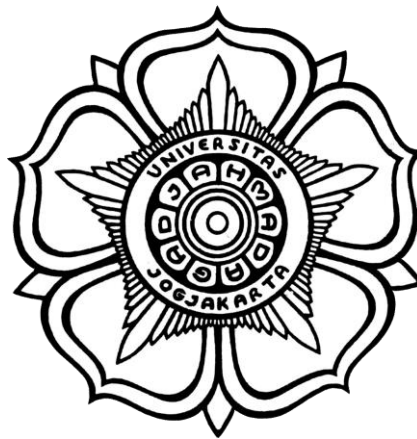


SKRIPSI

**IDENTIFIKASI TWEET BANJIR DI JAKARTA PADA DATA TWITTER
MENGUNAKAN ALGORITMA MULTINOMIAL NAIVE BAYES DAN
SUPPORT VECTOR MACHINE**

***FLOOD TWEETS IDENTIFICATION IN JAKARTA ON TWITTER DATA
USING MULTINOMIAL NAÏVE BAYES AND SUPPORT VECTOR
MACHINE ALGORITHMS***



**MUCHAMMAD ERSAN RAMADHAN
12/334666/PA/14899**

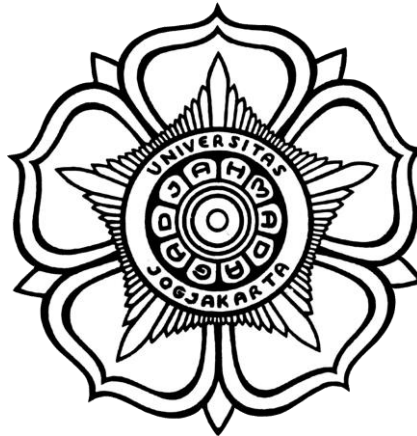
**PROGRAM STUDI ILMU KOMPUTER
DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS GADJAH MADA
YOGYAKARTA
2017**

SKRIPSI

**IDENTIFIKASI TWEET BANJIR DI JAKARTA PADA DATA TWITTER
MENGUNAKAN ALGORITMA MULTINOMIAL NAIVE BAYES DAN
SUPPORT VECTOR MACHINE**

***FLOOD TWEETS IDENTIFICATION IN JAKARTA ON TWITTER DATA
USING MULTINOMIAL NAÏVE BAYES AND SUPPORT VECTOR
MACHINE ALGORITHMS***

Diajukan untuk memenuhi salah satu syarat memperoleh derajat
Sarjana Komputer



MUCHAMMAD ERSAN RAMADHAN
12/334666/PA/14899

**PROGRAM STUDI ILMU KOMPUTER
DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS GADJAH MADA
YOGYAKARTA
2017**

HALAMAN PENGESAHAN

SKRIPSI

IDENTIFIKASI TWEET BANJIR DI JAKARTA PADA DATA TWITTER MENGUNAKAN ALGORITMA MULTINOMIAL NAIVE BAYES DAN SUPPORT VECTOR MACHINE

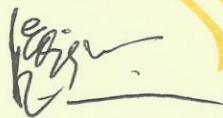
Telah dipersiapkan dan disusun oleh

MUCHAMMAD ERSAN RAMADHAN

12/334666/PA/14899

Telah dipertahankan didepan Tim Penguji
pada tanggal 12 Juni 2017

Susunan Tim Penguji



Moh Edi Wibowo, S.Kom, M.Kom., Ph.D

Pembimbing



Azhari SN, Drs., M.T., Dr

Ketua Penguji

Mengetahui
a.n. Dekan FMIPA-UGM
Wakil Dekan Bidang Akademik dan
Kongregasi



Dr. rer. nat. Nurul Hidayat Aprilia, M.Ed.
NIP. 197304071998031002



Isna Alfi Bustoni, M.Eng

Anggota Penguji

PERNYATAAN

Dengan ini saya menyatakan bahwa Skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, 13 Juni 2017



Muchammad Ersan Ramadhan

HALAMAN MOTTO DAN PERSEMBAHAN

“Sebaik-baik manusia diantaramu adalah paling banyak memberi manfaat bagi orang lain“

(HR.Bukhari Muslim)

“Ketakutanmu terhadap kegagalan itulah yang membuatmu gagal“

(Habib Syekh bin Abdulqodir Assegaf)

Bismillahirrohmanirrohim

Karya ini penulis persembahkan kepada

Bapak dan Mama tercinta,

Kakak tersayang,

Kawan-kawan Ilmu Komputer 2012

PRAKATA

Puji dan syukur kehadiran Tuhan Yang Maha Esa atas limpahan rahmat, berkat, karunia dan petunjuk-Nya sehingga penulis dapat menyelesaikan skripsi berjudul “Identifikasi Tweet Banjir di Jakarta pada Data Twitter Menggunakan Multinomial Naïve Bayes dan Support vector Machine ” telah terselesaikan dengan baik.

Selama penyusunan skripsi ini, penulis menyadari bahwa penulisan skripsi ini tidak lepas dari bimbingan, bantuan, dan dukungan dari berbagai pihak baik secara langsung maupun tidak langsung. Oleh karena itu, penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada :

1. Kedua orang tua penulis, Bapak Muchamad Munawir dan Mama Heni Ernawati tercinta yang tiada henti mendoakan, memberikan kasih sayang, memberikan dorongan moril dan spiritual kepada penulis. Muchammad Ervan Rozin selaku kakak yang selalu memberikan nasehat dan support positif.
2. Bapak Moh. Edi Wibowo Ph.D. selaku dosen pembimbing skripsi yang telah berbagi ilmu dan pengetahuan, memberikan arahan dan dorongan dari awal sampai akhir penulisan skripsi.
3. Bapak I Gede Mujiyatna, M.I.Kom selaku dosen pembimbing akademik yang telah memberikan bimbingan selama penulis menempuh pendidikan S1 Ilmu Komputer.
4. Seluruh dosen dan civitas akademika Departemen Ilmu Komputer dan Elektronika UGM yang telah memberikan ilmu pengetahuan dan memotivasi selama berada di bangku perkuliahan.
5. Aldi, Haikal, Kiki, Denis, Deni, Vita, Amel, Mbak Em, Luna, Tya, Iman, Mufti selaku kawan-kawan yang selalu ada, menemani hari-hari penulis di perkuliahan baik itu suka maupun duka, juga memberikan semangat dan motivasi dalam penyusunan skripsi.
6. Audiza yang selalu ada dan memberikan semangat serta motivasi dalam penyusunan skripsi.

7. Rilut, Azam, Danis serta teman-teman Ilmu Komputer UGM angkatan 2012 lainnya, yang sudah menemani hari-hari penulis selama masa perkuliahan, memberikan semangat dan motivasi dalam penulisan skripsi.
8. Teman-teman HIMAKOM UGM, yang telah menjadi keluarga kedua penulis selama menuntut ilmu di Yogyakarta. Terima kasih atas didikan dan pengalaman berharga yang diberikan kepada penulis.
9. Semua pihak yang berkontribusi dan tidak dapat disebutkan satu-persatu.

Akhir kata penulis berharap semoga skripsi ini dapat bermanfaat bagi pembaca dan semua pihak yang berkepentingan dengan skripsi ini.

Yogyakarta, 13 Juni 2017

Muchammad Ersan Ramadhan

DAFTAR ISI

HALAMAN PENGESAHAN.....	iii
PERNYATAAN.....	iv
HALAMAN MOTTO DAN PERSEMBAHAN.....	v
PRAKATA.....	vi
DAFTAR ISI.....	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xii
INTISARI.....	xiii
ABSTRACT.....	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah.....	3
1.4 Tujuan Penelitian.....	4
1.5 Manfaat Penelitian.....	4
1.6 Sistematika Penulisan.....	4
BAB II TINJAUAN PUSTAKA.....	6
BAB III LANDASAN TEORI.....	10
3.1 Data Mining.....	10
3.1.1 Text mining	10
3.2 Twitter	11
3.2.1 Web Scraping	11
3.3 Banjir	12
3.4 Data Pre-Processing	12
3.4.1 Cleansing.....	13
3.4.2 Case Folding	13
3.4.3 Stemming	13
3.4.4 Tokenisasi	17
3.4.5 Pembuangan <i>Stopword</i>	17
3.4.6 N-gram	17
3.5 Fitur dan Pembobotan	18
3.5.1 Term Presence (TP).....	18

3.5.2	Term Frequency (TF).....	19
3.5.3	Term Frequency Inverse Document Frequency (TF-IDF).....	19
3.6	Klasifikasi.....	20
3.6.1	Multinomial Naive Bayes	20
3.6.2	Support Vector Machine	22
3.7	Evaluasi Performa Model.....	24
3.7.1	K-Fold Cross Validation	25
BAB IV ARSITEKTUR DAN PERANCANGAN SISTEM		27
4.1	Rancangan Sistem	27
4.2	Deskripsi Sistem.....	28
4.3	Data dan Kelas Data	30
4.3.1	Data Stopword.....	30
4.3.2	Data <i>Tweet</i>	31
4.3.3	Data Kata Dasar	33
4.4	<i>Preprocessing</i> Data (Prapemrosesan Data).....	33
4.4.1	Perancangan <i>Cleansing</i>	34
4.4.2	Perancangan <i>Case Folding</i>	35
4.4.3	Perancangan <i>Stemming</i>	36
4.4.4	Perancangan Tokenisasi	38
4.4.5	Perancangan Pembuangan <i>Stopword</i>	38
4.4.6	Perancangan Running Time <i>Preprocessing</i>	40
4.5	Perancangan Pembuatan <i>Vocabulary</i>	40
4.6	Perancangan TF-IDF	40
4.7	Klasifikasi.....	40
4.7.1	Pelabelan Data.....	41
4.7.2	Perancangan <i>Training</i>	41
4.7.3	Perancangan <i>Testing</i>	43
4.7.4	Perancangan Evaluasi dan Validasi Klasifikasi	44
BAB V IMPLEMENTASI SISTEM.....		45
5.1	Spesifikasi Sistem Pengembangan	45
5.2	Implementasi Sistem	45
5.3	Implementasi Pelabelan Data	46
5.3.1	Implementasi <i>Cleansing</i> dan <i>Case Folding</i>	46
5.3.2	Implementasi <i>Stemming</i>	47

5.3.3	Implementasi Tokenisasi.....	48
5.3.4	Implementasi Pembuangan <i>Stopword</i>	48
5.4	Implementasi Running Time Preprocessing	49
5.5	Implementasi Pembuatan <i>Vocabulary</i>	49
5.6	Implementasi Pembobotan TF-IDF.....	50
5.7	Implementasi <i>Training</i>	51
5.8	Implementasi <i>Testing</i>	52
5.9	Implementasi Klasifikasi dan Evaluasi Model	52
5.9.1	Model Evaluasi Multinomial Naïve Bayes	53
5.9.2	Model Evaluasi Support Vector Machine	55
BAB VI HASIL PENELITIAN DAN PEMBAHASAN.....		57
6.1	Pengujian Sistem dan Hasil	57
6.2	Hasil Preprocessing	57
6.3	Hasil Vocabulary	58
6.4	Hasil Pembobotan TF-IDF	60
6.5	Hasil Identifikasi	61
6.6	Hasil Pengujian.....	64
BAB VII KESIMPULAN DAN SARAN		65
6.7	Kesimpulan.....	65
6.8	Saran	65
DAFTAR PUSTAKA		67
LAMPIRAN.....		70
A.	Rata-rata Akurasi Hasil Pengujian Multinomial Naïve Bayes	70
B.	Rata-rata Akurasi Hasil Pengujian Support Vector Machine.....	70
C.	Data Stopwords	71

DAFTAR GAMBAR

Gambar 1. 1 Contoh <i>tweet</i> bukan bencana.....	2
Gambar 4. 1 Flowchart proses perancangan dan analisis sistem	28
Gambar 4. 2 Diagram alur deskripsi sistem	30
Gambar 4. 3 Contoh data yang terdiri dari <i>Username</i> , tanggal dan <i>Tweet</i>	31
Gambar 4. 4 Diagram alur <i>parsing</i> HTML	32
Gambar 4. 5 Diagram alur prapemrosesan.....	34
Gambar 4. 6 Alur proses <i>cleansing</i>	35
Gambar 4. 7 Diagram alur proses <i>case folding</i>	36
Gambar 4. 8 Data berita sebelum di proses <i>case folding</i> dan <i>cleansing</i>	36
Gambar 4. 9 Data berita setelah diproses <i>case folding</i> dan <i>cleansing</i>	36
Gambar 4. 10 Alur proses perancangan <i>stemming</i>	37
Gambar 4. 11 Contoh hasil tokenisasi.....	38
Gambar 4. 12 Alur proses perancangan tokenisasi	38
Gambar 4. 13 Flowchart pembuangan <i>stopword</i>	39
Gambar 4. 14 Proses pembuangan <i>stopword</i>	40
Gambar 4. 15 Flowchart proses <i>training</i>	42
Gambar 4. 16 Flowchat proses <i>testing</i>	43
Gambar 5. 1 Data training yang telah diberi label	46
Gambar 5. 2 Kode proses <i>cleansing</i> dan <i>case folding</i>	47
Gambar 5. 3 Kode proses <i>stemming</i>	48
Gambar 5. 4 Kode proses tokenisasi	48
Gambar 5. 5 proses pembuangan <i>stopword</i>	49
Gambar 5. 6 kode proses <i>running time preprocessing</i>	49
Gambar 5. 7 Kode proses pembuatan <i>vocabulary</i>	50
Gambar 5. 8 Kode Pembobotan <i>TF-IDF</i>	51
Gambar 5. 9 Kode Implementasi <i>training</i>	52
Gambar 5. 10 Kode Implementasi <i>testing</i>	52
Gambar 5. 11 Kode model evaluasi Multinomial NB	54
Gambar 5. 12 kode model evaluasi support vector machine	56

DAFTAR TABEL

Tabel 2. 1 Perbandingan penelitian	8
Tabel 3. 1 Contoh pemotongan N-gram berbasis karakter	17
Tabel 3. 2 Contoh pemotongan N-gram berbasis kata	18
Tabel 3. 3 <i>Confusion matrix</i>	25
Tabel 6. 1 Komposisi jumlah <i>tweet training</i> pada setiap label	57
Tabel 6. 2 Cuplikan <i>tweet</i> sebelum <i>dipreprocessing</i>	57
Tabel 6. 3 Cuplikan <i>tweet</i> setelah <i>dipreprocessing</i>	58
Tabel 6. 4 Percobaan terhadap nilai parameter α	62
Tabel 6. 5 Percobaan terhadap nilai parameter C	62
Tabel 6. 6 Validasi Hasil SVM dengan Fakta	63
Tabel 6. 7 Rata-rata nilai akurasi model multinomial naïve bayes	64
Tabel 6. 8 Rata-rata nilai akurasi model Support Vector Machine	64

INTISARI

IDENTIFIKASI TWEET BANJIR DI JAKARTA PADA DATA TWITTER MENGUNAKAN ALGORITMA MULTINOMIAL NAIVE BAYES DAN SUPPORT VECTOR MACHINE

Oleh

Muchammad Ersan Ramadhan
12/334666/PA/14899

Twitter merupakan salah satu media sosial yang umum digunakan oleh masyarakat. Masyarakat menuliskan informasi tentang banjir melalui media *twitter* dalam bentuk *tweet*. Beberapa *tweet* yang dituliskan tidak menunjukkan secara nyata bencana banjir yang sedang terjadi. Terdapat *tweet* tentang banjir menunjukkan *tweet* yang bukan kategori bencana, sebagai contoh banjir hadiah. Padahal informasi bencana banjir secara nyata dan cepat dibutuhkan untuk mengantisipasi banjir yang berkelanjutan. Oleh karena itu diperlukan sistem yang secara otomatis mengidentifikasi secara otomatis *tweet* banjir termasuk dalam kategori bencana atau banjir dalam arti lain.

Pada penelitian ini proses identifikasi *tweet* banjir menggunakan algoritma *Multinomial Naïve Bayes* dan *Support Vector Machine*. Data yang digunakan merupakan *tweet* yang mengandung kata banjir yang berlokasi di DKI Jakarta. Data *tweet* banjir diidentifikasi menjadi 2 kategori yaitu bencana dan bukan bencana. Jumlah *tweet* sebanyak 7789 sebagai data *training* dan 2327 sebagai data *testing* yang diambil pada 12 November 2014 – 30 Januari 2015.

Pengujian dilakukan dengan membandingkan algoritma *Multinomial Naïve Bayes* dan *Support Vector Machine*. Pengujian juga dilakukan dengan menggunakan metode *K-fold cross validation* dengan jumlah iterasi 10 kali. Dari hasil pengujian didapatkan nilai akurasi terbesar dengan menggunakan algoritma *Support Vector Machine* dengan rata-rata akurasi 78.9%

Kata Kunci : Banjir, *Multinomial Naïve Bayes*, *Support Vector Machine*.

ABSTRACT

FLOOD TWEETS IDENTIFICATION IN JAKARTA ON TWITTER DATA USING MULTINOMIAL NAÏVE BAYES AND SUPPORT VECTOR MACHINE ALGORITHMS

Oleh

Muchammad Ersan Ramadhan

12/334666/PA/14899

Twitter is one of the most common media sosial used by the society which is effective and fast in reporting the current situation, such as flood in DKI Jakarta. Flood tweets will be automatically identified so that it can provide information whether it is flooded or not flooded in certain area. The use of Twitter as a source of information regarding flood is potentially helpful to anticipate the occurrence of long-standing flood. That is why a system is needed to automatically categorized a flood containing tweets as flood as an event or other meaning

The identification process of flood tweets used in this research is Multinomial Naïve Bayes and Support Vector Machine algorithms. The data used are the tweets containing the word “banjir” located in DKI Jakarta. The data is identified into 3 categories, namely flooded, not flooded and unknown. Total tweets of 7789 is used as data training and 2327 tweets is used as data testing taken from November 12, 2014 - January 25, 2015.

The test is done by comparing Multinomial Naïve Bayes dan Support Vector Machine algorithms. The test is also done by using K-Fold Cross Validation method with 10 times iteration. The test result shows that the greatest accuracy obtained by using Support Vector Machine algorithms with average accuracy of 78.9%.

Keyword : Banjir, *Multinomial Naïve Bayes*, *Support Vector Machine*.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Saat ini, *microblogging* menjadi sangat populer untuk alat komunikasi antara pengguna internet. Setiap hari jutaan pesan muncul di website penyedia *microblogging* diantaranya Twitter, Facebook, dan Tumblr. User tersebut menuliskan pesan tentang kehidupan sehari-harinya, opini dari beberapa topik bahkan membicarakan tentang isu-isu terkini yang sedang hangat. Karena format pesan yang gratis dan mudah untuk diakses, pengguna internet cenderung meninggalkan alat komunikasi yang tradisional (blog, mail) dan beralih menggunakan *microblogging*, salah satunya twitter. Hingga bulan Oktober 2016 terdapat lebih dari 500 juta pengguna terdaftar di twitter, 317 juta diantaranya adalah pengguna aktif. Pada Oktober 2016, pengguna Twitter mengirimkan lebih dari 500 juta kicauan per hari (Morscheck, 2016). Sedangkan Indonesia menjadi negara yang menduduki peringkat ke 4 dalam mengakses situs twitter hingga saat ini. Twitter merupakan salah satu media sosial yang efektif dan cepat dalam mengabarkan keadaan yang sedang terjadi di ibu kota DKI Jakarta salah satunya ketika terjadi bencana. Bencana yang sering melanda ibu Kota Jakarta adalah bencana Banjir.

Banjir menurut KBBI berarti berair deras dan banyak, kadang kadang meluap yang disebabkan oleh sungai karena hujan yang turun terus menerus. Menurut fakta (data.jakarta.co.id), Tahun 2016, 25 Kecamatan di DKI Jakarta masih terendam banjir. Rata-rata waktu banjir di DKI Jakarta yaitu 2 hari. Angka tersebut telah berkurang jika dibanding tahun-tahun sebelumnya dimana banjir bisa terjadi selama 20 hari.

Masyarakat menuliskan informasi tentang banjir melalui media *twitter* dalam bentuk *tweet*. Media sosial termasuk *twitter* memiliki kredibilitas yang rendah sebagai penyedia informasi (Oktafiani dkk, 2012). Informasi yang tidak benar dapat menimbulkan masalah. Seperti contoh

Gambar 1.1 menunjukkan *tweet* tentang banjir menunjukkan *tweet* yang bukan kategori bencana.



Gambar 1. 1 Contoh *tweet* bukan bencana

Padahal informasi bencana banjir secara nyata dan cepat dibutuhkan untuk mengantisipasi banjir yang berkelanjutan. Akan tetapi, tidak terstruktur data pada *tweet* membuat *tweet* tidak bisa secara langsung dipergunakan. Oleh karena itu diperlukan sistem yang dapat mengubah data *tweet* yang tidak terstruktur menjadi data yang lebih terstruktur dengan mengenali entitas yang ada pada *tweet* sehingga dapat mengidentifikasi secara otomatis *tweet* banjir termasuk dalam kategori bencana atau banjir dalam arti lain.

