Klasifikasi Berdasarkan *Question* Dalam Stack Overflow Menggunakan Algoritma Naïve Bayes

Bagus Geriansyah Putra[1], Naim Rochmawati[2],

[1,2] Jurusan Teknik Informatika/Teknik Informatika, Universitas Negeri Surabaya

[1]bagusputra1@mhs.unesa.ac.id

[2]Naimrochmawati@unesa.ac.id

***Abstrak*—*Stackoverflow* merupakan sebuah website yang menyediakan banyak informasi tentang pemrograman. Pengguna dapat berinteraksi dengan pengguna lainnya dalam sebuah forum diskusi yang diajukan. Pengguna dapat mengajukan sebuah pertanyaan yang kemudian akan ditanggapi oleh pengguna lain. Ketika mengajukan sebuah pertanyaan, pengguna harus memasukkan kategori yang tepat pada pertanyaan yang diajukan agar mendapatkan respons atau jawaban yang sesuai. Berdasarkan beberapa kasus yang terjadi masih banyak pengguna *website* mengalami kebingungan ketika memilih kategori pertanyaan yang diajukan. Akibatnya, pertanyaan yang diajukan tidak mendapat respons yang tepat atau kurang sesuai. Sehingga, penelitian ini diajukan untuk membantu proses pengkategorian pertanyaan pada *website Stackoverflow*. Penelitian menggunakan Algoritma Naïve Bayes untuk memprediksi kategori pertanyaan yang diajukan. Pada penelitian ini dilakukan beberapa proses, dimulai dengan proses *input* dataset dilanjutkan dengan pembacaan *file* dataset. Kemudian dataset akan melalui *preprocessing* yang dilanjutkan dengan pembobotan dan proses ekstraksi fitur dengan Algoritma TF-IDF. Selanjutnya, data diproses menggunakan Algoritma Naïve Bayes yang akan menghasilkan kategori pertanyaan. Selanjutnya dilakukan proses evaluasi model untuk menentukan model terbaik yang akan digunakan untuk tampilan antarmuka aplikasi. Hasil yang didapat dari tahap evaluasi model dengan 4 kali percobaan menggunakan 10.000-40.000 data menghasilkan nilai akurasi, *precision, recall,* dan f1-*score* tertinggi sebesar 75%, 75%, 75% dan 74%. Dari hasil pengujian yang telah dilakukan Algoritma Naïve Bayes dapat digunakan sebagai klasifikasi text dan menghasilkan nilai yang cukup baik.**

Kata Kunci— text mining, Algoritma Naïve Bayes, stackoverflow, Algoritma TF-IDF

I. PENDAHULUAN

Pada zaman modern ini perkembangan teknologi berkembang pesat. Akibatnya, semakin mudah mendapatkan informasi yang dibutuhkan. Salah satunya adalah seperti pada *website stack overflow* yang menyediakan banyak informasi tentang pemrograman. *Stackoverflow* merupakan sebuah forum tanya jawab bagi para *programmer* yang memiliki fitur menarik, meliputi memberikan komentar dan jawaban pada halaman yang sama dari pertanyaan oleh pengguna lain [1]. Pada *website stack overflow* setiap pengguna dapat menanyakan sebuah permasalahan mengenai pemrograman misalnya mengenai *error code* yang ditemuinya. *Stack overflow* akan memberikan solusi terkait permasalahan tersebut. Selain itu, pengguna dapat memberikan masukan atau komentar pada pengguna lain yang sedang mengajukan pertanyaan di *website* ini.

Ketika mengajukan sebuah pertanyaan, pengguna diminta untuk memilih kategori bahasa pemrograman sesuai permasalahan yang terkait. Pemilihan kategori ini berfungsi untuk mempermudah pengguna lain dalam memberikan tanggapan atau komentar. Permasalahan utamanya adalah kurangnya mekanisme untuk menetukan validitas konten yang dibagikan oleh individu, seperti pemilihan kategori pertanyaan. Pemilihan kategori yang salah akan berakibat pada solusi atau jawaban yang kurang sesuai. Perlu diingat bahwa penting bagi pengguna untuk mengetahui kategori yang sesuai untuk menemukan jawaban yang benar. Setiap orang pasti menginginkan sebuah solusi yang terbaik untuk menyelesaikan permasalahan mereka dan itulah tantangannya.

Untuk mengatasi permasalahan tersebut diperlukan sebuah solusi agar dapat meminimalisir kesalahan. Sehingga pengguna *website* bisa memperoleh tanggapan atau solusi yang lebih akurat. Metode klasifikasi kategori dinilai lebih efektif dan efisien jika dibandingan dengan permasalahan perkiraan kategori oleh pengguna. Cara ini dianggap lebih menguntungkan karena pengguna tidak perlu memperkirakan kategori permasalahan yang dihadapi sehingga tanggapan yang dihasilkan juga lebih akurat.

