BAB IV
IMPLEMENTASI SISTEM

4.1. Implementasi Sistem

Implementasi sistem merupakan tahap di mana sistem telah selesai dibangun dan akan diimplementasikan kepada pengguna, sehingga sistem dapat diketahui kinerja dan kelayakannya pada sisi pengguna.

4.1.1 Implementasi *Download* Tweet

Download tweet merupakan tahap pertama yang dilakukan. Proses ini dilakukan dengan memanfaatkan Twitter API. Proses ini dibangun dengan menggunakan bahasa pemrograman *Python 3*. Pertama-tama dibutuhkan *consumer key*, *consumer secret*, *access key*, dan *access secret*, di mana keempat hal tersebut merupakan kunci untuk dapat mengakses Twitter API. Tanpa keempat hal tersebut, maka *developer* tidak dapat mengakses dan menggunakan Twitter API. Setelah mendapatkan 4 kunci di atas, selanjutnya adalah menginisialisasi *tweepy*. Hal ini merupakan proses untuk melakukan otorisasi terhadap Twitter API. *Twepy* merupakan *library* python yang digunakan untuk mengakses Twitter API.

Sebelum melakukan pengunduhan, terlebih dahulu sistem mengambil *list* kota dan *list* bencana yang sebelumnya sudah dimasukan ke dalam *database* .

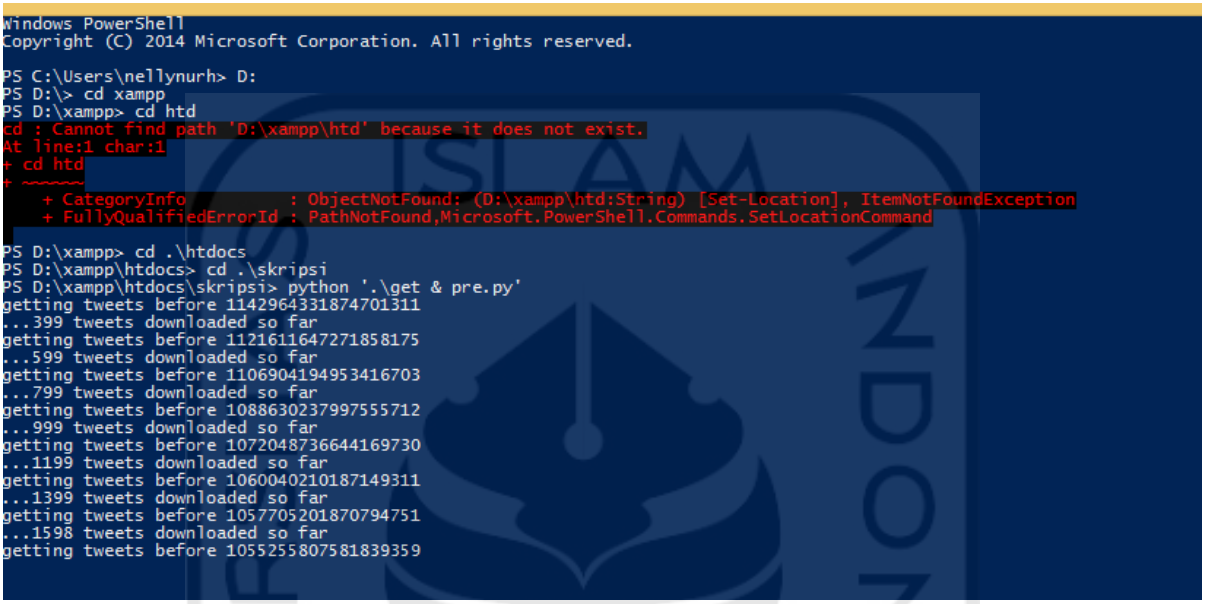
Setelah mengambil *list* kota dan bencana, selanjutnya sistem akan melakukan *download* tweet dan dilakukan secara *realtime* selama 1x24 jam. Data yang di *download* merupakan tweet dari akun BMKG dan BNPB pusat, yakni @info_BMKG dan @BNPD_Indonesia. Kode program implementasi *download tweet* dapat dilihat pada gambar 4.1.

```
while True :
    outtweets = get_all_tweets("@BNPB_Indonesia")
    labelling(outtweets)

    outtweets = get_all_tweets("@infoBMKG")
    labelling(outtweets)
```

Gambar 4.1 Kode program tahap *download* tweet

Proses *download* tweet menghasilkan kurang lebih 2000-3000 data dalam satu kali *download*. Hasil *download* tweet akan langsung tersimpan ke dalam *database*. Data yang ter*download* merupakan tweet-tweet dari akun BMKG dan BNPB pusat. Proses *running download* tweet dapat dilihat pada gambar 4.2



Gambar 4.2 Proses *running download* tweet

Setelah seluruh data ter*download* dan tesimpan, secara otomatis sistem akan melakukan proses pelabelan dan *preprocessing* secara otomatis. Hasil *download* data dapat dilihat pada gambar 4.3.

