasikan algoritma *Naive Bayes* menjadi suatu metode dalam mengklasifikasikan terhadap *query* serangan *SQL Injection* dan *query plain*, dan juga melakukan analisis terhadap performasi dari sistem yang telah di bangun.

Organisasi Tulisan

Struktur penulisan pada paper ini akan dibagi menjadi 5 bagian. Pada bagian 1 akan menjelaskan latar belakang, batasan masalah dan tujuan dari paper ini. Kemudian bagian 2 akan menjelaskan penelitian sebelumnya terkait dengan penelitian ini. Metodologi penelitian akan dijelaskan pada bagian 3. Hasil dan evaluasi penelitian yang telah dilakukan dengan menggunakan metodelogi penelitian yang telah dilakukan akan dijelaskan pada bagian 4. Terakhir, kesimpulan dan saran akan dijelaskan pada bagian 5 pada paper ini.

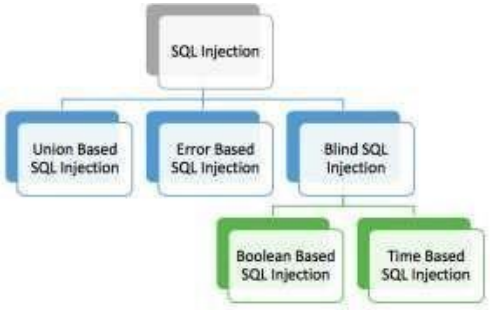
2. Studi Terkait

2.1 SQL Injection

SQL Injection adalah serangan yang melakukan percobaan untuk mendapatkan akses secara tidak sah ke database dengan cara menyuntikkan kode dan mengeksploitasi *query SQL* [4]. Contoh sederhananya adalah terdapat sebuah situs website perbankan yang memungkinkan user untuk masuk kedalam website dengan cara memasukkan nama user dan password user. Ketika user memasukkan nama dan password yang sesuai, otentikasiakan berhasil, dan user akan diizinkan untuk masuk.

2.2 Tipe SQL Injection

Kemudian SQL Injection secara luas dapat diklasifikasikan kedalam tiga kategori [4]. Kategori tersebut dapat dilihat pada gambar 2.



Gambar 2. Tipe SQL Injection (Sumber: Tesis SQL Injection Detection Using Machine Learning)

2.2.1 Tautology

Tujuan serangan: Melewati otentikasi, kemudian mengidentifikasi parameter yang dapat disuntikkan, lalu mengekstraksi data.

Deskripsi: Tujuan umum serangan berbasis tautologi adalah untuk menginjeksi kode dalam satu atau lebih pernyataan kondisional sehingga selalu mengevaluasi menjadi benar. Konsekuensi dari serangan ini tergantung pada bagaimana hasil *query* digunakan dalam aplikasi. Penggunaan yang paling umum adalah untuk melewati halaman otentikasi dan mengekstrak data. Di dalam jenis injeksi ini, penyerang mengeksploitasi bidang injeksi yang digunakan dalam kondisi *WHERE query*. Mengubah kondisional menjadi *tautologi* menyebabkan semua baris dalam tabel database ditargetkan oleh *query* yang akan dikembalikan. Secara umum, untuk serangan berbasis *tautology* untuk bekerja, penyerang harus mempertimbangkan tidak hanya parameter yang dapat disuntikkan/rentan, tetapi juga konstruksi pengkodean yang mengevaluasi hasil *query*. Biasanya, serangan berhasil ketika kode menampilkan semua catatan yang dikembalikan atau melakukan beberapa tindakan jika setidaknya satu catatan dikembalikan [6].

2.2.2 Union

Tujuan serangan*:* Melewati Otentikasi, kemudian mengekstraksi data.

Deskripsi*:* Dalam serangan *query* gabungan, penyerang mengeksploitasi indikator yang lemah untuk mengubah kumpulan data yang kemudian dibalikan untuk *queary* tertentu. Dengan teknik ini, penyerang dapat mengelabui aplikasi agar mengembalikan data dari tabel yang berbeda dari tabel yang dimaksudkan oleh pengembang. Penyerang melakukan ini dengan menyuntikkan pernyataan bentuk: *UNION SELECT* <sisa query yang disuntikkan>. Karena penyerang sepenuhnya mengontrol query kedua/yang disuntikkan, mereka dapat menggunakan query itu untuk mengambil informasi dari tabel tertentu. Hasil dari serangan ini adalah database mengembalikan kumpulan data itu adalah gabungan dari hasil kueri pertama asli dan hasil dari *query* kedua yangdisuntikkan [6].

2.2.3 Blind

Deskripsi*:* Banyak Web aplikasi yang menonaktifkan pesan untuk menampilkan mysql atau kesalahan sql lainnya. Dalam serangan ini informasi disimpulkan dengan bertanya pertanyaan benar/salah. Jika titik injeksi benar-benar blind maka satu-satunya cara untuk menyerang adalah dengan menggunakan *WAIT FOR DELAY* atauperintah *BENCHMARK* [7].