Metode klasifikasi merupakan salah satu metode pengelompokkan data sesuai dengan ciri-ciri atau karakteristik data tersebut [2]. Metode klasifikasi yang diusulkan pada penelitian ini

adalah pengolahan teks atau *text mining*. *Text mining* merupakan salah satu proses untuk menemukan hubungan baru dari sebuah kata atau pola, dengan cara memilah sejumlah dokumen, menggunakan algoritma penalaran maupun teknik statistik dan matematika [3]. Umumnya, *text mining* akan mengambil ekstraksi atau fitur dari sebuah informasi yang berharga dalam sebuah dokumen, dengan menganalisis pola atau hubungan dari dokumen-dokumen yang ada.

Ada berbagai cara melakukan klasifikasi teks salah satunya menggunakan Algoritma Naïve Bayes. Algoritma Naïve bayes merupakan sebuah algoritma klasifikasi probabilistik yang menggunakan teorema Bayes, dengan menghitung probabilitas dari penjumlahan frekuensi dan kombinasi nilai dari sekumpulan data/dokumen yang ada. Algoritma ini mengasumsikan bahwa semua atribut yang diperoleh dari nilai dalam variabel kelas tidak memiliki hubungan atau keterikatan (independen) [4].

Naïve Bayes juga diasumsikan sebagai metode *classifier* yang diusulkan oleh seorang ilmuwan Inggris bernama Thomas Bayes. Pada penemuannya yang dikenal dengan Teorema Bayes, dia memprediksi peluang masa yang akan datang dengan menggunakan pengalaman masa lampau. Sehingga, Naïve Bayes dianggap sebagai metode klasifikasi menggunakan probabilitas dan ilmu statistik [5].

Penelitian mengenai *text* mining dengan Algoritma Naïve Bayes telah banyak dilakukan, salah satunya oleh Admaja, dkk. Admaja (2019) ingin mengetahui keefektifan Algoritma Naïve Bayes terhadap klasifikasi sumber belajar berbasis teks. Pada penelitian ini, Admaja menggunakan sebanyak 235 data yang terbagi sesuai kelasnya masing-masing. Pengujian dilakukan sebanyak 4 kali dengan skema pengujian berdasarkan pembagian datanya. Berdasarkan hasil pengujian, didapatkan akurasi tertinggi sebesar 81,48% pada data uji keempat sebanyak 120 data [6].

Penelitian lainnya dilakukan oleh Denny, dkk (2019), penelitian ini menggunakan Algoritma Naïve Bayes untuk klasifikasi berita lokal radar Malang. Pengujian dilakukan sebanyak 5 kali percobaan, dengan skema pengujian pembagian data uji dan data latih yang memiliki persentase tertentu. Dengan total keseluruhan jumlah data sebanyak 764 data 7 kategori, didapatkan nilai akurasi tertinggi sebesar 78,66% pada skema pengujian pertama [7].

Akhmad (2016) juga menguji keefektifan Algoritma Naïve Bayes dalam klasifikasi dokumen teks untuk mengidentifikasi konten/isi dari E-*Government*. Berdasarkan pengujian yang dilakukannya, teknik klasifikasi dengan Naïve Bayes dan ekstraksi fitur TF-IDF menghasilkan nilai yang pasti dan akurat. Dengan menggunakan data yang terdiri dari 222 dokumen ekonomi dan 260 dokumen politik serta 40 data *testing* didapatkan akurasi sebesar 85% [3].

Alfin (2020) melakukan pengujian menggunakan Algoritma Naïve Bayes dan TF-IDF dalam prediksi kepribadian *Myres-Briggs Type Indicator* menggunakan data dari pengguna twitter. Hasil yang didapatkan dengan 100 data yang digunakan yaitu nilai akurasi sebesar 71%, dengan prediksi kategori benar sejumlah 71 data dan prediksi kategori salah sejumlah 29 data [8].

Ada juga penelitian yang membahas tentang analisis kategori pertanyaan berbahasa indonesia menggunakan Algoritma *Support Vector Machine* pada *question and answering system*, dari penelitian tersebut mereka mendapatkan rata rata akurasi 97% dengan menggunakan 900 data pertanyaan [9].