Dalam penelitian ini akan dilakukan analisis dengan menggunakan teknik klasifikasi multinomial naïve bayes dan support vector machine (SVM). Multinomial naïve bayes adalah metode dengan memperhitungkan frekuensi masing-masing kemunculan dalam sebuah dokumen dan probabilitas. Menurut Huang (2013) Algoritma naïve bayes memiliki akurasi yang bagus ketika terbatasnya memori dan CPU. Multinomial naïve bayes memiliki keunggulan pada kemudahan komputasi dan komputasi yang tidak memakan banyak sumber daya. Kelebihan algoritma ini sangat cocok di diterapkan untuk mengolah data dari twitter untuk mengidentifikasi *tweet* banjir yang membutuhkan komputasi yang cepat. Sedangkan klasifikasi SVM memiliki beberapa kelebihan tersendiri diantaranya lain generalisasi atau kemampuan mengklasifikasi suatu pattern. (Nugroho, 2003). Menurut Joachim (1998) masalah kategori teks seperti identifikasi *tweet* dapat dipisahkan secara linier menggunakan SVM. Keuntungan lainnya ialah SVM memiliki sedikit fitur yang tidak relevan karena SVM menganggap setiap fitur memiliki bobot yang tidak nol.

Untuk mengetahui akurasi dari identifikasi *tweet* banjir yang dilakukan, maka diperlukan performa evaluasi yang dapat mengetahui

seberapa baik performa dari algoritma yang digunakan. Salah satu metode yang digunakan adalah *k-fold cross validation*. Dengan menggunakan metode tersebut akan diketahui nilai akurasi, *precision*, *recall*, dan *f-measure* untuk mengetahui performa dari algoritma klasifikasi yang digunakan.

Dalam penelitian ini diharapkan akan diketahui identifikasi banjir di ibu kota Jakarta, apakah kata banjir yang di posting setiap user masuk kedalam kategori bencana atau bukan bencana. Dengan menggunakan algoritma *naïve bayes classification* dan *support vector machine* yang akan dibandingkan akurasi.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat diambil sebuah permasalahan utama yaitu bagaimana mengidentifikasi *tweet* banjir masuk kedalam kategori bencana atau bukan bencana dan seberapa banyak jumlah banjir kategori bencana atau bukan.

1.3 Batasan Masalah

Adapun batasan-batasan masalah pada penelitian ini adalah:

1. Data yang digunakan adalah data yang diambil dari *tweet* dalam kurun waktu 12 November 2014 sampai 25 Januari 2015 di *Twitter*.
2. Setiap posting *tweet* tersebut mengandung kata “banjir” yang terjadi di Jakarta
3. Identifikasi banjir hanya dilakukan kepada *tweet* berbahasa Indonesia.
4. Data *tweet* yang dilakukan merupakan data *offline* (tidak terintegrasi langsung dengan twitter secara *online*).
5. Fitur yang digunakan adalah *unigram* dengan seleksi fitur menggunakan metode *TF-IDF*
6. Proses identifikasi menggunakan algoritma *multinomial naive bayes* dan dilakukan komparasi terhadap algoritma *support vector machine*

7. Evaluasi performansi dilakukan dengan cara menghitung nilai akurasi, *precision*, *recall*, dan *f-measure* dari hasil analisis identifikasi *tweet* banjir

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini adalah:

- Membuat analisis untuk mengidentifikasi *tweet* banjir masuk dalam kategori bencana atau banjir dengan arti lain menggunakan metode *multinomial naïve bayes* dan *support vector machine*.
- Mengetahui dan membandingkan akurasi yang dihasilkan dari algoritma *multinomial naïve bayes* dan *support vector machine*.

1.5 Manfaat Penelitian

Manfaat penelitian ini berupa kontribusi pada identifikasi *tweet* banjir menggunakan *multinomial naïve bayes* dan *support vector machine*, sedangkan manfaat kepada masyarakat adalah memberikan kemudahan dalam mengidentifikasi *tweet* banjir sehingga dapat memberikan informasi banjir yang nyata ketika ada banjir di ibu kota.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam penyusunan tugas akhir ini adalah sebagai berikut;

BAB I: PENDAHULUAN

Bab ini menjelaskan tentang latar belakang, perumusan masalah, batasan masalah, tujuan penelitian, dan manfaat penelitian.

BAB II: TINJAUAN PUSTAKA

Bab ini memuat uraian mengenai penjelasan penelitian sebelumnya yang terkait dengan penelitian ini. Selain itu juga menjadi referensi dalam penelitian ini.

BAB III: LANDASAN TEORI

Bab ini memuat pengertian-pengertian yang diperlukan untuk pembahasan di bab-bab berikutnya.

BAB IV: ANALISIS DAN PERANCANGAN SISTEM

Bab ini menguraikan analisis sistem yang akan dibangun serta penjelasan mengenai perancangan sistem berdasarkan hasil analisis yang telah dilakukan.

BAB V: IMPLEMENTASI SISTEM

Bab ini menjelaskan tentang implementasi dari sistem sesuai dengan rancangan yang telah dibangun dan berdasarkan *tools* yang dipakai.

BAB VI: HASIL DAN PEMBAHASAN

Pada bab ini berisi hasil penelitian berdasarkan pengujian yang dilakukan.

BAB VII: KESIMPULAN DAN SARAN

Pada bab ini berisi kesimpulan yang diambil dari hasil pembahasan serta saran-saran untuk pengembangan sistem selanjutnya.

BAB II TINJAUAN PUSTAKA

Putro (2011) melakukan klustering untuk memperkirakan terjadinya banjir di Bandung. Metode yang digunakan yaitu *density based spartial clustering* dengan noise (DBSCAN). Data yang digunakan yaitu data klimatologi yang dihitung perhati sejak Januari 2016 hingga Desember 2010 dari BMKG kota Bandung. Parameter yang digunakan adalah curah hujan, temperature dan tekanan udara. Hasil yang didapatkan DBSCAN mampu memperlihatkan pola banjir, namun kurang cocok digunakan untuk melakukan prediksi. Prediksi dilakukan dengan membandingkan data hasil DBSCAN yang dibandingkan dengan hasil fakta dilapangan dengan akurasi 80%.

Gokulaksirhman, dkk (2012) melakukan klasifikasi tetapi dalam hal analisis sentiment terhadap data twitter untuk mengetahui konten emosial setiap *tweet*. Data *tweet* tersebut dikumpulkan dan diberi label positif , negatif dan netral, lalu dilakukan preprocessing. Data yang tidak menggunakan Bahasa Inggris mereka berikan label *irrelevant*. Setelah itu 8500 data negative dan 41000 data positif dibandingkan dengan menggunakan aplikasi Weka, dan dilakukan beberapa algoritma seperti *Multinomial Naive bayes*, *Support Vector Machine*, *Complement Naïve Bayes*, *Random Forest*, *J48*, *SMO*, *Filtered Classifier* dan *DMNBtext* dimana hasil rata-rata hasil algoritma *Multinomial Naïve Bayes* adalah 74.99% sedangkan untuk rata-rata akurasi algoritma *Support Vector Machine* adalah 72.70%. Penelitian tersebut menyimpulkan bahwa yang paling konsisten akurasinya pada berbagai keadaan yang dilakukan yaitu DMNB Text.

Rodiyansyah (2013) melakukan penelitian tentang klasifikasi *posting* kemacetan lalu lintas kota bandung dengan algoritma *naïve bayesian classification*. Data yang dikumpulkan adalah data *tweet* dengan jangka waktu tertentu yang terdapat kata “macet” pada setiap *posting tweet*. Data yang terkumpul akan dibagi menjadi *training* dan *testing*, selanjutnya dilakukan *preprocessing* untuk setiap *tweet* dan setelah itu dilakukan klasifikasi *tweet* apakah jalan masuk kategori macet atau lancar. Penelitian ini lalu dilanjutkan dengan memvisualisasikan macet kedalam API

Google Map sehingga letak kemacetan bisa terpampang jelas dalam bentuk peta. Hasil pengujian diperoleh akurasi terbaik 91.60% dengan sampel data 13106 untuk algoitma klasifikasi *Naïve Bayesian*.

Ilyas (2014) melakukan pemanfaatan twitter untuk manajemen bencana. Ilyas merancang micro filter (sebuah *machine learning*) yang mana data gambar yang diambil dari data tweet akan di klasifikasikan. Klasifikasi ini akan menghilangkan gambar yang tidak menunjukkan kerusakan langsung, sehingga tidak berguna untuk usaha penyelamatan. Klasifikasi menggunakan *naïve bayes* dan *support vector machine*. Hasilnya klasifikasi yaitu nilai presisi 70%, recall 88% dan F1 78%.

Soebroto dkk (2015) melakukan prediksi tinggi muka air untuk deteksi dini bencana banjir menggunakan support vector regression (SVR) dan metode *time variant inertia weight particle swarm optimization* (TVIWPSO). Objek yang digunakan Tinggi Muka Air. Data pengujian didapat dari 10 bulan dengan rincian 6 grafik berasal dari bulan pada musim penghujan dan 4 grafik berasal dari bulan pada musim kemarau. Hasil yang pengujian didapatkan nilai error terkecil sebesar 0.00755 dengan menggunakan mean absolute error.

Binawan (2016) melakukan klasifikasi tingkat kerawanan banjir pada kota bandung dengan metode *weighted product*. Kriteria yang digunakan adalah curah hujan, kemiringan, ketinggian, tutupan lahan, dan limpasan sungai. Dengan menggunakan bobot kriteria, sistem memberikan akurasi kecocokan antaran data BPBD dan data hasil hitung aplikasi sebesar 63.4%

Dari penelitian yang telah di bahas, setiap penelitian memiliki fokus yang berbeda. Pada penelitian Putro (2011) penelitian difokuskan pada klustering banjir yang akan terjadi. Parameter yang digunakan yaitu curah hujan, temperature dan tekanan udara. Soebroto dkk. (2015) memfokuskan pada deteksi dini banjir dengan parameter yg digunakan yaitu tinggi muka air. Ilyas (2014) berfokus pada pengambilan gambar pada *tweet* bencana lalu diklasifikasikan berdasar kerusakan daerah bencana. Sedangkan Binawan (2015) berfokus pada klasifikasi banjir dengan parameter curah hujan, kemiringan, ketinggian, tutupan lahan, dan limpasan sungai. Lalu penelitian yang dilakukan Rodiyansyah (2013) berfokus pada

klasifikasi *posting tweet* macet. Penelitian Gokulaksirhman (2012) berfokus pada perbandingan beberapa algoritma.

Pada penelitian ini dilakukan identifikasi tweet banjir menggunakan algoritma *multinomial naïve bayes* dan *support vector machine*. Identifikasi banjir berdasar pada sentimen, dan akan di kategorikan menjadi 2 kategori yaitu banjir dalam artian bencana dan banjir yang bukan berarti bencana. Hasil identifikasi banjir nantinya akan dikelompokkan berdasar kategori masing-masing dalam kurun waktu tertentu (time series) dan dibandingkan dengan fakta yang sedang terjadi.

Tabel 2. 1 Perbandingan penelitian

Peneliti	Topik penelitian	Dataset penelitian	Metode Klasifikasi	Domain
Putro (2011)	Implementasi Density Based Spatial Clustering Application with Noise (DBSCAN) dalam perkiraan banjir di bandung	BMKG kota bandung	<i>DBSCAN</i>	Banjir
Gokulaksirhman, dkk (2012)	<i>Opinion Mining and Sentiment analysis on a Twitter Data Stream</i>	<i>Twitter</i>	<i>Multinomial Naive bayes, Support Vector Machine, Complement Naïve Bayes, Random Forest, J48,</i>	Umum (<i>public</i>)

			<i>SMO, Filtered Classifier dan DMNBtext</i>	
Rodiansyah (2013)	Klasifikasi <i>posting</i> kemacetan lalu lintas kota bandung dengan algoritma <i>naïve bayesian classification</i>	<i>Twitter</i>	<i>Naïve bayesian classification</i>	Macet
Ilyas (2014)	MicroFilters : Harnessing Twitter for Disaster Management	<i>Twitter</i>	<i>Naïve Bayes Classifier dan Support vector machine</i>	Gambar Bencana
Soebroto dkk (2015)	Prediksi tinggi muka air untuk deteksi dini bencana banjir menggunakan SVR-TVIWPSO	BMKG	SVR dan SVR-TVIWPSO	Banjir
Binawan (2016)	Klasifikasi Tingkat Kerawanan Banjir pada Kota Bandung dengan Metode <i>Weighted Product</i>	BPBD, BAPPE DA, BPS	<i>Weighted Product</i>	Banjir

BAB III

LANDASAN TEORI

3.1 Data Mining

Data mining ialah proses untuk menemukan pola dalam data yang berjumlah besar. Pola yang dihasilkan melalui *data mining* dapat dikatakan berguna apabila dapat dilakukan prediksi data baru berdasar pola tersebut. Pola tersebut di representasikan dalam suatu struktur yang dapat dimengerti, dapat dianalisa dan dapat digunakan dalam membuat keputusan. Pola terstruktur tersebut dapat representasikan dalam berbagai bentuk, salah satunya ialah bentuk *decision rule* dan aturan. (Witten dkk, 2011)

Menurut Hand dkk (2001) *data mining* adalah analisis data pengamatan yang biasanya berjumlah besar untuk menemukan sesuatu yang tidak berhubungan dan meringkas data dengan cara yang baru yang bermanfaat dan mudah dipahami untuk pemilik data.

Liu (2011) *data mining* adalah proses untuk menemukan pola dalam data. Pola dalam *data mining* harus *valid*, dapat digunakan dan mudah dipahami. Data mining mencakup disiplin berbagai keilmuan antara lain *machine learning*, statistika, *database*, kecerdasan buatan dan visualisasi. Terdapat beberapa tugas dalam *data mining* yang dapat dilakukan. Tugas tugas tersebut antara lain adalah *supervised learning* (klasifikasi), *unsupervised learning* (*clustering*), *association rule mining*, dan *sequential pattern mining*.

3.1.1 Text mining

Text mining merupakan cabang ilmu dari data mining. *Text mining* adalah penambang yang dilakukan komputer untuk mendapatkan sesuatu yang baru, sesuatu yang tidak diketahui sebelumnya atau menemukan kembali informasi yang tersirat secara implisit, yang berasal dari informasi yang di ekstrak secara otomatis dari sumber-sumber data teks yang berbeda-beda (Feldman & Sanger, 2007). Tahap *text mining* secara umum adalah *preprocessing* dan *feature selection*. Teknik-teknik yang dapat digunakan

dalam text mining adalah teknik-teknik yang biasa digunakan pada *machine learning*. *Machine learning* dalam artian yang sederhana adalah mesin yang diatur supaya dapat belajar secara mandiri, tentunya menggunakan algoritma tertentu.

3.2 Twitter

Twitter merupakan salah satu media sosial yang populer dikalangan masyarakat. Twitter sendiri memiliki konsep *microblogging*. *Microblogging* ialah suatu bentuk blog yang memungkinkan para penggunanya dapat menuliskan pesan pembaharuan secara singkat lalu mempublikasikan. Dalam hal ini, Twitter membatasi jumlah karakter yang dapat dituliskan hanya sebesar 140 karakter, oleh karena disebut makro. Pesan singkat yang dikirim oleh pengguna twitter disebut dengan *tweet*.

3.2.1 Web Scraping

Web Scaping adalah usaha pemisahan konten yang penting dalam *website* dengan konten yang tidak penting. Konten yang telah dipisahkan selanjutnya diproses untuk kepentingan tertentu.

Menurut Turland (2010), yang dikutip dari Josi (2014), *Web Scraping* adalah proses pengambilan sebuah dokumen semi-terstruktur dari internet, umumnya berupa halaman-halaman web dalam bahasa *mark up* seperti HTML atau XHTML, dan menganalisis dokumen tersebut untuk diambil data tertentu dari halaman tersebut untuk digunakan bagi kepentingan lain. Menurut Mardi (2012) *web scraping* adalah salah satu cara memisahkan konten halaman situs dengan bagian-bagian yang tidak berhubungan dengan isi. Fokus dari *web scraping* adalah memperoleh data melalui pengambilan dan ekstraksi data dengan ukuran data yang bervariasi. Manfaat dari *web scraping* adalah agar informasi yang diambil lebih terfokus sehingga memudahkan dalam melakukan pencarian sesuatu.

3.3 Banjir

Banjir adalah debit aliran sungai yang secara relatif lebih besar dari biasanya akibat hujan turun di hulu atau suatu tempat tertentu secara terus menerus, sehingga air limpasan tidak dapat ditampung oleh alur/palung sungai yang ada, maka air melimpah keluar dan menggenangi daerah sekitarnya (Peraturan Dirjen RLPS No.04 thn 2009).

Menurut Doswell (2003) banjir adalah air yang meluap ke suatu tempat yang biasanya kering. Banjir biasanya terjadi karena hujan deras, tetapi banjir dapat bisa timbul dengan cara yang tidak langsung tergantung dengan cuaca yang sedang terjadi. Banyak faktor yang mengakibatkan terjadinya banjir, antara lain:

1. Kesadaran masyarakat yang kurang baik
2. Dataran rendah
3. Curah hujan yang tinggi
4. Daerah resapan yang sedikit
5. Mengendapnya lumpur di sungai dan pendangkalan sungai

.Banjir sudah menjadi ancaman musiman untuk berbagai wilayah di Indonesia, salah satunya nya DKI Jakarta. Menurut fakta (data.jakarta.co.id), Tahun 2016, 25 Kecamatan di DKI Jakarta masih terendam banjir. Rata-rata banjir di DKI Jakarta yaitu 2 hari. Angka tersebut telah berkurang jika dibanding tahun-tahun sebelumnya dimana banjir bisa terjadi selama 20 hari (2014). Sedangkan untuk tahun 2015 dampak banjir yang terjadi di Jakarta mengakibatkan korban jiwa 5 orang dengan jumlah pengungsi sekitar 230.000 jiwa.

3.4 Data Pre-Processing

Sebelum data memasuki tahap klasifikasi, data harus melalui proses *preprocessing*. *Preprocessing* adalah proses untuk membersihkan dan mempersiapkan data sebelum dilakukan proses klasifikasi. Teks yang diambil dari web biasanya banyak mengandung *noise* dan teks yang tidak informatif, seperti tag-tag HTML, *script-script*, dan *advertisements*. Selain

itu, banyak kata dalam teks yang diambil tidak mempunyai pengaruh yang besar pada proses klasifikasi. Tujuan dari proses *preprocessing* adalah untuk membersihkan serta melakukan penyeragaman kata/fitur sehingga kata/fitur tersebut siap untuk diekstraksi ke tahap selanjutnya. Proses umum yang dilakukan pada tahap *preprocessing* adalah *cleansing*, *stemming*, *filtering*, pembuangan *stopword* dan tokenisasi.

3.4.1 Cleansing

Cleansing adalah proses menghilangkan noise kata/fitur pada dokumen yang nantinya apabila tidak dihilangkan akan mengakibatkan proses klasifikasi menjadi tidak efisien. Tujuan dari proses *cleansing* adalah untuk memperbaiki dan meningkatkan kualitas data. Menurut Rahm dan Do (2000, dikutip dalam Rianto, 2016). Proses cleansing pada tweet dapat berupa menghilangkan *retweet*, menghilangkan *username*, menghilangkan tanda baca, menghilangkan angka-angka, menghilangkan URL, menghilangkan *white spaces*

3.4.2 Case Folding

Case folding mengubah semua huruf menjadi huruf kecil. Dimana hanya huruf 'a' sampai 'z' yang diterima dan karakter selain huruf dihilangkan dan dianggap sebagai *delimiter* (Maninng dkk. 2008). Tujuan dari proses ini adalah menghilangkan karakter-karakter selain huruf pada saat pengambilan informasi

3.4.3 Stemming

Proses Stemming berguna untuk mengubah kata berimbuhan menjadi kata dasar. Setiap proses *stemming* memiliki aturan-aturan yang berbeda yang berbeda. Aturan Bahasa diterapkan untuk menanggalkan imbuhan-umbuan itu. Variasi imbuhan dapat berupa dapat berupa awalan (*prefix*), sisipan (*infix*), akhiran (*suffix*).