2.3 Penelitian Terkait

Studi klasifikasi *SQL Injection* menggunakan machine learning sudah banyak dilakukan dan biasanya pengklasifikasian berdasarkan *query, query* tersebut merupakan serangan atau bukan banyak algorima yang digunakan sebagai *classifier* diantaranya *Decision Tree, SVM, Neural Network, Random Forest, KNN,* dan *Naive Bayes.* Seperti penelitian [4]. didalam penelitian tersebut melakukan percobaan klasifikasi *SQL Injection* menggunakan *machine learning* dengan metode *Naive Bayes*, pada penelian tersebut menggunakan dua data set yang pertama data set bukan serangan berupa plain text yang terdiri dari kombinasi *URL*, karakter khusus, data tekstual dan data numerik, sedangan untuk data set yang merupakan serangan terdapat tiga jenis *SQL Injection* didalamnya yaitu *Union, Error Based* dan *Blind*. Kemudian pada penelitian ini menghasilkan tingkat *Akurasi* sebesar 92,8%.

Kemudian pada penelitian [8] memaparkan metode untuk mendeteksi *query SQL* berbahaya berdasarkan kombinasi dua pengklasifikasi yaitu *Naive Bayes* dan *RBC.* Pada dataset terdiri dari *SQL* berbahaya dan *SQL* tidak berbahaya, kemudian mereka membagi kueri menjadi elemen penting yang disebut token. Lalu, mereka menggunakan indeks token untuk diproses lebih lanjut mereka menyebut metode mereka sebagai metode tokenisasi. Dalam fase klasifikasi, sistem yang diusulkan membaca dataset dari file teks dan mengirimkan setiap data ke pengklasifikasi. Hasilnya menunjukkan bahwa pengklasifikasi yang diusulkan dapat mencapai *akurasi* 93,3%. Keterbatasan utama dari penelitian ini adalah keterbatasan dataset uji yang digunakan.

Kemudian pada penelitian [9] menggunakan *Naive Bayes* sebagai model untuk mendeteksi dan mengkategorisasi jenis serangan *SQL Injection* kemudian model yang diusulkan dilatih dan di evaluasi dengan 16.050 contoh dataset yang terdiri dari halaman web yang rentan dan tidak rentan kemudian pada penelitian ini menunjukan Akurasi pada deteksi dan pengkategorian masing-masing 98% dan 99%. Lalu pada penelitian [10] dibuatlah suatu sistem yang dapat mengidentifikasi pola serangan dan dapat belajar mendeteksi pola yang baru dari berbagai pola serangan yang terjadi. Tujuan dari penelitian ini ialah membuat sistem yang bertindak sebagai *proxy* untuk mencegah serangan *SQL Injection* menggunakan *Hybrid Metode* yang merupakan kombinasi dari *SQL Injection Free Secure* (*SQL-IF*) dan metode *Naive Bayes* . Pengujian dilakukan untuk mengetahui tingkat *akurasi*, pengaruh *konstanta* (K) pada *SQL-IF*, dan jumlah dataset pada *Naive Bayes*

pada *akurasi* dan efisiensi (waktu muat rata-rata) halaman web. Hasil pengujian menunjukkan bahwa Metode *Hybrid* dapat meningkatkan *accucarcy* pencegahan serangan *SQL Injection.* Jika nilai K lebih kecil dan lebih besar pada dataset kemudian akan menghasilkan yang lebih baik. Metode *Hybrid* menghasilkan waktu rata-rata yang lebih panjang untuk membuka halaman web dari pada hanya menggunakan metode *SQL-IF* atau *Naïve Bayes.* Kemudian Metode *Hybrid* yang merupakan integrasi dari *SQL Injection Free Secure* (*SQL-IF*) dan metode *Hybrid* dapat meningkatkan *akurasi* pencegahan serangan *SQL Injection* dengan nilai *akurasi* 90%. Nilai *akurasi* ini lebih tinggi dari nilai *akurasi* metode *SQL-IF* yang memiliki nilai 83% akan tetapi metode *Naive Bayes* mempunyai skor *Akurasi* yang jauh lebih baik dibandingkan oleh metode *SQL-IF* dengan mendapatkan nilai 88%.

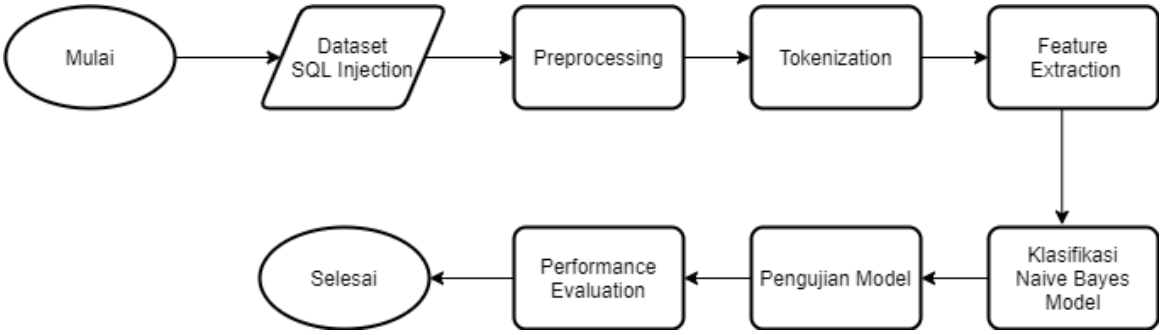
Kemudian pada penelitian [11] penelitian ini melakukan *Eksprimental* untuk mendeteksi serangan *SQL Injection* menggunakan *machine learning.* Penelitian ini menggunakan dataset yang terdiri dari kode *SQL* yang rentan dan aman, kemudian dataset di proses menggunakan *features extraction*, kemudian hasil dari klasifikasi digunakan untuk menentukan apakah itu termaksud kode rentan. Pada penelitian ini menggunakan lima algoritma *classifier,* kemudian didapatkan hasil pada *Naive Bayes Classifier* memperoleh *akurasi* sebesar 77%. Jauh lebih baik dan lebih besar dari metode *SVM* yang mendapatkan nilai *Akurasi* sebesar 57%.