Adapun penelitian yang menggunakan *stack overflow* telah dilakukan oleh peneliti yang lain diantaranya adalah:

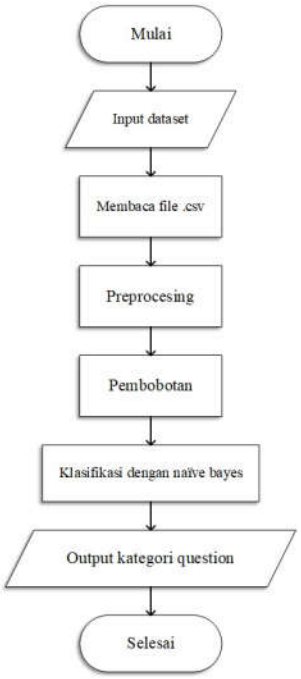
Kamel (2019) melakukan penelitian untuk memprediksi bahasa pemrograman pada *stack overflow*, penelitian tersebut menggunakan metode *Random Forest* dan XGBoost, pada penelitian tersebut mengunakan 232.727 data pertanyaan, dari penelitian yang dilakukan akurasi yang diperoleh adalah 78,9% [10].

Youshuai (2020) melakukan penelitian untuk memprediksi tingkat bug dengan acuan *stack overflow*, penelitian tersebut menggunakan metode *logistic regression*, dari penelitian tersebut mendapat peningkatan akurasi sebesar 23.03%, 21.86%, dari metode *k-Nearest Neighbor algorithm* (KNN), and *Long Short-Term Memory* (LSTM) [11].

Sedangkan penelitian ini bertujuan untuk mengetahui hasil klasifikasi *question* pada *stack overflow* menggunakan Algoritma Naïve Bayes dengan bantuan Algoritma TF-IDF dalam pembobotan fitur dengan menggunakan 10.000-40.000 dataset untuk mendapatkan akurasi terbaik serta mengevaluasi kualitas penggunaan Algoritma Naïve Bayes terhadap *Text Mining*.

II. METODOLOGI PENELITIAN

Penelitian ini menggunakan Algoritma probabilistik Naïve Bayes dengan bantuan Algoritma TF-IDF dalam pembobotan fitur. Dalam penelitian ini dataset akan dibagi menjadi 80% data *training* dan 20% data *testing* yang selanjutnya akan dilakukan *text processing*. Gambaran dari alur sistem klasifikasi dapat dilihat pada Gambar 1.



Gambar 1. Alur Diagram Penelitian

Gambar 1 menjelaskan tentang alur diagram penelitian menggunakan Naïve Bayes, yang di awali dengan *input* data, lalu dilanjutkan dengan pembacaan file data set yang berformat .csv, data tersebut akan melalui *preprocesing*, pembobotan serta klasifikasi sehingga menghasilkan kategori pertanyaan.

A. *Input* dataset

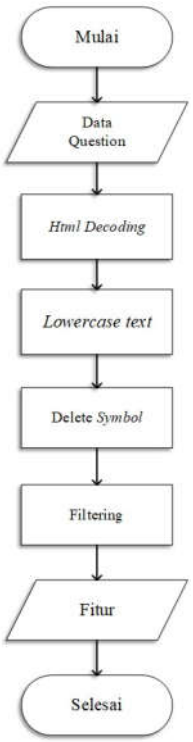
Dataset yang digunakan yaitu dataset pertanyaan yang disimpan dengan format .csv. dengan jumlah data sebanyak 40.000 data pertanyaan dengan menggunakan Bahasa inggris.

B. Membaca file dataset

Langkah selanjutnya adalah membaca file dataset yang memiliki tujuan mengambil isi data set tersebut yang berupa daftar pertanyaan yang akan menghasilkan fitur.

C. Preprocessing

Setelah data berhasil terbaca oleh program, selanjutnya sistem akan melakukan *prepocessing text*. Tahap ini merupakan awal dari klasifikasi di mana data pertanyaan akan melalui berbagai proses dan menghasilkan fitur yang akan digunakan untuk proses selanjutnya. Alur proses *prepocessing text* dapat dilihat pada Gambar 2.



Gambar 2 Alur *Prepocessing Text*

Gambar 2 menjelaskan tentang alur *prepocessing text*. Dataset yang telah dibaca akan diproses melalui *Html decoding,* dilanjutkan dengan *Lowercase text, Delete symbol,* dan filtering sehingga menghasilkan fitur.