Menurut Sastrawi (2014), yang dikutip dari Jelita (2007), dalam proses *stemming* Bahasa Indonesia terdapat beberapa persoalan, di antaranya yaitu:

1. Imbuhan pada Bahasa Indonesia yang cukup banyak dan kompleks, terdiri dari :
 - a. Prefiks, imbuhan didepan kata, seperti **ber**-empat
 - b. Suffiks, imbuhan diakhir kata, seperti minum-**an**
 - c. Konfiks, imbuhan didepan dan akhir kata, seperti **per**-empat-**an**
 - d. Infiks, imbuhan ditengah kata, seperti ge-**me**-tar
 - e. Imbuhan Bahasa asing, seperti sosial-**isasi**
 - f. Aturan perubahan prefiks, seperti me- menjadi **meng-**
2. Kata ambigu, yaitu kata yang memiliki dua makna, sebagai contoh kata Berikan menjadi **Ber**-ikan atau kata Berikan menjadi Beri-**kan**
3. *Overstemming*, sebagai contoh kata “berikan” dapat dipenggal **ber-i-kan**, untuk mengatasi hal itu maka digunakan daftar kata dasar, jika proses yang dipenggal terdapat kata dasar maka proses dihentikan atau selesai.
4. *Understemming*, sebagai contoh kata “mengecek” ketika dipenggal menjadi **meng**-ecek, padahal sebenarnya arti lain yaitu **menge**-cek, hal tersebut dikarenakan kata ecek terdapat dalam kata dasar.
5. Ketergantungan terhadap kamus / daftar kata dasar.
6. Penggunaan Bahasa Indonesia tidak konsisten dalam menentukan sistem, seperti contoh kata “adalah” merupakan kata memiliki kata arti sendiri akan tetapi dapat di penggal menjadi ada-**lah**. begitu juga kata “bagian”
7. Kata bentuk jamak
8. Kata serapan dari Bahasa asing
9. Kesalahan sistem penulisan
10. Akronim, seperti contoh “pemilu” dipenggal menajdi **pe**-milu

11. Noun (nama benda), sebagai contoh nama orang

Penelitian tentang *stemming* Bahasa Indonesia pertama kali dilakukan oleh Nazief –Adriani (1996) yang sekarang disebut algoritma Nazief – Adriani, akan tetapi masih ada kekurangan. Dikutip dari Tahitoe dan Purwitasai (2010). Beberapa algoritma juga telah dibuat untuk menyempurnakan algoritma Nazief – Adriani. Algoritma itu antara lain :

1. Algoritma *Stemming* Nazief –Adriani

Algoritma ini kembangkan berdasar aturan morfologi Bahasa Indonesia yang mengelompokkan imbuhan menjadi awalan, sisipan, akhiran, dan gabungan awalan dan akhiran. Aturan *morfologi* Bahasa Indonesia mengelompokkan imbuhan kedalam beberapa kategori sebagai berikut:

- 1) *Inflection suffix* yaitu kelompok akhiran yang tidak merubah karakter bentuk dasar. Sebagai contoh masuklah yang dapat dipenggal menjadi masuk-**lah**. kelompok ini dibagi menjadi 2 bagian :
 - a. *Particle* (P) yaitu termasuk didalamnya “-lah”, “-kah”, “-tah”, dan “-pun”.
 - b. *Possessive prounon* (PP) atau kata ganti kepunyaan, yaitu “-mu”, “-ku”, dan “-nya”.
- 2) *Derivation Suffixes* (DS) yakni kumpulan akhiran asli Bahasa Indonesia yang secara langsung ditambahkan pada kata dasar yaitu akhiran “-i”, “-kan”, dan “-an”.
- 3) *Derivation Prefixes* (DP) yaitu kumpulan awalan yang dapat langsung diberikan pada kata dasar murni, atau kata dasar yang sudah mendapatkan penambahan sampai dengan 2 awalan.
 - a. Awalan yang dapat bermorfologi (“me-“, “be-“, “pe-“, dan “te-“)
 - b. Awalan yang tidak dapat bermorfologi (“di-“, “ke-“, dan “se-“).

Berdasar pengklasifikasian imbuhan-imbuhan diatas, maka bentuk kata berimbuhan dalam Bahasa Indonesia dapat dimodelkan sebagai berikut:

$$[DP+[DP+[DP+]]] \text{ KATA DASAR } [[+DS][+PP][+P]] \quad (3.1)$$

Dengan model Bahasa Indonesia diatas serta aturan-aturan dasar *morfologi* Bahasa Indonesia, aturan dipergunakan dalam proses *stemming* algoritma Nazief – Adriani sebagai berikut :

- 1) Tidak semua kombinasi awalan dan akhiran diperbolehkan. Kombinasi tersebut adalah “be-i” , “di-an”, “ke-i”, “ke-kan”, “me-an”, “se-i”, “se-kan”, dan “te-an”
 - 2) Penggunaan imbuhan yang sama secara berulang tidak diperbolehkan.
 - 3) Jika suatu kata hanya terdiri dari satu atau dua huruf, maka proses *stemming* tidak dilakukan
 - 4) Penambahan suatu awalan tertentu dapat mengubah bentuk asli dasar kata, ataupun awalan yang telah diberikan sebelumnya pada kata bermorfologi. Sebagai contoh “me-” berubah menjadi “meng”, “meny”, “mem-” dan “men-”
2. Algoritma *Confix Stripping Stemmer* (CS Stemmer)
- Algoritma ini dikembangkan oleh Jelita (2007) dengan tujuan untuk memperbaiki kesalahan-kesalahan *stemming* yang masih dilakukan. Beberapa perbaikan yang dilakukan antara lain adalah penambahan kamus kata dasar lengkap, menambahkan aturan *stemming* untuk kata ulang, dan menambahkan proses pengecekan *ruleprecedence*. Proses pengecekan *ruleprecedence* akan menentukan proses *stemming* akan melakukan penghilangan akhiran atau awalan dulu.
3. Algoritma *Enhanced Confix Stripping Stemmer* (ECS stemmer)
- Algoritma ECS *stemmer* melakukan perbaikan dengan cara menambahkan algoritma untuk mengatasi kesalahan pemenggalan akhiran yang harusnya tidak dilakukan. Algoritma

ini disebut *loop* Pengembalian Akhiran, dan dilakukan apabila proses *recording* gagal.

3.4.4 Tokenisasi

Tokenisasi adalah tahap pemotongan *string* berdasarkan kata yang menyusun yang lalu disebut dengan token (Manning dkk. 2008). Tokenisasi juga menghilangkan *delimiter* dan tanda baca seperti tanda titik (.), koma(,), dan spasi. Proses tokenisasi mengandalkan karakter spasi pada dokumen teks untuk melakukan pemisahan sehingga hasil dari proses tokenisasi adalah kumpulan kata saja, baik itu kata yang penting maupun tidak penting.

3.4.5 Pembuangan *Stopword*

Stopword adalah kata umum yang biasanya muncul dalam jumlah besar disetiap dokumen dan dianggap tidak memiliki makna (Manning dkk, 2008). Proses penghapusan *stopword* dilakukan untuk setiap dokumen, apabila didalam dokumen ditemukan kata yang termasuk kedalam daftar *stopword* maka kata tersebut dihapus, sehingga dimensi dokumen menjadi berkurang

3.4.6 N-gram

Menurut Canvar dan Trenkle (1994) Ngram adalah potongan n-karakter dalam suatu *string*. Misalnya dalam kata “Banjir” akan didapatkan n-gram seperti Tabel 3.1

Tabel 3. 1 Contoh pemtongan N-gram berbasis karakter

Nama	n-gram karakter
Uni-gram	B, A, N, J, I, R
Bi-gram	_B, BA, AN, NJ, JI, IR, R_
Tri-gram	_BA, BAN, ANJ, NJI, JIR, IR_, R_ _

Karakter blank “_” digunakan untuk menunjukkan spasi didepan dan diakhir kata. Contoh untuk n-gram berbasis kata contohnya dalah sebagai berikut.

Kalimat “ Banjir sudah menggenang sebagian ruas jalanan di Jakarta”

Tabel 3. 2 Conoth pemotongan N-gram berbasis kata

Nama	n-gram kata
Uni-gram	Banjir, sudah, menggenang, sebagian, ruas, jalanan, di, Jakarta
Bi-gram	Banjir sudah, sudah menggenang, menggenang sebagian, sebagian ruas, ruas jalanan, jalanan di, di Jakarta
Tri-gram	Banjir sudah menggenang, sudah menggenang sebagian, menggenang sebagian ruas, sebagian ruas jalanan, ruas jalanan di, jalanan di Jakarta

3.5 Fitur dan Pembobotan

Pembobotan adalah metode mengubah input data menjadi suatu fitur vektor. Hal ini bertujuan untuk mengurangi kompleksitas dokumen dan membuat dokumen lebih mudah untuk diolah dalam proses identifikasi. Proses pembobotan kata ini memberikan nilai atau bobot ke sebuah kata atau fitur berdasarkan kemunculan dalam suatu teks dokumen. Metode ini menggunakan *bag-of-feature* yang mana sederetan fitur pada vector $\{f_1, f_2, f_3, \dots, f_n\}$ merupakan sekumpulan fitur-fitur sebanyak n yang sudah ditentukan sebelumnya.

3.5.1 Term Presence (TP)

Menurut O'Keefe dan Koprinska (2009) *Term Presense* adalah proses pembobotan yang kata yang paling sederhana dimana melihat keberadaan daftar kata-kata (*term*) terhadap satu dokumen teks. Pemberian nilai berdasar ada tidaknya *term* pada setiap dokumen teks. Nilai "0" diberikan jika *term i* tidak terdapat pada dokumen teks, sedangkan nilai "1" diberikan jika *term i* terdapat pada dokumen teks. Rumus yang dipakai dari fitur i pada dokumen j ditulis pada Persamaan 3.2

$$tp(t_i, d_j) = \begin{cases} 0 & : \text{term } i \text{ tidak terdapat pada dokumen } j \\ 1 & : \text{tem } i \text{ terdapat pada dokumen } j \end{cases} \quad (3.2)$$

3.5.2 Term Frequency (TF)

Menurut O’Keefe dan Koprinska (2009) *Term frequency* memiliki kesamaan dengan TP, tapi yang membedakan adalah metode pembobotan kata dengan menghitung frekuensi kemunculan kata atau *term* (t) pada dokumen teks. Misalkan suatu fitur berupa kata “banjir” muncul sebanyak 15 kali maka nilai fitur tersebut pada *feature vector* adalah 15 kali. Rumus TF dapat dilihat pada Persamaan 3.3

$$tf(t_i, d_j) = \text{frekuensi kemunculan term } i \text{ pada dokumen teks } j \quad (3.3)$$

Menurut Raschka (2014), term frequency sering di normalisasi dengan membagi TF dengan jumlah dokumen teks j . Normalisasi itu digunakan untuk menjaga dari bias dokumen yang panjang. Nilai TF lebih besar menghasilkan kemungkinan bahwa *term* itu kata kunci semakin besar. Rumus normalisasi TF dapat dilihat pada Persamaan 3.4

$$TF \text{ dinormalisasi} = \frac{tf(t_i, d_j)}{nd_j} \quad (3.4)$$

Dimana :

$tf(t_i, d_j)$ = *term frequency* (jumlah term t i pada dokumen d j)

nd_j = jumlah semua *term* dalam dokumen d teks j

3.5.3 Term Frequency Inverse Document Frequency (TF-IDF)

Term Frequency – Inverse Document Frequency (TF-IDF) adalah algoritma pembobotan yang disusun dari dua nilai yang berasal dari dua algoritma pembobotan yang berbeda yaitu TF dan IDF. IDF adalah perebaran *term* di kumpulan dokumen yang bernilai algoritma dari jumlah kumpulan dokumen (D) yang ada dibagi dengan banyaknya dokumen yang memiliki *term* (t_i) itu. Rumus IDF dapat dilihat pada Persamaan 3.5

$$idf(t_i, d_j) = \log \frac{|D|}{|D(t_i)|} \quad (3.5)$$

Dimana :

$|D|$ = jumlah kumpulan dokumen

$|D(t_i)|$ = banyaknya dokumen dimana suatu *term* t_i muncul

Setelah mendapatkan hasil IDF maka rumus TF-IDF dapat dilihat pada Persamaan 3.6

$$tfidf(t_i, d_j) = tf(t_i, d_j) \times idf(t_i, d_j) \quad (3.6)$$

Dimana :

i = kata ke $-i$

$tf-idf(t_i, d_j)$ = bobot kata ke $-i$ dalam dokumen d_j

$tf(t_i, d_j)$ = *term frequency* kata ke- i dalam dokumen d_j

$idf(t_i, d_j)$ = *inverse document frequency* kata ke- i dokumen d_j

3.6 Klasifikasi

3.6.1 Multinomial Naïve Bayes

Algoritma *naïve bayes* merupakan algoritma yang digunakan untuk mencari nilai probabilitas tertinggi untuk mengklasifikasi data uji pada kategori yang paling tepat (Feldman & Sanger 2007). *Naïve bayes* adalah algoritma yang menggunakan aturan Bayesian dan menganggap (secara naif) setiap kejadian yang bersifat lepas. *Naïve bayes* bekerja sangat efektif saat diuji pada dataset aktual, terutama dikombinasikan dengan prosedur pemilihan atribut (Witten et al., 2011). Walaupun *naïve bayes* sering dikalahkan oleh algoritma lain seperti SVM, *Random forest*, dan lain-lain. *Naïve bayes* memiliki keunggulan pada kemudahan komputasi dan komputasi yang tidak memakan banyak sumber daya. Algoritma *naïve bayes* juga bagus digunakan ketika keterbatasan memori dan CPU yang digunakan (Huang et al, 2003). Kelebihan algoritma ini sangat cocok di diterapkan untuk mengolah data penelitian ini.

Naïve bayes pada klasifikasi teks dilakukan dengan membandingkan nilai peluang kejadian bersyarat masing-masing kelas pada dokumen yang ingin diklasifikasi. Nilai peluang kejadian bersyarat yang tinggi maka yang

dipilih. Peluang kejadian bersyarat dalam kelas c pada dokumen d dapat dilihat pada Persamaan 3.7

$$P(c|d) = \prod_{1 \leq k \leq n_d} P(t_k|c) \quad (3.7)$$

Dimana

$P(c|d)$ = Peluang kejadian bersyarat kelas c di dokumen d

$P(t_k|c)$ = Peluang kejadian bersyarat *term* t_k pada kelas c

n_d = Jumlah token t pada dokumen d

Persamaan 3.7 menunjukkan untuk menghasilkan peluang kejadian bersyarat kelas c pada dokumen d , proses *naïve bayes* dilakukan dengan mengalikan peluang kemunculan c pada dokumen d dengan hasil pengalian dari peluang kejadian bersyarat kelas c pada dokumen d .

Pengalian peluang kejadian diatas memberikan hasil yang sangat kecil sehingga sulit untuk dipahami dan susah untuk komputer menyimpannya. Oleh karena itu Persamaan 3.6 dimodifikasi yang hasilnya seperti Persamaan 3.8

$$\log P(c|d) = \log P(c) + \sum_{1 \leq k \leq n_d} \log P(t_k|c) \quad (3.8)$$

Setiap kondisi parameter $\log P(t_k|c)$ adalah sebuah bobot yang menunjukkan seberapa bagus *term* t_k pada kelas c . Sama seperti $\log P(c)$ yang merupakan bobot yang menunjukkan frekuensi relative dari c .

Algoritma *multinomial naïve bayes* dilakukan dengan menghitung peluang kejadian *term* t pada kelas c dengan membagi jumlah kemunculan *term* t pada kelas c dengan jumlah kemunculan *term* t pada kelas c untuk seluruh *term* yang berada di dalam *corpus*. Algoritma MNB dapat digunakan untuk menghitung peluang kejadian yang ditunjukkan pada Persamaan 3.9

$$P(t|c) = \frac{T_{ct}}{\sum_{t \in V} T_{ct}} \quad (3.9)$$

Dimana

T_{ct} = Jumlah kemunculan *term t* di dalam dokumen *training* pada kelas *c*

$\sum_{t \in V} T_{ct}$ = Jumlah kemunculan *term t* pada kelas *c* yang berada didalam *corpus (vocab)*

Pada *multinomial Naïve Bayes*, setiap *term* yang muncul akan ditambahkan dalam *corpus*. Oleh karena itu didalam satu dokumen, *term* yang muncul lebih dari satu maka akan disimpan sesuai kemunculan *term* tersebut. Pada Persamaan 3.8 akan ada masalah ketika *term* tidak ada pada dokumen, yang otomatis $P(t|c)$ akan bernilai 0. Oleh karena itu dilakukan *laplace smoothing* yang akan menghasilkan Persamaan 3.10

$$P(t|c) = \frac{T_{ct} + \alpha}{(\sum_{t \in V} T_{ct}) + \alpha \cdot B} \quad (3.10)$$

Dimana :

T_{ct} = Jumlah kemunculan *term t* di dalam dokumen *training* pada kelas *c*

$\sum_{t \in V} T_{ct}$ = Jumlah kemunculan *term t* pada kelas *c* yang berada didalam *corpus*

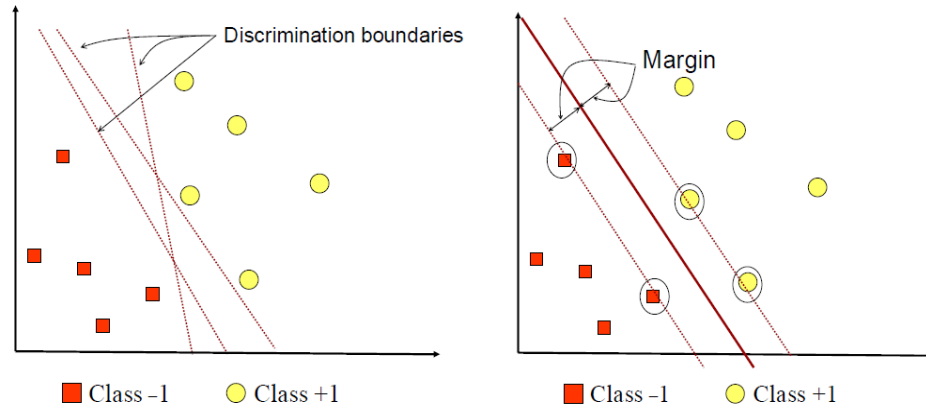
α = sebuah parameter *addictive smoothing* ($\alpha=1$ untuk *laplace smoothing*)

B = jumlah *corpus* yang berada diseluruh kelas (jumlah vocabulary)

3.6.2 Support Vector Machine

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas pada *input space*. Pola yang merupakan anggota dari dua buah kelas : +1 dan -1 dan berbagi alternatif garis pemisah (*discrimination boundaries*). *Margin* adalah jarak antara *hyperplane* tersebut dengan pola terdekat dari masing-masing kelas. Pola yang paling dekat ini disebut sebagai *support*

vector. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran SVM



Gambar 3. 2 Bagaimana SVM bekerja mencari hyperlane terbaik

Gambar 3.2 menunjukkan bagaimana cara SVM bekerja yaitu dengan menemukan *hyperlane* terbaik, persebaran data ditunjukkan oleh warna merah dan warna kuning (Nugroho, 2003). Data berwarna merah menunjukkan kelas -1 dan data berwarna kuning menunjukkan kelas +1. *Hyperlane* terbaik didapatkan dari jarak *margin* terbaik. *Margin* adalah jarak *hyperlane* dengan data terdekat tiap kelas.

Menurut Nugroho (2003), data yang tersedia dinotasikan sebagai $\vec{x} \in R^d$ sedangkan label masing-masing dinotasikan $y_i \in \{-1, +1\}$ untuk $I = 1, 2, \dots, l$, yang mana l adalah banyaknya data. Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh hyperplane berdimensi d , yang didefinisikan:

$$\vec{w} \cdot \vec{x} + b = 0 \quad (3.11)$$

Pattern \vec{x}_i yang termasuk kelas -1 (sampel negatif) dapat dirumuskan sebagai pattern yang memenuhi pertidaksamaan:

$$\vec{w} \cdot \vec{x} + b \leq -1 \quad (3.12)$$

Sedangkan \vec{x}_i yang termasuk kelas +1 (sampel positif) memenuhi pertidaksamaan:

$$\vec{w} \cdot \vec{x} + b \geq 1 \quad (3.13)$$

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara hyperplane dan titik terdekatnya, yaitu $1/||w||$. Hal ini dapat dirumuskan sebagai *Quadratic Programming (QP) problem*, yaitu mencari titik minimal Persamaan 3.14, dengan memperhatikan constraint Persamaan 3.15.

$$\min_{\vec{w}} \tau(\vec{w}) = \frac{1}{2} ||\vec{w}||^2 \quad (3.14)$$

$$y_i(\vec{x}_i \cdot \vec{w} + b) \geq 1, \forall i \quad (3.15)$$

Penjelasan diatas berdasarkan asumsi bahwa kedua belah kelas dapat terpisah secara sempurna oleh hyperlane. Akan tetapi, pada kenyataannya dua buah class tidak terpisah secara sempurna. Hal ini menyebabkan nilai *constraint* pada Persamaan 3.15 tidak dapat terpenuhi. Untuk mengatasi ini, SVM dirumuskan ulang dengan memperkenalkan teknik *softmargin*. Dalam *softmargin*, Persamaan 3.15 dimodifikasi dengan memasukkan *slack variable* S_i ($S_i > 0$) menjadi:

$$y_i(\vec{x}_i \cdot \vec{w} + b) \geq 1 - S_i, \quad \forall i \quad (3.16)$$

Dengan demikian persamaan 3.14 dapat diubah menjadi:

$$\min_{\vec{w}} \tau(\vec{w}, S) = \frac{1}{2} ||\vec{w}||^2 + C \sum_{i=1}^l S_i \quad (3.17)$$

Parameter C dipilih untuk mengontrol *tradeoff* antara margin dan klasifikasi error S. Nilai C yang besar berarti akan memberikan penalti yang lebih besar terhadap klasifikasi tersebut.

