# KLASIFIKASI *HATE SPEECH* BERBAHASA INDONESIA DI TWITTER MENGGUNAKAN *NAIVE BAYES* DAN SELEKSI FITUR *INFORMATION GAIN* DENGAN NORMALISASI KATA

**SKRIPSI**

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Ivan
NIM: 155150207111170

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

# PENGESAHAN

KLASIFIKASI *HATE SPEECH* BERBAHASA INDONESIA DI TWITTER MENGGUNAKAN *NAIVE BAYES* DAN SELEKSI FITUR *INFORMATION GAIN* DENGAN NORMALISASI KATA

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Ivan
NIM: 155150207111170

Skripsi ini telah diuji dan dinyatakan lulus pada
10 Mei 2019
Telah diperiksa dan disetujui oleh:

| Pembimbing I | Pembimbing II |
|---|---|
| Yuita Arum Sari, S.Kom, M.Kom | Putra Pandu Adikara, S. Kom., M.Kom. |
| NIK: 201609 880715 2 001 | NIP: 19850725 200812 1 002 |

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

# PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 10 Mei 2019


Ivan

NIM: 155150207111170

# PRAKATA

Puji syukur Penulis panjatkan kepada Tuhan Yang Maha Esa yang telah melimpahkan rahmat, karunia, serta hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi *Hate Speech* Berbahasa Indonesia di Twitter Menggunakan Metode *Naive Bayes* dan Seleksi Fitur *Information Gain* dengan Normalisasi Kata".

Skripsi ini merupakan salah satu syarat kelulusan yang harus ditempuh di Fakultas Ilmu Komputer, Program Studi Teknik Informatika Universitas Brawijaya Malang. Penulis ingin mengucapkan rasa terima kasih yang sebesar-besarnya kepada pihak-pihak yang telah memberikan bantuan selama pengerjaan skripsi ini dari awal hingga terselesaikannya laporan skripsi ini dengan baik, antara lain:

1. Yuita Arum Sari, S. Kom., M.Kom., selaku dosen pembimbing 1 dan Putra Pandu Adikara, S. Kom., M.Kom., selaku dosen pembimbing 2 yang telah memberikan bimbingan, saran, serta arahan selama penyusunan skripsi ini.
2. Bapak Agus Wahyu Widodo, S.T., M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Seluruh dosen Fakultas Ilmu Komputer yang telah mendidik dan memberikan ilmu serta wawasannya selama menempuh pendidikan dan menyelesaikan skripsi ini.
5. Seluruh civitas akademika Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberikan bantuan serta dukungan kepada penulis selama menempuh pendidikan dan menyelesaikan skripsi ini.
6. Papa, Mama, cece Nathania, dan Koko Jonathan yang telah memberikan dukungan, doa, nasihat, kasih sayang, perhatian, dan kesabarannya di dalam proses pengerjaan skripsi ini sehingga penulis tetap semangat dan dapat menyelesaikan skripsi ini.
7. Keluarga guru sekolah minggu GBI Suropati yaitu Ester Florida, Andam Catur Wulandari, Kristina Yovina, Eunike Yustina, Ricky Marten, Afandi Bukit, Ketrin Gracia, David V., Magareth Viryawan, Venny Gracelia, Yery Permata, Johana Amelia, Jenny Kristine, yang selalu memberikan bantuan, motivasi dan waktu selama pengerjaan skripsi ini.
8. Teman-teman kontrakan dalam berbagi pengalaman dan ilmu selama ini Ahmad Efriza Irsad, Juan Ozora Cafonda, Muhammad Aulia Rahman, Ivan Primananda, Stefan Levianto, dan Ibnu Dwi Kisnanto.
9. Teman-teman *First Team* sejak mahasiswa baru yaitu Said Atharillah Alifka, Muhammad Fauzan Ziqroh, Raditya Angkasa Megananda, Indrajati Yudistira, Tubagus Agung, Enggar Septrinas, Rhomzy Oesman, Muhammad Farhan, Alfin

Manurung, Haryaputra Arbisono yang telah memberikan warna dan pengalaman tak terlupakan dalam masa perkuliahan.

10. Seluruh teman-teman Teknik Informatika Kelas V 2015 yang telah berbagi ilmu dan pengalaman, serta memberikan bantuan selama pengerjaan skripsi ini.
11. Seluruh teman-teman Teknik Informatika Universitas Brawijaya angkatan 2015 serta semua pihak yang tidak dapat penulis sebutkan satu per satu yang telah membantu penulis selama pendidikan sampai terselesaikannya skripsi ini baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa skripsi ini memiliki banyak kekurangan baik pada format penulisan maupun isi bahasannya, oleh karena itu segala bentuk saran dan kritik yang membangun sangat diharapkan oleh penulis. Semoga skripsi ini dapat memberikan manfaat bagi semua pihak baik untuk pembaca maupun penelitian selanjutnya.

Malang, 10 Mei 2019

Penulis
Email: ivanliu@student.ub.ac.id

# ABSTRAK

**Ivan, Klasifikasi *Hate Speech* Berbahasa Indonesia di Twitter Menggunakan *Naive Bayes* dan Seleksi Fitur *Information Gain* dengan Normalisasi Kata**

**Pembimbing: Yuita Arum Sari S.Kom., M.Kom dan Putra Pandu Adikara S.Kom., M.Kom.,**

*Hate speech* atau ujaran kebencian adalah tindakan yang sering dilakukan oleh sebagian kelompok di masyarakat untuk memprovokasi kebencian dan tindakan kekerasan terhadap seseorang atau kelompok lain karena berbagai alasan. Kasus *hate speech* sangat sering kita jumpai di media sosial, salah satunya di Twitter. Tujuan penelitian ini adalah untuk membuat sebuah sistem yang mampu mengklasifikasikan sebuah *tweet* pada Twitter ke dalam kelas *hate speech* ataupun kelas *non hate speech*. Metode yang digunakan adalah *Naïve Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata. Normalisasi kata digunakan untuk mengatasi permasalahan pada Twitter seperti banyaknya penyingkatan kata, penggunaan bahasa gaul, kesalahan eja, dan penggunaan bahasa yang tidak sesuai dengan standar yang ada. Normalisasi kata yang digunakan berasal dari Pujangga *Indonesian Natural Language Processing* REST API. Data yang digunakan pada penelitian ini berjumlah 250 data *tweet hate speech* berbahasa Indonesia dengan perbandingan 80% untuk data latih dan 20% untuk data uji. *Threshold* yang digunakan pada penelitian ini adalah sebesar 20%, 40%, 60%, 80%, dan 90%. *Threshold* adalah ambang batas yang ditentukan untuk menyimpan kumpulan *term* atau kumpulan kata yang akan digunakan untuk menyeleksi kata-kata yang memiliki nilai tinggi pada proses seleksi fitur *Information Gain*. Hasil akurasi terbaik diperoleh dengan menggunakan normalisasi kata pada tahap *pre-processing* dan menggunakan seleksi fitur *Information Gain* dengan *threshold* 80%. Hasil akurasi terbaik yang didapatkan adalah sebesar 98%, nilai *precision* sebesar 100%, nilai *recall* sebesar 96,15%, dan nilai *f-measure* sebesar 98,03%. Berdasarkan hasil yang diperoleh, dapat diambil kesimpulan bahwa pada saat melakukan klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naïve Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata mampu meningkatkan hasil akurasi menjadi lebih baik.

**Kata kunci:** *Hate speech,* Twitter*, Naïve Bayes, Threshold, Information Gain,* Normalisasi

# ABSTRACT

**Ivan, *Classification of Indonesian Hate Speech on* Twitter *Using Naive Bayes and Information Gain Feature Selection with Word Normalization***

***Supervisors*: Yuita Arum Sari S.Kom., M.Kom and Putra Pandu Adikara S.Kom., M.Kom.,**

*Hate speech is an action often carried out by some groups in the community to provoke hatred and acts of violence against other people or other group for varios reasons. The cases of hate speech are very often encountered on social media, one of which is on Twitter. The purpose of this study is to create a system that is able to classify a tweet on Twitter into a class of hate speech or non hate speech. The method used in this study is Naïve Bayes and Information Gain feature selection with word normalization. Word normalization is used to solve problems on Twitter such as the number of words abbreviated, the use of slang, misspellings, and the use of languages that are not in accordance with existing standards.Word normalization comes from Indonesian Natural Language Processing REST API. The data used in this study were 250 data tweets of hate speech in Indonesian with a ratio of 80% for training data and 20% for testing data. The threshold used in this study is 20%, 40%, 60%, 80%, and 90%. Threshold is a limit that is determined to store a collection of terms or a collection of words that will be used to select words that have high values in the Information Gain feature selection process. The best accuracy results obtained by using word normalization in the pre-processing stage and using Information Gain feature selection with an 80% threshold. The best accuracy result is 98%, precision value is 100%, recall value is 96.15%, and f-measure value is 98.03%. Based on the results obtained, it can be concluded that when doing hate speech classifications in Indonesian on Twitter using Naïve Bayes and Information Gain feature selection with word normalization can improve better accuracy of the results.*

***Keywords**: Hate Speech,* Twitter, *Naïve Bayes, Threshold, Information Gain, Normalization*

# DAFTAR ISI

PENGESAHAN ..... ii
PERNYATAAN ORISINALITAS ..... iii
PRAKATA ..... iv
ABSTRAK ..... vi
ABSTRACT ..... vii
DAFTAR ISI ..... viii
DAFTAR GAMBAR ..... xi
DAFTAR TABEL ..... xii
DAFTAR LAMPIRAN ..... xiii
BAB 1 PENDAHULUAN ..... 1
1.1 Latar belakang ..... 1
1.2 Rumusan masalah ..... 4
1.3 Tujuan ..... 4
1.4 Manfaat ..... 4
1.5 Batasan masalah ..... 5
1.6 Sistematika pembahasan ..... 5
BAB 2 LANDASAN KEPUSTAKAAN ..... 6
2.1 Kajian Pustaka ..... 6
2.2 *Hate Speech* ..... 7
2.3 Twitter ..... 8
2.4 *Text Mining* ..... 8
2.5 Normalisasi ..... 9
2.6 *Text Pre-processing* ..... 9
2.7 *Naïve Bayes Classifier* ..... 11
2.8 *Multinomial Naive Bayes* ..... 12
2.9 *Information Gain* ..... 12
2.10 Evaluasi ..... 13
BAB 3 METODOLOGI ..... 15
3.1 Tipe Penelitian ..... 15
3.2 Strategi Penelitian ..... 15

3.3 Lokasi Penelitian ..... 15
3.4 Teknik Pengumpulan Data ..... 15
3.5 Data Penelitian ..... 15
3.6 Implementasi Algoritme ..... 16
3.7 Teknik Analisis Data ..... 17
3.8 Peralatan Pendukung ..... 17
BAB 4 PERANCANGAN ..... 18
4.1 Formulasi Permasalahan ..... 18
4.2 Penyelesaian Permasalahan ..... 18
4.2.1 *Naive Bayes* dengan *Information Gain* sebagai seleksi fitur ..... 19
4.2.2 *Pre-processing* ..... 19
4.2.3 Seleksi Fitur *Information Gain* ..... 25
4.2.4 Klasifikasi Menggunakan *Naive Bayes Classifier* ..... 27
4.3 Perhitungan Manual ..... 28
4.4 Perancangan Pengujian ..... 39
4.4.1 Skenario pengujian pengaruh normalisasi kata ..... 39
4.4.2 Skenario pengujian pengaruh seleksi fitur *Information Gain* ..... 39
BAB 5 IMPLEMENTASI ..... 41
5.1 Implementasi Algoritme ..... 41
5.1.1 Implementasi *Pre-processing* ..... 41
5.1.2 Implementasi Proses Pembobotan Kata ..... 43
5.1.3 Implementasi Algoritme Seleksi Fitur *Information Gain* ..... 44
5.1.4 Implementasi Algoritme *Naive Bayes Classifier* ..... 48
5.2 Tampilan Program ..... 50
5.2.1 Tampilan dari Tahap *Pre-processing* ..... 50
5.2.2 Tampilan Kelas dari Data Uji ..... 51
BAB 6 PENGUJIAN DAN ANALISIS ..... 52
6.1 Skenario Pengujian ..... 52
6.1.1 Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi ..... 52
6.1.2 Pengujian Pengaruh Seleksi Fitur *Information Gain* ..... 52
6.2 Hasil Pengujian ..... 53
6.2.1 Hasil Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi ..... 54

6.2.2 Hasil Pengujian Pengaruh Seleksi Fitur *Information Gain* ..... 55
6.2.3 Hasil dari seluruh pengujian..... 57
6.3 Analisis Hasil..... 60
BAB 7 PENUTUP ..... 62
7.1 Kesimpulan..... 62
7.2 Saran ..... 62
DAFTAR REFERENSI ..... 64
LAMPIRAN A *DATABASE* PENDUKUNG ..... 66
A.1 *Database Stopword* Bahasa Indonesia..... 66
A.2 *Database Tweet* Data Latih *Hate Speech* Berbahasa Indonesia..... 70
A.3 *Database Tweet* Data Uji *Hate Speech* Berbahasa Indonesia ..... 77
A.4 *Database* Kamus Normalisasi *(FormalizationDict)* Pujangga *Indonesian Natural Language Processing* REST API..... 79

# DAFTAR GAMBAR

Gambar 3.1 Diagram alir gambaran umum sistem ..... 17
Gambar 4.1 Diagram alir metode *Naïve Bayes* dengan seleksi fitur ..... 18
Gambar 4.2 Diagram Alir Algoritme *Naïve Bayes* dengan *Information Gain* sebagai seleksi fitur ..... 19
Gambar 4.3 Diagram Alir *Pre-processing* ..... 20
Gambar 4.4 Diagram Alir *Case Folding* ..... 21
Gambar 4.5 Diagram Alir Proses *Cleaning* ..... 21
Gambar 4.6 Diagram Alir Normalisasi ..... 22
Gambar 4.7 Diagram Alir *Filtering* ..... 23
Gambar 4.8 Diagram Alir *Stemming* ..... 24
Gambar 4.9 Diagram Alir *Tokenizing* ..... 25
Gambar 4.10 Diagram Alir Seleksi Fitur *Information Gain* ..... 26
Gambar 4.11 Diagram Alir Proses *Multinomial Naive Bayes* ..... 28
Gambar 5.1 Tampilan dari Tahap *Pre-processing* ..... 51
Gambar 5.2 Tampilan Kelas dari Data Uji ..... 51
Gambar 6.1 Hasil Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi ..... 54
Gambar 6.2 Hasil Pengujian Pengaruh *Information Gain* Tanpa Normalisasi ..... 55
Gambar 6.3 Hasil Pengujian Pengaruh *Information Gain* Dengan Normalisasi ..... 56
Gambar 6.4 Hasil Nilai *Accuracy* Dari Seluruh Pengujian ..... 57
Gambar 6.5 Hasil Nilai *Precision* Dari Seluruh Pengujian ..... 58
Gambar 6.6 Hasil Nilai *Recall* Dari Seluruh Pengujian ..... 59
Gambar 6.7 Hasil Nilai *f-measure* Dari Seluruh Pengujian ..... 59
Gambar 6.8 Ilustrasi Proses Pengujian ..... 60

# DAFTAR TABEL

Tabel 2.1 Contoh Proses Normalisasi ..... 9
Tabel 2.2 Tabel *Confusion Matrix* Dua Kelas ..... 14
Tabel 4.1 *Data set Tweet Hate Speech* ..... 29
Tabel 4.2 *Data set Tweet Hate Speech* setelah *Case Folding* ..... 30
Tabel 4.3 *Data set Tweet Hate Speech* setelah Cleaning ..... 30
Tabel 4.4 *Data set Tweet Hate Speech* setelah dilakukan Normalisasi ..... 31
Tabel 4.5 *Database stopword* Bahasa Indonesia ..... 31
Tabel 4.6 Contoh Proses *Filtering Tweet Hate Speech* ..... 31
Tabel 4.7 Hasil *Filtering* Data Latih *Tweet Hate Speech* ..... 32
Tabel 4.8 Hasil *Filtering* Data Uji *Tweet Hate Speech* ..... 32
Tabel 4.9 Contoh Proses *Stemming Tweet Hate Speech* ..... 32
Tabel 4.10 Hasil *Stemming* Data Latih *Tweet Hate Speech* ..... 33
Tabel 4.11 Hasil *Stemming* Data Uji *Tweet Hate Speech* ..... 33
Tabel 4.12 Contoh Hasil *Tokenizing Tweet Hate Speech* ..... 33
Tabel 4.13 Hasil *Tokenizing* Data Latih *Tweet Hate Speech* ..... 34
Tabel 4.14 Hasil *Tokenizing* Data Uji *Tweet Hate Speech* ..... 34
Tabel 4.15 Frekuensi kemunculan kata ..... 34
Tabel 4.16 Hasil Perhitungan *Information Gain* ..... 35
Tabel 4.17 Hasil Pengurutan Term atau Kata dari *Information Gain* ..... 36
Tabel 4.18 Proses Menyimpan dan Membuang Term Berdasarkan *Information Gain* ..... 36
Tabel 4.19 Frekuensi kemunculan kata ..... 37
Tabel 4.20 Perhitungan Nilai *Prior* ..... 37
Tabel 4.21 Perhitungan Nilai *Conditional Probability* ..... 37
Tabel 4.22 Perhitungan *Total conditional probability* ..... 38
Tabel 4.23 Perhitungan *Posterior* kelas *Hate Speech* dan *Non Hate Speech* ..... 38
Tabel 4.24 Skenario pengujian pengaruh normalisasi kata ..... 39
Tabel 4.25 Skenario pengujian pengaruh seleksi fitur *Information Gain* ..... 40
Tabel 6.1 Hasil Pengujian Normalisasi dan Tanpa Normalisasi ..... 52
Tabel 6.2 Hasil Pengujian Pengaruh Seleksi Fitur *Information Gain* ..... 53