1. Data *input* berupa data pertanyaan.
2. *Html Decoding* merupakan proses menghilangkan format file *html* pada teks pertanyaan.
3. *Lowercase text* merupakan proses mengubah seluruh kata yang menggunakan huruf kapital menjadi huruf kecil.
4. Menghapus *Symbol* merupakan proses menghilangkan seluruh *symbol* pada data pertanyaan.
5. *Filtering* merupakan proses mengambil kata penting dalam data pertanyaan. Proses *filtering* ini membutuhkan algoritma *stoplist* atau *wordlist*. *Stoplist* digunakan untuk menghilangkan kata yang dianggap kurang penting dalam dokumen. Sedangkan *wordlist* digunakan untuk menyimpan kata yang dianggap sebagai kata penting dalam dokumen. *Stopword* merupakan kumpulan kata yang lazim digunakan dan sering muncul pada dokumen namun tak bermakna, sehingga harus dihilangkan. Penggunaan *stopword* diperlukan agar informasi yang didapat lebih akurat karena data pertanyaan akan lebih fokus pada kata-kata penting.

D. Pembobotan (TF-IDF)

Pembobotan merupakan proses pemberian *weight*/bobot untuk setiap kata yang ada pada sebuah teks dilihat dari jumlah munculnya kata tersebut. Pada penelitian ini menggunakan algoritma pembobotan *TF-IDF* yang merupakan kepanjangan dari *Term Frequency – Inverse Document Frequency*. Algoritma TF-IDF merupakan algoritma pemberian bobot (*weight*) pada suatu term/kata [12].

Berikut merupakan alur proses pembobotan dengan algoritma TF-IDF.

1. Hitung nilai tf yaitu total kemunculan kata dalam sebuah dokumen.

2. Hitung nilai df yaitu keseluruhan dokumen yang mengandung kata tersebut. Jumlah dokumen yang digunakan dalam penelitian ini akan sangat memengaruhi nilai df yang dihasilkan.

3. Hitung nilai idf yaitu hasil inverse/kebalikan dari nilai df. Nilai idf didapatkan menggunakan rumus (1).

$$IDF_{(t)} = log_2 (N/df_t) \quad (1)$$

Dimana N merupakan total keseluruhan dokumen yang digunakan dalam penelitian.

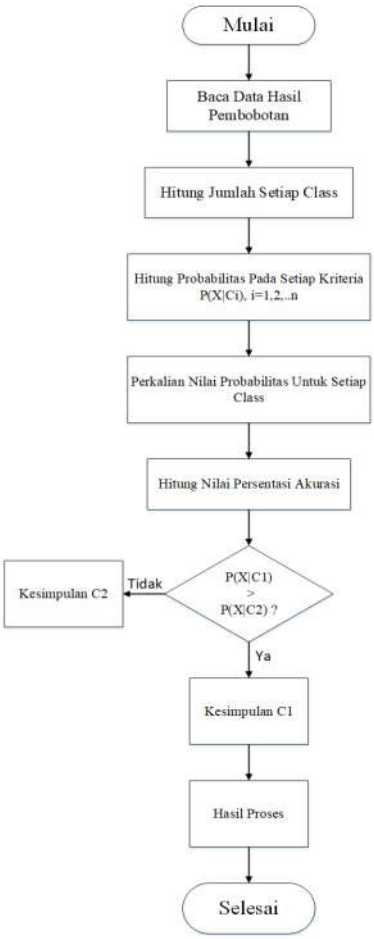
4. Hitung tf-idf untuk setiap kata dalam dokumen menggunakan rumus (2).

$$TF\text{-}IDF = tf * log_2 (N/df_t) \quad (2)$$

Sehingga pada proses TF-IDF ini akan didapatkan bobot untuk setiap kata yang diambil dari hasil perkalian nilai tf dan inverse dari idf.

E. Proses Klasifikasi dengan Naïve Bayes

Proses klasifikasi menggunakan algoritma *Naïve Bayes* dilakukan setelah tahapan *prepocessing text*. Gambaran proses klasifikasi dengan Algoritma Naïve Bayes dapat dilihat pada Gambar 3.



Gambar 3. Alur Klasifikasi Naïve Bayes

Gambar 3 menjelaskan tentang alur klasifikasi naïve bayes. Data hasil dari pembobotan akan di olah melalui beberapa proses sehingga mendapatkan hasil proses.

1. Masukkan data hasil pembobotan yang telah dilakukan *preprocessing* dan pembobotan *TF-IDF*.

2. Hitung probabilitas P(C) pada setiap *class* menggunakan rumus probabilitas seperti rumus (3).

$$P(C_i) = \frac{jumlah\ class\ i}{total\ data\ uji} \quad (3)$$

Dimana i adalah jumlah kategori dokumen.