2.4 Naive Bayes

Algoritma Naive Bayes merupakan sebuah algoritma klasifikasi yang memiliki nilai probabilitas serta statistik [12]. Kemudian penelitian ini dapat melakukan klasifikasi *SQL Injection* menggunakanalgoritma Naive Bayes. Alasan mengunakan *Naive Bayes* dikarenakan sudah banyak penelitian terkait yang menggunakan *Naive Bayes*, dan mempunyai performasi yang cukup baik. Kemudian algoritma Naive Bayes dipilih karena memiliki performasi yang baik dengan jumlah data latih minimal [13]. Kemudian Keunggulanlain dalam menggunakan Naive Bayes adalah hanya memerlukan jumlah data latihan yang minim yang bertujuan untuk menentukan index yang digunakan dalam proses klasifikasi [14]. menurut penelitian
[4] terdapat benefit menggunakan model Naive Bayes diantaranya yaitu, pertama dapat dilatih dengan dataset yang minim/kecil. Pada penelitian ini menggunakan data yang terbilang sedikit atau minim, karna pada pembelajaran Machine Learning jumlah dari data mempengaruhi hasil dari performasi yang dihasilkan. Alasan inilah yang menyebabkan memilih metode menggunakan Naive Bayes. Kemudian pada penelitian [4] dan [8] menyebutkan bahwa keterbatasan utama dalam penelitian ini terdapat pada ketersedian dataset.

3. Sistem yang Dibangun

Terdapat beberapa tahapan yang dilakukan dalam membangun rancangan sisitem ini, yaitu preprocessing, tokenization, feature extraction, model klasifikasi Naive Bayes, serta pengujian model. Gambar perancangan model dapat dilihat pada gambar 3.



Gambar 3. Perancangan Model

3.1 Dataset SQL Injection

Pada penelitian ini menggunakan dataset yang terdiri dari plan *SQL* dan *SQLI*. Kemudian datset di berikan tiga label yaitu label 0 menandakan bukan 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 yangberbahaya dan jenis yang tidak berbahaya. Contoh dataset dapat dilihat pada tabel 1.

Tabel 1. 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

Dataset yang akan gunakan pada penenlitian ini berasal dari website bernama kaggle kenapa penelitian ini mengambil dataset dari website tersebut. Dataset ini terdiri dari kalimat teks biasa dan berisi sekitar 3000 baris. Pada dataset plain ini merupakan kumpulan data yang terdiri dari kombinasi URL, karakter khusus, data tekstual dan data numerik. Kemudian yang kedua yaitu dataset *SQLI* atau dataset yang merupakan serangan dataset ini berisi sekitar 3000 baris *SQL Injection* yang berasal dari 3 jenis yaitu *Union*, *Tautology*, dan *Blind*.

3.2 Preprocessing

Preprocessing merupakan proses awal dalam melakukan pengolahan kata sebelum memasuki pemodelan[15].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*.

Case folding adalah proses untuk menyetarakan seluruh kata menjadi sama atau setiap huruf besar atau *uppercase* menjadi huruf kecil atau *lowercase*[15]. Hasil dari *case folding* dapat dilihat pada tabel 2.

Tabel 2. Case Folding

Sebelum	Sesudah
1" Waitfor delay '0:0:5'	1" waitfor delay '0:0:5'
To believe thought, believe true private heart true men, — genius	to believe thought, believe true private heart true men, — genius

Stopword removal adalah proses menyingkirkan sebuah kata yang dianggap tidak memiliki keterkaitan pada suatu kalimat atau kurang penting [15]. Hasil dari Stopwords removal dapat lihat pada tabel 3.

Tabel 3. Stopword Removal

Sebelum	Sesudah
to believe thought, believe true private heart true men, — genius	believe thought, believe true private heart true, — genius

Tokenization adalah proses untuk memecah *query* menjadi elemen bermakna yang disebut token[8]. Hasil Tokenization dapat dilihat pada tabel 4.

Tabel 4. Tokenization

Sesudah	Sebelum
believe thought, believe true private heart true, — genius	|,| |-|
1" waitfor delay '0:0:5'	|"| |waitfor| |'|

Setelah *tokenization* kemudian *feature extraction* dilakukan pada data menggunakan *TF-IDF*

3.3 Feature Extraction

Feature Extraction berguna dalam mengenali karekteristik dalam query pada dataset dan menjadikan sebagai fitur [16]. Dalam penggunaan machine learning fitur-fitur digunakan untuk menghasilkan performasi yang baik sehingga feature extraction dianggap penting [17]. Pada penelitian ini dilakukan feature extraction menggunakan TF-IDF.

3.3.1 TF-IDF

Setelah melalui tahapan tokenization pada tahap 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* [17]. Nilai term yang tinggi pada sebuah dokumen akan berpengaruh terhadap nilai yang didapat [18]. 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. TF-IDF dapat di rumuskan sebagai berikut [19]:

$$W_{ij} = TF_{ij} \; x \; IDF_j \qquad (1)$$

$$W_{ij} = TF_{ij} \; x \; log(D/df_j) \qquad (2)$$

Dimana *Wij* merupakan bobot term dari sebuah dokumen, *TFij* merupakan total kemunculan *term* pada dokumen, dan *IDFj* menyatakan untuk banyaknya dokumen yang mengandung *term*. D menyatakan banyaknya dokumen yang ada dalam data, sedangkan *df j* adalah dokumen yang mengandung term.