# DAFTAR LAMPIRAN

Lampiran 1 *Database Stopword* Bahasa Indonesia ..... 66

Lampiran 2 *Database Tweet* Data Latih *Hate Speech* Berbahasa Indonesia ..... 70

Lampiran 3 *Database Tweet* Data Uji *Hate Speech* Berbahasa Indonesia..... 77

Lampiran 4 *Database* Kamus Normalisasi (*FormalizationDict)* Pujangga *Indonesian Natural Language Processing* REST API ..... 79

# BAB 1
# PENDAHULUAN

## 1.1 Latar belakang

*Hate speech* atau ujaran kebencian adalah suatu bentuk ekspresi yang menyebar, menghasut, mempromosikan atau membenarkan kebencian, kekerasan dan diskriminasi terhadap seseorang atau sekelompok orang karena berbagai alasan (Davidson, Warmsley, Macy, & Weber, 2017). *Hate speech* dapat menimbulkan bahaya besar bagi kohesi masyarakat demokratis, perlindungan hak asasi manusia dan supremasi hukum. Jika dibiarkan tanpa penanganan, itu dapat menyebabkan aksi kekerasan dan konflik pada skala yang lebih luas. Dalam pengertian ini, *hate speech* adalah bentuk ekstrem dari intoleransi yang berkontribusi terhadap kejahatan kebencian. Banyak dampak yang dapat ditumbulkan karena *hate speech,* antara lain adalah memicu perpecahan, membuat generasi muda menjadi intoleran dan diskriminatif, menguntungkan pihak tertentu, mengakibatkan fakta tidak lagi dipercaya, konflik horizontal hingga terjadinya genosida. Kasus *hate speech* sering terjadi, khususnya pada media sosial.

Saat ini kebebasan berbicara dan mengemukakan pendapat di media sosial sudah sangat luas dan bebas, bahkan bisa sampai mengarah ke dalam unsur-unsur *hate speech.* Suatu komunikasi apapun yang meremehkan seseorang atau suatu kelompok berdasarkan beberapa karakteristik seperti ras, warna kulit, etnis, jenis kelamin, orientasi seksual, kebangsaan, agama, atau karakteristik lainnya sudah termasuk ke dalam unsur *hate speech*. Pada zaman sekarang ini, media sosial sering disalahgunakan demi tujuan yang buruk seperti menghina, mencemarkan nama baik, menista, melakukan perbuatan tidak menyenangkan, memprovokasi, menghasut, menyebarkan berita bohong, dan masih banyak lagi. Semuanya itu termasuk ke dalam tindak pidana *hate speech* dan sudah menjadi hal yang perlu diwaspadai dan perlu diperhatikan dalam menggunakan media sosial.

Menurut Kepala Subdit IT dan *Cyber Crime* Direktorat Tindak Pidana Ekonomi Khusus Badan Reserse Kriminal Kepolisian Negara Republik Indonesia (Bareskrim Polri), Kombes Pol Himawan Bayu Aji dikatakan bahwa konten berisi *hate speech* merupakan jenis tindak pidana yang paling banyak diadukan masyarakat ke polisi (Direktorat Tindak Pidana Siber Bareskrim Polri, 2017). Dari tahun ke tahun laporan mengenai tindak pidana *hate speech* cenderung meningkat. Pada 2015, jumlah laporan yang masuk berkaitan dengan *hate speech* sebanyak 671 kasus. Pada 2016, jumlah laporan yang masuk meningkat menjadi 1829 kasus. Pada 2017, jumlah laporan yang masuk meningkat sebesar 44,99% menjadi 3325 kasus. Selama 2017, Polri berhasil menyelesaikan 2108 kasus *hate speech.* Dari 3325 kasus sepanjang tahun 2017, tindak pidana *hate speech* mengenai penghinaan paling sering diadukan dengan 1657 kasus, kemudian disusul dengan kasus *hate speech* mengenai perbuatan tidak menyenangkan sebanyak 1224 kasus dan kasus *hate speech* mengenai pencemaran nama baik sebanyak 444 kasus. Sepanjang

tahun 2018, Polri berhasil menangkap 122 orang terkait mengenai *hate speech.* Setidaknya ada 3000 akun yang terdeteksi oleh Polri secara aktif untuk menyebarkan *hate speech* di media sosial.

Twitter merupakan salah satu jejaring sosial yang kontemporer dan populer sampai saat ini. Sebagai sistem *microblogging,* Twitter sering digunakan untuk tujuan seperti membuat status, memulai percakapan, membuat sebuah konten, dan mempromosikan sebuah produk (Benevenuto et al., 2010). Pada Twitter, teks atau pesan yang dibuat oleh pengguna disebut dengan *tweet* atau dalam bahasa Indonesia disebut sebagai kicauan. Melalui *tweet* ini, setiap orang bebas mengekspresikan perasaan dan emosi mereka dalam setiap peristiwa secara *real time* (Burnap & Williams, 2014).

Twitter sebagai sebuah situs jejaring sosial memberikan akses kepada penggunanya untuk mengekspresikan perasaan dan emosi mereka melalui *tweet* yang terdiri dari 280 karakter. Penggunaan karakter yang terbatas pada Twitter membuat sebuah *tweet* sering mengalami penyingkatan kata, penggunaan bahasa gaul, penggunaan bahasa yang tidak sesuai dengan standar yang ada ataupun terjadi kesalahan eja pada sebuah *tweet.* Hal itu menyebabkan perlu adanya suatu proses yang harus dilakukan agar dapat mengatasi permasalahan tersebut. Normalisasi kata adalah proses yang dilakukan untuk mengubah kata-kata yang tidak baku menjadi kata-kata yang baku sesuai dengan kamus yang ada. Normalisasi kata akan berguna untuk mengatasi permasalahan seperti banyaknya penyingkatan kata, penggunaan bahasa gaul atau slang, terjadinya kesalahan eja pada suatu kalimat, atau penggunaan bahasa yang tidak sesuai dengan standar yang ada. Dengan adanya proses normalisasi kata, permasalahan tersebut dapat diselesaikan dan akan membuat sebuah *tweet* dapat diproses dan dapat dianalisis dengan baik.

Permasalahan *hate speech* sudah diangkat menjadi beberapa topik penelitian. Salah satunya, penelitian mengenai deteksi *hate speech* pada Twitter dalam Bahasa Indonesia untuk pemilihan kepala daerah DKI Jakarta tahun 2017 (Alfina, Mulia, Fanany, & Ekanata, 2017). Penelitian ini membahas tentang perbandingan algoritme *Naïve Bayes* (NB), *Bayesian Logistic Regression* (BLR), *Random Forest Decision Tree* (RFDT), dan *Support Vector Machine* (SVM) dalam mengidentifikasi *hate speech* berbahasa Indonesia pada Twitter. Hasil penelitian ini menunjukkan *f-measure* superior dicapai ketika menggunakan kata *N-gram*, terutama ketika dikombinasikan dengan RFDT (93,5%), BLR (91,5%), dan NB (90,2%). Algoritme SVM ketika menggunakan kata *N-gram* hanya menghasilkan *F-measure* sebesar (86,5%). Penelitian yang dilakukan oleh (Kiema et al, 2018) juga membahas tentang deteksi *tweet* kebencian menggunakan algoritme *Naïve Bayes*. Hasil penelitian ini menunjukkan bahwa *Naive Bayes* memiliki akurasi di atas 80% dan mengungguli algoritme lain yaitu *NU Support Vector Classification* (NUSVC), *Logistic Regression*, *Linear Support Vector Classification* (Linear SVC), dan *Stochastic Gradient Descent Classification* (SGD*)*.

Beberapa peneliti sebelumnya juga telah melakukan penelitian mengenai metode *Naive Bayes* dan normalisasi kata. Penelitian yang dilakukan oleh (Saputra,

Adji & Permanasari, 2015) membahas tentang analisis sentimen data presiden Jokowi dengan *pre-processing* normalisasi dan *stemming* menggunakan metode *Naive Bayes* dan *Support Vector Machine*. Hasil Penelitian ini menunjukkan bahwa dengan melakukan normalisasi dan *stemming*, hasil akurasi yang didapatkan menjadi lebih tinggi. Hasil akurasi terbaik didapatkan dengan menggunakan metode *Unigram* pada *Support Vector Machine* sebesar 89,26% dan dengan menggunakan metode *N-Gram* pada *Naive Bayes* sebesar 88,70%. Kemudian penelitian oleh (Azis, 2013) membahas tentang sistem pengklasifikasian entitas pada pesan Twitter menggunakan ekspresi regular dan *Naive Bayes*. Hasil Penelitian ini menunjukkan bahwa proses normalisasi teks menggunakan fungsi penggantian kata tidak baku menghasilkan akurasi yang cukup baik sebesar 91,1% (9111 dari 1000 sampel data). Pada penelitian ini, sistem pengklasifikasian *tweet* menggunakan *Naive Bayes* memberikan akurasi yang cukup besar sebesar 96,75% untuk model *Multinomial Naive Bayes* dan 96,33% untuk model *Bernoulli Naive Bayes*.

Penelitian mengenai penggunaan seleksi fitur *Information Gain* dengan *Naive Bayes* juga sudah diangkat menjadi beberapa topik penelitian. Penelitian yang dilakukan oleh (Andilala, 2016) membahas tentang sentimen analisis pada ulasan film dengan menggunakan metode *Naïve Bayes* berbasis seleksi fitur. Hasil penelitian ini menunjukkan bahwa *Naïve Bayes* tanpa seleksi fitur mampu memberikan akurasi sebesar 95,15%. Kemudian setelah menambahkan *Information Gain* sebagai seleksi fitur maka akurasi dalam pengklasifikasian dokumen meningkat sebesar 0,9% dan menjadi 95,70%. Penelitian yang dilakukan oleh (Rozzaqi, 2015) juga membahas tentang penerapan algoritme *Naïve Bayes* dan *Information Gain* sebagai seleksi fitur untuk prediksi ketepatan kelulusan mahasiswa. Hasil penelitian ini menunjukkan bahwa algoritme *Naïve Bayes* dan seleksi fitur *Information Gain* untuk prediksi kelulusan menghasilkan akurasi tertinggi sebesar 89,79% dan *Area Under Curve* (AUC) tertinggi dengan hasil 0,875 dengan menggunakan *K=3*.

Metode *Naïve Bayes* dalam beberapa penelitian sebelumnya terbukti memiliki akurasi yang cukup tinggi dan waktu komputasi yang cepat jika diimplementasikan untuk menyelesaikan permasalahan yang ada. Berdasarkan beberapa penelitian sebelumnya dapat dikatakan juga bahwa proses normalisasi pada tahap *pre-processing* dan proses seleksi fitur *Information Gain* yang digunakan untuk menyeleksi fitur-fitur atau kata-kata yang memiliki nilai informasi yang tinggi dapat berguna untuk meningkatkan hasil akurasi dalam mengklasifikasikan *hate speech* berbahasa Indonesia pada Twitter.

Berdasarkan dari penjelasan di atas bahwa pada penelitian ini diberi judul "Klasifikasi *Hate Speech* Berbahasa Indonesia di Twitter Menggunakan Metode *Naive Bayes* dan Seleksi Fitur *Information Gain* dengan Normalisasi Kata". Pada penelitian ini diharapkan normalisasi *pre-processing* data Twitter menggunakan metode *Naive Bayes* dengan seleksi fitur *Information Gain* dapat mengklasifikasikan *hate speech* dengan tepat dan sesuai serta dapat meningkatkan hasil akurasi, sehingga tujuan utama dari penelitian ini dapat

terpenuhi yaitu untuk mengetahui pengaruh normalisasi kata pada tahap *pre-processing* untuk klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain*.

## 1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka rumusan masalah untuk penelitian ini adalah:

1. Bagaimana pengaruh normalisasi *pre-processing* data Twitter berbahasa Indonesia dengan menggunakan algoritme *Naive Bayes* dan seleksi fitur *Information Gain* untuk melakukan klasifikasi *hate speech* pada dokumen Twitter?
2. Bagaimana hasil akurasi dari perhitungan normalisasi *pre-processing* data Twitter berbahasa Indonesia dengan menggunakan algoritme *Naive Bayes* dan seleksi fitur *Information Gain* untuk melakukan klasifikasi *hate speech* pada dokumen Twitter?

## 1.3 Tujuan

Setelah rumusan masalah ditentukan, maka selanjutnya ditentukan juga tujuan akhir dari penelitian yang akan dilakukan, antara lain:

1. Mengetahui pengaruh normalisasi *pre-processing* data Twitter berbahasa Indonesia dengan menggunakan algoritme *Naive Bayes* dan seleksi fitur *Information Gain* untuk melakukan klasifikasi *hate speech* pada dokumen Twitter.
2. Mengetahui hasil akurasi dari proses normalisasi *pre-processing* data Twitter berbahasa Indonesia dengan menggunakan algoritme *Naive Bayes* dan seleksi fitur *Information Gain* untuk melakukan klasifikasi *hate speech* pada dokumen Twitter.

## 1.4 Manfaat

Pada penelitian yang akan dilakukan sangat diharapkan akan memiliki manfaat yang berguna, antara lain:

1. Memberikan kontribusi pengetahuan mengenai pengaruh penggunaan metode *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata untuk klasifikasi *hate speech* berbahasa Indonesia di Twitter.
2. Mendapatkan proses normalisasi kata dan seleksi fitur dengan *threshold* yang tepat sehingga dapat meningkatkan nilai akurasi untuk klasifikasi *hate speech* berbahasa Indonesia.

## 1.5 Batasan masalah

Pada penelitian yang akan dilakukan, permasalahan yang diangkat memiliki batasan-batasan tertentu yang telah dijabarkan sebagai berikut.

1. Penelitian yang akan dilakukan berfokus pada klasifikasi *hate speech* berbahasa Indonesia hanya pada sosial media Twitter.
2. Data yang digunakan berupa *tweet* dengan format data hanya teks yang berasal dari *tweet* pengguna Twitter yang sudah dipilih sesuai dengan kategori *hate speech* dan *non hate speech*.
3. Kategori kelas hanya terdiri dari dua kelas yaitu *hate speech* dan *non hate speech*.

## 1.6 Sistematika pembahasan

**BAB I – PENDAHULUAN**

Bab ini berisi mengenai latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, dan sistematika pembahasan.

**BAB II – LANDASAN KEPUSTAKAAN**

Bab ini berisi mengenai kajian pustaka dan dasar teori yang berkaitan serta menunjang proses penelitian.

**BAB III – METODOLOGI**

Bab ini berisi mengenai metode dan langkah kerja penelitian yang terdiri dari tipe penelitian, strategi penelitian, lokasi penelitian, teknik pengumpulan data, data penelitian, implementasi algoritme, teknik analisis data, dan peralatan pendukung pada penelitian.

**BAB IV – PERANCANGAN**

Bab ini berisi mengenai perancangan sistem menggunakan diagram alir dan perhitungan pada sampel data seperti perhitungan manualisasi dan evaluasi hasil penelitian.

**BAB V – IMPLEMENTASI**

Bab ini berisi mengenai bagaimana sistem diimplementasikan dalam bentuk program, perangkat lunak, dan perangkat keras apa saja yang digunakan serta hasil implementasi berupa *source code*.

**BAB VI – PENGUJIAN DAN ANALISIS**

Bab ini berisi mengenai bagaimana pengujian yang didapatkan dari penelitian yang telah dilakukan dan menganalisis hasil pengujian dari penelitian yang telah dilakukan.

**BAB VII – PENUTUP**

Bab ini berisi mengenai kesimpulan dari penelitian yang telah dilakukan dan saran dalam pengembangan penelitian selanjutnya.