3. Hitung Probabilitas $P(X|C_i)$ pada setiap kriteria menggunakan rumus (4).

$$P(X|C_i) = \frac{P(Ci|X)\ \ P(X)}{P(Ci)} \quad (4)$$

4. Kalikan nilai probabilitas untuk setiap *class*.
5. Hitung persentase akurasi tiap *class*.

6. Membandingkan hasil akurasi dari setiap *class*.
7. Memilih *class* dengan hasil akurasi yang lebih tinggi.

F. Proses Evaluasi Metode

Proses evaluasi metode dilakukan untuk mengetahui tingkat kinerja metode yang diusulkan terhadap studi kasus tersebut. Proses evaluasi ini menggunakan tabel *confusion* matriks untuk mengetahui nilai presisi, *recall* serta *F-1 Score* [9].

TABEL I
CONFUSION MATRIKS

		Actual Class	
		+	-
Prediksi Kelas	+	*True Positive*	*False Positive*
	-	*False Negative*	*True Negative*

Tabel I menunjukkan empat kriteria pengkategorian dokumen, yaitu :

1. *True Positive* (TP), merupakan hasil dari kelas prediksi positif dan sesuai dengan kelas sebenarnya positif.
2. *True Negative* (TN), merupakan hasil dari kelas prediksi negatif dan sesuai dengan kelas sebenarnya negatif.
3. *False Positive* (FP), merupakan hasil dari kelas prediksi positif namun kelas sebenarnya negatif.
4. *False Negative* (FN), merupakan hasil dari kelas prediksi negatif namun kelas sebenarnya positif.

Sesuai dengan empat kriteria pada Tabel I, proses evaluasi metode dapat menggunakan rumus berikut:

1. Akurasi adalah hasil perbandingan dari prediksi benar bernilai positif maupun negatif dari total keseluruhan data. Akurasi didapatkan dari rumus (5).

$$Akurasi = \frac{TP+TN}{TP+FP+FN+TN} \quad (5)$$

2. *Precision* adalah hasil perbandingan dari prediksi benar bernilai positif dengan hasil keseluruhan prediksi positif. *Precision* didapatkan dari rumus (6).

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

3. *Recall* adalah hasil perbandingan prediksi benar bernilai positif dengan data keseluruhan prediksi benar bernilai positif. *Recall* didapatkan dari rumus (7).

$$Recall = \frac{TP}{TP+FN} \quad (7)$$

4. F1-*Score* merupakan perhitungan kombinasi nilai yang dijadikan sebagai nilai pengukuran. Perhitungan F1-*Score* menggunakan rumus (8).

$$F1 - score = 2\ x\ \frac{Precission.Recall}{Precission+Recall} \quad (8)$$

III. HASIL DAN PEMBAHASAN

A. Dataset

Dataset yang digunakan dalam penelitian ini adalah data pertanyaan dari hasil crawling data situs https://stackoverflow.com/. Berikut merupakan rincian deskripsi data pertanyaan.

TABEL II
DESKRIPSI DATA PENELITIAN

No	Kategori	Training	Test	Total
1	.net	1592	408	2000
2	android	1604	396	2000
3	angularjs	1602	398	2000
4	asp.net	1605	395	2000
5	c	1605	395	2000
6	c#	1624	376	2000
7	c++	1598	402	2000
8	css	1593	407	2000
9	html	1576	424	2000
10	ios	1617	383	2000
11	iphone	1602	398	2000
12	java	1600	400	2000
13	javascript	1595	405	2000
14	jquery	1625	375	2000
15	mysql	1617	383	2000
16	objective-c	1605	395	2000
17	php	1603	397	2000
18	python	1582	418	2000
19	ruby-on-rails	1577	423	2000
20	sql	1578	422	2000
Total Data				40000

Data yang digunakan disimpan dalam bentuk file csv bernama *stackoverflow.csv*

B. Baca Data

Data pertanyaan yang telah dikonversi ke dalam bentuk file csv akan diolah dan menghasilkan *output* yang diinginkan. Contoh dataset yang siap digunakan dapat dilihat pada Gambar 4.

[2]:

	post	tags
0	what is causing this behavior in our c# datet...	c#
1	have dynamic html load as if it was in an ifra...	asp.net
2	how to convert a float value in to min:sec i ...	objective-c
3	.net framework 4 redistributable just wonderi...	.net
4	trying to calculate and print the mean and its...	python
...	...	...
39995	different output if at end of function rather ...	c++
39996	multiple arrays is there a way to access/stor...	iphone
39997	c - how to differentiate a second same key pre...	c
39998	state.go not working (#! & url is being append...	angularjs
39999	understanding the mechanisms of intentservice ...	android

40000 rows × 2 columns

Gambar 4. Hasil proses baca data

Gambar 4 menunjukan gambar dari dataset yang terdiri dari 40.000 data.