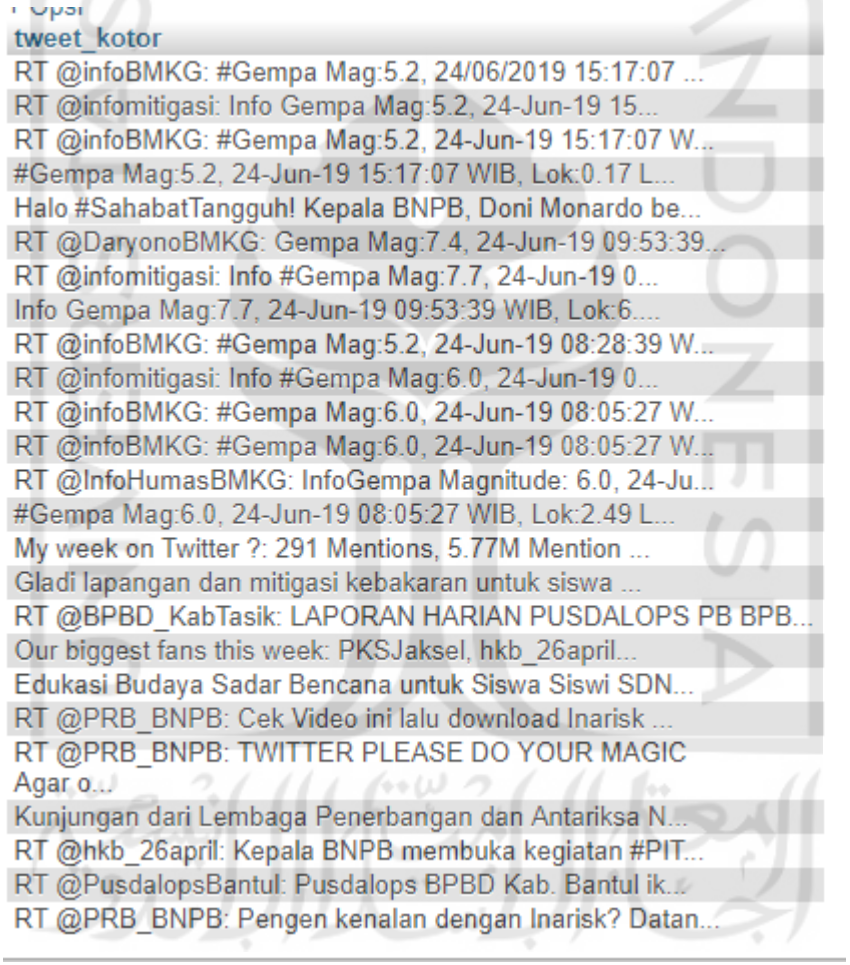
+ Opsi

	id	tweet	tweet_kotor	lokasi	label	jenis_bencana	date
Ubah Salin Hapus	1	gempa mag 5 2 24 06 2019 15 17 07 pusat gempa dila...	RT @infoBMKG: #Gempa Mag:5.2, 24/06/2019 15:17:07 ...	NULL	1	1	2019-06-24
Ubah Salin Hapus	2	info gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17...	RT @infomitigasi: Info Gempa Mag:5.2, 24-Jun-19 15...	NULL	1	1	2019-06-24
Ubah Salin Hapus	3	gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17 ls 1...	RT @infoBMKG: #Gempa Mag:5.2, 24-Jun-19 15:17:07 W...	NULL	1	1	2019-06-24
Ubah Salin Hapus	4	gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17 ls 1...	#Gempa Mag:5.2, 24-Jun-19 15:17:07 WIB, Lok:0.17 L...	NULL	1	1	2019-06-24
Ubah Salin Hapus	5	halo sahabattangguh kepala bnpb doni monardo cpns ...	Halo #SahabatTangguh! Kepala BNPB, Doni Monardo be...	NULL	0	NULL	2019-06-24
Ubah Salin Hapus	6	gempa mag 7 4 24 jun 19 09 53 39 wib lok 6 51 ls 1...	RT @DaryonoBMKG: Gempa Mag:7.4, 24-Jun-19 09:53:39...	NULL	1	1	2019-06-24
Ubah Salin Hapus	7	info gempa mag 7 7 24 jun 19 09 53 39 wib lok 6 51...	RT @infomitigasi: Info #Gempa Mag:7.7, 24-Jun-19 0...	NULL	1	1	2019-06-24
Ubah Salin Hapus	8	info gempa mag 7 7 24 jun 19 09 53 39 wib lok 6 51...	Info Gempa Mag:7.7, 24-Jun-19 09:53:39 WIB, Lok:6....	NULL	1	1	2019-06-24

Gambar 4.3 Hasil *download* data

4.1.2 Implementasi Text Preprocessing

Text Preprocessing merupakan proses pengubahan bentuk data yang belum terstruktur menjadi data yang terstruktur sesuai dengan kebutuhan. Tujuan dari *text preprocessing* yakni penyeragaman dan kemudahan pembacaan serta proses *LSA*. Selain itu, *text preprocessing* mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya (Andini, 2013). Data yang dipakai merupakan data hasil *download* dari akun BMKG dan BNPB pusat. Tahapan dari text Preprocessing yang digunakan dalam sistem ini sebagai berikut. Data tweet sebelum dilakukan tahap *preprocessing* dapat dilihat pada gambar 4.4