# BAB 2
# LANDASAN KEPUSTAKAAN

## 2.1 Kajian Pustaka

Berdasarkan topik penelitian mengenai klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata, terdapat beberapa penelitian terdahulu yang relevan dengan topik tersebut. Penelitian mengenai *hate speech* sudah diangkat menjadi beberapa topik penelitian, salah satunya penelitian mengenai deteksi *tweet* kebencian menggunakan algoritme *Naive Bayes* (Kiema et al, 2018). Penelitian ini mengumpulkan data pelatihan dari 45645 *tweet* yang didapatkan dari *Application Programming Interface* (API) Twitter dan menguji data dari 2.228 *tweet*. Tujuan dari penelitian ini adalah untuk mengevaluasi kinerja untuk klasifikasi sentimen dalam hal akurasi, presisi dan daya ingat. Dalam penelitian ini, ada beberapa algoritme yang dibandingkan untuk melakukan analisis sentimen dan deteksi *tweet* kebencian di Twitter, antara lain adalah *NU Support Vector Classification* (NUSVC), *Logistic Regression*, *Linear Support Vector Classification* (Linear SVC), dan *Stochastic Gradient Descent Classification* (SGD*)*. Hasil penelitian ini menunjukkan bahwa *Naive Bayes* memberikan akurasi di atas 80% dan menghasilkan hasil yang lebih baik untuk tinjauan *tweet* kebencian serta mengungguli algoritme lain.

Penelitian yang dilakukan oleh (Alfina, Mulia, Fanany, & Ekanata, 2017) juga membahas tentang perbandingan algoritme *Naïve Bayes* (NB) dengan beberapa algoritme lain. Dalam penelitian ini, *Naive Bayes* (NB) dibandingkan dengan *Bayesian Logistic Regression* (BLR), *Random Forest Decision Tree* (RFDT), dan *Support Vector Machine* (SVM) dalam mengidentifikasi *hate speech* berbahasa Indonesia pada Twitter. Penelitian ini menunjukkan beberapa hasil penelitian, yang pertama adalah *f-measure* superior dicapai ketika menggunakan kata *N-gram*, terutama ketika dikombinasikan dengan RFDT (93,5%), BLR (91,5%), dan NB (90,2%). Algoritme SVM ketika menggunakan kata *N-gram* hanya menghasilkan *f-measure* sebesar (86,5%). Kedua, hasil penelitian ini juga menemukan bahwa fitur kata *N-gram* lebih unggul dari karakter *N-gram*. Ketiga, hasil penelitian ini juga menunjukkan bahwa menggunakan kata *Unigram* saja jauh lebih baik daripada menggunakan kata bigram atau bahkan menyatukan kata *Unigram* dan kata *Bigram*.

Penggunaan metode *N-gram* dan *Unigram* akan memberikan hasil yang lebih baik dibandingkan metode *Bigram* dan *Trigram* juga dibuktikan oleh Saputra, Adji & Permanasari (2015). Penelitian ini membahas tentang analisis sentimen data presiden Jokowi dengan *pre-processing* normalisasi dan *stemming* menggunakan metode *Naive Bayes* dan SVM. Hasil pada penelitian ini menunjukkan bahwa nilai akurasi terbaik didapatkan dengan menggunakan metode *Unigram* pada SVM sebesar 89,26% dan dengan menggunakan metode *N-Gram* pada *Naive Bayes* sebesar 88,70%. Pada penelitian ini juga memperlihatkan bahwa dengan

melakukan normalisasi dan *stemming*, maka hasil akurasi yang didapatkan akan menjadi lebih tinggi atau lebih optimal.

Penelitian mengenai normalisasi teks juga dilakukan oleh Aziz (2013) dengan tujuan untuk membuat sistem pengklasifikasian entitas pada pesan Twitter menggunakan ekspresi regular dan *Naive Bayes*. Hasil penelitian ini menunjukkan bahwa proses normalisasi teks menggunakan fungsi penggantian kata tidak baku menghasilkan akurasi yang cukup baik sebesar 91,1% (9111 dari 1000 sampel data). Proses normalisasi teks dengan penggantian kata baku juga membutuhkan waktu yang lebih cepat dibandingkan dengan menggunakan fungsi jarak *Levenshtein*. Hasil dari sistem pengklasifikasian *tweet* menggunakan *Naive Bayes* memberikan akurasi yang cukup besar yaitu 96,33% untuk model *Bernoulli Naive Bayes* dan 96,75% untuk model *Multinomial Naive Bayes*.

*Multinomial Naive Bayes* merupakan model pengembangan dari algoritme Bayes yang sangat cocok digunakan dalam pengklasifikasian teks atau dokumen. Namun, *Multinomial Naive Bayes* memiliki kekurangan yaitu sangat sensitif pada fitur yang terlalu banyak. Seleksi fitur adalah salah satu cara yang dapat digunakan untuk mengurangi bias dan mengurangi dimensi atribut agar dapat meningkatkan akurasi dalam proses klasifikasi menggunakan *Multinomial Naive Bayes*. Penelitian yang dilakukan oleh Andilala (2016) membuktikan bahwa seleksi fitur mampu meningkatkan hasil akurasi dari analisis sentimen pada ulasan film. Seleksi Fitur yang digunakan adalah *Information Gain*. Hasil penelitian ini menunjukkan bahwa akurasi yang didapatkan dengan menggunakan metode *Naive Bayes* tanpa seleksi fitur mampu memberikan akurasi sebesar 95,15%. Kemudian akurasi yang dihasilkan setelah memberikan seleksi fitur *Information Gain* terhadap algoritme *Naive Bayes* adalah sebesar 95,70%. Hasil akurasi dalam pengklasifikasian dokumen dengan penambahan seleksi fitur meningkat sebesar 0,9%.

Penerapan algoritme *Naive Bayes* dan *Information Gain* sebagai seleksi fitur juga dilakukan oleh Rozzaqi (2015) untuk melakukan prediksi ketepatan kelulusan mahasiswa. Hasil penelitian ini menunjukkan bahwa algoritme *Naive Bayes* dan seleksi fitur *Information Gain* untuk melakukan prediksi kelulusan mahasiswa menghasilkan akurasi tertinggi sebesar 89,79% dengan menggunakan *K=3*. Pada peneletian ini juga diperoleh *Area Under Curve* (AUC) tertinggi dengan hasil 0,875 dengan *K=3*.

Berdasarkan referensi dari beberapa penelitian sebelumnya, maka algoritme *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata akan digunakan pada penelitian ini karena algoritme *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata dapat dipadukan dan terbukti dapat memberikan hasil akurasi yang baik serta memiliki waktu komputasi yang cepat dibandingkan algoritme lainnya.

## 2.2 *Hate Speech*

*Hate speech* adalah suatu bentuk khusus dari bahasa ofensif yang digunakan untuk mengekspresikan ideologi kebencian. *Hate speech* didefinisikan sebagai

komunikasi apapun yang meremehkan seseorang atau suatu kelompok berdasarkan beberapa karakteristik seperti ras, warna kulit, etnis, jenis kelamin, orientasi seksual, kebangsaan, agama, atau karakteristik lainnya (Nockleby, 2000). Ada beberapa tindakan *hate speech* yang termasuk dalam tindak pidana berdasarkan Kitab Undang Undang Hukum Pidana (KUHP), antara lain:

1. melakukan penghinaan,
2. melakukan pencemaran nama baik,
3. melakukan penistaan,
4. melakukan perbuatan tidak menyenangkan,
5. melakukan provokasi,
6. melakukan hasutan, dan
7. melakukan penyebaran berita bohong.

*Hate Speech* di Indonesia juga mengacu pada Undang Undang Nomor 11 Tahun 2008 mengenai Informasi dan Transaksi Elektronik atau yang dikenal dengan UU ITE. Hukum yang berkaitan dengan *hate speech* tercantum dalam Pasal 27 ayat 3 Undang Undang Informasi dan Transaksi Elektronik yang berbunyi: "Setiap orang dengan sengaja dan tanpa hak mendistribusikan dan/atau mentransmisikan dan/atau membuat dapat diaksesnya Informasi Elektronik dan/atau Dokumen Elektronik yang memiliki muatan penghinaan dan/atau pencemaran nama baik". Hukuman pidana bagi seseorang yang melanggar Pasal 27 ayat 3 adalah membayar denda maksimal seharga 1 miliar ataupun menerima hukuman pidana penjara maksimal 6 tahun lamanya.

## 2.3 Twitter

Twitter merupakan salah satu jejaring sosial yang kontemporer dan populer sampai saat ini. Sebagai sistem *microblogging,* Twitter sering digunakan untuk tujuan seperti membuat status, memulai percakapan, membuat sebuah konten, mempromosikan sebuah produk, dan bahkan untuk mengirim sebuah *spam* (Benevenuto et al., 2010). Pada Twitter, teks atau pesan yang dibuat oleh pengguna disebut dengan *tweet* atau dalam bahasa Indonesia yang berarti kicauan. Setiap *tweet* yang dibuat oleh pengguna memiliki batas karakter yaitu sebanyak 280 huruf. Penggunaan karakter yang terbatas pada Twitter membuat sebuah *tweet* sering mengalami penyingkatan kata, penggunaan bahasa gaul, ataupun terjadi kesalahan eja pada sebuah kata (Agarwal, et al., 2014).

## 2.4 *Text Mining*

*Text mining* adalah sebuah proses ekstraksi sebuah pola (informasi dan pengetahuan yang bermanfaat) dari sejumlah sumber data besar yang tidak terstruktur (Ian H. Witten, 2011). Tujuan utama dari *text mining* adalah untuk menyusun dokumen teks. Ada beberapa teknik pada *text mining*, seperti ekstraksi informasi, pengambilan informasi, pemrosesan bahasa alami, pemrosesan kueri,

kategorisasi, dan pengelompokkan teks. Data yang akan diolah pada *text mining* adalah data yang tidak (atau kurang) terstruktur seperti dokumen *Word*, PDF, kutipan teks, *tweet*, dan lain-lain. Sementara pada data mining, data yang akan diolah adalah data terstruktur (Ronen Feldman, 2007). Pada umumnya, proses *text mining* memiliki langkah-langkah, sebagai berikut:

1. mengubah *input* teks yang tidak terstruktur menjadi *database* terstruktur,
2. melakukan identifikasi pola dan *trends* dari data terstruktur,
3. melakukan analisis dan menafsirkan pola dan *trends,* dan
4. melakukan ekstraksi informasiyang berguna dari teks.

## 2.5 Normalisasi

Normalisasi kata adalah proses untuk melakukan pengubahan kata dari yang tidak baku menjadi kata yang baku. Proses normalisasi kata (Buntoro, et al., 2014), meliputi:

1. merenggangkan tanda baca (*punctuation*) dan simbol selain alfabet,
2. mengubah semua huruf dalam dokumen menjadi huruf kecil,
3. mengubah kata yang tidak baku menjadi kata yang baku sesuai dengan kamus,
4. menghilangkan huruf yang berulang, dan
5. menghilangkan *emoticon.*

Pada penelitian ini digunakan proses normalisasi menggunakan Pujangga *Indonesian Natural Language Processing* REST API untuk mengatasi permasalahan kata-kata yang tidak baku pada bahasa gaul (slang), singkatan kata, dan salah eja. Normalisasi menggunakan Pujangga *Indonesian Natural Language Processing* REST API dilakukan dengan melakukan pengecekan setiap kata yang ada pada sebuah kalimat dengan menggunakan kamus *formalizationDict*. Jika terdapat kata-kata yang tidak baku pada *formalizationDict* maka nanti akan langsung diubah menjadi kata-kata yang baku sesuai dengan ketentuan pada *formalizationDict.* Contoh proses normalisasi kata terdapat pada Tabel 2.1

**Tabel 2.1 Contoh Proses Normalisasi**

| Sebelum Normalisasi | Setelah Normalisasi |
|---|---|
| adlh | adalah |
| akku | aku |
| boljug | boleh juga |

## 2.6 *Text Pre-processing*

Pada *text mining*, teks diolah menjadi lebih terstruktur melalui beberapa tahapan. *Text pre-processing* adalah suatu tahapan pada *text mining* untuk mempersiapkan sebuah data yang tidak terstruktur supaya menjadi data yang

telah siap untuk diolah dan dilakukan analisis (Hadna, et al., 2016). Pada data yang belum dilakukan *pre-processing* masih akan menjadi data yang mentah, yaitu data yang masih belum siap untuk dilakukan analisis, karena masih banyak mengandung kata-kata yang tidak memiliki arti dan kata yang belum terstruktur. Oleh karena itu, diperlukan *text pre-processing* untuk mengolah teks menjadi lebih terstruktur.

Pada proses *text pre-processing* terdapat beberapa tahapan umum yang dilakukan seperti *case folding*, *tokenzing*, *filtering*, dan s*temming* (Triawati, 2009). Pada penelitian ini, *pre-processing* yang dilakukan menggunakan *library* dari Pujangga *Indonesian Natural Language Processing* REST API yang mempunyai tahapan sedikit berbeda dan akan dijelaskan sebagai berikut:

1. *Case Folding*

   *Case folding* adalah tahapan untuk mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya alfabet yang diubah menjadi huruf kecil semua, untuk karakter selain huruf akan dihilangkan dan dianggap sebuah pembatas (Feldman & Sanger, 2007).

2. *Cleaning*

   *Cleaning* adalah tahapan yang dilakukan untuk menghilangkan *noise* (karakter yang tidak penting) pada dokumen teks. Pada dokumen teks *tweet* terdapat beberapa variabel atau karakter yang perlu dihilangkan karena tidak memiliki pengaruh dalam pemrosesan teks. Beberapa variabel pada *tweet*, antara lain yaitu *link* URL (http), *username* (@), *hashtag* (#), dan *retweet* (RT).

3. Normalisasi

   Normalisasi kata adalah tahapan untuk mengubah kata-kata yang tidak baku menjadi kata-kata yang baku. Pada normalisasi kata ini dilakukan pencocokan kata terhadap kamus *formalizationDict* yang didapatkan dari Pujangga *Indonesian Natural Language Processing* REST API.

4. *Filtering*

   *Filtering* adalah proses penghapusan kata atau *term* pada *tweet* yang tidak memengaruhi suatu proses dalam sistem. Ada dua algoritme pada proses *filtering* yaitu *stoplist* (membuang kata yang kurang penting) dan *wordlist* (menyimpan kata yang penting). *Filtering* pada penelitian ini menggunakan *stoplist*. *Stoplist* atau *stopword* yang digunakan pada penelitian ini berasal dari Pujangga *Indonesian Natural Language Processing* REST API dan merupakan *stopword* dalam bahasa Indonesia.

5. *Stemming*

   *Stemming* adalah tahapan untuk mengubah kata-kata dalam dokumen menjadi kata dasar menggunakan aturan tertentu sesuai dengan algoritme yang digunakan. Contohnya adalah kata yang memiliki imbuhan yang

nantinya akan diubah menjadi sebuah kata dasar, seperti "cacian" menjadi caci, "bertanding" menjadi "tanding" dan masih banyak lagi. Pada penelitian ini, proses *stemming* menggunakan *library* dari Pujangga *Indonesian Natural Language Processing* REST API *stemming* dalam bahasa Indonesia.

6. *Tokenizing*

   *Tokenizing* adalah tahap untuk melakukan pemotongan *string input* berdasarkan kata yang menyusunnya. Cara pemotongannya dengan menggunakan *spasi* untuk memisahkan antar kata dalam dokumen.

## 2.7 *Naïve Bayes Classifier*

*Naive Bayes Classifier* adalah sebuah metode klasifikasi yang berakar pada teorema Bayes. Metode ini melakukan proses klasifikasi dengan menggunakan metode probabilitas dan statistik yaitu dengan memprediksi peluang berdasarkan pengalaman di masa sebelumnya (Teorema Bayes). Ciri utama *Naive Bayes Classifier* adalah asumsi yang sangat kuat (naif) akan ketergantungan dari masing-masing kondisi/kejadian. *Naive Bayes* memiliki beberapa keunggulan, antara lain proses komputasi yang cepat, mudah diimplementasikan dengan struktur yang sederhana, dan efektif (Sona Taheri, 2013).