3.7 Evaluasi Performa Model

Model klasifikasi yang dibuat ialah pemetaan dari suatu baris data dengan luaran sebuah prediksi kelas/target dari data tersebut. Klasifikasi yang memilikidua kelas sebagai keluarannya disebut dengan klasifikasi biner. Kedua kelas tersebut bisa direpresentasikan dalam $\{0,1\}$, $\{+1, -1\}$, atau $\{\text{positive}, \text{negative}\}$. (Rianto, 2016)

Dalam proses evaluasi klasifikasi data terdapat 4 kemungkinan dari hasilnya. Jika data positif dan prediksi positif maka diklasifikasikan sebagai

true positive. Jika data positif dan prediksi negatif maka diklasifikasikan sebagai *false negative*. Jika data negatif dan prediksi data negatif maka diklasifikasikan sebagai *true negative*. Jika data negatif dan prediksi data positif maka diklasifikasikan sebagai *false negative*. Hasil dari evaluasi data dan klasifikasinya dapat direpresentasikan dengan matrix 2x2 yang disebut *confusion matrix* (Fawcett, 2016). Tabel *confusion matrix* dapat dilihat pada Tabel 3.3

Tabel 3. 3 Confusion matrix

Prediksi data	Data kelas	
	<i>Positive</i>	<i>Negative</i>
<i>Positive</i>	<i>True positive</i>	<i>False positive</i>
<i>Negative</i>	<i>False negative</i>	<i>True negative</i>

Terdapat beberapa rumus umum yang dapat digunakan untuk menghitung performa klasifikasi. Antara lain adalah akurasi, *precision*, *recall*, dan *f-measure*.

$$akurasi = \frac{TP + TN}{TP + FP + FN + TN} \quad (3.18)$$

$$Precision = \frac{TP}{TP + FP} \quad (3.19)$$

$$recall = \frac{TP}{TP + FN} \quad (3.20)$$

$$f - measure = 2 \frac{precision \times recall}{precision + recall} \quad (3.21)$$

3.7.1 K-Fold Cross Validation

Cross validation dilakukan dengan membagi data menjadi beberapa *fold* (bagian). *Cross validation* ini dilakukan dengan melakukan sesuai jumlah k. Untuk mendapatkan hasil terbaik, jumlah k biasanya berjumlah 10, hal ini sudah dibuktikan dalam beberapa penelitian sebelumnya. (Witten et al. 2011)

Menurut kohavi (1995) *k-fold cross validation* diawali dengan membagi jumlah data dalam yang berjumlah k dengan bagian setiap data sama misalnya $d_1, d_2, d_3, \dots, d_k$, dan selanjutnya proses *testing* dan *training* dilakukan sebanyak k kali dengan ketentuan iterasi ke- i dalam data $d(i)$ akan menjadi data *testing* dan sisanya akan menjadi data *training*.

BAB IV

ARSITEKTUR DAN PERANCANGAN SISTEM

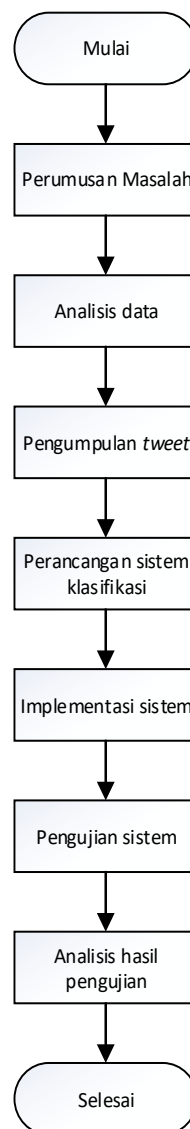
4.1 Rancangan Sistem

Pada Bab ini akan dijelaskan tentang analisis dan perancangan sistem yang akan dibangun untuk mengidentifikasi kata “banjir” dengan menggunakan data yang berasal dari *Twitter*. Dalam analisis dan perancangan sistem ini meliputi deskripsi sistem, persiapan data *Twitter*, *preprocessing*, pembobotan kata dan metode klasifikasi teks menggunakan pemodelan *multinomial naïve bayes* dan *support vector machine*.

Pada awalnya, sebelum pembuatan sistem dilakukan analisis terhadap data yang akan digunakan dalam sistem. Data set yang digunakan adalah data yang diambil dari *Twitter* yang setiap *tweet* mengandung kata “banjir” yang lokasinya berada di DKI Jakarta. Lalu dalam pengolahan sistem dilakukan identifikasi terhadap kata “banjir” dan dikategorikan menjadi 3 kelas yaitu banjir dengan artian sesungguhnya, tidak banjir atau *unknown*.

Kata banjir yang berada pada *tweet* akan diujikan pada sistem yang lalu apakah kata tersebut telah berhasil diklasifikasi sesuai dengan kategori yang telah ditentukan atau belum. Hasil pengujian akan dianalisis menggunakan tabel untuk mendapatkan nilai akurasi, *precision*, *recall* dan *fmeasure* untuk mengetahui keakuratan sistem.

Alur proses analisis dan perancangan penelitian dapat dilihat pada diagram alur dalam Gambar 4.1



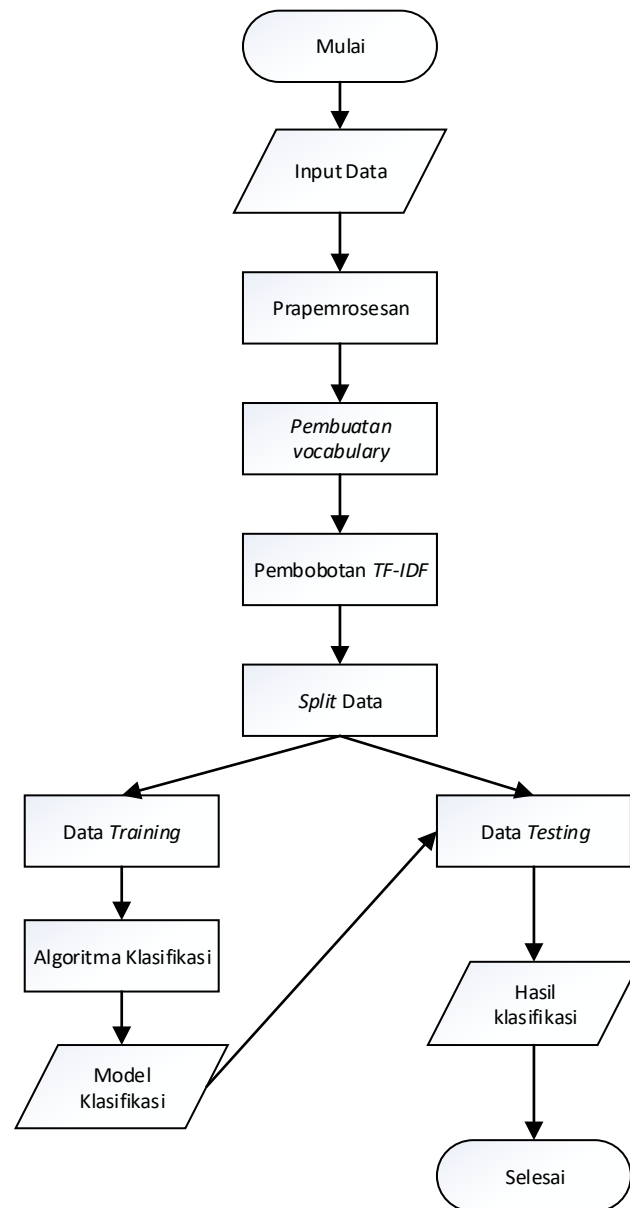
Gambar 4. 1 Flowchart proses perancangan dan analisis sistem

4.2 Deskripsi Sistem

Pada penelitian ini akan dilakukan klasifikasi kata banjir. Oleh sebab itu, diperlukan data *tweet* yang mengandung kata “banjir” yang selanjutnya akan dilakukan *preprocessing* pada setiap *tweet* yang telah dikumpulkan. Secara keseluruhan sistem yang akan dibuat terbagi menjadi beberapa tahap, yaitu *preprocessing*, pembobotan dengan TF-IDF, lalu mengklasifikasikan data *tweet* dengan menggunakan metode *multinomial naïve bayes* dan *support vector machine*.

Sebelum membangun sistem, dilakukan pengumpulan data *tweet* yang mengandung kata "banjir" lalu dilakukan pelabelan secara manual berdasarkan kategori yang sudah ditentukan. Dan setelah melakukan pelabelan *tweet* maka akan dilanjutkan dengan *preprocessing*. *Preprocessing* ini digunakan untuk mengurangi *noise* pada data yang dipakai. Adapun fungsi *preprocessing* yang dipakai antara lain *cleansing*, *casefolding*, *stemming*, *tokenizing*, pembuangan *stopword* kata dari Bahasa Indonesia. Setelah didapatkan hasil *preprocessing*, akan dilanjutkan dengan menentukan fitur-fitur yang mewakili setiap kata yang ada setiap dokumen yang mana dokumen ini adalah 1 baris *tweet*. Tahap penentuan fitur dilakukan dengan pembuatan daftar *vocabulary* serta pembobotan TF-IDF. Pembuatan *vocabulary* dilakukan untuk menentukan fitur apa saja yang digunakan berdasar dari berapa sering sebuah kata muncul didalam data set.

Setelah mendapat fitur yang diperoleh, dilakukan proses klasifikasi menggunakan model *multinomial naïve bayes* dan *support vector machine*. Dalam proses klasifikasi akan dibagi menjadi dua proses yaitu proses *training* dan proses *testing*. Proses *training* akan memberikan hasil berupa model klasifikasi yang selanjutnya digunakan dalam proses *testing*. Dalam menentukan model terbaik dan evaluasi model digunakan *K-Fold Cross Validation*. Dan dalam penelitian ini digunakan *10-Fold Cross Validation*. Berikut alur proses untuk deskripsi sistem dapat dilihat pada diagram alur dalam Gambar 4.2



Gambar 4. 2 Diagram alur deskripsi sistem

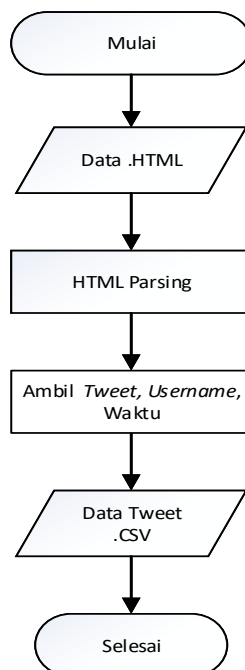
4.3 Data dan Kelas Data

4.3.1 Data Stopword

Dalam penelitian ini, *preprocessing* akan menyaring kata-kata dalam dokumen yang kurang penting. Penyaringan ini menggunakan kamus kata tertentu, yang disebut database *stopword*. Data *stopword* yang akan digunakan dalam penelitian ini menggunakan database *stopword* yang telah dikumpulkan oleh Tala (2013) yang berisikan kata-kata dalam Bahasa

Dalam penelitian ini data diambil dengan cara membuka situs *twitter.com/search-advanced* yang mana memasukkan kata kunci “banjir” pada kolom pencarian, memasukkan tanggal pencarian dan lokasi yang ditentukan yaitu DKI Jakarta. Setelah itu muncul halaman hasil pencarian seperti yang diisikan sebelumnya.

Halaman hasil pencarian akan menampilkan data yang terbatas, akan tetapi jika *scrolling* ke halaman bawah secara manual akan muncul data lanjutan, oleh karena itu dibuat program agar dapat melakukan *auto scrolling* untuk halaman hasil pencarian tersebut sampai waktu yang telah ditentukan sebelumnya dan menyimpan dalam format HTML. Setelah itu data dari HTML diambil sesuai kebutuhan menggunakan teknik *parsing*. Pada penelitian ini data yang diambil adalah *tweet*, *username*, dan waktu lalu menyimpan dalam dokumen CSV. Berikut diagram alur untuk *parsing* dapat dilihat pada Gambar 4.4



Gambar 4. 4 Diagram alur *parsing* HTML

4.3.3 Data Kata Dasar

Dalam penelitian ini, kata dasar yang akan digunakan merupakan kata yang diambil pada *library* Sastrawi (2014), yang mana data tersebut berjumlah 29932 kata dalam Bahasa Indonesia.

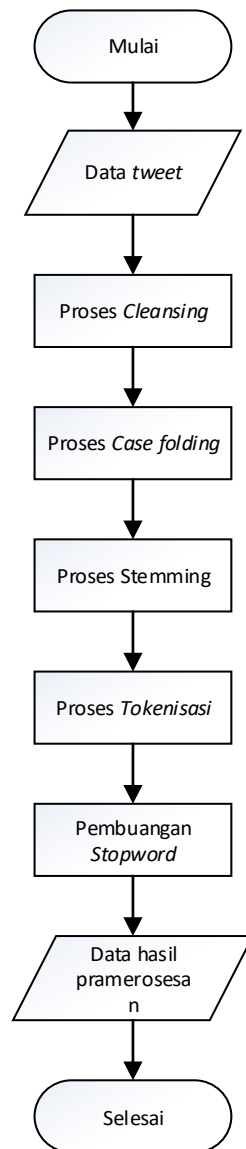
4.4 Preprocessing Data (Prapemrosesan Data)

Pada tahap ini, akan dilakukan *preprocessing* data yang mana akan dilakukan sebelum menjalankan metode klasifikasi dokumen teks. Data *tweet* yang didapatkan merupakan data yang masih asli dan tidak bisa dilakukan klasifikasi karena data banyak memuat tanda baca, angka dan kata-kata yang lain yang kurang bermakna dan tidak bisa dijadikan fitur. Selain itu, perlu juga dilakukan penyeragaman agar *feature space* menjadi kecil sehingga akan mempermudah proses klasifikasi. Maka dari itu sebelum fitur diekstrak dari data *tweet*, perlu dilakukan *preprocessing* data yang bertujuan untuk menghilangkan karakter-karakter dari huruf, mengurangi volume kosakata dan tentunya menyeragamkan bentuk kata.

Proses Preprocessing meliputi :

- a) Proses *cleansing*
- b) Proses *case folding*
- c) Proses *stemming*
- d) Proses tokenisasi
- e) Proses pembuangan stopword

Setiap proses preprocessing akan dijelaskan pada sub bab berikutnya. Alur dari prapemrosesan data dapat dilihat pada Gambar 4.5



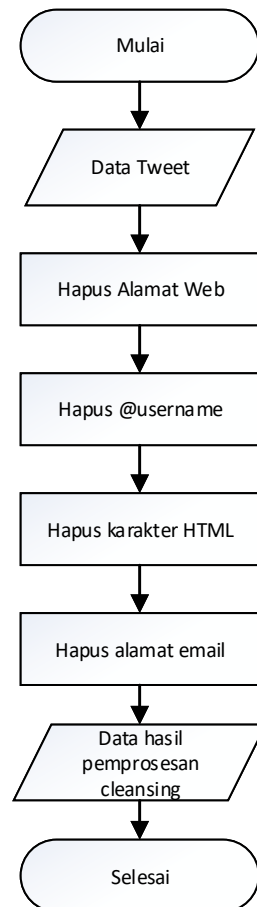
Gambar 4. 5 Diagram alur prapemrosesan

4.4.1 Perancangan *Cleansing*

Dalam tahap ini, akan dilakukan pembersihan data tweet dari noise sehingga lebih bermakna. Data *tweet* yang akan di *cleansing* merupakan data yang telah di simpan sebelumnya dalam bentuk .csv. Adapun kata-kata yang dihilangkan antara lain :

- Alamat web
- @username dalam tweet
- Karakter HTML
- Alamat email

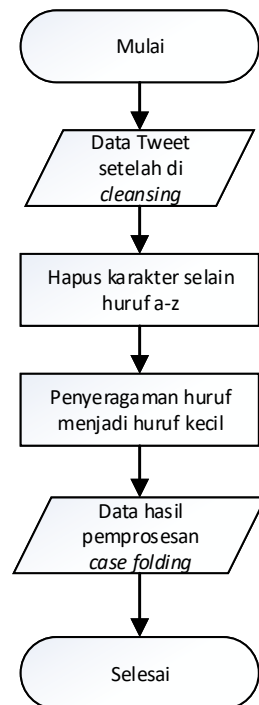
Alur proses *cleansing* dapat dilihat pada gambar 4.6



Gambar 4. 6 Alur proses *cleansing*

4.4.2 Perancangan *Case Folding*

Proses ini merupakan proses penyeragaman huruf serta penghilangan tanda baca. Dalam hal ini *casefolding* akan hanya mengeluarkan huruf latin a sampai dengan z. Karakter lain selain huruf dianggap *delimiter* sehingga karakter tersebut akan dihapus dari *tweet*. Diagram alur proses *case folding* dapat dilihat pada Gambar 4.7



Gambar 4. 7 Diagram alur proses *case folding*

Proses *case folding* dan *cleansing* setelah dilakukan maka akan menghasilkan data yang seragam. Contoh hasil proses *case folding* dapat dilihat pada Gambar 4.8 dan 4.9

#BeritaHarian: Hujan Deras di Mana Saja Titik Banjir Jakarta??
<http://bit.ly/1swfEB?>

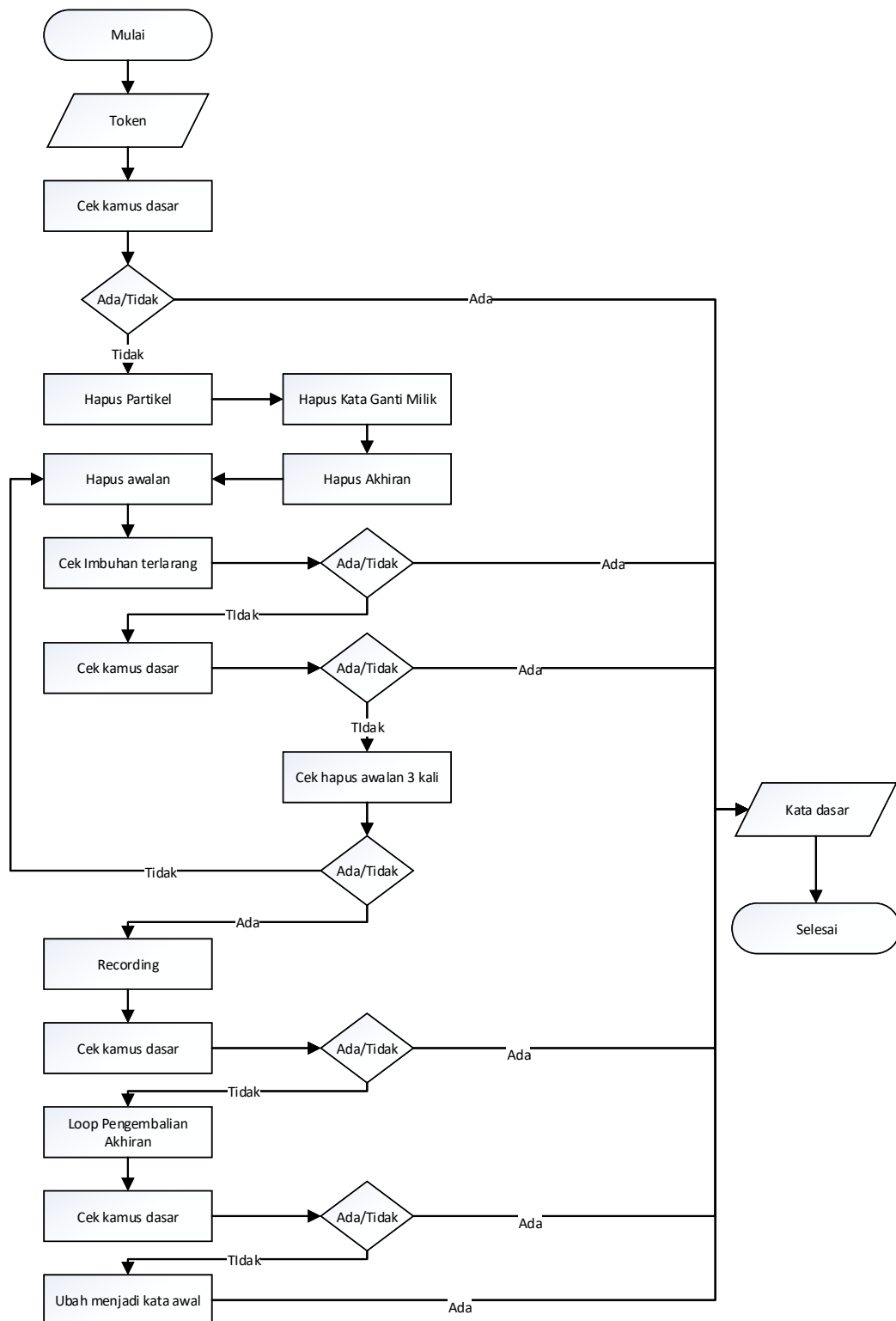
Gambar 4. 8 Data berita sebelum di proses *case folding* dan *cleansing*

beritaharian hujan deras di mana saja titik banjir Jakarta

Gambar 4. 9 Data berita setelah diproses *case folding* dan *cleansing*

4.4.3 Perancangan *Stemming*

Stemming adalah pemotongan imbuhan pada kata atau term. Proses *Stemming* berguna untuk mengubah kata atau term berimbuhan menjadi kata dasar. Algoritma yang digunakan pada penelitian ini ialah algoritma *Enhanced Confix Stripping Stemming* (ECSS). Alur proses *stemming* dapat dilihat pada Gambar 4.10



Gambar 4. 10 Alur proses perancangan *stemming*

4.4.4 Perancangan Tokenisasi

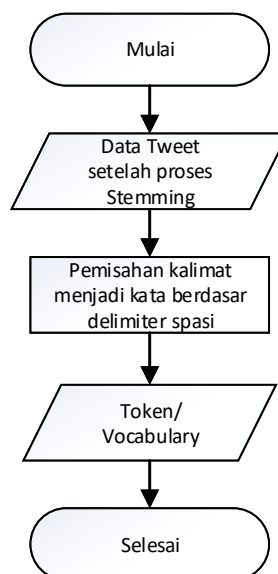
Tahap tokenisasi dilakukan proses penguraian teks yang semula berupa kalimat-kalimat menjadi kata-kata berdasarkan karakter spasi.