Gambar 4.4 Data tweet sebelum dilakukan tahap *preprocessing*

1. Remove Stopword

Tahap *stopword removal* merupakan tahapan yang dilakukan untuk menghilangkan kata-kata yang tidak deskriptif. Proses penghapusan *stopword* dilakukan dengan mendefinisikan kata-kata yang telah didaftarkan sebagai stopword terlebih dahulu. Dalam hal ini, seluruh

daftar stopword disimpan ke dalam sebuah *file* yang diberi nama *stopword_id*. *File* tersebut disimpan pada sebuah folder di dalam *corpora* yang terdapat pada *nltk_data*. Tahap *remove stopword* dilakukan dengan menggunakan bantuan *library* pada bahasa pemrograman *Python 3* bernama nltk. NLTK *(natural language toolkit)* merupakan *platform* yang digunakan untuk membangun program analisis teks. *Library* nltk perlu di instalasi terlebih dahulu. Setelah melakukan instalasi, *library* tersebut perlu dideklarasikan. Kemudian mengimplementasikan fungsi remove *stopword*.

2. Cleaning

Tahap ini bertujuan untuk membersihkan data tweet dari hal-hal yang tidak diperlukan, seperti simbol, *emoticon*, tanda baca,dan lain-lain supaya lebih efektif. Tahap-tahap yang dilakukan di dalam *cleaning*, yakni:

a. Menghapus karakter HTML

Pada tahap ini, karakter HTML perlu dihapus agar data tweet yang digunakan lebih mudah dibaca dan lebih efektif.

b. Menghapus *tag, username,* dan *unicode* karakter

Username,tag dan *unicode* merupakan karakter khusus pada percakapan di *Twitter.* Penghapusan elemen-elemen tersebut dideteksi dengan menggunakan *regular expression* (*regex*). *Regex* dilakukan dengan memanfaatkan *library re* dari *python*. Kata yang terdeteksi oleh *regex* maka akan diubah dengan teks kosong.

c.* Menghapus *emoticon

Penghapusan *emoticon* berfungsi untuk mempermudah pembacaan data dan melakukan tahap *preprocessing*. Penghapusan elemen-elemen tersebut sama seperti penghapusan *username, tag,* dan *unicode* karakter, yakni dengan mendeteksi menggunakan *regular expression* (*regex*). Kata yang terdeteksi oleh *regex* pun akan sama, yakni dengan mengubah menjadi teks kosong.

d.* Menghapus RT *(re-tweet)

Kata RT atau *re-tweet* akan dihapus dengan cara mendeteksi elemen menggunakan *regex*. *Regex* akan mendeteksi kata RT atau *re-tweet* yang terdapat pada sebuah tweet.

e. **Menghapus *link***

Proses penghapusan *link* dilakukan sama seperti yang penghapusan yang lain, yakni dengan mendeteksi elemen menggunakan *regex*. *Regex* mendeteksi *link* ketika menemukan kata dengan diawali dengan kemunculan kata *www*, *http*, atau *https* pada sebuah tweet.

f. Menghapus *colon* (:), *comma* (,)

Colon dan *comma* akan dihapus dengan cara mendeteksi elemen menggunakan *regex*. *Regex* akan mendeteksi *colon* dan *comma* yang terdapat pada sebuah tweet.

g. Menghapus kelebihan spasi

Kelebihan spasi pada tweet akan dideteksi oleh *regex. Regex* akan mendeteksi \s , yakni spasi, tab, ataupun baris yang berlebih pada suatu tweet.

h. Menjadikan *lowercase* (huruf menjadi kecil)

Proses *lowercase* di sini dilakukan untuk menyeragamkan keseluruhan teks menjadi huruf kecil. Penyeragaman teks menjadi huruf kecil dilakukan dengan fungsi *lower()* dari *python*.

Implementasi tahap-tahap tersebut hampir semua dilakukan dengan memanfaatkan *regular expression (regex)* untuk mendeteksi beberapa hal yang perlu dihilangkan dari sebuah data tweet. Hasil tahap *preprocessing* dapat dilihat pada gambar 4.5.

tweet
gempa mag 5 2 24 06 2019 15 17 07 pusat gempa dila...
info gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17...
gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17 ls 1...
gempa mag 5 2 24 jun 19 15 17 07 wib lok 0 17 ls 1...
halo sahabattangguh kepala bnpb doni monardo cpns ...
gempa mag 7 4 24 jun 19 09 53 39 wib lok 6 51 ls 1...
info gempa mag 7 7 24 jun 19 09 53 39 wib lok 6 51...
info gempa mag 7 7 24 jun 19 09 53 39 wib lok 6 51...
gempa mag 5 2 24 jun 19 08 28 39 wib lok 2 49 ls 1...
info gempa mag 6 0 24 jun 19 08 05 27 wib lok 2 49...
gempa mag 6 0 24 jun 19 08 05 27 wib lok 2 49 ls 1...
gempa mag 6 0 24 jun 19 08 05 27 wib lok 2 49 ls 1...
infogempa magnitude 6 0 24 jun 2019 08 05 27 wib l...
gempa mag 6 0 24 jun 19 08 05 27 wib lok 2 49 ls 1...
week twitter xf0 x9f x8e x89 291 mentions 5 77m me...
gladi lapangan mitigasi kebakaran siswa siswi sdn ...
laporan harian pusdalops pb bpbd kabupaten tasikma...
our biggest fans this week pksjaksel hkb 26april m...
edukasi budaya sadar bencana siswa siswi sdn 1 sel...
cek video download inarisk personal playstore
twitter please your magic nagar orang orang downlo...
kunjungan lembaga penerbangan antariksa nasional d...
kepala bnpb membuka kegiatan pitiabi2019 sentul bo...
pusdalops bpbd kab bantul ikuti lokalatih pusat pe...
kenalan inarisk datang pekan ilmiah tahunana keben...