*Naive Bayes* juga merupakan salah satu dari 10 algoritme teratas yang sering diterapkan dalam *data mining* (Wu et al., 2008). Hal itu dikarenakan model *Naive Bayes* sangat menarik dengan 3 hal utama yang dimiliki oleh algoritme ini yaitu *simplicity* (kesederhanaan), *elegance* (keanggunan), dan *robustness* (kekokohan). Algoritme *Naive Bayes* telah digunakan dengan cukup sukses dalam konteks beragam aplikasi dan sangat populer dalam konteks klasifikasi teks. Dalam melakukan klasifikasi teks, algoritme *Naive Bayes* mampu menghasilkan akurasi yang tinggi dan kompleksitas *run time* yang baik dengan jumlah data yang besar (Aggarwal, 2015). Proses klasifikasi menggunakan algoritme *Naive Bayes* terdapat pada Persamaan 2.1

$$P(H_j|X_i) = \frac{P(H_j)\ P(X_i|H_j)}{P(X_i)} \quad (2.1)$$

Keterangan:

$P(H_j|X_i)$ : peluang pada kelas *j* saat ada munculnya kata *i*

$P(H_j)$ : peluang munculnya sebuah kelas *j*

$P(X_i|H_j)$ : peluang kata *i* masuk ke dalam kelas *j*

$P(X_i)$ : peluang munculnya sebuah kata

Pada saat melakukan proses klasifikasi menggunakan algoritme *Naive Bayes*, peluang munculnya sebuah kata dapat dihilangkan karena tidak memiliki pengaruh pada hasil klasifikasi dari setiap kelas. Proses klasifikasi menggunakan algoritme *Naive Bayes* dapat disederhanakan pada Persamaan 2.2.

$$P(H_j|X_i) = \quad P(H_j)\, P(X_i|H_j) \tag{2.2}$$

Selanjutnya, pada saat melakukan klasifikasi menggunakan algoritme *Naive Bayes* perlu dilakukan perhitungan nilai *prior,* yaitu menghitung nilai peluang munculnya suatu kelas pada seluruh dokumen. Proses perhitungan nilai *prior* terdapat pada Persamaan 2.3.

$$P(H_j) = \frac{N_H}{N} \tag{2.3}$$

Keterangan:

$N_H$ : banyaknya dokumen dengan kelas $H_j$ pada dokumen data latih

*N* : total seluruh dokumen data latih yang digunakan dalam klasifikasi

Langkah selanjutnya adalah melakukan perhitungan nilai *posterior,* yaitu dengan mengalikan nilai *prior* dengan nilai *total conditional probability*. Perhitungan nilai *posterior* terdapat pada Persamaan 2.4.

$$P(H_j|X_i) = \quad P(H_j)\; x\; P(X_1|H_j)\; x\; P(X_2|H_j) \ldots P(X_n|H_j) \tag{2.4}$$

## 2.8 *Multinomial Naive Bayes*

*Multinomial Naive Bayes* merupakan model pengembangan dari algoritme Bayes yang sangat cocok digunakan dalam pengklasifikasian teks atau dokumen. Pada formula *Multinomial Naive Bayes*, kelas dokumen tidak hanya ditentukan dengan kata yang muncul tetapi juga jumlah frekuensi kemunculannya (Ian H. Witten, 2011). Proses perhitungan nilai *conditional probability* pada algoritme *Naive Bayes* untuk menghitung peluang sebuah kata atau kueri *i* yang terdapat pada kelas *j* terdapat pada Persamaaan 2.5.

$$P(X_i|H_j) = \frac{count(X_i, H_j) + 1}{\left(\sum_{X \epsilon V} count(X, H_j)\right) + |V|} \tag{2.5}$$

Keterangan :

$count(X_i, H_j)$ : total sebuah kata atau kueri yang terdapat pada suatu kelas. Penjumlahan dengan 1 diperlukan untuk menghindari hasil yang didapatkan bernilai 0

$\sum_{w \epsilon V} count(X, H_j)$ : total seluruh kata yang terdapat pada kelas $H_j$

$|V|$ : total seluruh kata unik yang terdapat pada seluruh kelas

## 2.9 *Information Gain*

*Information Gain* adalah salah satu metode seleksi fitur yang sukses dalam pengklasifikasian teks (Uysal & Gunal, 2012). *Information Gain* bekerja dengan mengukur berapa banyak informasi kehadiran dan ketidakhadiran dari suatu kata yang berperan untuk membuat keputusan klasifikasi yang benar dalam suatu

kelas. *Information Gain* merupakan salah satu algoritme seleksi fitur yang digunakan untuk memilih fitur terbaik. *Information Gain* dimanfaatkan untuk memberi peringkat pada kata-kata penting dari hasil reduksi fitur. Hasil dari proses *Information Gain* adalah kata penting yang bersifat informatif. Metode *Information Gain* dapat melihat setiap fitur untuk memprediksi label kelas yang benar karena memilih fitur dengan nilai yang tertinggi dan lebih efektif untuk mengoptimalkan hasil klasifikasi.

Terdapat 3 proses yang harus dilakukan untuk menyeleksi fitur-fitur mana saja yang akan digunakan pada metode *Information Gain*, antara lain:

1. Setiap atribut dilakukan perhitungan untuk mendapatkan nilai *Information Gain* untuk masing-masing atribut.
2. Menentukan sebuah *threshold* yang akan digunakan. *Threshold* yang ditentukan berguna untuk menentukan apakah sebuah atribut layak untuk disimpan atau harus dibuang.
3. Setiap atribut yang memenuhi *threshold* akan disimpan dan setiap atribut yang tidak memenuhi *threshold* akan dibuang, sehingga akan memperbaiki *data set* yang ada.

   Proses pemilihan fitur-fitur yang aan digunakan pada *Information Gain* terdapat pada Persamaan 2.6.

$$IG(t) = -\sum_{i=1}^{|C|} P(Ci)logP(Ci) + P(t)\sum_{i=1}^{|C|} P(Ci|t)logP(Ci|t) + P(\bar{t})\sum_{i=1}^{|C|} P(Ci|\bar{t})logP(Ci|\bar{t}) \quad (2.6)$$

Keterangan:

$P(Ci)$ : nilai peluang pada kelas *i* pada keseluruhan dokumen

$P(t)$ : nilai peluang suatu kata *t* yang terdapat dalam dokumen

$P(\bar{t})$ : nilai peluang suatu kata *t* yang tidak terdapat dalam dokumen

$P(Ci|t)$ : nilai peluang suatu kata *t* yang terdapat pada kelas *i* dalam dokumen

$P(Ci|\bar{t})$ : nilai peluang suatu kata *t* yang tidak terdapat pada kelas *i* dalam dokumen

## 2.10 Evaluasi

Hasil evaluasi dalam proses klasifikasi *tweet hate speech* dapat diukur berdasarkan nilai *accuracy,* nilai *recall*, nilai *precision,* dan nilai *f-measure*. Pengukuran utama adalah klasifikasi *accuracy,* yang merupakan jumlah kasus *tweet* yang diklasifikasikan dengan benar pada data uji dibagi dengan jumlah total kasus dalam data uji. *Recall* adalah nilai tingkat keberhasilan mengenali suatu kelas yang harus dikenali. *Precision* adalah nilai tingkat ketepatan hasil klasifikasi dari seluruh dokumen. *f-measure* merupakan nilai yang mewakili keseluruhan

kinerja sistem dan merupakan penggabungan nilai *recall* dan nilai *precision* (Destuardi dan Surya, 2009). Rumus perhitungan untuk mendapatkan nilai *accuracy*, nilai *precision*, nilai *recall* dan nilai *f-measure* terdapat pada Persamaan 2.7 sampai Persamaan 2.10.

$$accuracy = \frac{TN + TP}{TN + TP + FP + FN} \tag{2.7}$$

$$precision = \frac{TP}{TP + FP} \tag{2.8}$$

$$recall = \frac{TP}{TP + FN} \tag{2.9}$$

$$f - measure = \frac{2 \times Pecision \times Recall}{Precision + Recall} \tag{2.10}$$

Nilai *accuracy,* nilai *precision,* nilai *recall,* dan nilai *f-measure* dapat ditentukan menggunakan nilai yang sudah ditentukan pada *confusion matrix*. *Confusion matrix* merupakan klasifikasi aktual dan prediksi pada sistem klasifikasi. Pada penelitian ini terdapat dua kelas klasifikasi, maka evaluasi menggunakan *confusion matrix* yang ditunjukan pada Tabel 2.2.

**Tabel 2.2 Tabel *Confusion Matrix* Dua Kelas**

| | *Actual class (expectation)* | | |
|---|---|---|---|
| *Prediction Class (Observation)* | | *Hate Speech* | *Non Hate Speech* |
| | *Hate Speech* | TP | FP |
| | *Non Hate Speech* | FN | TN |

Keterangan:

*True Positive* (TP) : dokumen *tweet* pada kelas *Hate Speech* yang benar diprediksi oleh sistem dengan hasil *Hate Speech*

*True Negative* (TN) : dokumen *tweet* pada kelas *Non Hate Speech* yang benar diprediksi oleh sistem dengan hasil *Non Hate Speech*

*False Positive* (FP) : dokumen *tweet* pada kelas *Non Hate Speech* yang salah diprediksi oleh sistem dengan hasil *Hate Speech*

*False Negative* (FN) : dokumen *tweet* pada kelas *Hate Speech* yang salah diprediksi oleh sistem dengan hasil *Non Hate Speech*

# BAB 3
# METODOLOGI

## 3.1 Tipe Penelitian

Penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata memiliki tipe penelitian yaitu *non-implementatif* analitik. Penelitian ini berfokus pada suatu permasalahan yang menjadi dasar sebagai objek penelitian yang di dalamnya terdapat proses pengambilan keputusan untuk menyelesaikan masalah tersebut.

## 3.2 Strategi Penelitian

Penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata memiliki strategi penelitian yang bersifat kualitatif dengan studi kasus yang menjadi strateginya. Studi kasus memiliki tujuan untuk menganalisis dan menarik kesimpulan dari suatu kasus atau topik yang diangkat dalam penelitian yaitu klasifikasi *hate speech* berbahasa Indonesia pada Twitter.

## 3.3 Lokasi Penelitian

Lokasi penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata bertempat pada Laboratorium Riset Komputer Cerdas (KC) Fakultas Ilmu Komputer Universitas Brawijaya. Penelitian ini bertujuan untuk menerapkan normalisasi kata pada tahap *pre-processing* dan seleksi fitur *Information Gain* dengan menggunakan metode *Naïve Bayes* untuk klasifikasi *hate speech* berbahasa Indonesia pada Twitter.

## 3.4 Teknik Pengumpulan Data

Penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata memiliki teknik pengumpulan data yaitu dengan menggunakan data sekunder. Data tersebut diperoleh dari penelitian sebelumnya mengenai deteksi *hate speech* pada Twitter dalam Bahasa Indonesia untuk pemilihan kepala daerah DKI Jakarta tahun 2017 (Alfina, Mulia, Fanany, & Ekanata, 2017). *Database* ini berisi kumpulan *tweet hate speech* dan *non hate speech*.

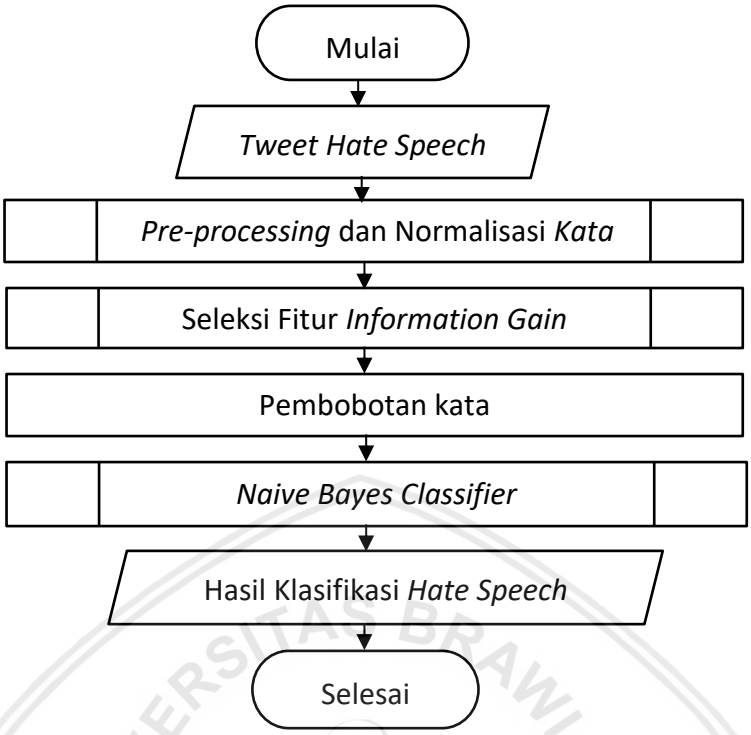
## 3.5 Data Penelitian

Penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata menggunakan data *tweet* pada Twitter mengenai *hate speech* berbahasa Indonesia. Total data *tweet hate speech* tersebut adalah sebanyak 250 data,

dengan total pembagian data *tweet hate speech* yaitu sebesar 80% untuk data latih dan 20% untuk data uji. Total data latih yang digunakan sebanyak 100 data dengan label *hate speech* dan 100 data dengan label *non hate speech*. Adapun untuk data uji akan digunakan 25 data dengan label *hate speech* dan 25 data dengan label *non hate speech*. Berdasarkan penelitian yang dilakukan oleh Fitri Cahyanti, Saptono & Widya Sihwi (2016), hasil akurasi terbaik sebesar 94,8% diperoleh dengan menggunakan skenario pengujian dengan melakukan perbandingan data latih dan data uji sebesar 80% dan 20%. Penelitian ini bertujuan untuk menentukan model terbaik pada metode *Naive Bayes Classifier* dalam menentukan status gizi balita dengan mempertimbangkan independensi parameter.

## 3.6 Implementasi Algoritme

Pada penelitian klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata, proses implementasi algoritme dilakukan dengan mengolah *tweet hate speech* pada Twitter kemudian melakukan perhitungan manualisasi menggunakan metode *Naive Bayes* dan seleksi fitur *Information Gain*. Proses implementasi dimulai dengan melakukan beberapa tahapan pada *text pre-processing.* Tahapan pertama adalah *case folding* yaitu mengubah seluruh *tweet* menjadi huruf kecil atau *lowercase*. Tahapan kedua adalah *cleaning* yaitu proses untuk membersihkan *tweet* dari karakter atau komponen yang kurang penting. Tahapan ketiga adalah normalisasi kata yaitu proses untuk mengubah kata-kata pada *tweet* yang semula tidak baku menjadi baku. Tahapan keempat adalah *filtering* yaitu proses untuk memisahkan kata pada *tweet* yang tidak memengaruhi suatu proses dalam sistem. Tahapan kelima adalah *stemming* yaitu proses mengubah kata-kata pada *tweet* menjadi kata dasar. Tahapan terakhir adalah *tokenizing* yaitu proses pemotongan *string input* berdasarkan kata yang menyusunnya. Setelah dilakukan tahap *pre-processing* akan dilanjutkan dengan melakukan penghitungan *Term Frequency* (TF). Kemudian sistem akan melakukan perhitungan seleksi fitur *Information Gain*, dan terakhir sistem akan melakukan proses klasifikasi menggunakan metode *Multinomial Naïve Bayes*. Implementasi algoritme diikuti dengan gambaran *flowchart* dari penggunaan metode *Naïve Bayes* dan seleksi fitur *Information Gain*, serta perancangan pengujian yang akan dilakukan. Pembuatan kode program untuk melakukan implementasi algoritme menggunakan bahasa pemrograman Python 3.7 dan memanfaatkan beberapa *library* yang dibutuhkan. Penjelasan gambaran umum sistem terdapat pada Gambar 3.1.




**Gambar 3.1 Diagram alir gambaran umum sistem**

## 3.7 Teknik Analisis Data

Teknik analisis data pada klasifikasi *hate speech* berbahasa Indonesia pada Twitter ini berupa pengujian dari hasil perhitungan normalisasi *pre-processing* data dengan seleksi fitur *Information Gain* dan metode *Multinomial Naïve Bayes.* Pengujian tersebut dilakukan dengan melakukan evaluasi terhadap hasil *accuracy*, *precision*, *recall*, dan *f-measure* yang dihasilkan oleh masing-masing pengujian.

## 3.8 Peralatan Pendukung

Penelitian normalisasi *pre-processing* data Twitter berbahasa Indonesia untuk klasifikasi *hate speech* dengan menggunakan metode *Naive Bayes* dan seleksi fitur *Information Gain* menggunakan beberapa peralatan pendukung yang berguna untuk membantu jalannya proses penelitian, antara lain:

1. *Hardware* atau perangkat keras berupa:
   - Processor Intel(R) Core(TM) i5-5200U CPU dengan RAM 4 GB
   - Harddisk sebesar 500 GB dan memori RAM sebesar 8192 MB.
2. *Software* atau perangkat lunak berupa:
   - Sistem operasi yang digunakan adalah Windows 10 Pro 64-bit.
   - Editor yang dipakai adalah Spyder Anaconda.
   - Bahasa pemrograman yang digunakan adalah Python 3.7.
   - *Database* yang digunakan adalah Mongo *Database*.
   - *Library* yang digunakan untuk normalisasi adalah Pujangga *Indonesian Natural Language Processing* REST API.