Contoh hasil tweet setelah tokenisasi dapat dilihat pada Gambar 4.11

Sebelum tokenisasi	beritaharian hujan deras di mana saja titik banjir Jakarta
Setelah tokenisasi	'beritaharian', 'hujan', 'deras', 'di', 'mana', 'saja', 'titik', 'banjir', 'Jakarta'

Gambar 4. 11 Contoh hasil tokenisasi

Alur proses perancangan tokenisasi dapat dilihat pada Gambar 4.12



Gambar 4. 12 Alur proses perancangan tokenisasi

4.4.5 Perancangan Pembuangan *Stopword*

Proses *stopword* merupakan proses untuk menghilangkan kata-kata yang kurang bermakna dalam Bahasa Indonesia dimana kata-kata tersebut berasal dari *database stopwords*. Apabila dalam dokumen terdapat kata yang termasuk didalam daftar *stopword*, maka otomatis akan dihapus karena kata-kata tersebut umum dan tidak mencirikan dokumen. Gambar 4.13 proses pembuangan *stopword*.



Gambar 4. 13 Flowchart pembuangan *stopword*

Pada Gambar 4.13 ditunjukkan alur proses dalam menghilangkan *stopword* dan berikut langkah-langkah prosesnya :

1. Kata yang telah diproses pada tokenisasi disimpan dalam bentuk array.
2. Setiap kata (array) tersebut kemudian akan dibandingkan dengan kata-kata yang ada di *database stopwords*.
3. Jika kata yang dibandingkan dengan kata yang terdapat dalam *database stopwords* ada, maka kata tersebut dihapus secara otomatis. Tapi jika kata yang dibandingkan tidak ada didalam *database stopwords* maka kata tersebut tidak dihapus. Sisa kata disimpan dalam bentuk array
4. Proses pembuangan *stopwords* selesai

Contoh proses pembuangan *stopwords* dapat dilihat pada Gambar 4.14

Sebelum Pembuangan <i>stopword</i>	beritaharian hujan deras di mana saja titik banjir Jakarta
Setelah pembuangan <i>stopword</i>	beritaharian hujan deras titik banjir jakarta

Gambar 4. 14 Proses pembuangan *stopword*

4.4.6 Perancangan Running Time Preprocessing

Dalam tahap ini dilakukan perhitungan waktu seberapa lama proses yang dibutuhkan data *training* dalam *preprocessing*. Untuk tahap *preprocessing* yang dilakukan yaitu *cleansing*, *case folding*, *stemming*, *tokenisasi*, dan pembuangan *stopword*.

4.5 Perancangan Pembuatan *Vocabulary*

Dalam proses pembuatan *vocabulary* dilakukan pencarian jumlah kata yang paling banyak muncul sesuai dengan metode yang digunakan, kata tersebut nantinya akan digunakan sebagai fitur untuk mewakili. Hal ini digunakan agar dalam proses TF-IDF tidak terjadi *over-features*.

4.6 Perancangan TF-IDF

Pada penelitian ini, pembobotan kata menggunakan metode TF-IDF. Proses ini dilakukan setelah setiap kata pada *tweet* telah di hilangkan imbuhan kata (*stemming*) dan pembuangan *stopword*. Pembobotan TF-IDF bertujuan untuk menentukan bobot kata yang ada pada *tweet*, sehingga didapat bobot kata yang dapat mepresentasikan informasi sebagai fitur yang akan digunakan untuk melakukan klasifikasi *tweet*.

4.7 Klasifikasi

Pada penelitian ini menggunakan metode klasifikasi *Multinomial Naïve Bayes* dan *Support Vector Machine*. Proses awal dalam Klasifikasi *tweet* adalah pelabelan data training *tweet*.

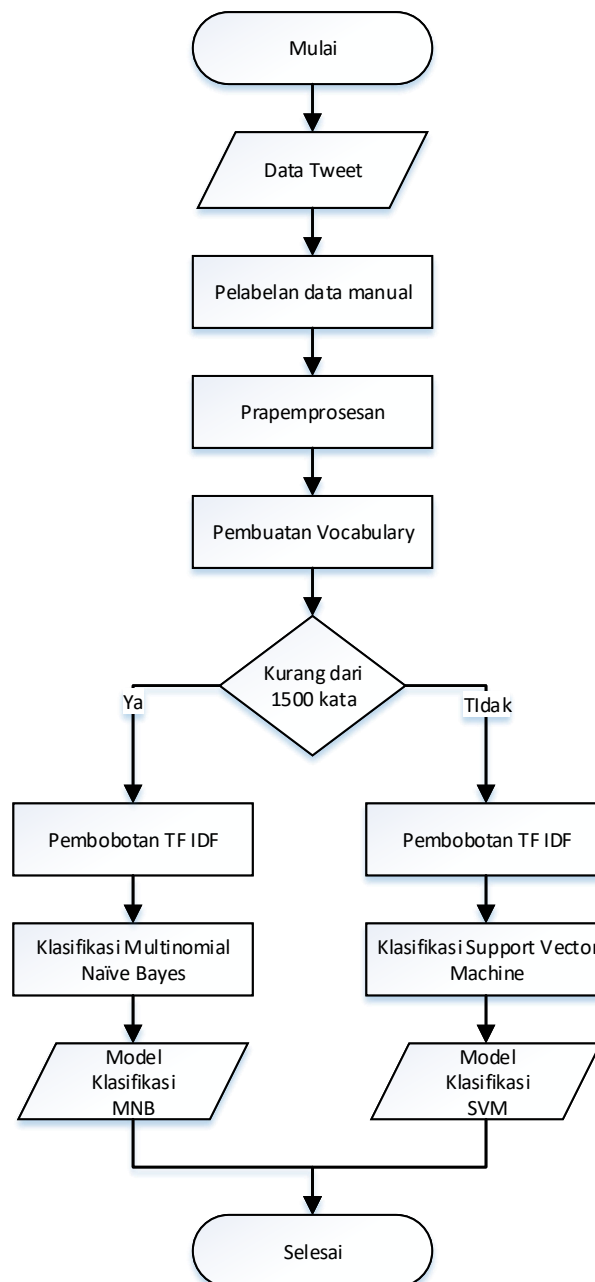
4.7.1 Pelabelan Data

Pada tahap ini, data *training* akan dilakukan pelabelan secara manual. Pelabelan dilakukan berdasar kategori kata kunci banjir pada *tweet* yang telah ditentukan. Kategori yang digunakan sebagai label dalam penelitian ini terdiri dari 3 kategori, yaitu:

1. Banjir (Bencana) : untuk menginformasikan keadaan suatu wilayah sedang terjadi banjir. Label yang digunakan yaitu “3”
2. Tidak banjir (Bencana) : untuk menginformasikan keadaan suatu tempat tidak terdapat genangan air, juga dapat menginformasikan banjir yang belum terjadi, fakta tentang banjir, dan banjir yang tidak terjadi pada waktu sekarang. Label yang digunakan yaitu “2”
3. Unknown (Bukan Bencana) : untuk *tweet* yang tidak mengandung ciri khas banjir dan tidak banjir, sebagai contoh banjir kejutan, banjir hadiah dan banjir sindiran. Label yang digunakan yaitu “1”

4.7.2 Perancangan *Training*

Data training yang telah digunakan akan diujikan dengan fitur yang telah didapatkan dari proses pembobotan setiap kata pada *twitter*. Untuk klasifikasi data akan digunakan metode *multinomial naïve bayes* dan *support vector machine*. Langkah-langkah proses *training* klasifikasi data *twitter* menggunakan metode klasifikasi *multinomial naïve bayes* dan *support vector machine* dapat dilihat pada Gambar 4.15. Untuk metode klasifikasi *support vector machine* prosesnya sama seperti *multinomial naïve bayes*.



Gambar 4. 15 Flowchart proses *training*

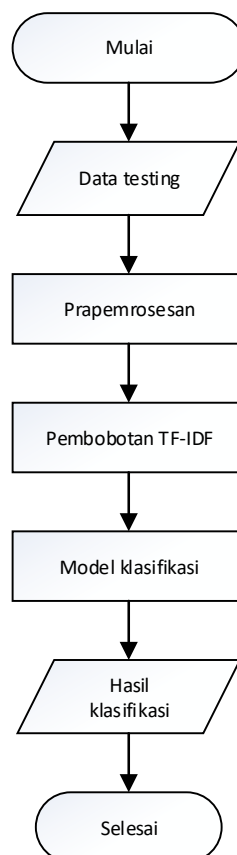
Pada Gambar 4.15 memperlihatkan proses training dari klasifikasi *multinomial naïve bayes*. Penjelasan dari Gambar tersebut sebagai berikut:

1. Data tweet yang mengandung kata banjir diberi label secara manual lalu dilakukan prapemrosesan.

2. Setelah data bersih maka dilakukan pembobotan kata dan seleksi fitur (*vocabulary*) yang mewakili sebagai representasi data yang digunakan
3. Pada proses seleksi fitur ditentukan, jika fitur kurang dari 1500 maka metode yang digunakan *multinomial naïve bayes* dan jika fitur yang digunakan 1500 maka digunakan metode *support vector machine*.
4. Setelah itu akan menghasilkan model klasifikasi untuk masing-masing metode yang nantinya digunakan untuk *testing*.
5. Proses *training* selesai.

4.7.3 Perancangan *Testing*

Data *testing* diklasifikasikan menggunakan model klasifikasi yang merupakan hasil dari proses *training*. Langkah langkah proses *testing* klasifikasi data twitter dapat dilihat pada Gambar 4.16.



Gambar 4. 16 Flowchat proses *testing*

Penjelasan dari Gambar 4.16 sebagai berikut:

1. Data *testing* merupakan data *tweet* yang mengandung kata banjir yang belum diketahui kelasnya atau labelnya. Selanjutnya data akan dilakukan prapemrosesan.
2. Tahap selanjutnya yaitu dilakukan pembobotan kata dan seleksi fitur yang dapat merepresentasikan data.
3. Data akan dicocokkan dengan model klasifikasi terbaik sehingga muncul hasil kelas atau labelnya.

Hasil dari klasifikasi dikelompokkan bedasar hari dan divalidasi dengan data kenyataan yang terjadi. Data yang digunakan untuk validasi adalah data yang bersumber dari situs data.jakarta.co.id (Data Jakarta, 2015).

4.7.4 Perancangan Evaluasi dan Validasi Klasifikasi

Pengujian pada penelitian ini dilakukan untuk mendapatkan fitur dan model klasifikasi terbaik dalam klasifikasi kata banjir pada *tweet*. Fitur terbaik didapatkan dengan memilih parameter-parameter terbaik dalam proses ekstraksi fitur. Sedangkan untuk model klasifikasi terbaik diperoleh dengan memilih parameter-parameter terbaik dalam proses *training* yang sehingga mendapatkan akurasi yang terbaik.

Untuk pengujian, digunakan metode *k-fold cross validation* dengan nilai $k = 10$. Proses evaluasi dengan *10-fold cross validation* dilakukan dengan membagi data set menjadi 10 bagian dengan persentase sama untuk masing-masing kelas, lalu diambil 9/10 untuk data training dan sisanya untuk pengujian. Evaluasi performa dari proses klasifikasi *tweet* ini dengan membandingkan hasil data pengujian model klasifikasi dengan data pelatihan yang dilabeli secara manual.

BAB V

IMPLEMENTASI SISTEM

Dalam bab ini akan dijelaskan mengenai implementasi dari sistem yang telah dirancang sebelumnya. Dalam bab ini penjelasan mengenai hasil implementasi dibagi menjadi sub bab yang terdiri dari spesifikasi sistem pengembangan, dan implelementasi dari masing masing *activity* yang telah dirancang pada bab sebelumnya.

5.1 Spesifikasi Sistem Pengembangan

Lingkungan implementasi klasifikasi kata banjir meliputi perangkat keras dan perangkat lunak yang digunakan untuk proses implementasi. Perangkat keras yang digunakan dalam proses implementasi sebagai berikut:

1. Processor : Intel Core i5-5200
2. Hardisk : 1 TB
3. Memory : 4 GB

Sedangkan untuk lingkungan perangkat lunak yang digunakan dalam proses implementasi sebagai berikut:

1. Sistem operasi : Microsoft Windows
2. Bahasa pemrograman : Python 3.5.1
3. IDE : Anaconda 2.4.1
4. Web browser : Google chrome

5.2 Implementasi Sistem

Pada bagian ini dijelaskan tentang implementasi dari sistem yang telah dirancang. Secara umum, implementasi sistem dibagi menjadi 5 bagian yaitu implementasi *preprocessing*, implementasi TF-IDF, implementasi klasifikasi menggunakan metode *multinomial naïve bayes* dan *support vector machine*, implementasi pengujian dengan data *testing*, dan implementasi pengujian sistem menggunakan *k-fold cross validation*. Proses implementasi *preprocessing* sudah di jelaskan pada Diagram 4.4.

Model klasifikasi yang digunakan yaitu *multinomial naïve bayes* dan *support vector machine* yang mana telah disediakan oleh python.

5.3 Implementasi Pelabelan Data

Terdapat beberapa label data yang telah dijelaskan pada sub bab 4.3.1 bahwa data *tweet* yang digunakan berjumlah 7789 *tweet* dalam kurun waktu 12 November 2014 sampai 30 Desember 2014. Pelabelan dilakukan sebelum *tweet* di lakukan *preprocessing* dan pelabelan dilakukan secara manual dengan cara membaca setiap *tweet* yang mengandung kata banjir dan mengkatagorikannya kedalam kelas yang telah ditetapkan. Berikut Gambar 5.1 menunjukkan cuplikasi data yang telah diberikan label atau kelas yang tesimpan dalam file .csv

	A	B	C	D
1	@budi18	12/30/2014	Banjir? Banjir Yg mana? Ngga ada banjir hanya air tergenang kata metrotipu @estiningsihdwi: Kabarnya jakarta banjir p	3
2	@fatheen	12/30/2014	Apa ??u dikata masih ada yg butuh Pesta "@RIFQI: astagfirullah jakarta banjir ahok ramein pesta tahun baru pake dan	2
3	@rfmwl	12/30/2014	astagfirullah jakarta banjir ahok ramein pesta tahun baru pake dana 1 M. teganya dikau. Malaysia batalin itu krn banjir	3
4	@Alviann	12/30/2014	Jakarta suka banjir karena.....	2
5	@ahmadg	12/30/2014	Mesti semua ulah avatar. Tanah: Longsor banjarnegara. Api: kebakaran pasar klewer. Air: Banjir jakarta. Udara :Jatuhny	2
6	@PuisiSel	12/30/2014	#Sabtu Info Jakarta @infojekardah :Sabtu Pagi Jakarta Timur Banjir Lagi - http://Tempo.co?? http... http://goo.gl/P	3
7	@ardorefi	12/30/2014	Karena banya orang jones:D"@Di_Kepoin: #Dikepoin Jakarta suka banjir karena....."	2
8	@MFitrah	12/30/2014	@AntiLiberalNews: Batalkan Acara Sambut Tahun Baru Malaysia Alihkan Dana Untuk Korban Banjir http://tinyurl.com/	2
9	@DetikRu	12/30/2014	?Dhekhadekoyy? @dhekhadedis :Hujan Deras di Mana Saja Titik Banjir Jakarta? http://tempo.co/s/1631190?... http	2
10	@puisinic	12/30/2014	#Katanya Muhammadfikhans: yang jomblo kalau mlm minggu minta hujanwaeee": Jakarta suka banjir karena....." htt	2
11	@ramaele	12/30/2014	terlalu banyak bangunan"@NotersSobat: Jakarta suka banjir karena....."	2
12	@billysbs	12/30/2014	banjir "@Bobotoh_Kepoo: #Gimel jakarta identik dengan..."	2
13	@Khulafa	12/30/2014	Pada tahun 1621 untuk pertama kalinya Batavia (Jakarta) mengalami banjir besar.	2
14	@syafniar	12/30/2014	Djarot Sesumbar Banjir di Jakarta Tidak Lama PIYUNGAN ONLINE http://fb.me/4bEqLBfNh?	2
15	@KINGYU	12/30/2014	JAKARTA BANJIR \nbilang jokowi ahok bohong \npdhal smua salah manusianya MUARA DIBIKIN RUMAH \nair ngalir r	2
16	@dianrap	12/30/2014	20 Januari 2015 Diprediksi Puncak Banjir Jakarta: Badan Nasional Penanggulangan Bencana (BNPB) memprediksi pun...	2
17	@Jrkengk	12/30/2014	Resolusinya 2014 pengen bantuin jakarta bebas banjir pake gayung min?@TheComment_NET #TheCommentHampirAk	2
18	@saronda	12/30/2014	SEBENERNYA KAMI BISA SAJA MEMBERI SOLUSI YG MUNGKIN BISA SEDIKIT MENGURANGI PENGANGURAN DI JAKARTA C	2
19	@diiniiii	12/30/2014	Tanah: bencana longsor banjarnegara Air: banjir diaceh bandung jakarta Api: kebakaran dipasar klewer solo Udara: hil	2
20	@Burung	12/30/2014	Thx geng huruhara yuhuu RT @BIGREDS_SERANG: Juara sebenarnya adalah semua regional perusak NASA. bocorborcor.	2
21	@BIGRED	12/30/2014	Juara sebenarnya adalah semua regional perusak NASA...bocor...bocor...fokus Jakarta ajur keseret banjir :))	2
22	@sameol	12/30/2014	Effect kelantan n jakarta banjir "@areefzakwan: Akibat tak beradab dengan tumbuhan."	2
23	@anitaba	12/30/2014	Siapin pelampung dan perahu karet @KicauSunyi:Wuih @PORTALKBR:BNPB Tentukan 20 Januari Tanggap Darurat Banji	2