Gambar 4.5 Hasil tahap *preprocessing*

4.2 Implementasi Klasifikasi

Setelah proses pengunduhan dan proses *preprocessing* berhasil dilakukan, selanjutnya akan dilakukan klasifikasi. Proses klasifikasi dilakukan untuk mengetahui status dari *tweet-tweet* yang telah diunduh dari akun-akun yang telah ditentukan sebelumnya. Proses–proses yang terdapat di dalam klasifikasi terbantu oleh *library* pada bahasa pemrograman *python3*, yakni *scikit-learn* untuk klasifikasi dan *numpy* serta *pandas* untuk pembacaan data serta modul *pipeline* untuk meringkas banyaknya tahap yang kemudian akan dijadikannya sebagai fungsi. Tahapan yang terdapat pada modul *pipeline* yakni *countVeectorizer* yang memiliki parameter *n-gram range, Tfidf* yang memiliki parameter perhitungan.

Sebelum proses klasifikasi dijalankan, terlebih dahulu proses *labelling* secara otomatis dilakukan. Proses ini dilakukan dengan menggunakan beberapa kondisi. Seperti yang sudah dijelaskan sebelumnya, dilakukan tahap mengambil *list* kota dan bencana dari *database*

terlebih dahulu sebelum mengunduh data tweet. Di mana nanti secara otomatis sistem akan membaca data dan disesuaikan dengan kondisi yang sudah ditentukan.

Terdapat tiga kondisi, yang pertama yakni apabila terdapat kalimat yang berhubungan dengan bencana seperti contoh "gempa", "tsunami", dan lain-lain maka sistem akan membaca kondisi tersebut sebagai bencana dengan label 1. Kondisi yang kedua yakni apabila terdapat kalimat "peringatan dini" di dalam data tweet maka sistem akan membaca bahwa kondisi tersebut merupkan peringatan dengan label 2. Namun, sistem hanya mampu membaca bencana sesuai dengan yang terdapat di dalam *list* bencana saja. Kondisi yang terakhir yakni ketika di dalam data tweet tersebut tidak terdapat kalimat selain kondisi satu dan dua, maka sistem akan membaca kondisi tersebut sebagai bukan bencana dengan label 0.

Setelah proses *preprocessing* dan *labelling* dilakukan, proses klasifikasi dijalankan. Adapun pada proses klasifikasi diberikan status, di antaranya 0 untuk status bukan bencana, 1 untuk status bencana, dan 2 untuk status *warning*. Selain itu, dalam proses klasifikasi ini kita dapat mengetahui nilai dari *accuracy, F1 score, precission score, recall score,* dan *confusion matriks.*

Accuracy merupakan tingkat kedekatan antara nilai prediksi denan nilai actual. *F1 score* merupakan rata–rata dari presisi dan *recall.* Seedangkan *precission score* merupakan tingkat ketepatan antara informasi yang diminta dengan jawaban yang diberikan oleh sistem. Yang terakhir yakni *confusion matriks* berfungsi untuk mengevaluasi keakuratan dari sebuah klasifikasi.

Metode yang digunakan dalam klasifikasi ini merupakan 2 metode klasifikasi *multinomial naive bayes*. *Multinomial Naive Bayes* atau biasa disebut dengan *Multinomial NBC* merupakan model pengembangan dari algoritma *bayes* yang cocok dalam pengklasifikasian teks atau dokumen. Pada formula *Multinomial NBC,* kelas dokumen tidak hanya ditentukan dengan kata yang muncul tetapi juga jumlah kemunculannya. Yang membedakan yakni fungsi dan parameter yang digunakan dalam *multinomial naïve bayes* tersebut. Untuk *multinomial naïve bayes* yang pertama menggunakan n_gram = unigram, sedangkan yang satu lagi menggunakan n_gram = bgram.

Langkah pertama yang dilakukan yakni meng*install library* yang diperlukan, setelahnya mendeklarasikan *library* yang dibutuhkan.

Kemudian mengambil dataset dari *database* yang akan dijadikan sebagai data *training* menggunakan *library* pandas. Pada klasifikasi ini,menggunakan data hasil *preprocessing* yang sudah ter*labelling* secara otomatis.Sedangkan data *testing* yakni 30% dari data *training*.