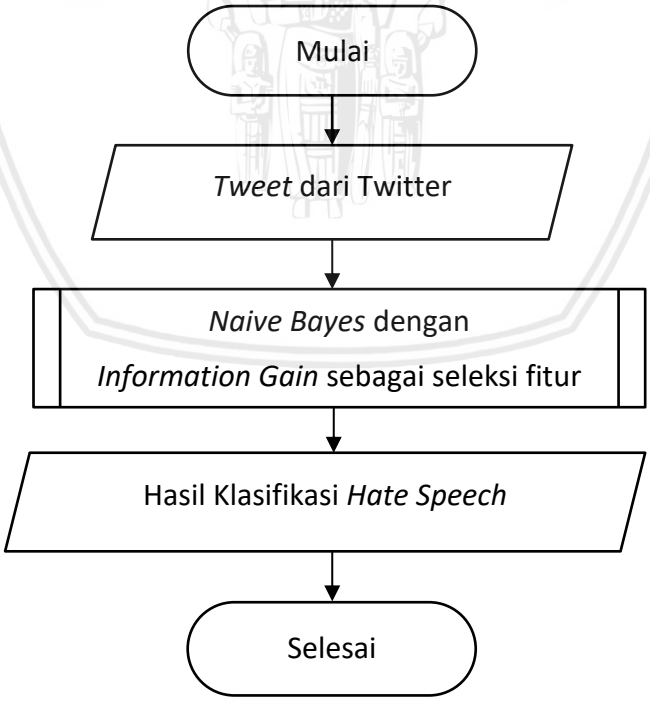
# BAB 4
# PERANCANGAN

## 4.1 Formulasi Permasalahan

Pada penelitian ini, permasalahan yang ingin diselesaikan yaitu klasifikasi *hate speech* berbahasa Indonesia di Twitter menggunakan *Naive Bayes* dan seleksi fitur *Information Gain* dengan normalisasi kata. Data yang digunakan untuk klasifikasi *hate speech* berbahasa Indonesia diperoleh dari penelitian sebelumnya yang di dalamnya terdapat *database tweet* yang mengandung *hate speech* dan *non hate speech* (Alfina, Mulia, Fanany, & Ekanata, 2017). Perbandingan pembagian *tweet* data latih *hate speech* dan *tweet* data uji *hate speech* yang digunakan pada penelitian ini adalah 80% untuk *tweet* data latih *hate speech* dan 20% *tweet* data uji *hate speech*. Hasil klasifikasi kemudian akan dilakukan evaluasi menggunakan perhitungan C*onfusion Matrix* dan *F-Measure* untuk melihat tingkat akurasinya.

## 4.2 Penyelesaian Permasalahan

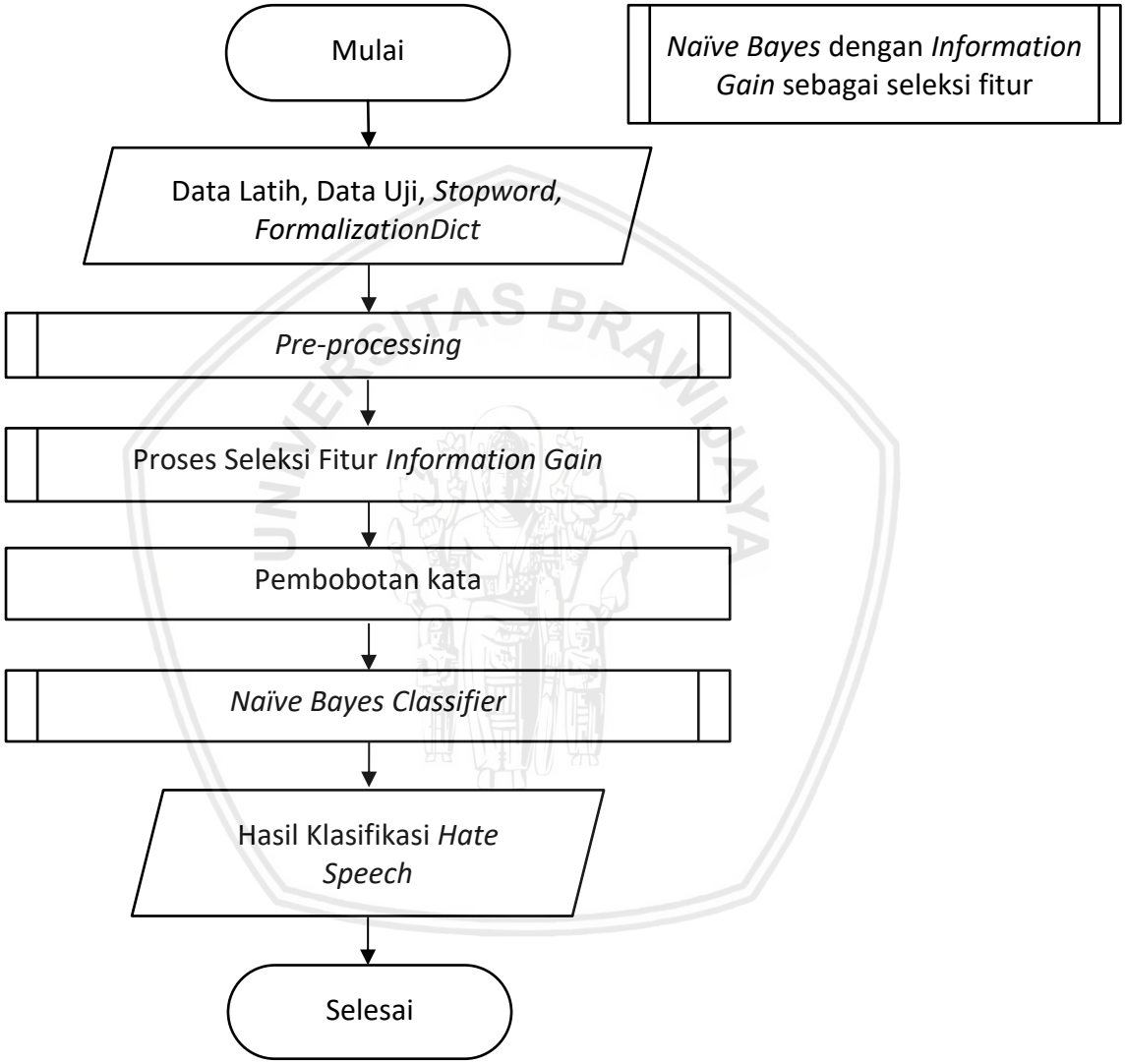
Metode *Naive Bayes* dan *Information Gain* sebagai seleksi fitur digunakan untuk menyelesaikan permasalahan di atas dengan tujuan mendapatkan tingkat akurasi yang lebih baik. Proses penyelesaian permasalahan menggunakan metode *Naive Bayes* dan *Information Gain* sebagai seleksi fitur secara garis besar dapat dilihat pada Gambar 4.1.




**Gambar 4.1 Diagram alir metode *Naïve Bayes* dengan seleksi fitur**

### 4.2.1 *Naive Bayes* dengan *Information Gain* sebagai seleksi fitur

Gambar 4.2 menunjukkan langkah-langkah pemrosesan data yang dilakukan. Proses algoritme *Naive Bayes* dengan *Information Gain* sebagai seleksi fitur dibagi menjadi tiga proses, yakni proses *pre-processing* yang didalamnya terdapat proses normalisasi, proses seleksi fitur *Information Gain*, dan proses klasifikasi menggunakan *Multinomial Naïve Bayes*. Untuk diagram alir dari algoritme dapat dilihat pada Gambar 4.2.

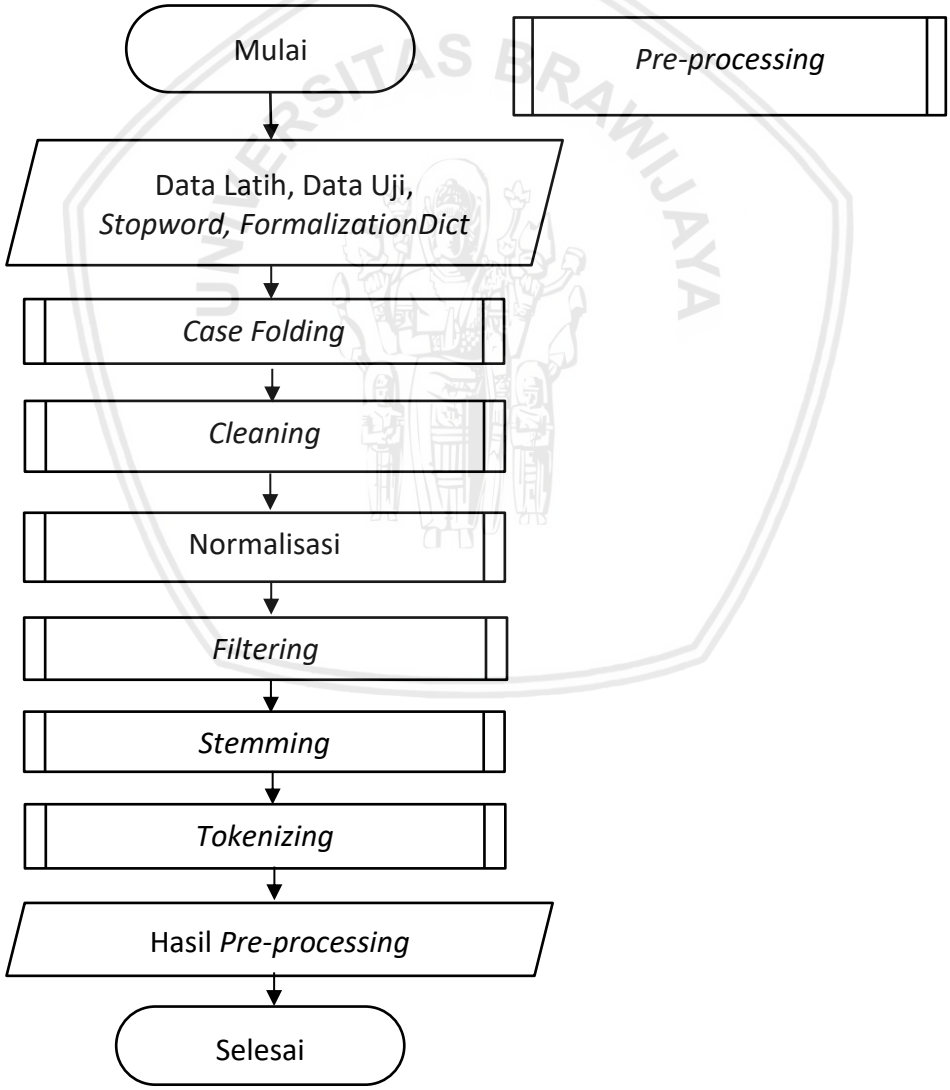



**Gambar 4.2 Diagram Alir Algoritme *Naïve Bayes* dengan *Information Gain* sebagai seleksi fitur**

### 4.2.2 *Pre-processing*

*Pre-processing* dilakukan dengan tujuan untuk mempersiapkan data agar bisa diolah dan dianalisis lebih lanjut pada dokumen *tweet hate speech*. Dalam melakukan *pre-processing* terdapat beberapa tahapan atau proses yang harus dilalui. Pada penelitian ini, langkah pertama dalam *pre-processing* adalah melakukan proses *case folding*, yaitu tahapan atau proses mengubah semua huruf

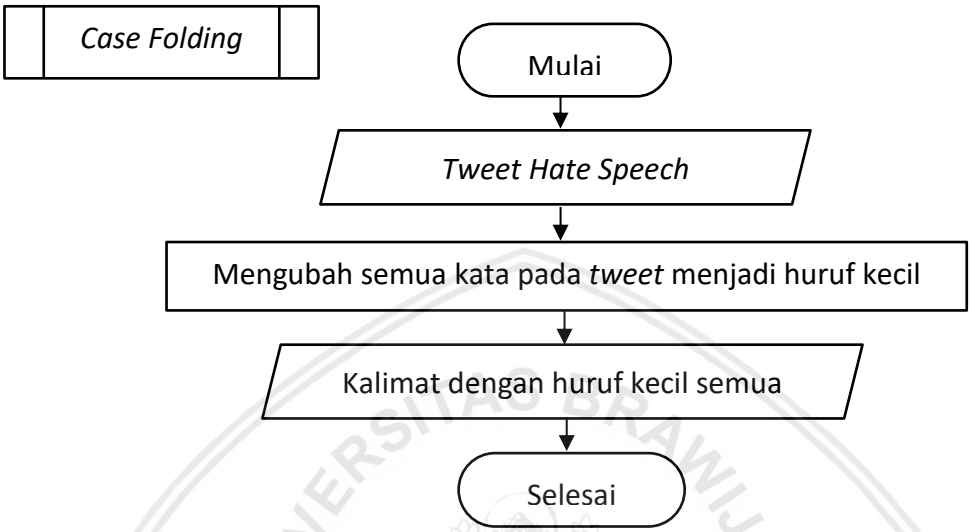
pada dokumen *tweet hate speech* menjadi huruf kecil semua. Tahapan kedua dalam *pre-processing* ini adalah melakukan proses *cleaning,* yaitu proses pembersihan dokumen *tweet hate speech* dari sebuah *username* (@), *hashtag (#)*, *link* (URL) dan *retweet* (RT). Tahap ketiga dalam *pre-processing* ini adalah melakukan proses normalisasi kata atau melakukan pengubahan kata-kata yang tidak baku dalam *tweet hate speech* menjadi kata-kata yang baku berdasarkan *formalizationDict.* Tahapan keempat pada penelitian ini adalah *filtering,* yaitu proses memisahkan kata dari sebuah karakter ataupun huruf yang tidak terlalu berdampak atau berguna dalam suatu kalimat. Tahapan kelima pada penelitian ini adalah *stemming,* yaitu proses mengubah suatu kata menjadi bentuk dasarnya. Selanjutnya tahap terakhir adalah proses *tokenizing*. Pada tahap ini, setiap *tweet hate speech* akan dipecah menjadi beberapa kata yang berdiri sendiri atau dikenal dengan istilah token. Untuk lebih lengkapnya tahap *pre-processing* ini disajikan ke dalam diagram alir pada Gambar 4.3.




**Gambar 4.3 Diagram Alir *Pre-processing***

#### *4.2.2.1 Case Folding*

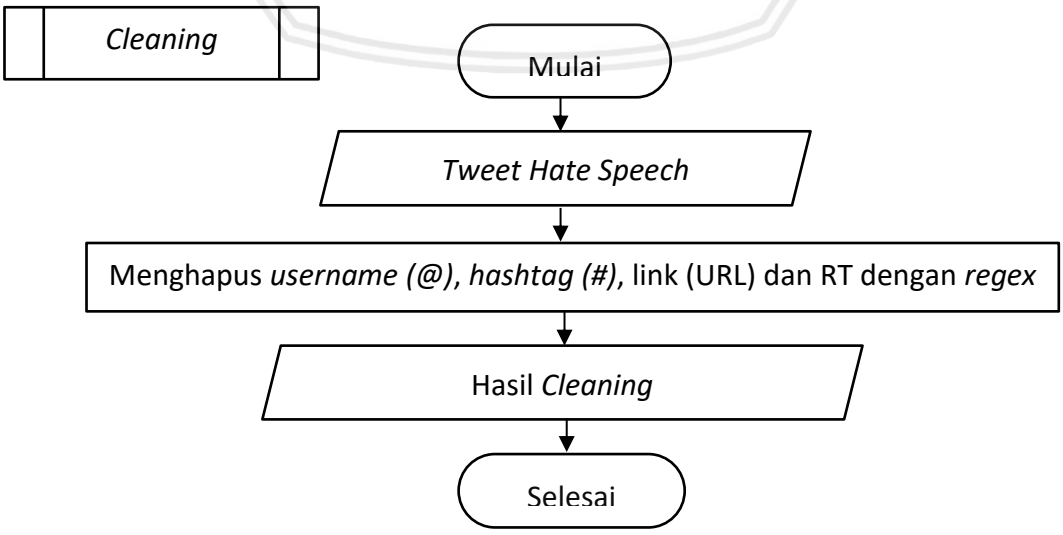
Langkah pertama dalam proses *pre-processing* pada penelitian ini adalah *case folding. Case folding* adalah proses untuk mengubah seluruh *tweet hate speech* menjadi huruf kecil atau yang dikenal dengan nama *lowercase.* Penjelasan alir proses dari tahapan *case folding* ditunjukkan pada Gambar 4.4.