3.4 Naive Bayes

Naive Bayes merupakan algoritma pada machine learning dengan metode supervised learning[12]. Pada penelitian ini algoritma Naive Bayes bekerja menghitung probabilitas. Berikut alur kerja algoritma naiva bayes.

$$P(\blacksquare\blacksquare|\blacksquare\blacksquare) = \frac{P(\blacksquare|C) \cdot P(c)}{P(\blacksquare)} \quad (3)$$

Keterangan :

x : Data kelas yang belum diketahui
c : Hipotesis data dari class spifik
P(C|X) : Probabilitas berdasarkan kondisi (Postior Probability)
P(c) : Probabilitas Hipotesisi (Prior Probability)
P(x|c) : Probabilitas berdasarkan kondisi hipotesisi
P(x) : Probabilitas c

3.5 Perhitungan Persentase

Untuk menentukan persentase terkelasifikasinya *SQL Injection*, maka diperlukan perhitungan keberhasilan terkelasifikasinya data berdasarkan pengujian yang dilakukan. Dalam pengklasifikasian, data yang diuji akan di cocokan dengan data yang sudah menjadi model, dengan pengujian seperti itu maka akan mengetahui seberapa persentase SQL Injection yang berhasil di klasfikasikan. Persentase dapat di rumuskan sebagai berikut:

$$\blacksquare\blacksquare c\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare c\blacksquare = \frac{TP}{T\blacksquare\blacksquare + F\blacksquare\blacksquare + F\blacksquare\blacksquare + TN} \quad (4)$$

Dimana True Positives (TP) melambangkan data yang diprediksi memiliki kelas positif dan klasifikasi aslinya adalah kelas positif, True Negatives (TN) melambangkan data yang diperediksi memiliki kelas negatif dan klasifikasiaslinya adalah kelas negatif, False Positives (FP) melambangkan data yang diprediksi memiliki kelas positif namun sebenarnya pada klasifikasi aslinya merupakan data kelas negatif, sedangkan False Negative (FN) melambangkan datayang diprediksi memiliki kelas negatif namun pada klasifikasi aslinya merupakan kelas positif.

4. Evaluasi.

4.1 Skenario Pengujian Fungsionalitas

Pada rancangan yang di buat terhadap pengklasifikasian SQL Injection kemudian diimplementasikanpada suatu website dimana pada website tersebut memiliki empat menu diantaranya:

4.1.1 Preprocessing

Pada tampilan preprocessing ditampilkan keseluruhan proses preprocessing data yang akandigunakan pada percobaan ini, diantaranya:

1. Case Folding

Menampilkan data yang sudah melalui pemrosesan *case folding*, *case folding*, pada proses *case folding* melakukan pengubahan teks/query *uppercase*, *capitalize* menjadi *lower case* yangbertujuan menyetarakan semua data agar menjadi satu kesatuan sumber. Hasil pemprosesan casefolding dapat dilihat pada gambar 4.

Gambar 4. Case Folding

2. Stopwords Removal

Menampilkan data yang sudah melalui pemrosesan stopwords removal, pada proses ini menyingkirkan kata yang dianggap tidak terkait. Hasil pemprosesan *stopwords removal* dapat dilihat pada gambar 5.

Gambar 5. Stopwords Removal

3. Tokenisasi

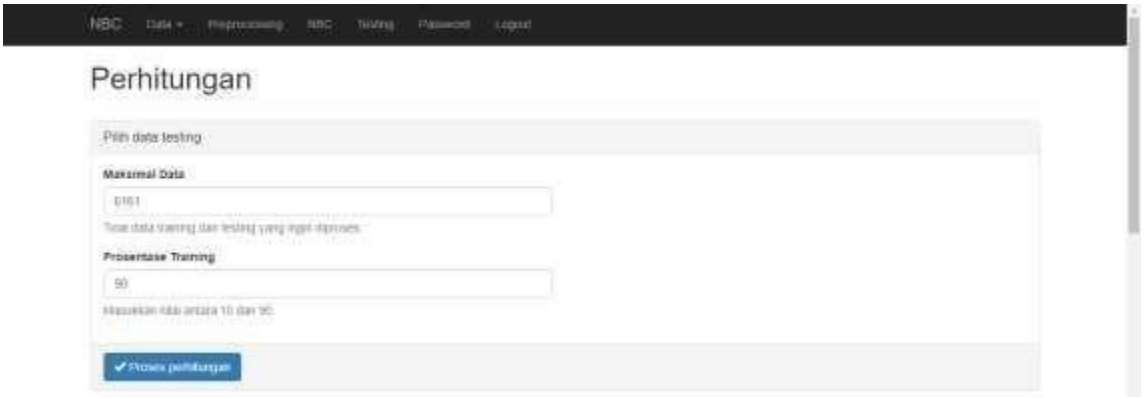
Menampilkan data yang sudah di proses tokenisasi, mamfaat dari tokenisasi ini sendiri yaitu memecah query menjadi elemen yang bermakna. Hasil pemprosesan *tokenisasi* dapat dilihat pada gambar 6.