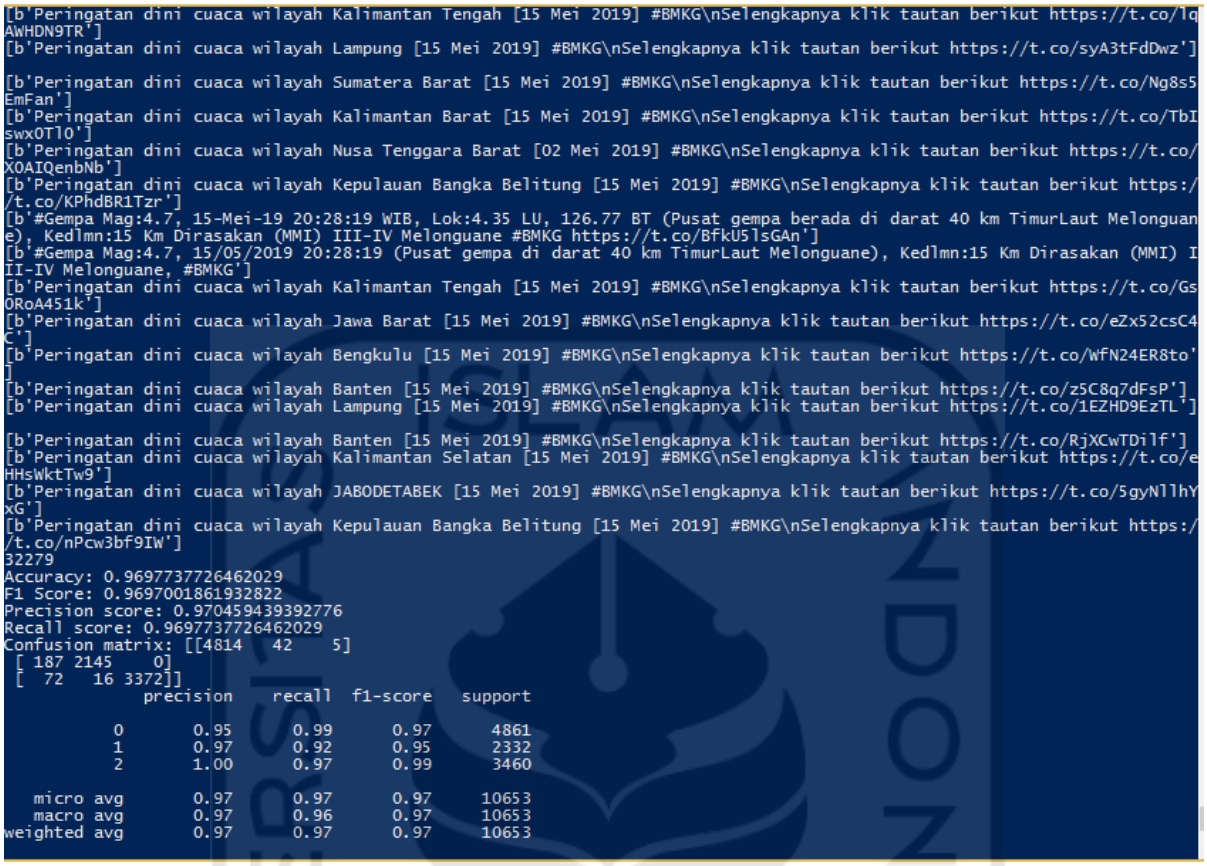
Proses selanjutnya membuat *class pipeline* yang di dalamnya terdapat 3 urutan tahapan yaitu mengubah dataset ke dalam sebuah representasi *vector* menggunakan library *CountVectorizer* dilanjut dengan pembobotan *word vector* dengan menggunakan *library TfidfTransformer* dan yang ketiga baru dilakukan klasifikasi dengan menggunakan library *Multinomial Naïve Bayes. CountVectorizer* berfungsi merubah teks menjadi *vector*. Sedangkan *TfudfTransformer* merupakan *termfrekuensi* atau menghitung frekuensi kata di dalam sebuah dokumen.

Seperti yang sudah dijelaskan di atas, klasifikasi menggunakan 2 metode klasifikasi *multinomial NBC.* Yang membedakan keduanya adalah parameter yang dijalankan dalam metode tersebut. Pada klasifikasi *Multinomial NBC* 1 , menggunakan *countVectorizer* untuk mengkonversi teks menjadi matriks, *TfidfTransformer* yang memiliki bobot *use_idf = true* yang berarti menggunakan idf dan *smooth_idf = true* yang bertipe *Boolean* dan berfungsi menambahkan angka 1 pada perhitungan frekuensi dokumen agar terhindar dari pembagi menjadi 0. Sedangkan *(alpha=1)* pada *Clf,MultinomialNB* berfungsi sebagai parameter yang berarti perlu melakukan penghalusan.

Pada klasifikasi terdapat *X-train* sebagai parameter dan *Y-train* sebagai kategori. Di mana data akan dilakukan *train-test-split* (dilihat-diuji-dibagi). *Test_size* atau data yang diuji berjumlah 33% dari keseluruhan data yang ada. Pada pengujian, menggunakan *method astype ('U')* yang berarti melakukan pengubahan tipedata dari *int* menjadi 'U' atau *string.*

Hampir sama dengan *multinomialNBC 1*, *multinomialNBC 2* juga memiliki parameter yang dijalankan dalam metode tersebut. Untuk *countVectorizer* di sini menggunakan *ngram-range* = 2,2 yang berarti melakukan pemecahan kata menjadi 2 kata dalam dokumen, *TfidfTransformer* serta *clf* yang berbobot sama seperti *multinomialNBC 1.*