**Gambar 4.4 Diagram Alir *Case Folding***

#### *4.2.2.2 Cleaning*

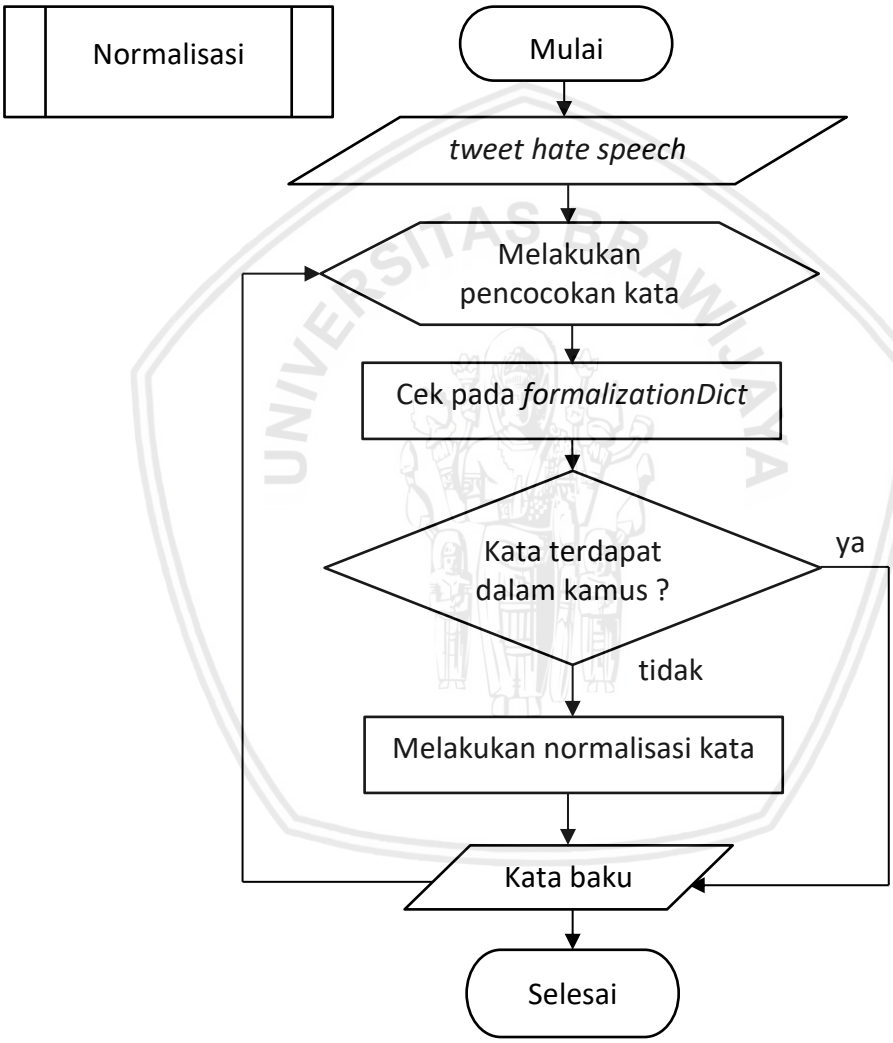
Langkah kedua dalam proses *pre-processing* pada penelitian ini adalah *cleaning. Cleaning* adalah proses yang dilakukan untuk membersihkan *tweet hate speech* dari karakter atau komponen yang kurang penting. Pada *tweet hate speech* terdapat beberapa variabel atau karakter seperti *username* (@), *hashtag (#)*, *link* (URL) dan *retweet* (RT) yang perlu dihilangkan karena tidak memberikan pengaruh dalam melakukan pemrosesan *tweet hate speech*. Penjelasan alir proses dari *cleaning* ditunjukkan pada Gambar 4.5.




**Gambar 4.5 Diagram Alir Proses *Cleaning***

#### *4.2.2.3* Normalisasi

Langkah ketiga dalam proses *pre-processing* pada penelitian ini adalah normalisasi. Normalisasi kata merupakan proses untuk melakukan pengubahan kata yang tidak baku menjadi kata yang baku. Pada normalisasi kata ini dilakukan pencocokan kata terhadap kamus *formalizationDict* yang didapatkan dari Pujangga *Indonesian Natural Language Processing* REST API. Ketika kata-kata yang tidak baku terdapat pada *formalizationDict* maka nanti akan langsung diubah menjadi kata-kata yang baku sesuai dengan ketentuan pada *formalizationDict.* Penjelasan alir proses normalisasi kata ditunjukkan pada Gambar 4.6.

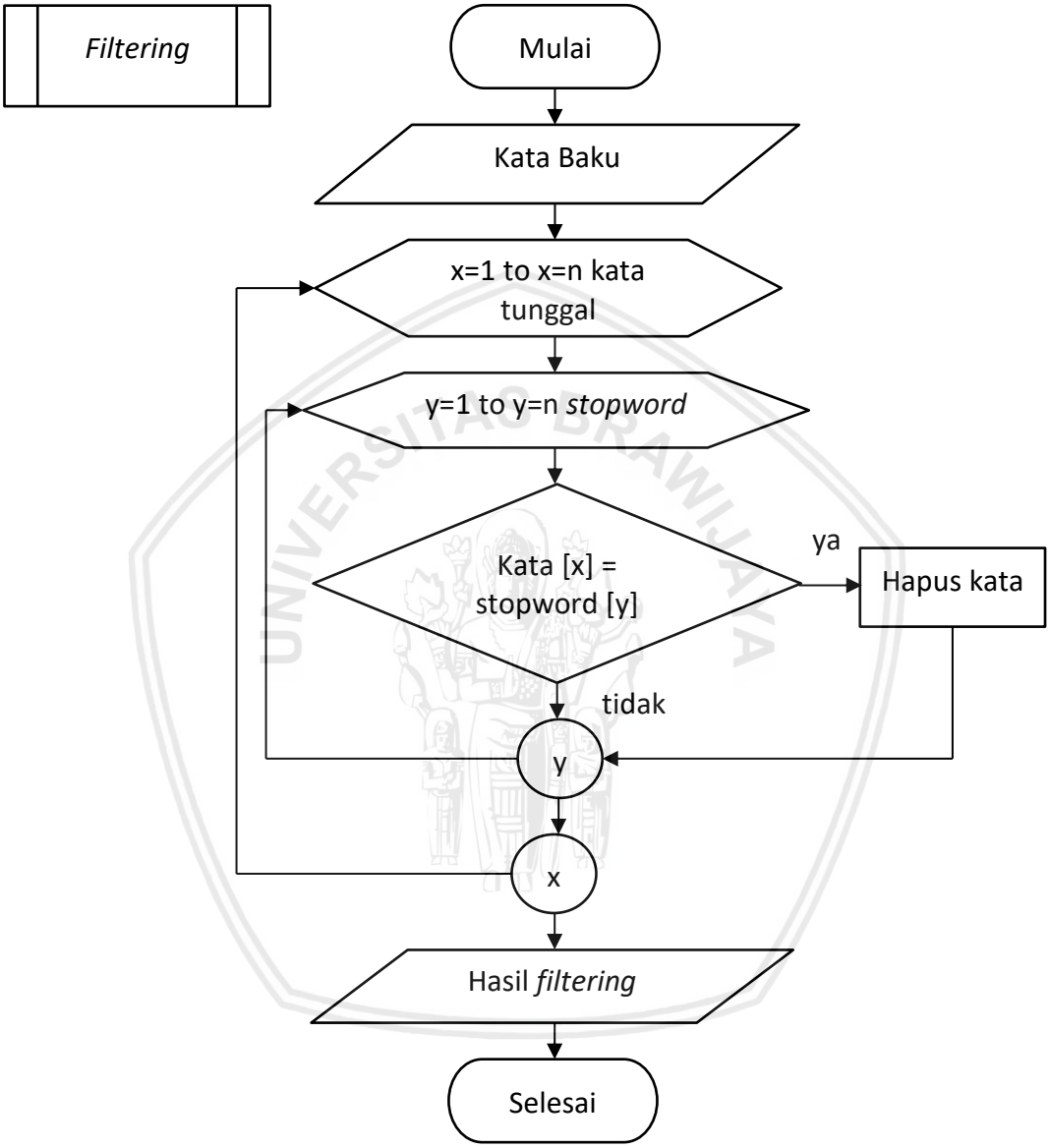



**Gambar 4.6 Diagram Alir Normalisasi**

#### *4.2.2.4 Filtering*

Langkah keempat dalam proses *pre-processing* pada penelitian ini adalah *filtering. Filtering* adalah proses untuk melakukan pemisahan kata dari sebuah huruf ataupun karakter yang tidak memengaruhi suatu proses dalam sistem. *Filtering* pada penelitian ini menggunakan metode *stopword removal*. Pada metode ini, setiap kata atau token akan dibandingkan dengan *stopword list* dan

jika kata tersebut terdapat pada *stopword list* maka kata tersebut akan dibuang. *Stopword* yang digunakan pada penelitian ini berasal dari Pujangga *Indonesian Natural Language Processing* REST API dan merupakan *stopword* dalam bahasa Indonesia. Berikut ini adalah diagram alir proses *filtering* yang bisa dilihat pada Gambar 4.7.

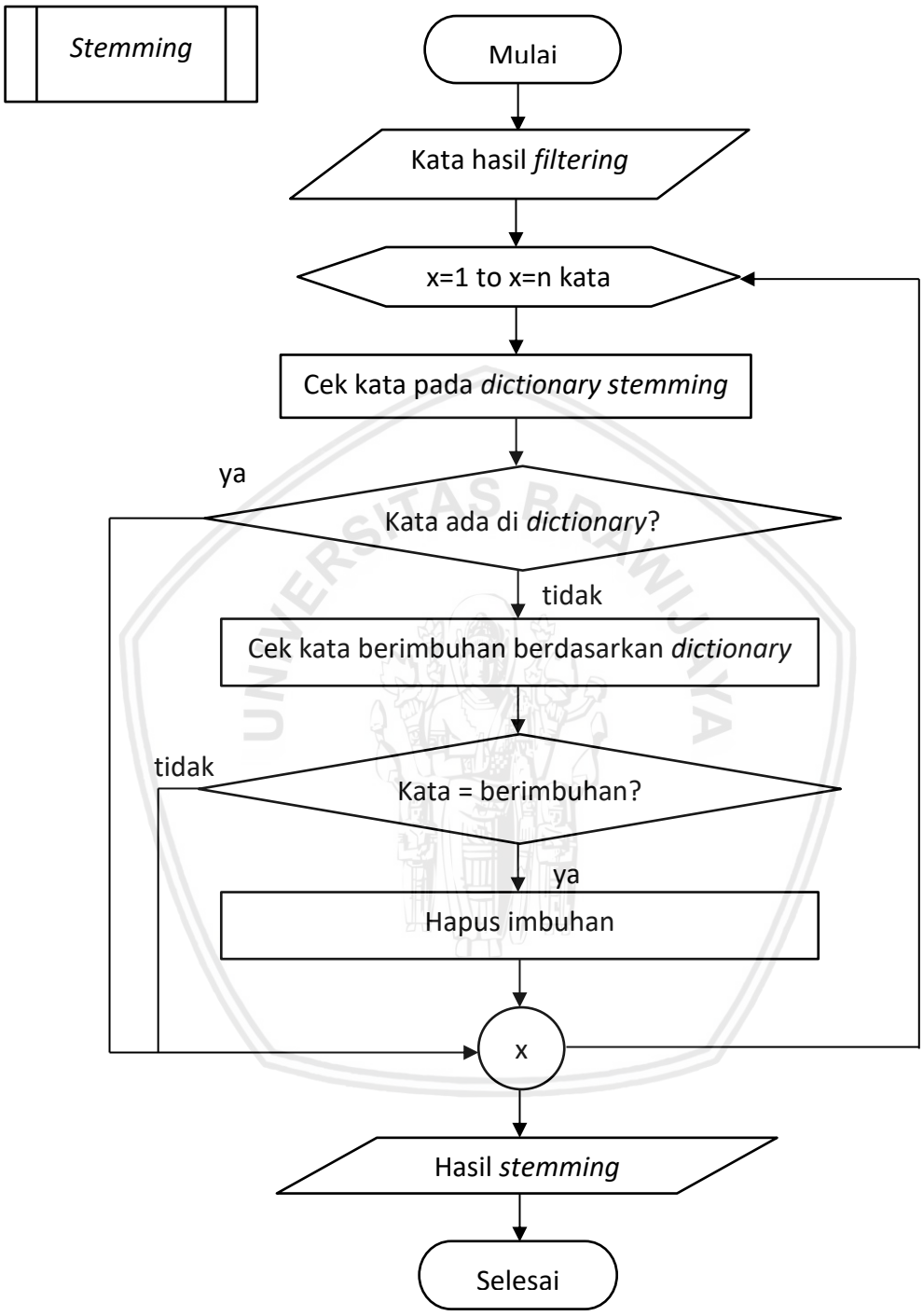



**Gambar 4.7 Diagram Alir *Filtering***

### *4.2.2.5 Stemming*

Langkah kelima dalam proses *pre-processing* pada penelitian ini adalah *stemming*. *Stemming* adalah proses untuk mengubah sebuah kata menjadi kata dasar. Contohnya adalah kata yang memiliki imbuhan yang nantinya akan diubah menjadi sebuah kata dasar, seperti "makanan" menjadi makan, "berlari" menjadi "lari" dan masih banyak lagi. Pada penelitian ini, proses *stemming* menggunakan *library* dari Pujangga *Indonesian Natural Language Processing* REST API *stemming*

dalam bahasa Indonesia. Penjelasan alir proses dari *stemming* ditunjukkan pada Gambar 4.8.

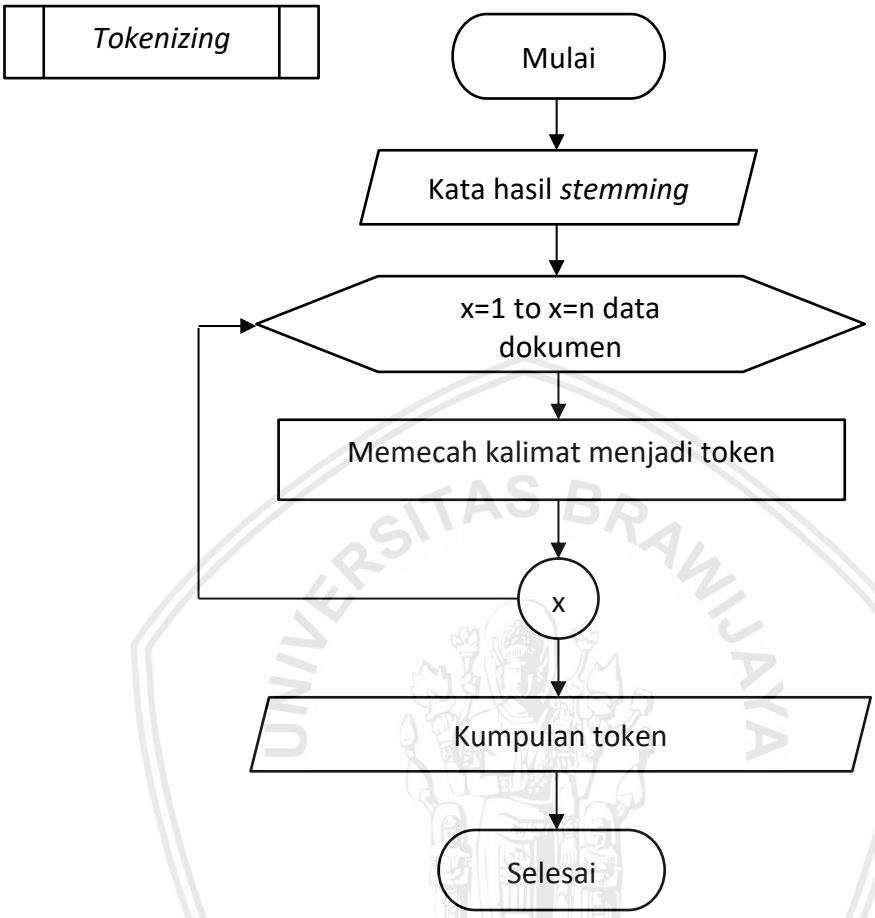



**Gambar 4.8 Diagram Alir *Stemming***

### *4.2.2.6* Proses *Tokenizing*

Langkah keenam dalam proses *pre-processing* pada penelitian ini adalah *tokenizing. Tokenizing* adalah proses untuk memecah *tweet-tweet hate speech* menjadi kata tunggal atau yang dikenal dengan istilah token. Hasil dari *tokenizing* yang telah dipecah menjadi sebuah token atau kata tunggal nantinya akan