C. *Preprocessing* Data

Dataset yang digunakan pada proses klasifikasi akan dilakukan *preprocessing* terlebih dahulu. Pada tahap preprocessing ini, dataset akan melalui beberapa sub-tahap *preprocessing* diantaranya *HTML Decoding* (penghilangan tag HTML pada dataset), *Lowercase text* (proses pengubahan huruf besar menjadi huruf kecil), menghilangkan seluruh *symbol* pada data pertanyaan, dan *filtering* (proses mengambil kata penting dalam data). Contoh *preprocesing* ditunjukkan seperti ilustrasi pada Tabel II dan Tabel III.

TABEL IIII
ILUSTRASI SEBELUM PREPROCESSING DATA

Sebelum *Preprocessing* Data :
what is causing this behavior in our c# datetime type <pre><code>[test] public void sadness() { var datetime = datetime.utcnow; assert.that(datetime is.equalto(datetime.parse(datetime.tostring( )))); } </code></pre> failed : <pre><code> expected: 2011-10-31 06:12:44.000 but was: 2011-10-31 06:12:44.350 </code></pre> i wish to know what is happening behind the scenes in tostring() etc to cause this behavior. edit after seeing jon s answer : <pre><code>[test] public void newsadness() { var datetime = datetime.utcnow; assert.that(datetime is.equalto(datetime.parse(datetime.tostring(o))); } </code></pre> result : <pre><code>expected: 2011-10-31 12:03:04.161 but was: 2011-10-31 06:33:04.161 </code></pre> same result with capital and small o . i m reading up the docs but still unclear.

TABEL IVII
ILUSTRASI SETELAH PREPROCESSING DATA

Setelah *Preprocessing* Data :
causing behavior c# datetime type test public void sadness var datetime datetimeutcnow assertthat datetime isequalto datetimeparse datetimetostring failed expected 20111031 061244000 20111031 061244350 wish know happening behind scenes tostring etc cause behavior edit seeing jon answer test public void newsadness var datetime datetimeutcnow assertthat datetime isequalto datetimeparse datetimetostring result expected 20111031 120304161 20111031 063304161 result capital small reading docs still unclear

D. Pembobotan TF-IDF

Proses pembobotan TF-IDF merupakan proses pemberian bobot atau inisialisasi bobot pada setiap karakter teks dataset. Proses ini menghasilkan matriks bobot berupa fitur. Dari keseluruhan fitur yang dihasilkan ini akan dilakukan proses ektraksi fitur. Proses ektraksi fitur dilakukan untuk mendapatkan fitur penting dari keseluruhan fitur yang ada. Proses ini dilakukan agar proses komputasi pada Algoritma klasifikasi lebih cepat dengan hasil yang optimal. Proses ekstraksi fitur ini akan berpengaruh pada proses klasifikasi nantinya.

E. Klasifikasi Naïve Bayes

Aplikasi klasifikasi data pertanyaan *stack overflow* yang dibangun ini berfungsi untuk memudahkan pengguna *stackoverflow* untuk menentukan kategori pertanyaan yang diinputkan pada situs *stackoverflow*.

F. Evaluasi Metode

Tahap evaluasi metode ini digunakan untuk menguji kinerja algoritma Naïve Bayes dalam mengkategorikan data pertanyaan. Tahap evaluasi dilakukan sebanyak 4 kali percobaan dengan perbedaan jumlah data yang digunakan.

1. Percobaan 1
   Percobaan 1 dilakukan pemilihan data acak sebanyak 10.000 data. Total 10.000 data akan dibagi menjadi data *training* sebanyak 8000 dan *testing* sebanyak 2000 data.

2. Percobaan 2
   Percobaan 2 dilakukan dengan pemilihan data secara acak sebanyak 20.000 data. Total 20.000 data akan dibagi menjadi data *training* sebanyak 16000 dan *testing* sebanyak 4000 data.

3. Percobaan 3
   Percobaan 3 dilakukan dengan pemilihan data secara acak sebanyak 30.000 data. Total 30.000 data akan dibagi menjadi data *training* 24000 data dan *testing* sebanyak 6000 data.

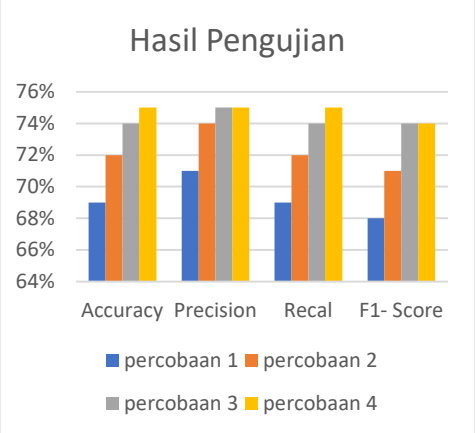
4. Percobaan 4
   Percobaan 4 menggunakan total data keseluruhan sebanyak 40.000 data. Total 40.000 data akan dibagi menjadi data *training* sebanyak 32000 data dan *testing* sebanyak 8000 data.