Proses *download* tweet, dilanjutkan dengan proses *preprocessing* dan klasifikasi dapat dilihat pada gambar 4.6



```
[b'Peringatan dini cuaca wilayah Kalimantan Tengah [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/lq
AWHDN9TR']
[b'Peringatan dini cuaca wilayah Lampung [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/syA3tFdDwz']

[b'Peringatan dini cuaca wilayah Sumatera Barat [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/Ng8s5
EmFan']
[b'Peringatan dini cuaca wilayah Kalimantan Barat [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/TbI
swxOTlO']
[b'Peringatan dini cuaca wilayah Nusa Tenggara Barat [02 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/
XOAIQenbNb']
[b'Peringatan dini cuaca wilayah Kepulauan Bangka Belitung [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https:/
/t.co/KPhdBR1Tzr']
[b'#Gempa Mag:4.7, 15-Mei-19 20:28:19 WIB, Lok:4.35 LU, 126.77 BT (Pusat gempa berada di darat 40 km TimurLaut Melonguan
e), Kedlmn:15 Km Dirasakan (MMI) III-IV Melonguane #BMKG https://t.co/BfkU5lsGAn']
[b'#Gempa Mag:4.7, 15/05/2019 20:28:19 (Pusat gempa di darat 40 km TimurLaut Melonguane), Kedlmn:15 Km Dirasakan (MMI) I
II-IV Melonguane, #BMKG']
[b'Peringatan dini cuaca wilayah Kalimantan Tengah [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/Gs
ORoA451k']
[b'Peringatan dini cuaca wilayah Jawa Barat [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/eZx52csC4
C']
[b'Peringatan dini cuaca wilayah Bengkulu [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/WfN24ER8to'
]
[b'Peringatan dini cuaca wilayah Banten [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/z5C8q7dFsP']
[b'Peringatan dini cuaca wilayah Lampung [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/1EZHD9EzTL']

[b'Peringatan dini cuaca wilayah Banten [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/RjXCwTDilf']
[b'Peringatan dini cuaca wilayah Kalimantan Selatan [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/e
HHsWktTw9']
[b'Peringatan dini cuaca wilayah JABODETABEK [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https://t.co/5gyNllhY
xG']
[b'Peringatan dini cuaca wilayah Kepulauan Bangka Belitung [15 Mei 2019] #BMKG\nSelengkapnya klik tautan berikut https:/
/t.co/nPcw3bf9IW']
32279
Accuracy: 0.9697737726462029
F1 Score: 0.9697001861932822
Precision score: 0.970459439392776
Recall score: 0.9697737726462029
Confusion matrix: [[4814   42    5]
 [ 187 2145    0]
 [  72   16 3372]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      4861
           1       0.97      0.92      0.95      2332
           2       1.00      0.97      0.99      3460

   micro avg       0.97      0.97      0.97     10653
   macro avg       0.97      0.96      0.97     10653
weighted avg       0.97      0.97      0.97     10653
```

Gambar 4.6 Proses *download* tweet, proses *preprocessing*, dan proses klasifikasi

Hasil dari klasifikasi-klasifikasi tersebut akan otomatis tersimpan dalam bentuk file.xls. Semua proses *download tweet, preprocessing,* dan klasifikasi akan berjalan *realtime* 1 x 24 jam. Contoh hasil klasifikasi *multinomialNBC 1* dapat dilihat pada gambar 4.7.

A	B	C	D	E	F
	Accuracy	F1 Score	Precision score	Recall score	Confusion matrix
0	0.9686479014827846	0.9685407641467402	0.9688224778638558	0.9686479014827846	[[4014 33 171] [207 2177 0] [71 17 9226]]

Gambar 4.7 Contoh hasil klasifikasi *multinomialNBC 1*

Dari hasil klasifikasi dengan *multinomialNBC 1* dan m*ultinomialNBC* 2 dapat diketahui bahwa hasil akurasi keduanya berbeda. Pada *multinomialNBC 1* hasil akurasi lebih

tinggi dibandingkan dengan *multinomialNBC 2*. Hasil klasifikasi *multinomialNBC 1* kurang lebih diangka 0,965 – 0,968. Sedangkan *multinomialNBC 2* memilliki hasil klasifikasi kurang lebih diangka 0,993 – 0,995.

4.3 Hasil Implementasi Sistem

4.3.1 Implementasi Halaman Pemetaan Bencana

Halaman ini merupakan halaman awal sistem. Pada halaman ini, terdapat informasi bencana yang terjadi dalam kurun waktu 3 hari terakhir. Di mana informasi tersebut ditampilkan dalam bentuk pointer merah yang terletak pada wilayah terjadinya bencana. Informasi tersebut berupa *text* tweet dari akun BMKG(@info_BMKG) dan BNPB (BNPB_Indonesia) pusat. Implementasi halaman pemetaan bencana atau beranda dapat dilihat pada gambar 4.8.