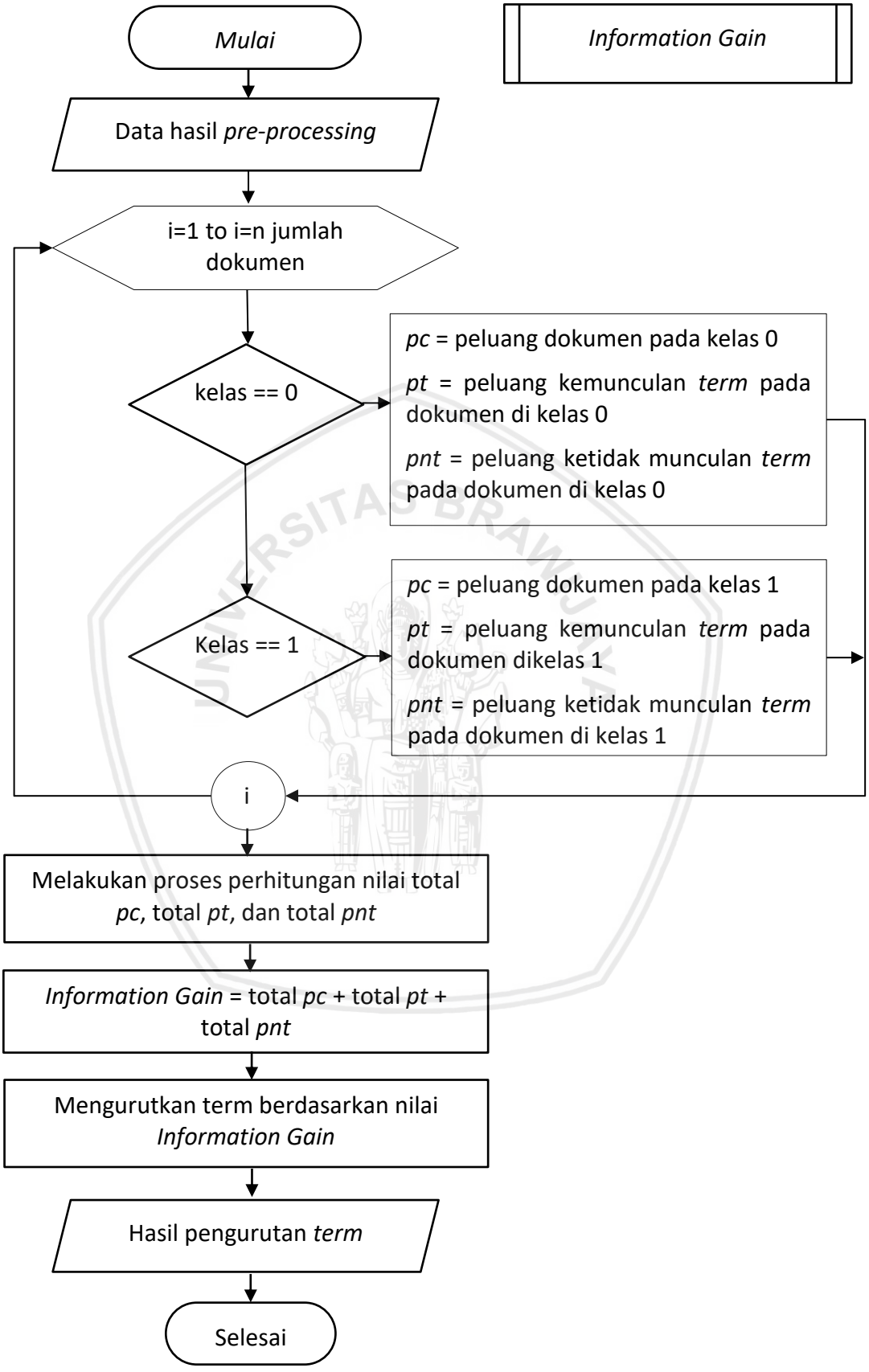
diproses lebih lanjut yaitu melakukan seleksi fitur dengan *Information Gain*. Penjelasan alir proses dari *tokenizing* ditunjukkan pada Gambar 4.9.




**Gambar 4.9 Diagram Alir *Tokenizing***

### 4.2.3 Seleksi Fitur *Information Gain*

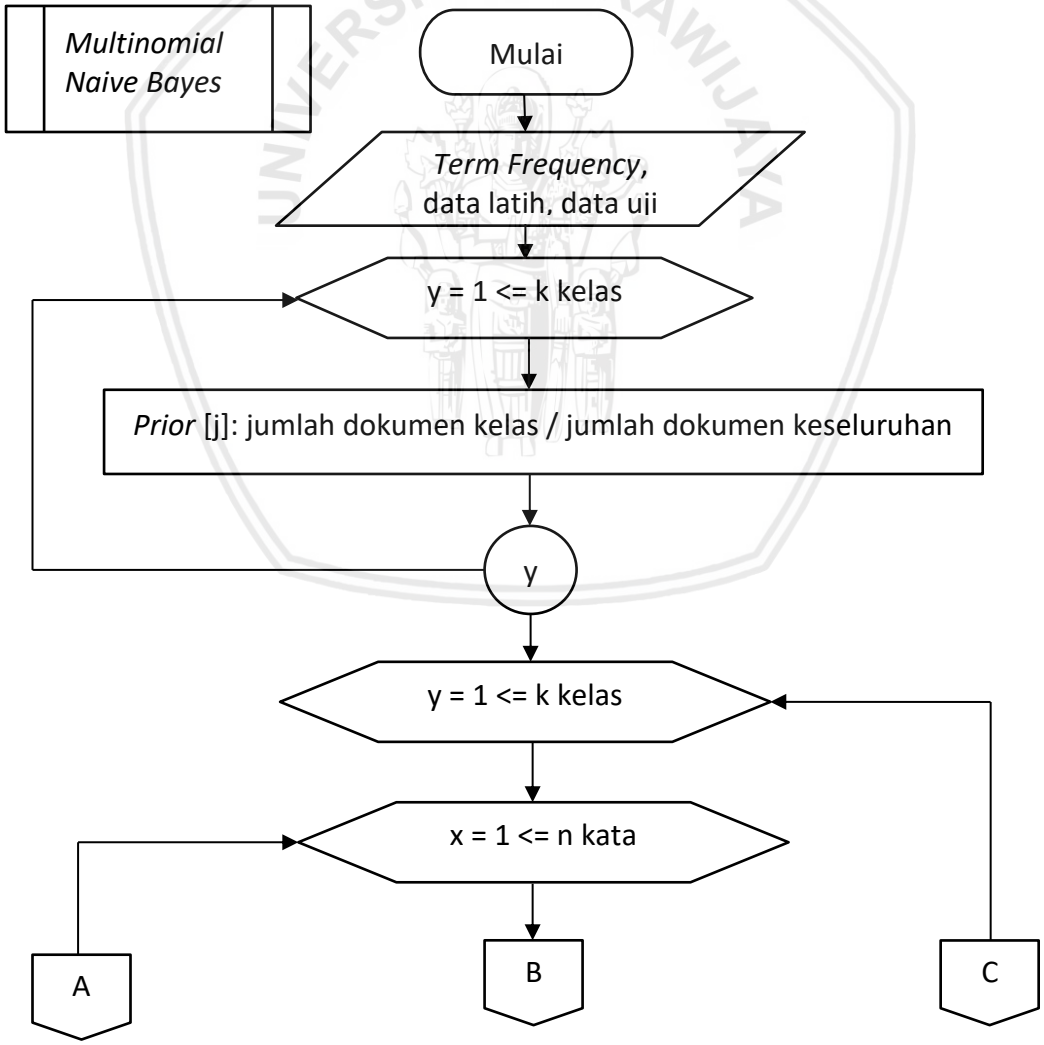
Pada seleksi fitur berfungsi untuk memilih fitur yang akan digunakan pada proses selanjutnya yaitu pembobotan kata. Pada penilitian ini metode yang digunakan adalah *Information Gain*. Cara kerja *Information Gain* adalah menghitung peluang dokumen pada setiap kelas, kemudian menghitung sebuah *term* yang muncul pada dokumen di kelas tertentu, dan yang terakhir adalah menghitung sebuah *term* yang tidak muncul pada dokumen di kelas tertentu. Nantinya ketiga proses tersebut dilakukan proses perhitungan dan akan dijumlahkan untuk mendapatkan nilai *Information Gain* dari setiap *term* nya. Kemudian setiap *term* yang telah dihitung akan di urutkan berdasarkan nilai *Information Gain* tertinggi dan akan dipilih sejumlah *term* dengan batas *threshold* yang ditentukan untuk digunakan dalam proses selanjutnya yaitu proses pembobotan kata. Penjelasan alir proses dari seleksi fitur *Information Gain* ditunjukkan pada Gambar 4.10.

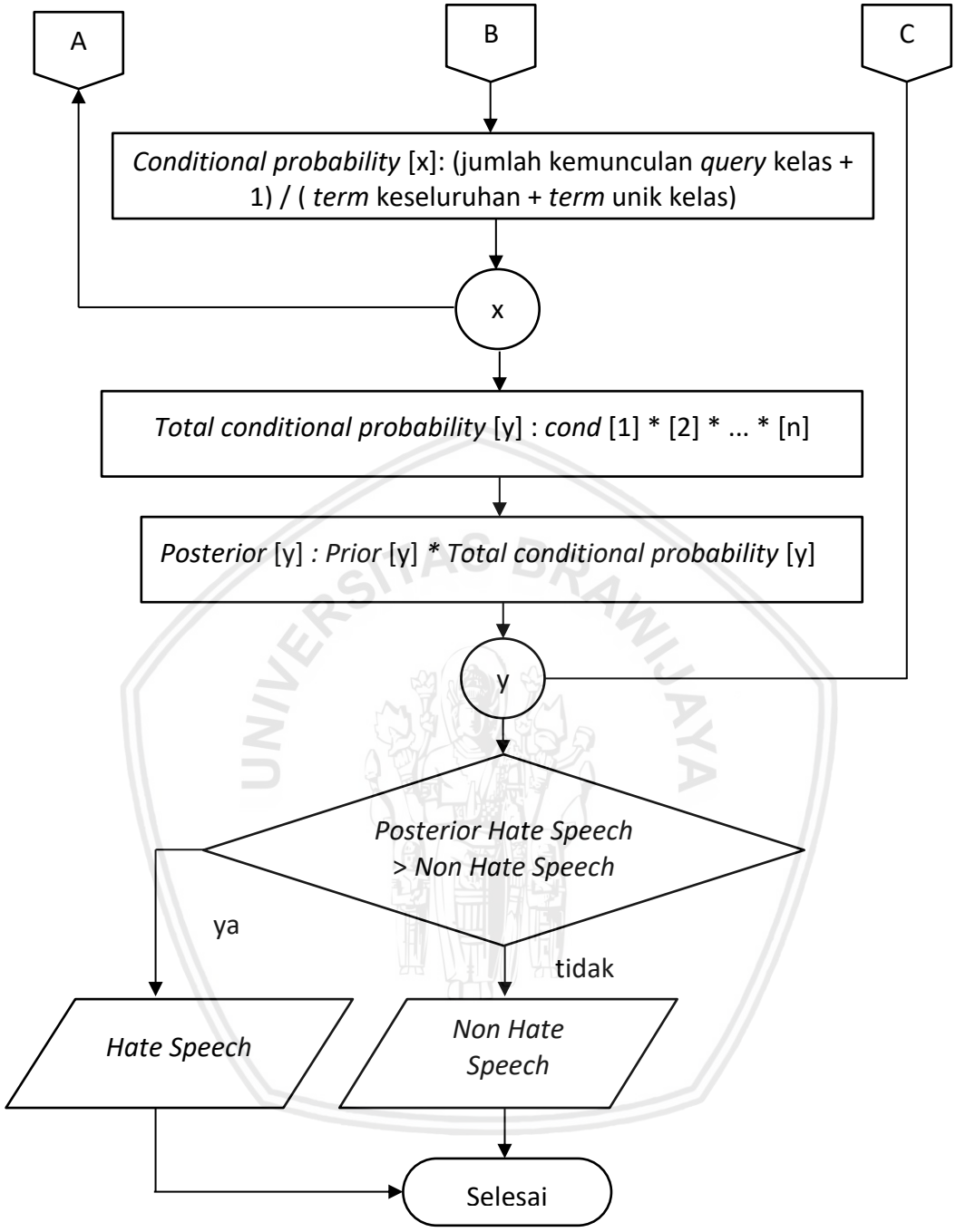



**Gambar 4.10 Diagram Alir Seleksi Fitur *Information Gain***

### 4.2.4 Klasifikasi Menggunakan *Naive Bayes Classifier*

Pada proses klasifikasi menggunakan *Naive Bayes Classifier* dilakukan dengan *Multinomial Naive Bayes* yaitu proses perhitungan yang digunakan untuk melakukan klasifikasi *hate speech* pada pemrosesan teks pada *tweet hate* speech berbahasa Indonesia. Pada *Multinomial Naive Bayes*, terdapat 3 hal penting yang harus dilakukan. Hal pertama yaitu melakukan proses perhitungan nilai *prior*. Nilai *prior* bisa dihitung dari nilai probabilitas antara kelas *hate speech* dan kelas *non hate speech*. Hal kedua yang perlu dilakukan adalah menghitung nilai *conditional probability*. Nilai ini didapatkan dengan menghitung banyaknya kemunculan suatu kata pada kelas *hate speech* dan *non hate speech*. Hal ketiga yang perlu dilakukan adalah mencari nilai *posterior.* Nilai ini didapat dengan mengalikan seluruh hasil perhitungan nilai *conditional probability* dari masing-masing kata pada kelas *hate speech* maupun kelas *non hate speech*. Hasil *posterior* suatu kelas dengan nilai tertinggi akan menentukan suatu *tweet* masuk ke dalam kelas *hate speech* ataupun kelas *non hate speech*. Penjelasan alir proses dari klasifikasi menggunakan *Naive Bayes Classifier* ditunjukkan pada Gambar 4.11.






**Gambar 4.11 Diagram Alir Proses *Multinomial Naive Bayes***

## 4.3 Perhitungan Manual

Perhitungan manual dari penelitian normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain* untuk klasifikasi *hate speech* berbahasa Indonesia akan dijelaskan pada sub bab ini untuk memberikan gambaran umum terkait proses perhitungan dari sistem yang akan dibuat.

Langkah-langkah proses perhitungan manual pada penelitian normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain* untuk klasifikasi *hate speech* berbahasa Indonesia adalah:

**Langkah 1.** Menentukan *data set* yang dipakai

Perhitungan manual yang dilakukan hanya menggunakan 5 *tweet hate speech* berbahasa Indonesia yang terdiri dari 4 data latih dan 1 data uji dengan 2 kelas data yaitu *hate speech* (HS) dan *non hate speech* (NONHS).

*Data set tweet hate speech* berbahasa Indonesia yang digunakan untuk perhitungan manual dapat dilihat pada Tabel 4.1.

**Tabel 4.1 *Data set Tweet Hate Speech***

| No | Label | Tweet |
|---|---|---|
| 1 | HS | SI KUTIL BABI TELAH MELECEHKAN PRIBUMI DAN UMAT ISLAM #IklanAhokJahat |
| 2 | HS | Kapolda Babi! Biadap dan Bodoh! Gak punya otak kali. |
| 3 | NONHS | Drama bgt itu :(( #DebatFinalPilkadaJKT |
| 4 | NONHS | Tolong adu program |
| 5 | ? | @TPK_RI ITULAH si ahok kutil babi |

**Keterangan:**

No 1 - 4: Data Latih No 5: Data Uji

**Tahap *Pre-processing***

**Langkah 2.** Melakukan proses *Case Folding*

*Case folding* adalah proses untuk mengubah seluruh *tweet hate speech* menjadi huruf kecil atau yang dikenal dengan nama *lowercase*. Berikut adalah contoh proses *case folding* pada data *tweet* nomor 1 yang memiliki label *hate speech* yang terdapat pada *data set*.

Sebelum dilakukan *case folding*:

*"SI KUTIL BABI TELAH MELECEHKAN PRIBUMI DAN UMAT ISLAM #IklanAhokJahat"*

Sesudah dilakukan *case folding*:

*"si kutil babi telah melecehkan pribumi dan umat islam #iklanahokjahat"*

Pada Tabel 4.2 terdapat hasil seluruh *data set tweet hate speech* yang telah dilakukan proses *case folding*.

**Tabel 4.2 *Data set Tweet Hate Speech* setelah *Case Folding***

| No | Label | Tweet |
|---|---|---|
| 1 | HS | si kutil babi telah melecehkan pribumi dan umat islam #iklanahokjahat |
| 2 | HS | kapolda babi! biadap dan bodoh! gak punya otak kali. |
| 3 | NONHS | drama bgt itu :(( #debatfinalpilkadajkt |
| 4 | NONHS | tolong adu program |
| 5 | ? | @tpk_ri itulah si ahok kutil babi |

**Langkah 3.** Melakukan proses *cleaning. Cleaning* adalah proses yang dilakukan untuk membersihkan *tweet hate speech* dari karakter atau komponen yang kurang penting. Pada *tweet hate speech* terdapat beberapa variabel atau karakter seperti *username* (@), *hashtag (#), link* (URL) dan *retweet* (RT) yang perlu dihilangkan karena tidak memberikan pengaruh dalam melakukan pemrosesan *tweet hate speech*

Pada Tabel 4.3 terdapat hasil seluruh *data set tweet hate speech* yang telah dilakukan proses *cleaning*.