Hasil evaluasi model pada percobaan 1 hingga percobaan 4 dapat dilihat pada Tabel III.

TABEL VII
RINGKASAN HASIL EVALUASI MODEL

Percobaan ke-	*Accuracy*	*Precision*	*Recall*	*F1-score*
1	69%	71%	69%	68%
2	72%	74%	72%	71%
3	74%	75%	74%	74%
4	**75%**	**75%**	**75%**	**74%**

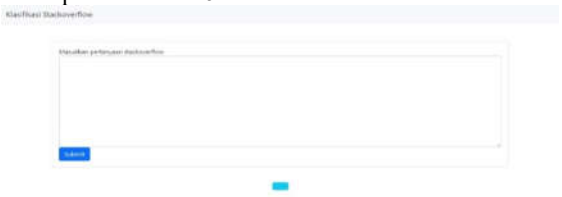
Berdasarkan hasil pengujian pada Tabel III, untuk mempermudah dalam melihat perbandingan hasil dari Percobaan pertama hingga percobaan keempat, maka dapat direpresentasikan secara visual melalui grafik pada Gambar 5.



Gambar 5. Grafik Hasil Pengujian

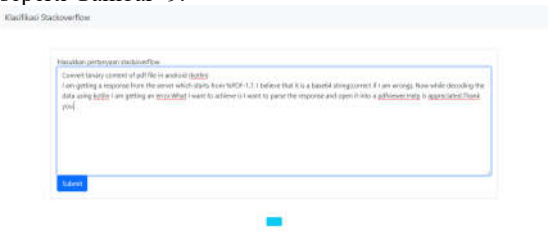
Berdasarkan hasil evaluasi model pada Tabel III dan Gambar 5, dapat dilihat bahwa hasil percobaan dengan perhitungan nilai akurasi, presisi, *recall* dan f1-*score* terbaik adalah pada Percobaan ke-4. Hasil percobaan ke-4 dengan nilai akurasi 75%, presisi 75%, *recall* 75%, dan f1-*score* 74% yang akan digunakan sebagai model algoritma untuk pembuatan tampllan antarmuka aplikasi.

G. Tampilan Antarmuka Aplikasi
   Penelitian ini menghasilkan sebuah aplikasi antarmuka berbasis website yang bernama Sistem Klasifikasi *Stackoverflow*. Tampilan awal aplikasi sistem ditunjukkan pada Gambar 6.



Gambar 6. Tampilan halaman *input* data pertanyaan

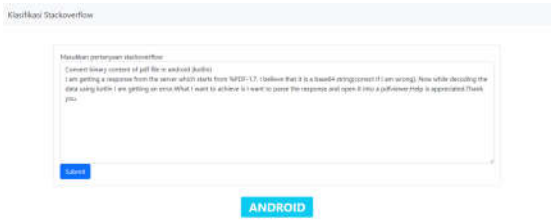
Pada Gambar 6 merupakan tampilan awal, di sini pengguna diminta untuk memasukkan data pertanyaan yang akan diprediksi kategorinya pada kolom "Masukkan pertanyaan *stackoverflow*". Selanjutnya, pengguna harus menekan tombol "submit" untuk melanjutkan proses prediksi kategori seperti yang ditunjukkan pada Gambar 7. Hasil prediksi kategori data pertanyaan akan ditampilkan seperti Gambar 9.



Gambar 7. Tampilan halaman setelah memasukan pertanyaan

Gambar 8. Tampilan halaman setelah memasukan pertanyaan

Gambar 7 dan Gambar 8 menunjukkan data pertanyaan yang telah dimasukan pada kotak yang sudah disediakan.



Gambar 9. Tampilan halaman hasil

Gambar 10. Tampilan halaman hasil

Gambar 9 dan Gambar 10 merupakan gambar yang menunjukkan hasil dari klasifikasi pertanyaan, Gambar 9 menunjukkan hasil klasifikasi dari pertanyaan Gambar 7 dan Gambar 10 menunjukkan hasil dari Gambar 8.