Gambar 4.8 Implementasi halaman pemetaan bencana

Selain itu, terdapat beberapa bagian lain, di antaranya kolom pencarian dan peta Indonesia. Kolom pencarian terdiri dari 2 kolom, yakni kolom pencarian untuk melakukan pencarian *tweet* berdasarkan nama wilayah atau nama kota dan kolom pencarian untuk melakukan pencarian *tweet* berdasarkan jenis bencananya. Hasil dari pencarian tersebut sama seperti halaman awal sistem, di mana secara langsung akan menampilkan pion kecil berwarna

merah pada peta Indonesia. Pion tersebut berfungsi untuk memberikan tanda kepada pengguna bahwa wilayah yang diduduki pion tersebut memiliki informasi. Selain berfungsi untuk memberikan tanda wilayah, pion tersebut juga berisi informasi-informasi bencana berupa *text tweet* yang berasal dari akun-akun yang telah ditentukan di awal. Impelentasi pencarian berdasarkan nama wilayah dapat dilihat pada gambar 4.9.

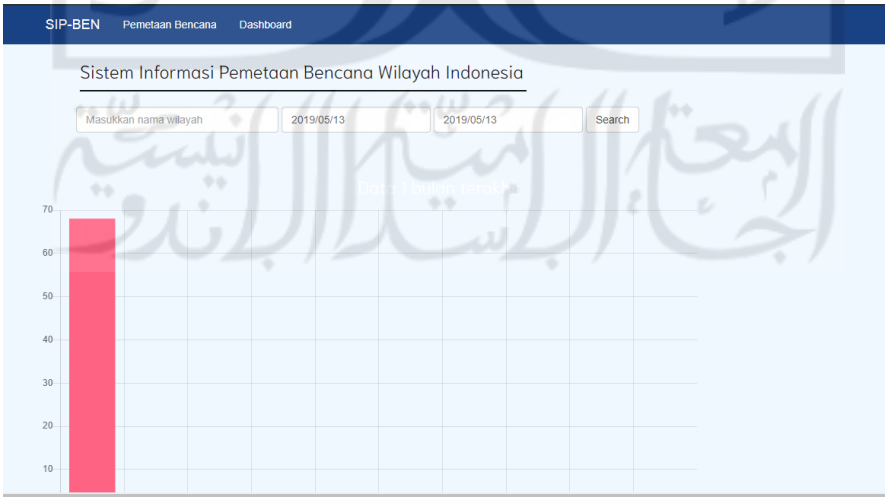
Gambar 4.9 Implementasi pencarian berdasarkan nama wilayah

Pada halaman ini, *user* juga dapat mencari informasi berdasarkan kategori bencana yang terdapat pada kolom. *User* hanya perlu memilih kategori bencana yang tersedia kemudian akan muncul informasi bencana dalam bentuk yang sama seperti pada halaman awal dan pencarian dengan nama wilayah. Implementasi pencarian berdasarkan kategori bencana dapat dilihat pada gambar 4.10.

Gambar 4.10 Implementasi pencarian berdasarkan kategori bencana

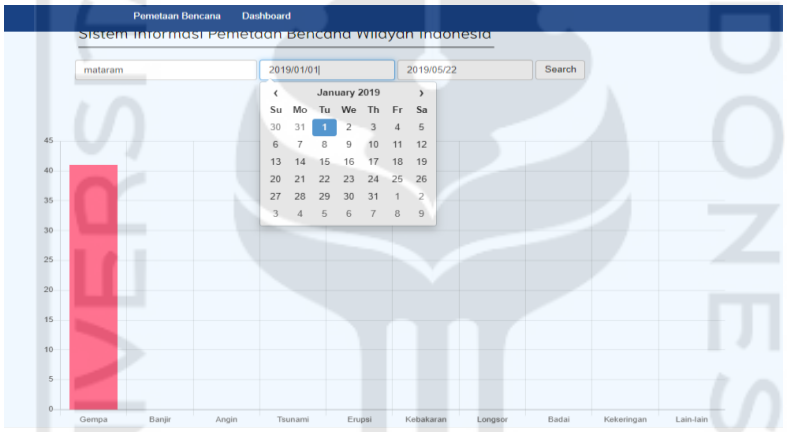
4.3.2 Implementasi Halaman *Dashboard*

Halaman *dashboard* berfungsi untuk mengetahui rekam jejak dari sebuah bencana. Rekam jejak tersebut dibuat ke dalam bentuk grafik. Tujuannya adalah memudahkan pengguna dalam mengetahui rekam jejak bencana. Saat mengunjungi halaman *dashboard*, *user* akan disuguhkan dengan grafik batang yang menampilkan informasi jumlah bencana yang terjadi dalam kurun waktu 1 bulan terakhir. Implementasi halaman *dashboard* dapat dilihat pada gambar 4.11.



Gambar 4.11 Implementasi halaman *dashboard*

Selain itu, pada halaman ini terdapat sebuah kolom pencarian, di mana pengguna diminta untuk memasukkan nama wilayah beserta rentang tanggal yang ingin dicari. Setelah pengguna menekan tombol *search*, maka secara otomatis akan muncul grafik. Proses hingga membentuk sebuah grafik, yakni pertama-tama seluruh data yang telah diunduh dengan memanfaatkan Twitter API disimpan ke dalam bentuk *excel*, kemudian dilakukan proses *preprocessing* pada *file* tersebut. Setelah itu dilakukan klasifikasi dan hasil dari klasifikasi dimasukkan ke dalam *database*. Sehingga, grafik yang muncul berisi tentang jumlah tweet yang telah dikelompokkan berdasarkan wilayah, rentang tanggal, dan jenis bencana yang tersimpan di dalam *database*. Implementasi halaman tampilan pencarian dengan nama wilayah dan rentang tanggal gambar 4.12.