Gambar 5. 1 Data training yang telah diberi label

5.3.1 Implementasi *Cleansing* dan *Case Folding*

Dalam tahap ini, dilakukan pembersihan data tweet dari *noise* sehingga lebih bermakna seperti yang telah dijelaskan pada sub bab 4.4.1. Data *tweet* yang akan di *cleansing* merupakan data yang telah di simpan sebelumnya dalam bentuk .csv. Selain itu akan dilakukan proses *case folding*. *Case folding* adalah proses penyeragaman kata dalam sebuah dokumen. Dalam hal ini peran case folding digunakan untuk mengkorvesi keseluruhan teks yang tedapat pada data set menjadi bentuk standar (biasanya huruf kecil). Pada proses ini dihilangkan tanda baca dan digit

yang terdapat pada set. *Library* Pandas telah disediakan oleh Bahasa pemrograman Python yang digunakan untuk analisis data beserta struktur datanya. Dan pada baris 2 terdapat *library* os yang digunakan untuk membuka file menjadi input data yang akan diolah selanjutnya. Pada baris 4 dilakukan proses *case folding* dimana mengubah kata pada *tweet* menjadi huruf kecil semuanya. Pada baris 6-25 dilakukan beberapa proses seperti mengubah www. Atau https menjadi URL, mengubah @username menjadi AT_USER, menghilangkan tanda # atau yang biasa disebut *hashtag*, menghilangkan emotikon, menghilangkan spasi kebawah, menghilangkan karakter yang ganda, menghilangkan tanda baca, dan menghilangkan tanda petik.

```

1. import pandas as pd
2. import os
3. def remove_hastagnall(tweet):
4.     #Convert to lower case
5.     tweet = tweet.lower()
6.     #Convert www.* or https?://* to URL
7.     tweet = re.sub
      ('(www\.([^\s]+)| (https?:/[^\s]+)| ((pic\.([^\s]+)))', '', tweet)
8.     #Convert @username to AT_USER
9. tweet = re.sub('@([^\s]+)', '', tweet)
10. #Replace #word with word
11. tweet = re.sub(r'#([^\s]+)', r'\1', tweet)
12. #Replace number with spaces
13. tweet = re.sub(r'(?:(?:\d+,?)+(?:\.?\d+)?)', '', tweet)
14. #Remove emoticon
15. tweet = re.sub(r'^'+emoticon_str+'$', '', tweet)
16. #Remove \n
17. tweet = tweet.replace('\n', ' ')
18. #remove duplicate char
19. tweet = re.sub(r'([a-z])\1+', r'\1', tweet)
20. #Remove punctuation
21. tweet = re.sub(r'^\w\s', ' ', tweet)
22. #Remove additional white spaces
23. tweet = re.sub('[\s]+', ' ', tweet)
24. #trim
25. tweet = tweet.strip('\'\"')
26. return tweet

```

Gambar 5. 2 Kode proses *cleansing* dan *case folding*

5.3.2 Implementasi *Stemming*

Pada sub bab 4.4.2 telah dijelaskan perancangan stemming dengan metode algoritma *Enhanced Confix Stripping Stemmer (ECSS)*. Cara

kerja algoritma ECSS telah dijelaskan pada sub bab 3.4.3. Proses implementasi stemming ini menggunakan library stemmer Sastrawi yang merupakan pustaka stemming Python *open-souce*. Pada baris 1 menunjukkan pemanggilan library *Stemmer* Sasrawi. Pada baris 2-6 menunjukkan proses *stemming* yang digunakan *stemmerFactory* dari library Sastrawi.

```
1. from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
2. def stemming(text):
3.     factory = StemmerFactory()
4.     stemmer = factory.create_stemmer()
5.     kata_dasar = stemmer.stem(text)
6.     return kata_dasar
```

Gambar 5. 3 Kode proses *stemming*

5.3.3 Implementasi Tokenisasi

Dalam sub bab ini, proses tokenisasi adalah proses pemecahan kalimat menjadi kata-kata yang terdapat pada *tweet*. Proses ini sangat penting dilakukan sebelum melakukan pembuangan *stopword*.

```
1. def tokenize(text):
2.     return text.split()
```

Gambar 5. 4 Kode proses tokenisasi

5.3.4 Implementasi Pembuangan *Stopword*

Pada bab 4.4.5 telah dijelaskan perancangan *stopword*. Dimana proses implelementasi diawali dengan pemanggilan *database stopwords* yang telah disimpan. Baris 2-3 merupakan pemanggilan *database stopwords* yang mana digunakan *absolute path* yang telah ditentukan sebelumnya. Pada penelitian ini telah di atur direktorinya yaitu D:/skripsi. File *database stopwords* disimpan dalam direktori D:/skripsi dengan nama file *stopword.txt*. Pada baris 4-5 dilakukan proses penghilangan *stopword* kata yang terdapat pada *tweet*. Lalu setelah itu kata yang telah dihilangkan *stopword* akan digabungkan kembali menjadi kalimat.

```
1. def hapus_stopword(text):
2.     StopWords = "stopword.txt"
3.     sw=open(StopWords,encoding='utf-8',
4.         mode='r');stop=sw.readlines();sw.close()
5.     stop=[kata.strip() for kata in stop];stop=set(stop)
```

```

5. kata = [item for item in text if item not in stop]
6. return ' '.join(kata)

```

Gambar 5. 5 proses pembuangan *stopword*

5.4 Implementasi Running Time Preprocessing

Pada bagian ini semua proses pra pemrosesan yang telah dilakukan akan dihitung waktunya. Pada baris 3 merupakan fungsi untuk memulai menghitung waktu dalam satuan detik. Pada baris ke 5 *tweetbersih* adalah *variable* untuk menyimpan hasil *preprocessing* dalam bentuk *array*. Pada baris 7-12 merupakan proses perulangan *preprocessing* untuk seluruh konten. Pada baris 17-20 merupakan proses dimana hasil *preprocessing* disimpan dalam bentuk *.csv*

```

1. import time
2.
3. start_time = time.time()
4.
5. tweetbersih = []
6. for a,b in tweets:
7.     text = remove_hastagnall(a)
8.     text2 = stemming(text)
9.     text3 = tokenize(text2)
10. text4 = filter_kata(text3)
11. text5 = hapus_stopword(text4)
12. tweetbersih.append((text5,b))
13.
14. print("--- %s seconds ---" % (time.time() - start_time))
15. print (tweetbersih)
16.
17. import csv
18. with open('preprocessing.csv','w', newline='') as csvfile:
19.     writer = csv.writer(csvfile)
20.     writer.writerows(tweetbersih)

```

Gambar 5. 6 kode proses *running time preprocessing*

5.5 Implementasi Pembuatan *Vocabulary*

Vocabulary diperoleh dari pencarian kata yang sering muncul didalam data set yang mana pembuatan *vocabulary* bertujuan untuk menentukan fitur yang akan digunakan. Pada baris pertama digunakan *library counter* yang digunakan untuk menghitung *hastable object*. *Hastable object* membuat sebuah objek dapat digunakan sebagai kamus dan anggota yang telah ditetapkan. Pada baris 3-4 dilakukan pemanggilan terhadap data *preprocessing*. Sedangkan pada baris 6-13 merupakan proses

dalam pembuatan *vocabulary*. Fungsi *most_common* pada baris ke 9 untuk menentukan kata yang sering muncul, pada penelitian ini digunakan 1000 kata yang paling sering muncul untuk keseluruhan dokumen didalam set

```

1. from collections import Counter
2.
3. df = pd.read_csv('preprocessing.csv', delimiter=',', names =
  ['Tweet', 'Sentimen'])
4. tweetbersih = df['Tweet'].tolist()
5.
6. def make_vocab(tweetbersih):
7.     counter = Counter ()
8.     [counter.update(tweet.split()) for tweet in
  tweetbersih]
9.     print(counter.most_common(1000))
10. vocab = counter.most_common(1000)
11. vocab = [word [:][0] for word in vocab]
12. print (len(vocab))
13. return vocab

```

Gambar 5. 7 Kode proses pembuatan *vocabulary*

5.6 Implementasi Pembobotan TF-IDF

Proses selanjutnya yaitu TF-IDF. Pada baris pertama digunakan *library scikit learn* yang mana untuk mengekstrak teks agar dapat digunakan untuk pembobotan TF-IDF dengan matriks. Pada tahap sebelumnya, vocab yang telah ditentukan akan disimpan dalam bentuk matriks. Baris 3-4 fungsi pembobotan dengan menggunakan *library*. Pada baris ke 3 terdapat beberapa parameter. Parameter yang pertama yaitu *analyzer='word'* dimana digunakan untuk memastikan bahwa input yang di masukkan adalah berupa kata. Lalu setelah itu ada *min_df=0.01*, *min_df* sendiri merupakan batas nilai dari pembangunan kosa kata (*vocab*). Dalam membangun kosa kata perlu diperhatikan frekuensi kemunculan dari *term*, apakah sebuah *term* tersebut muncul satu kali dalam dokumen atau bisa juga muncul pada seluruh dokumen. Penggunaan *min_df* ditunjukkan untuk hal tersebut, nilai 0,01 berarti dikalikan dengan jumlah dokumen (7789 dokumen) = 77,89 atau setara dengan 78. Frekuensi suatu kata itu harus muncul didalam dokumen minimal berjumlah 78. Jika kata muncul kurang dari 78 dokumen maka kata tersebut diabaikan. Parameter *Ngram_range=(1,1)* menunjukkan range n-gram dari n-gram(kata). Ini berarti pemotongan n-gram berbasis kata yang

digunakan adalah unigram seperti dijelaskan sub bab 3.4.6. Parameter selanjutnya adalah *vocabulary*. Parameter ini diisi dengan *vocab* yang telah dibuat pada tahap sebelumnya. Pada baris ke 4 menunjukkan pembelajaran pada *vocabulary* dan IDF pada data set. Setelah itu hasil dari TF-IDF akan disimpan dalam bentuk *array* dan akan digunakan sebagai input untuk proses klasifikasi.

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. def make_tfidf(tweetbersih):
3.     tfidf = TfidfVectorizer(analyzer='word', min_df=0.01,
4.                             ngram_range=(1,1), lowercase=False)
5.     tfidf_hasil = tfidf.fit_transform(tweetbersih)
6.     print(tfidf_hasil.toarray())
7.     return tfidf_hasil
```

Gambar 5. 8 Kode Pembobotan TF-IDF

5.7 Implementasi Training

Implementasi *training* dibuat berdasar rancangan pada sub bab 4.7.2. pada proses ini, *library scikit learn* digunakan untuk pemanggilan fungsi support vector machine dan multinomial naïve Bayes. Pada baris 9 merupakan pemodelan menggunakan SVM dengan parameter tertentu. Lalu pada baris 10 merupakan pemodelan menggunakan multinomial naïve bayes juga dengan parameter tertentu yang telah dijelaskan pada sub bab 3.6.1.

Pada baris ke-11 merupakan penggunaan fungsi *pipeline* yang bertujuan untuk mengumpulkan TF-IDF dan evaluasi yang lalu digunakan untuk dijadikan model pada proses *training*. Pada baris ke-12 merupakan fungsi untuk menyimpan x,y menjadi array data hasil *preprocessing* dan data kelas. Setelah itu pada baris ke-13 digunakan *pickle.dump* yang digunakan untuk menyimpan model *training* yang telah dibangun yang kemudian digunakan untuk memprediksi data *testing*.

```
1. from sklearn import svm
2. from sklearn.svm import SVR
3. from sklearn.naive_bayes import MultinomialNB
4. from sklearn.pipeline import Pipeline
5. from sklearn.externals import joblib
6. import pandas as pd
7. import pickle
8. def model_classifier(x,y):
9.     clf = svm.SVC(kernel='linear', C=1)
```

```

10. # clf = MultinomialNB(alpha=1,fit_prior=True,
    class_prior=None).fit(X_train, y_train)
11. tfidf_svm = Pipeline([('tfidf',tfidf),
    ('evaluasism',clf)])
12. tfidf_svm.fit(x,y)
13. svm_dump = pickle.dumps(tfidf_svm)
14. return svm_dump

```

Gambar 5. 9 Kode Implementasi *training*

5.8 Implementasi *Testing*

Data testing yang digunakan adalah data yang telah dilakukan proses *preprocessing*. Pada baris ke-9, *user* diminta untuk memasukkan path file dimana data *testing* tersebut berada. Baris 10 merupakan proses pengecekan pathfile. Baris 11 merupakan proses pembacaan pada file data *testing*, dimana diberi nama databaru yang berbentuk array. Selanjutnya baris ke-13 merupakan pemanggilan *pickle.loads* dengan menggunakan *model_classifier* yang telah di buat pada proses *training* sebelumnya. Tahap terakhir adalah prediksi hasil identifikasi yang terdapat pada baris ke-14. Output yang akan keluar berupa array hasil identifikasi yang terdiri dari kelas sesuai jumlah data yang dimasukkan.

```

1. import time
2. import os
3. import pandas as pd
4. from sklearn.externals import joblib
5. if __name__ == '__main__':
6.     print ('Hello , ini identifikasi banjir')
7.     print ('=====')
8.     time.sleep(1)
9.     user_input = input("masukkan path dari file: ")
10. assert os.path.exists(user_input), 'tidak ditemukan file
    dari, '+str(user_input)
11. databaru = pd.read_csv(user_input, sep=";", header=0,
    names = ['Username', 'Time', 'Tweet'])
12. start_time = time.time()
13. b = pickle.loads(model_classifier(tweetbersih,y))
14. result = b.predict(databaru)
15. print ("ini hasilnya : " +str(result))
16. print("--- %s seconds ---" % (time.time() - start_time))

```

Gambar 5. 10 Kode Implementasi *testing*

5.9 Implementasi Klasifikasi dan Evaluasi Model

Dalam pengujian model klasifikasi untuk identifikasi *tweet* banjir digunakan *k-fold cross validation* dengan menggunakan $k = 10$, sehingga

data set dibagi menjadi 10 bagian yang mana akan melakukan 10 kali perulangan dimana setiap perulangan membagi 1/10 bagian untuk *testing* dan sisanya untuk proses *training*. Algoritma yang digunakan dalam pengujian model ini menggunakan algoritma *multinomial naïve bayes* dan algoritma *support vector machine*.

5.9.1 Model Evaluasi Multinomial Naïve Bayes

Untuk implementasi, Pada baris 1-4 digunakan *library scikit learn* untuk beberapa proses. Proses tersebut adalah *naïve_bayes* untuk algoritma MultinomialNB, lalu ada *cross_validation* untuk implementasi *K-fold cross validation* dan *metric* untuk menghitung *fscore*, *precision*, *recall*, dan akurasi. Lalu baris ke 7 menunjukkan akan dilakukan perulangan sejumlah *n_fold =10* dengan urutan yang tidak tertentu (*shuffle=True*). Variable X merupakan input berupa hasil array dari pembobotan TF-IDF sebagai representasi term berupa vector, sedangkan variable y merupakan array dari label kelas. Pada baris 19 merupakan proses Multinomial naïve bayes dengan alpha 1. Lalu pada baris 21 adalah fungsi *score* untuk mendapatkan nilai akurasi, baris 22 adalah fungsi *f1score* untuk mendapatkan nilai f, lalu baris 23 fungsi *precision* untuk mendapatkan nilai presisi, dan baris 24 untuk mendapatkan nilai *recall*. Baris 40 - 58 adalah proses untuk mendapatkan akurasi dari setiap evaluasi, yaitu akurasi, *fscore*, *precision*, dan *recall*.

```

1. from sklearn import cross_validation
2. from sklearn.naive_bayes import MultinomialNB
3. from sklearn.cross_validation import KFold
4. from sklearn.metrics import f1_score, precision_score,
   recall_score, accuracy_score
5.
6. def evaluasiNB(X,y):
7.     kf = KFold(len(X), n_folds=10, shuffle=True,
   random_state=9999)
8.     model_train_index = []
9.     model_test_index = []
10. model = 0
11.
12. nilai_akurasiNB = []
13. nilai_f1NB =[]
14. nilai_precisionNB =[]
15. nilai_recallNB = []
16.

```

```

17. for k, (index_train, index_test) in enumerate(kf):
18.     X_train, X_test, y_train, y_test = X.ix[index_train,:],
        X.ix[index_test,:], y[index_train], y[index_test]
19.     clf = MultinomialNB(alpha=1, fit_prior=True,
        class_prior=None).fit(X_train, y_train)
20.
21.     score = clf.score(X_test, y_test)
22.     f1score = f1_score(y_test, clf.predict(X_test))
23.     precision = precision_score(y_test, clf.predict(X_test))
24.     recall = recall_score(y_test, clf.predict(X_test))
25.
26.     nilai_akurasiNB.append(score) #untuk mean
27.     nilai_f1NB.append(f1score)
28.     nilai_precisionNB.append(precision)
29.     nilai_recallNB.append(recall)
30.
31.     print('Model %d has accuracy %f with | f1score: %f |
        precision: %f | recall: %f'%(k,score, f1score, precision,
        recall))
32.     model_train_index.append(index_train)
33.     model_test_index.append(index_test)
34.     model+=1
35.     return score, f1score, precision, recall, nilai_akurasiNB,
        nilai_f1NB, nilai_precisionNB, nilai_recallNB
36.
37.     score, f1score, precision, recall, nilai_akurasiNB,
        nilai_f1NB, nilai_precisionNB, nilai_recallNB =
        evaluasiNB(X,y)
38.
39.     # evaluasiNB(X,y)
40.     mean_akurasiNB = 0
41.     for a in nilai_akurasiNB:
42.         mean_akurasiNB = mean_akurasiNB + a
43.     print ('rata-rata nilai akurasi Multinomial Naive bayes
        adalah %f' %(mean_akurasiNB/10))
44.
45.     mean_f1score = 0
46.     for a in nilai_f1NB:
47.         mean_f1score = mean_f1score + a
48.     print ('rata-rata nilai f1 Multinomial Naive bayes adalah
        %f'%(mean_f1score/10))
49.
50.     mean_precision = 0
51.     for a in nilai_precisionNB:
52.         mean_precision = mean_precision + a
53.     print ('rata-rata nilai precision Multinomial Naive bayes
        adalah %f'%(mean_precision/10))
54.
55.     mean_recall = 0
56.     for a in nilai_recallNB:
57.         mean_recall = mean_recall + a
58.     print ('rata-rata nilai recall Multinomial Naive bayes
        adalah %f'%(mean_recall/10))

```

Gambar 5. 11 Kode model evaluasi Multinomial NB

5.9.2 Model Evaluasi Support Vector Machine

Untuk implementasi, Pada baris 1-4 digunakan library scikit learn untuk beberapa proses. Proses tersebut adalah svm untuk algoritma support vector machine, lalu ada *cross_validation* untuk implementasi *K-fold cross validation* dan *metric* untuk menghitung *fscore*, *precision*, *recall*, dan akurasi. Lalu baris ke 7 menunjukkan akan dilakukan perulangan sejumlah *n_fold = 10* dengan urutan yang tidak tertentu (*shuffle=True*). Variable *X* merupakan input berupa hasil array dari pembobotan *tf idf* sebagai representasi term berupa vector, sedangkan variable *y* merupakan array dari label kelas. Pada baris 18 merupakan proses support vector machine dengan menggunakan linier kernel dan nilai *C=1*. Lalu pada baris 19 adalah fungsi *score* untuk mendapatkan nilai akurasi, baris 20 adalah fungsi *f1score* untuk mendapatkan nilai *f-measure*, lalu baris 21 fungsi *precision* untuk mendapatkan nilai presisi, dan baris 22 untuk mendapatkan nilai *recall*. Baris 39-57 adalah proses untuk mendapatkan akurasi dari setiap evaluasi, yaitu akurasi, *fscore*, *precision*, dan *recall*.