Gambar 6. Tokenissasi

4.2 Skenario Pengujian Performasi

4.2.1 Perhitungan

Pada proses pengujian performasi klasifikasi menggunakan Naive Bayes, terdapat beberapa proses yang dilakukan diantranya:



Gambar 7. NBC

Pada proses ini user memasukan data yang akan digunakan dalam percobaan kemudian user juga memasukan pesentase dari data training yang akan digunakan dalam percobaan yang dimana persentase ini akan dibagi dengan jumlah data yang dimasukan sebelumnya, setelah memasukan jumlah data dan presentase training, user mengklik proses perhitungan dimana setelah proses perhitungan diklik maka data akan di split sesuai presentase yang sudah di tentukan, data yang masuk dalam perhitungan merupakan data yang dipilih secara random, alasan merandom data pada perhitungan adalah untuk menghindari data yang monoton dan mengakibatkan overfitting dan membuat akurasi menjadi menurun. Setelah mespit data kemudian data training memasuki proses perhitungan yaitu:

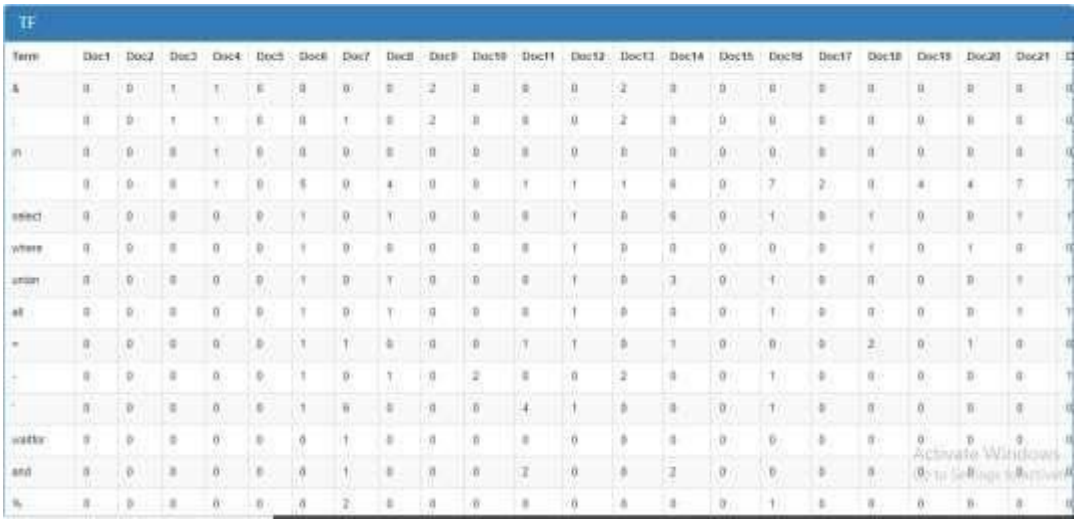
1. Kemunculan Term

Pada proses pertama pada Kemunculan term yaitu mencari term dari setiap dokumen data training, term ini didapat dari hasil tokenisasi sebelumnya. Hasil dari kemunculan term dapat dilihat pada gambar 8.

Gambar 8. Kemunculan Term

2. TF

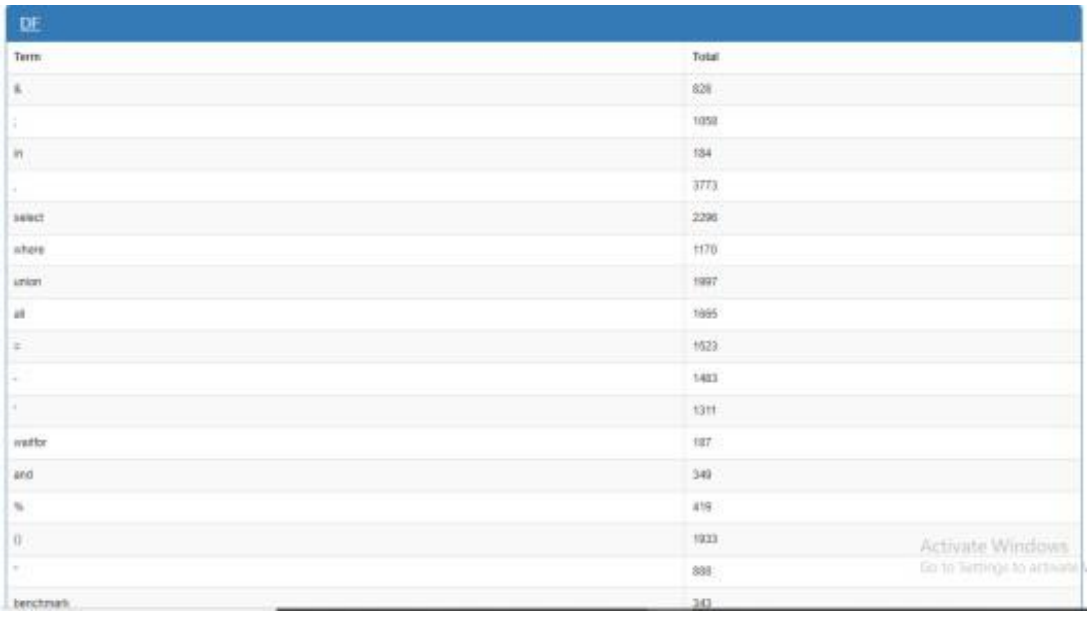
Langkah selanjutnya yaitu TF mencari term pada dokumen pada data training, dokument yang sebelumnya masih random, pada proses TF dokumen sudah tersusun dari dokumen pertama hingga dokumen terakhir. Hasil dari kemunculan term di setiap dokumen dapat dilihat pada gambar 9.