Gambar 4.12 Implementasi tampilan pencarian dengan nama wilayah dan rentang tanggal

4.3.3 Implementasi *Database*

Database yang digunakan yakni *database mysql*. *Database mysql* memiliki 3 relasi, yaitu:

1. Many to many

   Yakni relasi di mana tabel A mampu memiliki banyak relasi ke tabel lainnya dan tabel A mampu menampung banyak relasi dari tabel lainnya.

2. One to one

   One to one adalah relasi antar tabel yang datanya hanya sebatas antar tabel itu saja..

3. One to many / many to one

   One to many / many to one merupakan relasi antara tabel A yang mampu ke banyak tabel lain, namun tabel A tidak mampu menerima data banyak dari tabel lain

Di dalam *database* sistem ini, terdapat 5 tabel, yakni:

1. Tabel tweet

   Di dalam tabel ini terdapat struktut tabel yang dapat dilihat pada gambar 4.13.

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(10)			Tidak	*Tidak ada*		AUTO_INCREMENT	Ubah Hapus Lainnya
2	tweet	varchar(255)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
3	tweet_kotor	varchar(255)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
4	lokasi	int(3)			Ya	*Tidak ada*			Ubah Hapus Lainnya
5	label	varchar(1)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
6	jenis_bencana	int(3)			Ya	*Tidak ada*			Ubah Hapus Lainnya
7	date	date			Tidak	*Tidak ada*			Ubah Hapus Lainnya

Gambar 4.13 Struktur tabel tweet

Dari struktur tabel di atas, dapat diketahui bahwa:

a. Id = *primary key.*

b. Tweet = berisi data tweet yang sudah ter*preprocessing.*

c. Tweet_kotor = berisi data yang asli tweet atau data yang belum ter-*perprocesing*

d. Lokasi = berisi data kode kota atau kabupaten yang didapatkan dari tabel lokasi karena memiliki relasi *one to one.* Begitupun sebaliknya.

e. Label = berisi label data tweet.

f. Jenis_bencana = berisi kode jenis bencana yang didapatkan dari tabel jenis_bencana karena memiliki relasi *many to many*.

2. Tabel lokasi

   Pada tabel lokasi ini terdapat struktut tabel yang dapat dilihat pada gambar 4.14.

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(3)			Tidak	*Tidak ada*			Ubah Hapus Lainnya
2	nama_lokasi	varchar(255)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
3	latitude	varchar(13)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
4	longitude	varchar(13)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya

Gambar 4.14 Struktur tabel lokasi

Dari struktur tabel di atas, dapat diketahui bahwa:

a. Id = *primary key.*

b. Nama lokasi = berisi nama-nama kota dan kabupaten yang dimasukkan ke dalam *database* oleh peneliti.

c. *Latitude* = berisi data *latitude* dari sebuah kota atau kabupaten yang di inputkan ke dalam *database* oleh peneliti.

d. *Longitude* = berisi data *longitude* dari sebuah kota atau kabupaten yang di inputkan ke dalam *database* oleh peneliti.

3. Tabel kategori

Pada tabel kategori terdapat struktut tabel yang dapat dilihat pada gambar 4.15.

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(2)			Tidak	*Tidak ada*		AUTO_INCREMENT	Ubah Hapus Lainnya
2	kategori_bencana	varchar(20)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya

Gambar 4.15 Struktur tabel kategori

Dari struktur tabel di atas, dapat diketahui bahwa:

a. Id = *primary key.*

b. Kategori_bencana = kategori bencana yang sudah ditentukan oleh peneliti.

4. Tabel jenis bencana

Pada tabel jenis bencana terdapat struktur tabel yang dapat dilihat pada gambar 4.16.

#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
1	id	int(10)			Tidak	*Tidak ada*		AUTO_INCREMENT	Ubah Hapus Lainnya
2	jenis_bencana	varchar(255)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya
3	kategori	varchar(20)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya

Gambar 4.16 Struktur tabel jenis bencana

Dari struktur tabel di atas, dapat diketahui bahwa:

a. Id = *primary key.*

b. Jenis_bencana = berisi berbagai jenis bencana yang sudah di inputkan oleh peneliti.

c. Kategori = berisi kategori bencana yang merujuk pada tabel kategori.

5. Tabel *hashtag*

Pada tabel jenis *hashtag* terdapat struktut tabel yang dapat dilihat pada gambar 4.17.

	#	Nama	Jenis	Penyortiran	Atribut	Tak Ternilai	Bawaan	Komentar	Ekstra	Tindakan
☐	1	**id**	int(10)			Tidak	*Tidak ada*			Ubah Hapus Lainnya
☐	2	**hashtag**	varchar(255)	latin1_swedish_ci		Tidak	*Tidak ada*			Ubah Hapus Lainnya

Gambar 4.17 Struktur tabel *hashtag*

Dari struktur tabel di atas, dapat diketahui bahwa:

a. Id = *primary key.*

b. *Hashtag* = berisi data trending topic yang diambil dari twitter