IV. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan dapat ditarik kesimpulan bahwa Sistem Klasifikasi Berdasarkan *Question* Dalam *Stack Overflow* Menggunakan Algoritma Naïve Bayes berhasil dilakukan dan menghasilkan tampilan antarmuka aplikasi berbasis *website*. Proses klasifikasi data pertanyaan dimulai dari *prepocessing text* dengan sub-proses diantaranya *HTML Decoding, Lowercase text,* menghapus *symbol*, dan *filtering* yang menggunakan bantuan Algoritma *Stopword.* Proses dilanjutkan dengan pembobotan TF-IDF yang menghasilkan matriks bobot untuk diproses di Algoritma Naïve Bayes. Sehingga menghasilkan *output* sistem berupa kategori pertanyaan. Selanjutnya, dilakukan proses evaluasi metode yang digunakan untuk mengetahui pengaruh jumlah data yang digunakan terhadap hasil pengujian yang didapatkan. Proses evaluasi metode terbagi menjadi 4 model percobaan. Dari keempat model percobaan didapatkan hasil terbaik adalah pada percobaan ke-4. Hasil percobaan ke-4 dengan nilai *accuracy* 75%, *precision* 75%, *recall* 75%, dan *f1-score* 74% yang digunakan sebagai model algoritma untuk pembuatan tampllan antarmuka aplikasi.

REFERENSI

[1] A. Diyanati, B. S. Sheykhahmadloo, S. M. Fakhrahmad, M. H. Sadredini and M. H. Diyanati, "A proposed approach to determining expertise level of StackOverflow programmers based on mining of user comments," *Journal of Computer Languages,* pp. 1-8, 2020.

[2] E. Setiani and W. Ce, "Text Classification Services using Naïve Bayes for Bahasa Indonesia," *International Conference on Information Management and Technology (ICIMTech),* pp. 361 - 366, 2018.

[3] A. P. Wijaya and H. A. Santoso, "Naïve Bayes Classification on Document Classification to Identify E-Government Content," *Journal of Applied Intelligent System,* vol. 1, pp. 48 - 55, 2016.

[4] J. Kolluri and S. Razia, "Text classification using Naïve Bayes classifier," *Materials Today,* pp. 1 - 4, 2020.

[5] N. Rochmawati and S. C. Wibawa, "Opinion Analysis on Rohingya using Twitter Data," *IOP Conf. Series: Materials Science and Engineering,* pp. 1-6, 2018.

[6] A. D. Herlambang and S. H. Wijoyo, "ALGORITMA NAÏVEBAYES UNTUK KLASIFIKASI SUMBER BELAJAR BERBASIS TEKS PADA MATA PELAJARAN PRODUKTIFDI SMK RUMPUN TEKNOLOGI INFORMASI DAN KOMUNIKASI," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK),* Vols. 6, No.4, pp. 431-435, 2019.

[7] D. N. Chandra, G. Indrawan and I. N. Sukajaya, "KLASIFIKASI BERITA LOKAL RADAR MALANG MENGGUNAKAN METODE NAÏVE BAYES DENGAN FITUR N-GRAM," *Jurnal Ilmu KomputerIndonesia (JIKI),* vol. 4, pp. 10 - 20, 2019.

[8] W. Alfin and C. Maya, "Klasifikasi Kepribadian Myres-Briggs Type Indicator Berdasarkan Cuitan di Twitter Menggunakan Metode TF-IDF dan Naive Bayes Classifier," *Jurnal Linguistik Komputasional (JLK),* vol. 3, pp. 48-53, 2020.

[9] S. Sakina, J. Hartarto and J. Santoso, "Analisis Jenis Pertanyaan Berbahasa Indonesia pada Question and Answering System Menggunakan Metode Support Vector Machine(SVM)," *Jurnal Ilmiah Teknologi Sistem Informasi,* pp. 1-8, 2018.

[10] A. Kamel, D. Dhanush, M. Daniel, S. Venkatesh and A. T, "SCC++ Predicting the programming language of questions and snippets of Stack Overflow," *The Journal of Systems and Software,* pp. 1-11, 2019.

[11] T. Youshuai, X. Sijie, W. Zhaowei, Z. Tao, X. Zhou and L. Xiapu, "Bug severity prediction using question-and-answer pairs from Stack Overflow," *The Journal of Systems and Software,* pp. 1-14, 2020.

[12] R. Melita, V. Amrizl, H. B. Suseno and T. Dirjam, "PENERAPAN METODE TERM FREQUENCY INVERSE DOCUMENT FREQUENCY (TF-IDF) DAN COSINE SIMILARITY PADA SISTEM TEMU KEMBALI INFORMASI UNTUK MENGETAHUI SYARAH HADITS BERBASIS WEB (STUDI KASUS: SYARAH UMDATIL AHKAM)," *JURNAL TEKNIK INFORMATIKA NO. 2,* vol. 11, pp. 149-265, 2018.