Gambar 9. TF

3. DF

Langkah selanjutnya yaitu mencari DF, dimana DF merupakan dokumen yang mengandung term, dimana pada total meruoakan jumlah dokument yang mengandung term tersebut. Hasil daridokumen yang mengandung term dapat dilihat pada gambar 10.


DF

Term	Total
&	828
;	1059
in	184
.	3773
select	2296
where	1170
union	1997
all	1995
=	1523
-	1483
'	1311
waitfor	187
and	349
%	419
()	1933
*	888
benchmark	343

Gambar 10. DF

4. IDF

Sebelum masuk pada perhitungan 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. Hasil dari nilai IDF dapat dilihat pada gambar 11


IDF

Term	Total
&	0.826
;	0.719
in	1.479
.	0.167
select	0.383
where	0.676
union	0.443
all	0.522
=	0.561
-	0.573
'	0.626
waitfor	1.472
and	1.201
%	1.122
()	0.458
*	0.795
benchmark	1.209

Gambar 11. IDF

5. TF-IDF

Kemudian langkah selanjutnya yaitu mencari nilai TF-IDF. Nilai TF-IDF didapat dari nilai TF per term per dokument di kalikan nilai dari IDF per term per dokumen, Hasil dari nilai TF-IDF dapat dilihat pada gambar 12.

Gambar 12. TF-IDF

6. Probabilitas

Kemudian langkah selanjutnya setelah mendapatkan nilai TF-IDF dimana outputan dari proses tersebut akan masuk dalam perhitungan probabilitas dimana nilai dari setiap term/token yang ada akan dihitung probabilitasnya menggunakan Naive Bayes. Hasil dari perhitungan probabilitas dapat dilihat pada gambar 13

Gambar 13. Tabel Probabilitas Data Training

Setelah mendapatkan nilai dari probabilitas dari setiap term/token beserta probabilitas dari setiap jenis label yang di dapat. Nilai tersebut akan dilanjutkan pada proses pembuatan model dengan Naive Bayes.

7. Data Testing

Setelah mendapatkan model dari data train menggunakan Naive Bayes, selanjutnya model tersebut akan di uji dengan data testing, akan tetapi sebelum menguji data testing juga harus di cari nilai probabilitasnya, langkah pertama pada pemprosesan data testing yaitu mencari setiap term yang ada pada dokumen data testing, dokumen pada data testing terpilih secara random. Setelah mendapatkan term dari setiap dokumen pada data testing, kemudian data testing dilakukan proses yang sama pada training yaitu TF – DF – IDF – TF*IDF hingga mendapatkan nilai probabilitas.

Setelah mendapatkan nilai probabilitas dari setiap term/token yang didapat serta mendapatkan probabilitas dari label yang didapat. Kemudian nilai probabilitas tersebut akan menghasilkan model data testing akan tetapi model tersebut belum memiliki label, kemudian pola-pola dari datatesting akan di cek kesamaannya dengan model yang sudah didapat sebelumnya (data training) setelah mencocokan pola dari data testing dan training outputan dari proses tersebut adalah modelklasifikasi menggunakan Naive Bayes. Hasil dari klasifikasi dapat dilihat pada gambar 14.

Gambar 14. Hasil Klasifikasi Naive Bayes

8. Performasi

Setelah mendapatkan hasil dari model klasifikasi menggunakan Naive Bayes yang dicocokan/diklasifikasikan dengan label yang sebenarnya. Tahapan selanjutnya yaitu menghitung akurasi atau menganalisis dari performa yang dihasilkan. Hasil perhitungan akurasi dapat dilihat pada gambar 15.

Gambar 15. Hasil Performasi

9. Cross Validation

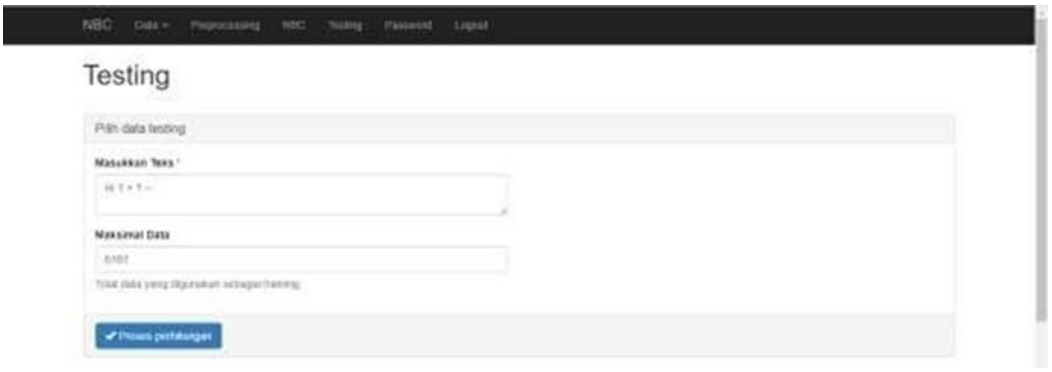
Pada gambar 15 menunjukan hasil akurasi terbaik yang dilakukan dalam 10 (fold) eksperimen pada cross validation. Setelah mendapatkan hasil dari setiap data fold kemudian hasil nilai akurasi dari setiap data fold dilakukan perhitungan rata-rata menggunakan mean. Untuk hasil akurasi lainya yang dilakukan dengan crossvalidation dapat dilihat pada tabel 6.