```

1. from sklearn import cross_validation
2. from sklearn import svm
3. from sklearn.cross_validation import KFold
4. from sklearn.metrics import f1_score, precision_score,
   recall_score, accuracy_score
5. def evaluasisvm(X,y):
6.     kf = KFold(len(X), n_folds=10, shuffle=True,
   random_state=9999)
7.     model_train_index = []
8.     model_test_index = []
9.     model = 0
10.     nilai_akurasisvm = []
11.     nilai_flsvm = []
12.     nilai_precisionsvm = []
13.     nilai_recallsvm = []
14.
15.     for k, (index_train, index_test) in enumerate(kf):
16.         X_train, X_test, y_train, y_test = X.ix[index_train,:],
   X.ix[index_test,:], y[index_train], y[index_test]
17.         clf = svm.SVC(kernel='linear', C=1).fit(X_train, y_train)
18.         score = clf.score(X_test, y_test)
19.         f1score = f1_score(y_test, clf.predict(X_test))
20.         precision = precision_score(y_test, clf.predict(X_test))
21.         recall = recall_score(y_test, clf.predict(X_test))
22.
23.         nilai_akurasisvm.append(score) #untuk mean
24.         nilai_flsvm.append(f1score)

```

```

25. nilai_precisionsvm.append(precision)
26. nilai_recallsvm.append(recall)
27.
28. print('Model %d has accuracy %f with | flscore: %f |
    precision: %f | recall: %f'%(k,score, flscore, precision,
    recall)
29. model_train_index.append(index_train)
30. model_test_index.append(index_test)
31. model+=1
32. return score, flscore, precision, recall,
    nilai_akurasisvm, nilai_flsvm, nilai_precisionsvm,
    nilai_recallsvm
33.
34. score, flscore, precision, recall, nilai_akurasisvm,
    nilai_flsvm, nilai_precisionsvm, nilai_recallsvm =
    evaluasisvm(X,y)
35. # evaluasisvm(X,y)
36.
37. mean_akurasisvm = 0
38. for a in nilai_akurasisvm:
39. mean_akurasisvm = mean_akurasisvm + a
40. print ('rata-rata nilai akurasi svm adalah %f'
    %(mean_akurasisvm/10))
41.
42. mean_flscore = 0
43. for a in nilai_flsvm:
44. mean_flscore = mean_flscore + a
45. print ('rata-rata nilai fl svm adalah
    %f'%(mean_flscore/10))
46.
47. mean_precision = 0
48. for a in nilai_precisionsvm:
49. mean_precision = mean_precision + a
50. print ('rata-rata nilai precision svm adalah
    %f'%(mean_precision/10))
51.
52. mean_recall = 0
53. for a in nilai_recallsvm:
54. mean_recall = mean_recall + a
55. print ('rata-rata nilai recall svm adalah
    %f'%(mean_recall/10))

```

Gambar 5. 12 kode model evaluasi support vector machine

BAB VI

HASIL PENELITIAN DAN PEMBAHASAN

6.1 Pengujian Sistem dan Hasil

Pada penelitian ini dilakukan identifikasi *tweet* banjir menggunakan klasifikasi Multinomial naïve bayes dan *support vector machine*. Data set yang digunakan sejumlah 7798 *tweet* sebagai data *training* dan 2327 *tweet* sebagai data *testing*. Data *training* dilabeli secara manual dan dikategorikan ke dalam 3 kelas, yaitu banjir, tidak banjir, dan *unknown*. Untuk komposisi data *training* ditunjukkan pada Tabel 6.1

Tabel 6. 1 Komposisi jumlah *tweet training* pada setiap label

Kelas		Kode pelabelan	Jumlah
Bencana	Banjir	3	3086
	Tidak banjir	2	4518
Bukan Bencana	<i>Unknown</i>	1	194

6.2 Hasil Preprocessing

Data *training* yang berjumlah 7789 *tweet*. Pada Tabel 6.2 merupakan cuplikan data sebelum dilakukan *preprocessing*.

Tabel 6. 2 Cuplikan *tweet* sebelum *dipreprocessing*

No	Tweet	Label
1	Banjir? Banjir Yg mana? Ngga ada banjir hanya air tergenang kata metrotipu @estiningsihdwi: Kabarnya jakarta banjir parah nggih?\nSemoga.....	3
2	Apa ??u dikata masih ada yg butuh Pesta "@RIFQI: astagfirullah jakarta banjir ahok ramein pesta tahun baru pake dana 1 M. @PintarPolitik"	2
3	astagfirullah jakarta banjir ahok ramein pesta tahun baru pake dana 1 M. teganya dikau. Malaysia batalin itu krn banjir @PintarPolitik	3
4	Jakarta suka banjir karena.....	2
5	Mesti semua ulah avatar. Tanah: Longsor banjarnegara. Api: kebakaran pasar klewer. Air: Banjir jakarta. Udara :Jatuhnya air asia.	2

Proses *preprocessing* data *training* yang berjumlah 7789 tweet memerlukan waktu selama 4509 detik atau setara dengan 1 jam 15 menit. Hasil dari *preprocessing* dapat dilihat pada Tabel 6.3.

Tabel 6. 3 Cuplikan *tweet* setelah *dipreprocessing*

No	Tweet
1	banjir banjir ngga banjir air genang metrotipu kabar jakarta banjir parah nggih moga
2	butuh pesta astagfirullah jakarta banjir ahok ramein pesta pake dana
3	astagfirullah jakarta banjir ahok ramein pesta pake dana tega dikau malaysia batalin krn banjir
4	jakarta suka banjir
5	mesti ulah avatar tanah longsor banjarnegara api bakar pasar klewer air banjir jakarta udara jatuh air asia

Pada pengujian ini juga dilakukan perhitungan *running time preprocessing* dari berbagai jumlah porsi data. Hasil pengujian *running time preprocessing* dapat dilihat pada Tabel 6.4.

Sampel	Waktu (detik)	Rata-rata <i>preprocessing</i>
7798	4505	1.729 <i>tweet/detik</i>
5000	3119	1.603 <i>tweet/detik</i>
2500	1584	1.578 <i>tweet/detik</i>
1000	647	1.574 <i>tweet/detik</i>

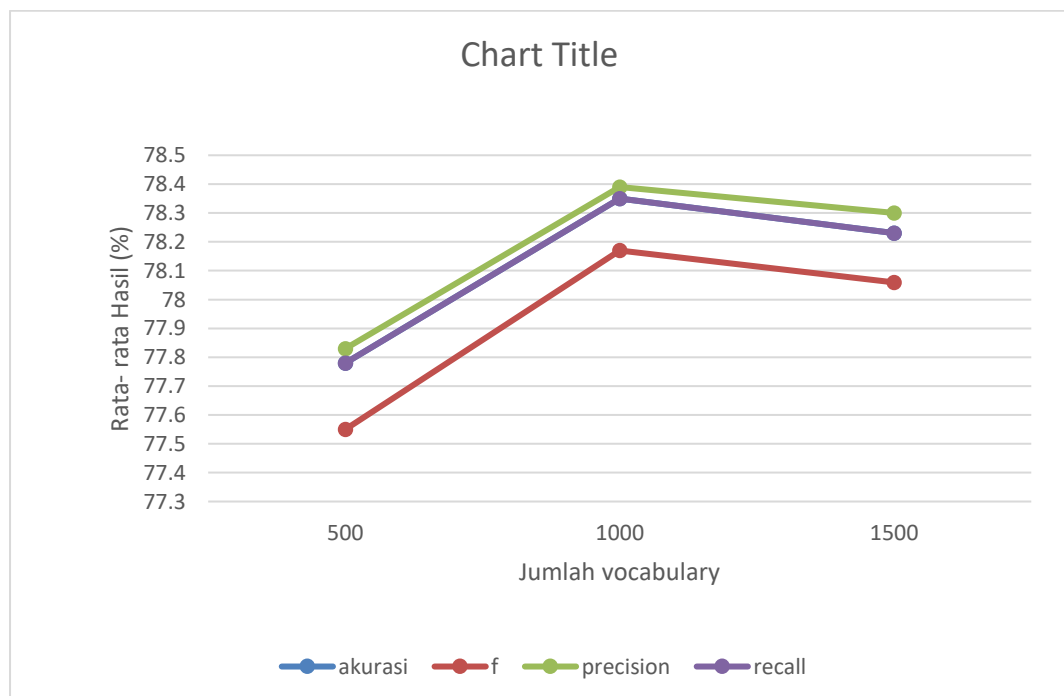
6.3 Hasil Vocabulary

Beberapa percobaan dilakukan untuk mengetahui jumlah kata yang sesuai untuk dijadikan sebagai fitur. Percobaan dilakukan sebanyak 3 kali , pertama dilakukan percobaan dengan jumlah kata 500 kata yang paling sering muncul, lalu 1000 kata yang sering muncul dan terakhir 1500 kata yang sering muncul. Masing-masing jumlah tersebut dicobakan ke dalam model klasifikasi data *training* dengan metode *multinomial naïve bayes* dan *support vector machine* sekaligus dievaluasi secara menyeluruh menggunakan *k-fold cross validation*. Gambar 6.1 menunjukkan cuplikan hasil pembuatan dari *vocabulary*.