Tabel 6. Cross Validation

No	Data testing	Prediksi benar	Akurasi
Data 1	616	489	79,38%
Data 2	616	497	80,68%
Data 3	616	516	83,77%
Data 4	616	483	78,41%
Data 5	616	478	77,60%
Data 6	616	486	78,90%
Data 7	616	496	80,52%
Data 8	616	491	79,71%
Data 9	616	490	79,55%
Data 10	616	491	79,71%
Rata-Rata			79,82%

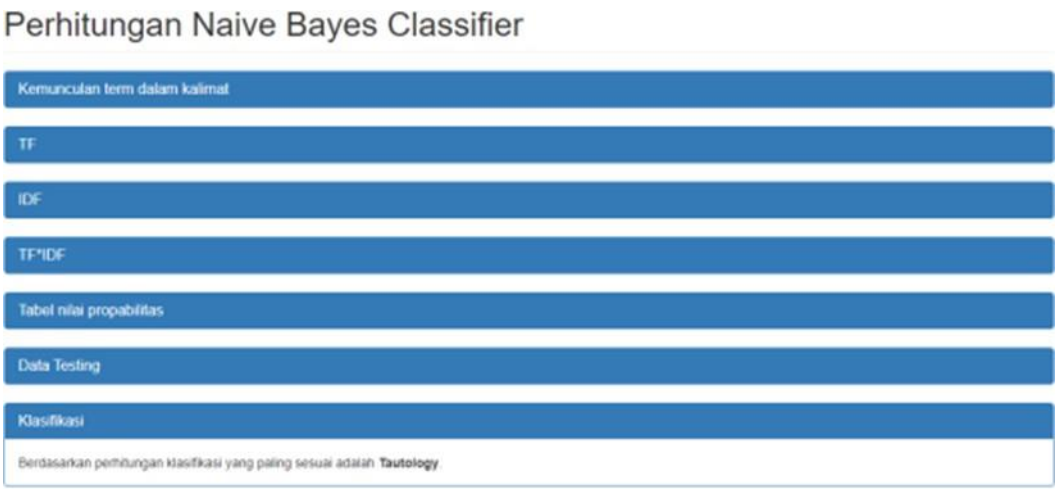
4.3 Skenario Pengujian Testing

dilakukan sebuah uji coba testing dengan menggunakan data yang sebenarnya, dikatakan data yang sebenarnya dikarenakan data secara langsung diinputkan oleh user yang ingin menguji. Seperti gambar 16.



Gambar 16. Testing Menggunakan Data Real

Setelah memasukan data berupa query/teks yang ingin diketahui query tersebut merupakan querydari SQL Injection jenis apa, user terlebih dahulu memasukan jumlah data yang ingin di jadikan data training, banyaknya data training yang dimasukan tentu saja akan mempengaruhi ketepatan dari klasifikasinya. Kemudian data training akan di proses hingga mendapatkan hasil dari probabilitasnya setelah mendapatkan hasil probabilitasnya, nilai dari probabilitas data training akan digunakan untuk membuat model yang nantinya model tersebut akan diuji dengan data testing. Data testing disini didapatkan dari inputan user sebelumnya.Setelah didapat model yang dihasilkan pada data training kemudian pada data testing dilakukan hal yang serupa untuk mencari nilai probabilitasnya, setelah mendapatkan nilai probabilitas dari data testing kemudian, model dari data training akan diuji dengan model yang dihasilkan oleh data testing. Outputan dari pengujian tersebut adalah jenis klasifikasi dari inputan data testing. Hasil klasifikasi dapat dilihat pada gambar 17.



Gambar 17. Klasifikasi Menggunakan Data Real

4.4 Hasil Pengujian dan Analisis Hasil pengujian

Dataset yang digunakan pada penelitian ini merupakan plan *query* dan *query SQL Injection* yang berjumlah 6161 data yang terdiri dari 3060 plan *query* dan 3101 *query SQL Injection*, Data pada serangan SQLI yang berjumlah 3101 dipecah menjadi 3 jenis serangan. kemudian data tersebut diolah melalui proses preprocesing yang bertujuan untuk menyiapkan data agar bisa melakukan klasifikasi, kemudian data dipecah menjadi 5544 data trainingdan 616 data test, yang kemudian dilanjutkan pada proses *feature extraction* yang bertujuan untuk melakukan pembobotan pada ciri atau karakteristik tertentu pada dokument, kemudian di lanjutkan untuk di klasifikasikan menggunakan *Naive Bayes.* Dilakukan cross validation untuk mengevaluasi kinerja model. Skenario pertama dilakukan untuk menguji fungsionalitas dari setiap fungsi yang ada, kemudian skenario kedua dilakukan untuk mendapatkan nilai akurasi dari model yang dibuat dimana model dibuat dengan metode *naive bayes* menggunakan 6161 data *SQL Injection*, dimana data dipecah menjadi

90% data latih dan 10% data uji yang kemudian saat pengujian dilakukan cross validation sebanyak 10 k (fold) dimana dari ke 10 eksperimen didapatkan hasil terbaik pada data 3 yaitu dengan akurasi sebesar 83,77%, dan hasil terburuk didapatkan pada data ke lime dengan akurasi sebesar 77,60%, setelah mendapatkan keseluruhan hasil akurasi darisetiap fold kemudian dilakukan perhitungan persentase dengan menggunakan mean, dimana jumlah akurasi yang didapatkan dijumlahkan kemudian dibagi dengan fold yang ada dan mendapatkan akurasi rata-rata sebesar 79,82%.

5. Kesimpulan

Berdasarkan hasil pengujian yang sudah dilakukan dalam penelitian ini, maka bisa disimpulkan bahwa motode *Naive Bayes* dapat mengklasifikasikan *SQL Injection* berdasarkan bukan serangan dan seranganya. Seranganya diantaranya, *Union*, *Tautology*, dan *Blind* dengan menghasilkan nilai akurasi sebesar 79,82%

Pada penelitian ini menghasilkan *akurasi* sebesar 79,82% pada pengklasifikasi *SQL Injection* menggunakan *Naive Bayes*. Jauh berbeda dengan nilai akurasi yang dihasilkan pada penelitian [4] sebesar 92,8% dan juga pada penelitian [8] yang mendapatkan nilai akurasi sebesar 93,3%. Hal ini disebabkan terbatasnya data yang digunakan pada penelitian ini yang menyebabkan kurangnya variasi terhadap pembentukan model. Pada dasarnya menggunakan metode *Machine Learning* membutuhkan banyak sumber daya atau banyaknya data yang digunakan, karna dari banyaknya data tersebut akan menghasilkan model yang jauh lebih baik sehingga ketika diuji dengan data yang baru model akan dengan baik mengenalinya sehingga dapat menghasilkan performasi yang baik.

Referensi

[1] Ali N S, dan Shibghatullah A S. 2016. Protection Web Application Using Real-Time Technique to Detect Structured Query Language Injection Attacks. International Journal of Computer Applications. 149:6 26 32.
[2] Alwan Z S, dan Younis M F. 2017. Detection and Prevention of SQL Injection Attack: A Survey. International Journal of Computer Science and Mobile Computing. 6:8 5-17.
[3] BSSN. 2018. Mengenal SQL Injection dan Cara Mencegahnya versi 1.3. Jakarta: BSSN.
[4] Mishra, S.. 2019. SQL Injection Detection Using Machine Learning. California: Spring.
[5] Krishnan, A S, Sabu A N, Sajan PP, dan Sreedeep A L. 2021. SQL Injection Detection Using Machine Learning. Geintec. 11:3 300-310.
[6] Halfond G J, Viegas J, dan Orso A. 2017. A Classification of SQL Injection Attacks and Countermeasure. Georgia: College of Computing Georgia Institute of Technology.
[7] Singh, J. P..2019. Analysis of SQL Injection Detection Techniques. Quebec: CIISE.
[8] Joshi A, dan Geetha V. 2014. SQL Injection Detection Using Machine Learning. International Conference on Control, Instrumentation, Communication and Computational Technologies. 1:2 1111-1115.
[9] Olalere M, Egigogo R A, Joseph O, dan Idris I. 2018. A Naïve Bayes Based Pattern Recognation Model for Detection and Categorization of Structured Query Language Injection Attack. International Journal of Cyber-Security and Digital Forensics. 7:2 189-199.
[10] Hernawan F Y, Hidayatulloh I, dan Adam I F. 2020. Hybrid Method Integrating SQL-IF and Naïve Bayes for SQL Injection Attack Avoidance. Journal of Engineering and Applied Technology. 1:2 85-96.
[11] Pham B, dan Subburaj V H. An Experimental Setup for Detecting SQLi Attacks Using Machine Learning Algorithms. Journal of The Colloqium for Information System Security Education. 8:1 1-5.
[12] Kaviani P, dan Dhotre S. 2017. Short Survey on Naïve Bayes Algorithm. International Journal of Advance Engineering and Research Development. 4:11 607-611.
[13] Devita R N, Herwanto W H, dan Wibawa A P. 2018. Perbandingan Kinerja Metode Naïve Bayes dan K-Nearest Neighbor untuk Klasifikasi Artikel Berbahasa Indonesia. Jurnal Teknologi Informasi dan Ilmu Komputer. 5:4 427-434.
[14] Saleh A. 2015. Implementasi Metode Klasifikasi Naïve Bayes dalam Memprediksi Besarnya Penggunaan Listrik Rumah Tangga. Citec Journal. 2:3 207-217.
[15] Syuriadi I K, Adiwijaya, Astuti W. Klasifikasi Teks Multi Label pada Hadis dalam Terjemahan Bahasa Indonesia Berdasarkan Anjuran, Larangan, dan Informasi menggunakan TF-IDFdan KNN. Bandung: Fakultas Informatika Universitas Telkom
[16] Cindo M, Rini P D, dan Ermatita. 2019. Studi Komparatif Metode Ekstraksi Fitur pada Analisis Sentimen Maskapai Penerbangan Menggunakan Support Vector Machine dan Maximum Entropy. Jurnal Rekayasa Sistem dan Teknologi Informasi. 3:3 402-407.

[17] Croft W B, Mertzler, dan Strohman T. 2015. Search Engines Information Retrieval in Practice. California: Pearson Education, Inc.
[18] Cahyanti E F, Adwijaya, dan Faraby S. 2020. Analisis Sentimen pada Review Film Menggunakan Support Vector Machine (SVM) dengan Ekstraksi Fitur Term Frequency – Inverse Document Frequency (TF-IDF) dan Latent Dirichlet Allocation (LDA). Bandung: Fakultas Informatika Universitas Telkom.
[19] Han J, Kamber M, dan Pei J. 2012. Data Mining Concept and Techniques 3rd Edition. Wyman Street: Simon Fraser University.