```
[('banjir', 1204), ('jakarta', 1110), ('hujan', 101), ('djarot', 100), ('liput', 91), ('metro', 90), ('kepong', 88), ('bandung', 84), ('ahok', 82), ('air', 74), ('netizen', 63), ('sabtu', 60), ('dki', 59), ('genang', 54), ('wagub', 53), ('sindir', 53), ('berita', 48), ('gubernur', 48), ('tinggal', 43), ('warga', 42), ('gak', 41), ('guyur', 41), ('timur', 40), ('aceh', 39), ('bebas', 38), ('malaysia', 37), ('jokowi', 36), ('macet', 34), ('aja', 33), ('mudik', 33), ('sesumbar', 33), ('batal', 33), ('kalo', 32), ('persija', 30), ('kampung', 30), ('pagi', 29), ('mati', 29), ('korban', 29), ('salah', 29), ('info', 28), ('pulo', 28), ('malam', 28), ('landa', 28), ('moga', 27), ('longsor', 27), ('bencana', 27), ('rumah', 25), ('jalan')]
```

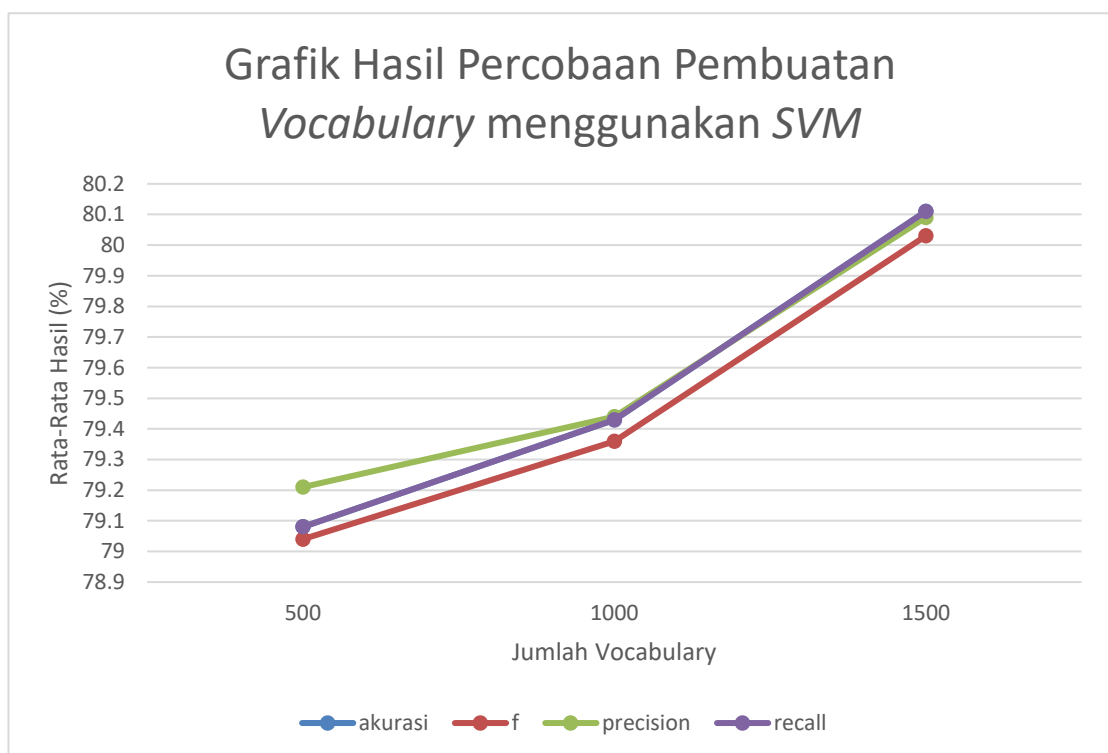
Gambar 6. 1 Cuplikan hasil dari pembuatan vobulary

Berikut Gambar 6.2 adalah diagram hasil dari ketiga percobaan untuk menghasilkan jumlah kata yang paling sering muncul yang akan digunakan sebagai fitur menggunakan metode multinomial naïve bayes dengan nilai $\alpha = 1$. Nilai α merupakan nilai *addictive laplace smoothing*. Dapat dilihat bahwa hasil terbaik ditunjukkan pada jumlah 1000 kata, dengan rata-rata hasil dari pengujian masing-masing berdasar nilai akurasi, *precision*, *recall*, dan *f* dalam satuan %.



Gambar 6. 2 Grafik hasil percobaan pembuatan vocabulary menggunakan metode MNB

Percobaan pembuatan *vocabulary* menggunakan metode *support vector machine* dengan parameter *kernel* = linier dan $C = 1$. Parameter C Berikut Gambar 6.3 merupakan grafik percobaan pembuatan *vocabulary* menggunakan metode *support vector machine*. Dapat dilihat bahwa hasil terbaik ditunjukkan pada jumlah kata 1500 kata, dengan rata-rata hasil dari pengujian masing-masing berdasar nilai akurasi, *precision*, *recall*, dan f dalam satuan %.



Gambar 6. 3 Grafik hasil percobaan pembuatan *vocabulary* menggunakan SVM

6.4 Hasil Pembobotan TF-IDF

Setelah didapatkan *vocabulary* sebagai fitur yang mewakili dengan metode, maka proses selanjutnya adalah pembobotan TF-IDF. Beberapa percobaan juga telah dilakukan pada pembobotan TF-IDF, seperti parameter yang digunakan sebelumnya yaitu pembuatan *vocabulary* yang akan mempengaruhi hasil dari pembobotan TF-IDF. Hasil dari TF-IDF matriks berukuran [7789,1000] untuk hasil pembobotan menggunakan *multinomial*

naïve bayes. Sedangkan matriks [7789,1500] untuk hasil pembobotan menggunakan metode *support vector machine*. Jumlah 7789 adalah merepresentasikan jumlah dokumen yang terdapat didalam data *training*. Sedangkan untuk 1500 dan 1000 merepresentasikan jumlah fitur dari *vocabulary*.

Gambar 6.4 merupakan cuplikan hasil dari pembobotan TF-IDF yang telah dilakukan.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	0.264651	0.067972	0	0	0	0	0	0	0	0.242874	0	0	0	0.259281	0	0	0	0	0	0	0
2	0.056808	0.058362	0	0	0	0	0	0	0.199349	0	0	0	0	0	0	0	0	0	0	0	0
3	0.106973	0.054949	0	0	0	0	0	0	0.187692	0	0	0	0	0	0	0	0	0	0	0	0
4	0.181585	0.18655	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0.046015	0.047273	0	0	0	0	0	0	0	0.337828	0	0	0	0	0	0	0	0	0	0	0
6	0.079705	0.163768	0	0	0	0	0	0	0	0	0.677471	0	0	0	0	0	0	0	0	0	0
7	0.078641	0.080792	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0.0671	0.068935	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0.123212	0.126582	0.411303	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0.063612	0.065352	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0.12183	0.125162	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0.09752	0.100187	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0.117548	0.120762	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0.103024	0.105842	0	0.348322	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0.09484	0.048716	0	0	0	0	0	0	0.166403	0.174072	0	0	0	0	0	0	0	0	0	0	0
16	0.055118	0.056626	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0.058764	0.060371	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0.091417	0.187833	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0.048368	0.04969	0	0	0	0	0	0.169132	0	0.177551	0	0	0	0	0	0	0	0	0	0	0
20	0.043843	0.045042	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0.043608	0.0448	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0.078727	0.080879	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0.062855	0.064574	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Gambar 6. 4 cuplikan hasil pembobotan TF-IDF

6.5 Hasil Identifikasi

Model yang digunakan untuk melakukan identifikasi adalah model klasifikasi dengan metode multinomial *naïve bayes*. Pada model multinomial *naïve bayes* dilakukan percobaan dengan mengubah nilai α (*addictive smoothing*) untuk mendapatkan hasil terbaik. Pada sub bab 3.6.1 telah dijelaskan tentang nilai α , dengan menggunakan *library scikit learn* dimana $\alpha = 0$ merupakan tanpa *smoothing*, sedangkan $\alpha = 1$ merupakan nilai default *laplace smoothing*. Dan hasil percobaan terhadap 7789 data didapatkan nilai α terbaik yaitu $\alpha = 1$. Nilai α digunakan untuk memberikan nilai probabilitas terhadap kata didalam data *testing* yang tidak muncul dalam data *training*. Percobaan terhadap nilai α dapat dilihat pada Tabel 6.4. Terdapat waktu dalam detik yang menunjukkan lamanya satu proses yang berjalan.

Alpha	Akurasi	Precision	Recall	Fmeasure	Waktu (detik)
0.1	0.781484	0.780976	0.781484	0.779935	1.023
0.2	0.782125	0.781796	0.782125	0.780616	1.005
0.3	0.782253	0.781968	0.782253	0.780778	1.061
0.4	0.782766	0.782555	0.782766	0.781316	1.039
0.5	0.783408	0.783189	0.783408	0.781953	1.086
0.6	0.783536	0.783397	0.783536	0.783536	1.109
0.7	0.783279	0.783221	0.783279	0.781731	1.049
0.8	0.78392	0.784019	0.78392	0.782371	1.01
0.9	0.783792	0.784019	0.783792	0.782188	1.01
1	0.784049	0.784458	0.784049	0.782307	1.193

Tabel 6. 4 Percobaan terhadap nilai parameter α

Percobaan selanjutnya yaitu dengan menggunakan metode *support vector machine* dengan mengubah nilai C. Pada sub bab 3.6.2 telah dijelaskan tentang parameter C pada *support vector machine*. Parameter C dipilih untuk mengontrol *tradeoff* antara margin dan klasifikasi error. Nilai C yang besar berarti akan memberikan penalti yang lebih besar terhadap klasifikasi. Dari hasil percobaan terhadap 7789 data didapatkan hasil C terbaik yaitu C = 1. Percobaan terhadap parameter C dapat dilihat pada Tabel 6.5.

C	Akurasi	Precision	Recall	Fmeasure	Waktu (detik)
0.001	0.579245	0.335795	0.579245	0.425056	1122.546
0.01	0.625925	0.728923	0.625925	0.524142	1137.553
0.1	0.775455	0.789794	0.775455	0.764881	921.804
1	0.789023	0.798261	0.798023	0.797318	728.528
10	0.784046	0.785278	0.784046	0.784227	672.838
100	0.776865	0.779559	0.776865	0.777601	935.135
1000	0.774172	0.776651	0.774172	0.774829	4546.042

Tabel 6. 5 Percobaan terhadap nilai parameter C

Data *testing* yang digunakan pada penelitian ini berjumlah *tweet* yang mengandung kata banjir didalamnya. Pada penelitian ini telah berhasil mengidentifikasi *tweet* banjir dengan menggunakan metode *multinomial naïve bayes* dan *support vector machine* dengan berdasar pada klasifikasi pada data *training*. Namun, data *testing* diuji dengan menggunakan metode terbaik dengan akurasi yang tinggi yaitu *support vector machine*. Tabel 6.6

menunjukkan hasil klasifikasi menggunakan *support vector machine*. Hasil dari klasifikasi dikelompokkan bedasar hari dan di validasi dengan data kenyataan yang terjadi. Data yang digunakan untuk validasi adalah data yang bersumber dari situs data.jakarta.co.id dan situs berita (Data Jakarta, 2015).

Tanggal Tweet	Bencana		Bukan bencana	Hasil SVM	Fakta
	Banjir	tidak banjir			
1 Januari 2015	5	23	3	Tidak Banjir	Tidak Banjir
2 Januari 2015	12	51	1	Tidak Banjir	Tidak Banjir
3 Januari 2015	17	66	0	Tidak Banjir	Tidak Banjir
4 Januari 2015	10	35	3	Tidak Banjir	Tidak Banjir
5 Januari 2015	2	21	0	Tidak Banjir	Tidak Banjir
6 Januari 2015	4	10	0	Tidak Banjir	Tidak Banjir
7 Januari 2015	2	20	2	Tidak Banjir	Tidak Banjir
8 Januari 2015	2	15	0	Tidak Banjir	Tidak Banjir
9 Januari 2015	6	17	2	Tidak Banjir	Tidak Banjir
10 Januari 2015	4	28	2	Tidak Banjir	Tidak Banjir
11 Januari 2015	32	51	2	Tidak Banjir	Tidak Banjir
12 Januari 2015	32	62	2	Tidak Banjir	Tidak Banjir
13 Januari 2015	16	38	4	Tidak Banjir	Tidak Banjir
14 Januari 2015	9	11	3	Tidak Banjir	Tidak Banjir
15 Januari 2015	7	15	1	Tidak Banjir	Tidak Banjir
16 Januari 2015	4	33	0	Tidak Banjir	Tidak Banjir
17 Januari 2015	21	31	0	Tidak Banjir	Tidak Banjir
18 Januari 2015	25	46	1	Tidak Banjir	Tidak Banjir
19 Januari 2015	21	27	0	Tidak Banjir	Tidak Banjir
20 Januari 2015	11	19	1	Tidak Banjir	Tidak Banjir
21 Januari 2015	65	106	3	Tidak Banjir	Tidak Banjir
22 Januari 2015	628	431	5	Banjir	Banjir, Liputan 6 (1), Liputan 6 (2).
23 Januari 2015	56	76	1	Tidak Banjir	Banjir, Liputan 6 (3)
24 Januari 2015	18	24	0	Tidak Banjir	Banjir, Liputan 6 (4)
25 Januari 2015	14	22	2	Tidak Banjir	Banjir

Tabel 6. 6 Validasi Hasil SVM dengan Fakta

6.6 Hasil Pengujian

Hasil pengujian dilakukan dengan menggunakan *10-fold cross validation*. Pada pengujian ini dilakukan pengujian rata-rata akurasi hasil dari *10-fold cross validation* terhadap data dengan berbagai porsi jumlah. Porsi jumlah data pada pengujian model terdiri dari 1000, 2500, 5000 dan

7789 sampel data. Untuk model *multinomial naïve bayes* digunakan parameter α terbaik yaitu $\alpha = 1$. Hasil pengujian dapat dilihat pada Tabel 6.6.

Jumlah sampel	Akurasi	Precision	Recall	Fmeasure	Waktu(s)
1000	81.9	82.37	81.9	81.25	0.174
2500	78.36	78.6	78.36	78.15	0.383
5000	77.38	77.68	77.38	77.35	0.991
7789	78.4	78.44	78.4	78.23	1.193

Tabel 6. 7 Rata-rata nilai akurasi model multinomial naïve bayes

Hasil pengujian dengan menggunakan *multinomial naïve bayes* didapatkan bahwa rata-rata akurasi terbaik adalah akurasi dengan jumlah porsi data yang minimal. Pengujian juga dilakukan dengan model *support vector machine* dengan parameter C terbaik yaitu $C = 1$. Dari hasil pengujian dapat dilihat pada Tabel 6.8

Jumlah sampel	Akurasi	Precision	Recall	Fmeasure	Waktu
1000	85.2	85.41	85.2	84.91	17.331
2500	80.24	80.46	80.24	80.25	89.648
5000	79.22	79.34	79.22	79.21	332.184
7789	78.9	79.82	79.8	79.73	728.528

Tabel 6. 8 Rata-rata nilai akurasi model Support Vector Machine

Dari Tabel 6.8 didapatkan bahwa rata-rata nilai akurasi dengan menggunakan support vector machine semakin menurun seiring dengan jumpah porsi data yang meningkat.

BAB VII

KESIMPULAN DAN SARAN

6.7 Kesimpulan

Berikut adalah kesimpulan yang diperoleh dari penelitian yang dilakukan.

1. Implementasi identifikasi *tweet* mengandung kata banjir menggunakan metode *multinomial naïve bayes* dan *support vector machine* telah berhasil dilakukan. Pengujian terhadap model klasifikasi *multinomial naïve bayes* menggunakan *10-fold cross validation* dengan menggunakan 7789 data *tweet* dan didapatkan hasil rata-rata nilai akurasi sebesar 78.4%. Sedangkan untuk pengujian menggunakan model klasifikasi *support vector machine* didapatkan hasil rata-rata nilai akurasi sebesar 78.9 %
2. Penentuan fitur untuk metode *multinomial naïve bayes* dilakukan dengan pembuatan *vocabulary* berdasar 1000 kata yang sering muncul pada seluruh dokumen didalam data set. Sedangkan *vocabulary* untuk metode *support vector machine* sebesar 1500 kata.
3. Parameter α (*addictive smoothing*) yang digunakan dalam klasifikasi *multinomial naïve bayes* mempengaruhi performa sistem, nilai parameter α terbaik pada penelitian ini didapatkan sebesar $\alpha = 1$.
4. Parameter C yang digunakan dalam klasifikasi *support vector machine* mempengaruhi performa pada sistem, nilai parameter C terbaik pada penelitian ini didapatkan sebesar $C = 1$

6.8 Saran

Berdasarkan penelitian yang telah dilakukan, beberapa saran yang dapat dikembangkan untuk penelitian selanjutnya sebagai berikut.

1. Diidentifikasi lokasi setiap *tweet* yang mengandung kata banjir sehingga dapat dilakukan visualisasi peta banjir di Jakarta
2. Perlu dilakukan analisis hubungan antar kata pada proses identifikasi *tweet* banjir.
3. Dilakukan perbandingan seleksi fitur menggunakan metode tertentu.

DAFTAR PUSTAKA

- Aliandu, P.,2012, Analisis Sentimen Tweet Berbahasa Indonesia di Twitter, *Tesis*, Progam Magister Ilmu Komputer Universitas Gadjah Mada, Yogyakarta.
- Asian, J.,2007, Effective Techniques for Indonesian Retrieval. *Thesis*, School of Computer Science and Information Technology, Science, Engineering, and TechnologyPortfolio, RMIT University, Melbourne, Victoria, Australia.
- Binawan, B. P., 2016, Klasifikasi Tingkat Kerawanan Banjir dengan Metode Weighted Product, *Tugas Akhir*, Teknik Fakultas Informatika Universitas Telkom, Bandung.
- Cavnar, W. B., dan Trenkle, J. M.,1994, N-gram Based Text Categorization, *Environmental Research Institute of Michigan*, 1-14.
- DataJakarta, 2015, Data Kejadian Banjir Tahun 2015 di DKI Jakarta, <http://data.jakarta.go.id/dataset/data-rekapitulasi-kejadian-banjir-bulan-januari-2015>, diakses 29 Juni 2017
- DataJakarta, 2016, Data Kejadian Banjir Tahun 2017 di DKI Jakarta, <http://data.jakarta.go.id/dataset/rekap-banjir-bulan-maret-2016>, diakses 4 Maret 2017
- Doswell, C. A.,2003, *Flooding*, Elsevier Science Ltd, Norman.
- Fawcett, T.,2006, An introduction to ROC analysis, *Pattern recognition Letters*, 861-874.
- Feldmen, R., & Sanger, J.,2007, *The Text Mining Handboo*., Cambridge University Press, New York.
- Gokulakrishnan, B., 2012, Opinion Mining and Sentiment Analisis on Twitter Data Stream, *The International Conference on Advances in ICT for emerging Regions*, 182-188.
- Hand, D., Mannila, H., & Smyth, P.,2011, *Principles of Data Mining*, Massachusetts Institute of Technology Press, Cambridge.
- Huang, J., Lu, J., & Ling, C. X.,2003. Comparing Naive Bayes, Decition Trees and SVM with AUC and Accuracy, *Third IEEE International Conference on Data Mining*, Melbourne.
- Ilyas, A., 2014, MicroFilters : Harnessing Twitter for Disaster Management, *IEEE 2014 global Humanitarian Technology Conference*, 417 - 424

- Joachims T.,1998, Text Categorization with Support Vector Machine: Learning with Many Relevant Features, *10th European Conference on Machine Learning*, 137-142
- Kohavi, R.,1995, A study of Cross Validation and Bootstraps for Accuracy Estimation and Model Selection, *International Joint Conference on Artificial Intelligence 14.12*, 1137-1143.
- Liu, B., 2011, *Web Data Mining*, Springer, Chicago.
- Liputan6-1., 2015. Hujan Deras Sejak Pagi, Genangan Air di Jakarta capai 50cm, <http://news.liputan6.com/read/2164777/hujan-deras-sejak-pagi-genangan-air-di-jakarta-capai-50-cm>, diakses pada 29 Juni 2017.
- Liputan6-2.,2015. Banjir Datangi Kampung Pulo dan Bukit Duri Lagi, <http://tv.liputan6.com/read/2164774/banjir-datangi-kampung-pulo-dan-bukit-duri-lagi>, diakses pada 29 Juni 2017.
- Liputan6-3., 2015. Kelapa Gading Masih Banjir 30cm, 2 Pompa Diturunkan, <http://news.liputan6.com/read/2165507/kelapa-gading-masih-banjir-30-cm-2-pompa-diturunkan>, diakses pada 29 Juni 2017.
- Liputan6-4., 2015, Kelapa Gading dan Cilincing Terendam Banjir, <http://tv.liputan6.com/read/2165792/kelapa-gading-dan-cilincing-terendam-banjir>, diakses pada 29 Juni 2017
- Manning C. D., Ragahvan. P. &Schutze, H., 2009, *An introduction to information Retrieval*, Cambridge University Press, Cambridge, England.
- Morscheck, P.,2016, 25 Fact About Twitter in 2016, <http://www.pettermorscheck.xyz/25-facts-twitter-2016/>, 12 November 2016, diakses 3 Februari 2017.
- Nugroho, A., Witarto, A. B., & Handoko, D., 2003, Support Vector Machine, <http://ilmukomputer.com> , diakses 3 Februari 2017
- O'keefe, T., & Koprinska.,2009, Feature Selection and Weighting methods in sentiment Analysis, *Australasian Document Computing Symposium*, 67.
- Oktafiani, P.M., Jariyah, A., Fitri, S.R., Takato, H.,2012, Social Media Analysis for Indonesian Language : Case study flood in Jakarta, *Advanced Computer Science and Information Systems (ICACSIS), 2012 International Conference*, 161-166.
- Putro, B.,M.,2011, Implementasi Density Based Spartial Clustering Application with Noise dalam perkiraan terjadinya banjir dijakarta, *Tugas Akhir*, Teknik Fakultas Informatika Universitas Telkom, Bandung.

- Rianto, B.,2016,Perbandingan metode pra-pemrosesan pada analisis sentimen tokoh masyarakat, *Skripsi*, Jurusan Ilmu Komputer dan Elektronika FMIPA UGM, Yogyakarta.
- Rodiyansyah, S. F.,2013, Klasifikasi Posting Twitter Kemacetan Lalu Lintas Kota Bandung Menggunakan Naive Bayesian Clasification, *IJCCS*, 13-22.
- Soebroto A.,A., Cholissodin, I., Wihandika, R., C., Frestantiya, M., T., Arief, Z.,E., 2015, Prediksi Tinggi Muka Air Untuk Deteksi Dini Bencana Banjir menggunakan SVR-TVIWPSO, *Jurnal Teknologi Informasi dan Ilmu Computer Vol. 2*, 79-86.
- Tahitoe, A. D., & Purwitasari, D.,2010, Implementasi Modifikasi Enhanced Confix Stripping Stemmer untuk Bahasa Indonesia dengan Metode Corpus Based Stemming. *Jurusan Teknik Informatika, Institut Teknologi Sepuluh November*, 1-15.
- Trisniantari, D. (2016). Klasifikasi Berita Ekonomi menggunakan metode Multinomial Naive Bayes. *Skripsi*. Jurusan Ilmu Komputer dan Elektronika FMIPA UGM, Yogyakarta.
- Turland, M., 2010, *php/architect's Guide to Web Scrapping with PHP*, Marco Tabini & Associates, Inc., Toronto.
- Utomo, M. S.,2012 Implementasi PHP sebagai Penghasil Konten Otomatis pada Halaman Situs, *Jurnal Teknologi Informasi DINAMIK volume 7 no 2*, 147-153.
- Wiryawan, F.,2014, Incremental Learning untuk Opinion Mining pada Tweet Berbahasa Indonesia Menggunakan Data Stream Twitter, *Skripsi*. Jurusan Ilmu Komputer dan Elektronika FMIPA UGM, Yogyakarta.
- Witten, I. H., Frank, E., & Hall, M. A.,2011, *Data Mining : Practical Machine Leaning Tools and Techniques*, Morgan Kaufmann, Burlington.

LAMPIRAN

A. Rata-rata Akurasi Hasil Pengujian Multinomial Naïve Bayes

alpha	Jumlah data			
	7789	5000	2500	1000
	akurasi %	akurasi %	akurasi %	akurasi %
0.1	78.14	77.26	78.32	77.26
0.2	78.21	77.72	78.44	77.72
0.3	78.22	78.02	78.48	78.02
0.4	78.27	77.94	78.4	77.94
0.5	78.34	77.9	78.36	77.9
0.6	78.35	77.82	78.28	77.82
0.7	78.32	77.7	78.44	77.7
0.8	78.39	77.54	78.24	77.54
0.9	78.37	77.42	78.24	77.42
1	78.4	77.38	78.36	77.38

B. Rata-rata Akurasi Hasil Pengujian Support Vector Machine

C	Jumlah data			
	7789	5000	2500	1000
	akurasi %	akurasi %	akurasi %	akurasi %
0.001	57.92	52.38	56.64	62.1
0.01	62.59	55.74	56.64	62.1
0.1	77.54	76.64	78.6	72.2
1	78.9	79.22	80.24	85.2
10	78.4	77.6	77.84	82.5
100	77.68	76.34	75.92	82
1000	77.41	75.6	76.04	82.3

C. Data Stopwords

Stopword	Stopword	Stopword	Stopword
ada	bagian	berikan	bukan
adalah	bahkan	berikut	bukankah
adanya	bahwa	berikutnya	bukanlah
adapun	bahwasanya	berjumlah	bukannya
agak	baik	berkali-kali	bulan
agaknya	bakal	berkata	bung
agar	bakalan	berkehendak	cara
akan	balik	berkeinginan	caranya
akankah	banyak	berkenaan	cukup
akhir	bapak	berlainan	cukupkah
akhiri	baru	berlalu	cukuplah
akhirnya	bawah	berlangsung	cuma
aku	beberapa	berlebihan	dahulu
akulah	begini	bermacam	dalam
amat	beginian	bermacam-	dan
amatlah	beginikah	macam	dapat
anda	beginilah	bermaksud	dari
andalah	begitu	bermula	daripada
antar	begitukah	bersama	datang
antara	begitulah	bersama-sama	dekat
antaranya	begitupun	bersiap	demi
apa	bekerja	bersiap-siap	demikian
apaan	belakang	bertanya	demikianlah
apabila	belakangan	bertanya-tanya	dengan
apakah	belum	berturut	depan
apalagi	belumlah	berturut-turut	di
apatah	benar	bertutur	dia
artinya	benarkah	berujar	diakhiri
asal	benarlah	berupa	diakhirinya
asalkan	berada	besar	dialah
atas	berakhir	betul	diantara
atau	berakhirilah	betulkah	diantaranya
ataukah	berakhirnya	biasa	diberi
ataupun	berapa	biasanya	diberikan
awal	berapakah	bila	diberikannya
awalnya	berapalah	bilakah	dibuat
bagai	berapapun	bisa	dibuatnya
bagaikan	berarti	bisakah	didapat
bagaimana	berawal	boleh	didatangkan
bagaimanakah	berbagai	bolehkah	digunakan
bagaimanapun	berdatangan	bolehlah	diibaratkan
bagi	beri	buat	diibaratkannya

diingat	disebutkannya	ibarat	kamulah
diingatkan	disini	ibaratkan	kan
diinginkan	disinilah	ibaratnya	kanan
dijawab	ditambahkan	ibu	kapankah
dijelaskan	ditandaskan	ikut	kapanpun
dijelaskannya	ditanya	ingat	karena
dikarenakan	ditanyai	ingat-ingat	karenanya
dikatakan	ditanyakan	ingin	kasus
dikatakannya	ditegaskan	inginkah	kata
dikerjakan	ditujukan	inginkan	katakan
diketahui	ditunjuk	ini	katakanlah
diketahuinya	ditunjuki	inikah	katanya
dikira	ditunjukkan	inilah	ke
dilakukan	ditunjukkannya	itu	keadaan
dilalui	ditunjuknya	itukah	kebetulan
dilihat	dituturkan	itulah	kecil
dimaksud	dituturkannya	jadi	kedua
dimaksudkan	diucapkan	jadilah	keduanya
dimaksudkannya	diucapkannya	jadinya	keinginan
dimaksudnya	diungkapkan	jangan	kelamaan
diminta	dong	jangankan	kelihatan
dimintai	dua	janganlah	kelihatannya
dimisalkan	dulu	jauh	kelima
dimulai	empat	jawab	keluar
dimulailah	enggak	jawaban	kembali
dimulainya	enggaknya	jawabnya	kemudian
dimungkinkan	entah	jelas	kemungkinan
dini	entahlah	jelaskan	kemungkinannya
dipastikan	guna	jelastah	kenapa
diperbuat	gunakan	jelastnya	kepada
diperbuatnya	hal	jika	kepadanya
dipergunakan	hampir	jikalau	kesampaian
diperkirakan	hanya	juga	keseluruhan
diperlihatkan	hanyalah	jumlah	keseluruhannya
diperlukan	hari	jumlahnya	keterlaluhan
diperlukannya	harus	justru	ketika
dipersoalkan	haruslah	kala	khususnya
dipertanyakan	harusnya	kalau	kini
dipunyai	hendak	kalaupun	kinilah
diri	hendaklah	kalaupun	kira
dirinya	hendaknya	kalian	kira-kira
disampaikan	hingga	kami	kiranya
disebut	ia	kamilah	kita
disebutkan	ialah	kamu	kitalah
kok	memerlukan	mengucapkannya	pada
kurang	memihak	mengungkapkan	padahal

lagi	meminta	menjadi	padanya
lagian	memintakan	menjawab	pak
lah	memisalkan	menjelaskan	paling
lain	memperbuat	menuju	panjang
lainnya	mempergunakan	menunjuk	pantas
lalu	memperkirakan	menunjuki	para
lama	memperlihatkan	menunjukkan	pasti
lamanya	mempersiapkan	menunjuknya	pastilah
lanjut	mempersoalkan	menurut	penting
lanjutnya	mempertanyakan	menuturkan	pentingnya
lebih	mempunyai	menyampaikan	per
lewat	memulai	menyangkut	percuma
lima	memungkinkan	menyatakan	perlu
luar	menaiki	menyebutkan	perlukah
macam	menambahkan	menyeluruh	perlunya
maka	menandaskan	menyiapkan	pernah
makanya	menanti	merasa	persoalan
makin	menanti-nanti	mereka	pertama
malah	menantikan	merekalah	pertama-tama
malahan	menanya	merupakan	pertanyaan
mampu	menanyai	meski	pertanyakan
mampukah	menanyakan	meskipun	pihak
mana	mendapat	meyakini	pihaknya
manakala	mendapatkan	meyakinkan	pukul
manalagi	mendatang	minta	pula
masa	mendatangi	mirip	pun
masalah	mendatangkan	misal	punya
masalahnya	menegaskan	misalkan	rasa
masih	mengakhiri	misalnya	rasanya
masihkah	mengapa	mula	rata
masing	mengatakan	mulai	rupanya
masing-masing	mengatakannya	mulailah	saat
mau	mengenai	mulanya	saatnya
maupun	mengerjakan	mungkin	saja
melainkan	mengetahui	mungkinkah	sajalah
melakukan	menggunakan	nah	saling
melalui	menghendaki	naik	sama
melihat	mengibaratkan	namun	sama-sama
melihatnya	mengibaratkannya	nanti	sambil
memang	mengingat	nantinya	sampai
memastikan	mengingatkan	nyaris	sampai-sampai
memberi	menginginkan	nyatanya	sampaikan
memberikan	mengira	oleh	sana
membuat	mengucapkan	olehnya	sangat
sangatlah	sejumlah	sendirinya	sudah
satu	sekadar	seolah	sudahkah

saya	sekadarnya	seolah-olah	sudahlah
sayalah	sekali	seorang	supaya
se	sekali-kali	sepanjang	tadi
sebab	sekalian	sepantasnya	tadinya
sebabnya	sekaligus	sepantasnyalah	tahu
sebagai	sekalipun	seperlunya	tahun
sebagaimana	sekarang	seperti	tak
sebagainya	sekarang	sepertinya	tambah
sebagian	sekecil	sepihak	tambahnya
sebaik	seketika	sering	tampak
sebaik-baiknya	sekiranya	seringnya	tampaknya
sebaiknya	sekitar	serta	tandas
sebaliknya	sekitarnya	serupa	tandasnya
sebanyak	sekurang-	sesaat	tanpa
sebegini	kurangnya	sesama	tanya
sebegitu	sekurangnya	sesampai	tanyakan
sebelum	sela	sesegera	tanyanya
sebelumnya	selain	sesekali	tapi
sebenarnya	selaku	seseorang	tegas
seberapa	selalu	sesuatu	tegasnya
sebesar	selama	sesuatunya	telah
sebetulnya	selama-lamanya	sesudah	tempat
sebisanya	selamanya	sesudahnya	tengah
sebuah	selanjutnya	setelah	tentang
sebut	seluruh	setempat	tentu
sebutlah	seluruhnya	setengah	tentulah
sebutnya	semacam	seterusnya	tentunya
secara	semakin	setiap	tepat
secukupnya	semampu	setiba	terakhir
sedang	semampunya	setibanya	terasa
sedangkan	semasa	setidak-tidaknya	terbanyak
sedemikian	semasih	setidaknya	terdahulu
sedikit	semata	setinggi	terdapat
sedikitnya	semata-mata	seusai	terdiri
seenaknya	semaunya	sewaktu	terhadap
segala	sementara	siap	terhadapnya
segalanya	semisal	siapa	teringat
segera	semisalnya	siapakah	teringat-ingat
seharusnya	sempat	siapapun	terjadi
sehingga	semua	sini	terjadilah
seingat	semuanya	sinilah	terjadinya
sejak	semula	soal	terkira
sejauh	sendiri	soalnya	terlalu
sejenak	sendirian	suatu	terlebih
terlihat	yakni		
termasuk	yang		

ternyata tersampaikan tersebut tersebutlah tertentu tertuju terus terutama tetap tetapi tiap tiba tiba-tiba tidak tidakkah tidaklah tiga tinggi toh tunjuk turut tutur tuturnya ucap ucapnya ujar ujarnya umum umumnya ungkap ungkapnya untuk usah usai waduh wah wahai waktu waktunya walau walaupun wong yaitu yakini			
--	--	--	--