**Tabel 4.3 *Data set Tweet Hate Speech* setelah Cleaning**

| No | Label | Tweet |
|---|---|---|
| 1 | HS | si kutil babi telah melecehkan pribumi dan umat islam |
| 2 | HS | kapolda babi! biadap dan bodoh! gak punya otak kali. |
| 3 | NONHS | drama bgt itu :(( |
| 4 | NONHS | tolong adu program |
| 5 | ? | itulah si ahok kutil babi |

**Langkah 4.** Melakukan proses normalisasi kata

Tahap normalisasi kata ini dilakukan untuk melakukan proses pengubahan kata-kata yang tidak baku menjadi baku. Berikut adalah contoh proses normalisasi pada data *tweet* nomor 3 yang memiliki label *non hate speech* yang terdapat pada *data set*.

Sebelum dilakukan normalisasi kata:

*“drama bgt itu :((”*

Sesudah dilakukan normalisasi kata:

*“drama banget itu :((”*

Pada Tabel 4.4 terdapat hasil seluruh data_set *tweet hate speech* yang telah dilakukan proses normalisasi kata.

**Tabel 4.4 *Data set Tweet Hate Speech* setelah dilakukan Normalisasi**

| No | Label | Tweet |
|---|---|---|
| 1 | HS | si kutil babi telah melecehkan pribumi dan umat islam |
| 2 | HS | kapolda babi! biadap dan bodoh! tidak punya otak kali. |
| 3 | NONHS | drama banget itu :(( |
| 4 | NONHS | tolong adu program |
| 5 | ? | itulah si ahok kutil babi |

**Langkah 5.** Melakukan proses *Filtering*

*Filtering* adalah proses untuk melakukan pemisahan kata dari sebuah huruf ataupun karakter yang tidak memengaruhi suatu proses dalam sistem. *Filtering* pada penelitian ini menggunakan metode *stopword removal*, di mana setiap kata atau token akan dibandingkan dengan *stopword list* dan jika kata tersebut terdapat pada *stopword list* maka kata tersebut akan dibuang. *Stopword* yang digunakan pada penelitian ini berasal dari Pujangga *Indonesian Natural Language Processing* REST API dan merupakan *stopword* dalam Bahasa Indonesia yang disajikan pada Tabel 4.5 dan untuk versi lengkapnya terdapat pada Lampiran 1.

**Tabel 4.5 *Database stopword* Bahasa Indonesia**

| No | Stopword |
|---|---|
| 1 | ada |
| 2 | adalah |
| 3 | adanya |
| 4 | adapun |
| ... | ... |
| 755 | yaitu |
| 756 | yakin |
| 757 | yakni |
| 758 | yang |

Berikut adalah contoh proses *filtering* pada data *tweet* nomor 1 yang memiliki label *hate speech* yang terdapat pada *data set* yang disajikan pada Tabel 4.6.

**Tabel 4.6 Contoh Proses *Filtering Tweet Hate Speech***

| Hasil Token | Hasil Filter |
|---|---|
| si | si |
| kutil | kutil |
| babi | babi |
| telah | - |
| melecehkan | melecehkan |
| pribumi | pribumi |
| dan | - |
| umat | umat |
| islam | islam |

Pada Tabel 4.7 dan Tabel 4.8 terdapat hasil seluruh *data set tweet hate speech* yang telah dilakukan proses *filtering*.

**Tabel 4.7 Hasil *Filtering* Data Latih *Tweet Hate Speech***

| No | Token |
|---|---|
| 1 | si |
| 2 | kutil |
| 3 | babi |
| 4 | melecehkan |
| 5 | pribumi |
| 6 | umat |
| 7 | islam |
| 8 | kapolda |
| 9 | babi |
| 10 | biadap |
| 11 | bodoh |
| 12 | otak |
| 13 | kali |
| 14 | tolong |
| 15 | adu |
| 16 | program |
| 17 | drama |
| 18 | banget |

**Tabel 4.8 Hasil *Filtering* Data Uji *Tweet Hate Speech***

| No | Token |
|---|---|
| 1 | si |
| 2 | ahok |
| 3 | kutil |
| 4 | babi |

**Langkah 6.** Melakukan proses *Stemming*

*Stemming* adalah proses untuk mengubah sebuah kata menjadi kata dasar. Contohnya adalah kata yang memiliki imbuhan yang nantinya akan diubah menjadi sebuah kata dasar, seperti "makanan" menjadi makan, "berlari" menjadi "lari" dan masih banyak lagi. Pada penelitian ini, proses *stemming* menggunakan *library* dari Pujangga *Indonesian Natural Language Processing* REST API *stemming* dalam Bahasa Indonesia. Berikut adalah contoh proses *stemming* pada *tweet* nomor 1 yang memiliki label *hate speech* yang terdapat pada *data set* yang disajikan pada Tabel 4.6.

**Tabel 4.9 Contoh Proses *Stemming Tweet Hate Speech***

| Hasil Filtering | Hasil Stemming |
|---|---|
| melecehkan | leceh |

Pada Tabel 4.10 dan Tabel 4.11 terdapat hasil seluruh data_set *tweet hate speech* yang telah dilakukan proses *stemming*.

**Tabel 4.10 Hasil *Stemming* Data Latih *Tweet Hate Speech***

| No | Token | No | Token | No | Token |
|---|---|---|---|---|---|
| 1 | si | 7 | islam | 13 | tolong |
| 2 | kutil | 8 | kapolda | 14 | adu |
| 3 | babi | 9 | biadap | 15 | program |
| 4 | leceh | 10 | bodoh | 16 | drama |
| 5 | pribumi | 11 | otak | 17 | banget |
| 6 | umat | 12 | kali | | |

**Tabel 4.11 Hasil *Stemming* Data Uji *Tweet Hate Speech***

| No | Token |
|---|---|
| 1 | si |
| 2 | ahok |
| 3 | kutil |
| 4 | babi |

**Langkah 7.** Melakukan proses *Tokenizing*

Proses *tokenizing* dilakukan setelah proses *filtering* dan dilakukan di tahap akhir *pre-processing*. *Tokenizing* adalah proses untuk memecah *tweet-tweet hate speech* menjadi kata tunggal atau yang dikenal dengan istilah token. Hasil dari *tokenizing* yang telah dipecah menjadi sebuah token atau kata tunggal nantinya akan diproses lebih lanjut yaitu melakukan seleksi fitur dengan *Information Gain*. Berikut adalah contoh proses *tokenizing* pada *tweet* nomor 1 yang memiliki label *hate speech* yang telah dilakukan sampai pada tahap *filtering*.

Sebelum dilakukan *tokenizing*:

*"si kutil babi leceh pribumi umat islam"*

Sesudah dilakukan *tokenizing*:

**Tabel 4.12 Contoh Hasil *Tokenizing Tweet Hate Speech***

| | |
|---|---|
| Token 1 | si |
| Token 2 | kutil |
| Token 3 | babi |
| Token 4 | leceh |
| Token 5 | pribumi |
| Token 6 | umat |
| Token 7 | islam |

Pada Tabel 4.13 dan Tabel 4.14 terdapat hasil seluruh *data set tweet hate speech* yang telah dilakukan proses *tokenizing*.

**Tabel 4.13 Hasil *Tokenizing* Data Latih *Tweet Hate Speech***

| | | |
|---|---|---|
| si | islam | tolong |
| kutil | kapolda | adu |
| babi | biadap | program |
| leceh | bodoh | drama |
| pribumi | otak | banget |
| umat | kali | |

**Tabel 4.14 Hasil *Tokenizing* Data Uji *Tweet Hate Speech***

| |
|---|
| si |
| ahok |
| kutil |
| babi |

**Tahap Proses Seleksi Fitur *Information Gain***

**Langkah 8.** *Information Gain* bekerja dengan menghitung peluang dokumen pada setiap kelas, kemudian menghitung sebuah *term* atau kata yang muncul pada dokumen di kelas tertentu, dan yang terakhir adalah menghitung sebuah *term* atau kata yang tidak muncul pada dokumen di kelas tertentu. Kemudian nantinya ketiga proses tersebut dilakukan proses perhitungan dan akan dijumlahkan untuk mendapatkan nilai *Information Gain* dari setiap *term* atau kata. Setiap *term* atau kata yang telah dihitung akan diurutkan berdasarkan nilai *Information Gain* tertinggi dan akan dipilih sejumlah *term* dengan *threshold* yang ditentukan untuk digunakan dalam proses selanjutnya yaitu proses pembobotan kata. Contoh *term* yang akan digunakan pada perhitungan *Information Gain* adalah "babi" yang dapat dilihat pada Tabel 4.15.

**Tabel 4.15 Frekuensi kemunculan kata**

| Kata | Frekuensi Term | | | |
|---|---|---|---|---|
| | *Hate Speech* | | *Non Hate Speech* | |
| | Dokumen 1 | Dokumen 2 | Dokumen 3 | Dokumen 4 |
| babi | 1 | 1 | 0 | 0 |

Data diatas akan digunakan untuk menghitung nilai *Information Gain* dari kata babi menggunakan rumus pada Persamaan 2.6.

$$IG(t) = -(\sum_{i=1}^{|c|} P(ci)\log P(ci) + P(t)\sum_{i=1}^{|c|} P(ci|t)\log P(ci|t) + P(\bar{t})\sum_{i=1}^{|c|} P(ci|\bar{t})\log P(ci|\bar{t})$$

$$= -\left(\frac{2}{4} log \frac{2}{4} + \frac{2}{4} \log \frac{2}{4}\right) + \left(\left(\frac{2}{4}\right) \frac{2}{2} log \frac{2}{2} + \frac{0}{2} log \frac{0}{2}\right) + \left(\left(\frac{2}{4}\right) \frac{0}{2} log \frac{0}{2} + \frac{2}{2} log \frac{2}{2}\right)$$

$$= -(-0{,}30103) + (0) + (0)$$

$$= 0{,}30103$$

Pada Tabel 4.16 terdapat seluruh hasil perhitungan nilai *Information Gain* pada setiap kata atau *term* pada data latih *tweet hate speech*.

**Tabel 4.16 Hasil Perhitungan *Information Gain***

| No | *Term* | *Hate Speech* | | *Non Hate Speech* | | *Information Gain* |
|---|---|---|---|---|---|---|
| | | Dok 1 | Dok 2 | Dok 3 | Dok 4 | |
| 1 | si | 1 | 0 | 0 | 0 | 0,093704052 |
| 2 | kutil | 1 | 0 | 0 | 0 | 0,093704052 |
| 3 | babi | 1 | 1 | 0 | 0 | 0,301029996 |
| 4 | leceh | 1 | 0 | 0 | 0 | 0,093704052 |
| 5 | pribumi | 1 | 0 | 0 | 0 | 0,093704052 |
| 6 | umat | 1 | 0 | 0 | 0 | 0,093704052 |
| 7 | islam | 1 | 0 | 0 | 0 | 0,093704052 |
| 8 | kapolda | 0 | 1 | 0 | 0 | 0,093704052 |
| 9 | biadap | 0 | 1 | 0 | 0 | 0,093704052 |
| 10 | bodoh | 0 | 1 | 0 | 0 | 0,093704052 |
| 11 | otak | 0 | 1 | 0 | 0 | 0,093704052 |
| 12 | kali | 0 | 1 | 0 | 0 | 0,093704052 |
| 13 | tolong | 0 | 0 | 1 | 0 | 0,093704052 |
| 14 | adu | 0 | 0 | 1 | 0 | 0,093704052 |
| 15 | program | 0 | 0 | 1 | 0 | 0,093704052 |
| 16 | drama | 0 | 0 | 0 | 1 | 0,093704052 |
| 17 | banget | 0 | 0 | 0 | 1 | 0,093704052 |

Setelah dilakukan perhitungan *Information Gain*, *term* akan diurutkan berdasarkan nilai *Information Gain* tertinggi hingga terendah. Kemudian ditentukan *threshold* yang akan digunakan untuk memilih *term* yang akan disimpan. Misal akan dipilih sebanyak 50% dari 17 *term* yang akan disimpan, sehingga yang diambil hanya 9 *term* dengan hasil *Information Gain* tertinggi yang dapat dilihat pada Tabel 4.17.

**Tabel 4.17 Hasil Pengurutan Term atau Kata dari *Information Gain***

| No | Term / Kata | Nilai *Information Gain* |
|---|---|---|
| 1 | babi | 0,301029996 |
| 2 | si | 0,093704052 |
| 3 | kutil | 0,093704052 |
| 4 | leceh | 0,093704052 |
| 5 | pribumi | 0,093704052 |
| 6 | umat | 0,093704052 |
| 7 | islam | 0,093704052 |
| 8 | kapolda | 0,093704052 |
| 9 | biadap | 0,093704052 |

**Langkah 9.** Melakukan proses menyimpan *term* dan membuang *term* sesuai dengan hasil seleksi fitur *Information Gain*. Jika *term* pada data uji tidak ditemukan pada hasil pengurutan *term* pada seleksi *Information Gain*, maka *term* tersebut akan dibuang dan sebaliknya jika *term* tersebut terdapat pada hasil pengurutan *term* pada seleksi *Information Gain* maka *term* tersebut akan disimpan. Proses menyimpan dan membuang *term* pada data uji *tweet hate speech* bisa dilihat pada Tabel 4.18.

**Tabel 4.18 Proses Menyimpan dan Membuang Term Berdasarkan *Information Gain***

| Term Data Uji | Term Hasil Seleksi Fitur | Keterangan |
|---|---|---|
| si | ada | simpan term |
| ahok | tidak ada | buang term |
| kutil | ada | simpan term |
| babi | ada | simpan term |

**Langkah 10.** Melakukan proses pembobotan kata

Pembobotan kata adalah suatu proses yang harus dilakukan terlebih dahulu sebelum melakukan proses klasifikasi *tweet hate speech* menggunakan *Multinomial Naive Bayes*. Proses pembobotan kata adalah proses perhitungan *term frequency* atau kemunculan kata pada data uji *tweet hate speech* yang masing-masing termnya sudah berhasil disimpan atau dibuang dengan menggunakan *Information Gain sebagai* seleksi fitur yang terdapat pada Tabel 4.19.

**Tabel 4.19 Frekuensi kemunculan kata**

| Kata | Frekuensi Term | | | | | |
|---|---|---|---|---|---|---|
| | **Kueri** | **Dok 1** | **Dok 2** | **Dok 3** | **Dok 4** | **Jumlah** |
| si | 1 | 1 | 0 | 0 | 0 | 1 |
| kutil | 1 | 1 | 0 | 0 | 0 | 1 |
| babi | 1 | 1 | 1 | 0 | 0 | 2 |

**Langkah 11.** Melakukan proses klasifikasi Multinomial Naïve Bayes

Setelah melakukan proses pembobotan kata, maka proses klasifikasi menggunakan Multinomial Naive Bayes bisa mulai dilakukan. Tahap pertama yang dilakukan adalah menghitung nilai *prior* dengan rumus Persamaan 2.3. Tabel perhitungan nilai *prior* terdapat pada Tabel 4.20.

**Tabel 4.20 Perhitungan Nilai *Prior***

| P(***Hate Speech***) | P(***Non Hate Speech***) |
|---|---|
| $\frac{2}{4} = 0{,}5$ | $\frac{2}{4} = 0{,}5$ |

Setelah didapatkan hasil dari prior *hate speech* dan *non hate speech* maka tahap kedua yang dilakukan adalah menghitung nilai *conditional probability* kelas *hate speech* dan kelas *non hate speech* dengan rumus Persamaan 2.4 yang ditunjukkan pada Tabel 4.21.

**Tabel 4.21 Perhitungan Nilai *Conditional Probability***

| **Kata** | ***Conditional Probability* (*Hate Speech*)** |
|---|---|
| si | $\mathrm{P}(Hate\ speech\|\mathrm{si}) = \frac{(1+1)}{(12+17)} = 0{,}06896551724$ |
| kutil | $\mathrm{P}(Hate\ Speech\|\mathrm{kutil}) = \frac{(1+1)}{(12+17)} = 0{,}06896551724$ |
| babi | $\mathrm{P}(Hate\ Speech\|\mathrm{babi}) = \frac{(2+1)}{(12+17)} = 0{,}1034482759$ |

| **Kata** | ***Conditional Probability* (*Non Hate Speech*)** |
|---|---|
| si | $\mathrm{P}(Non\ Hate\ speech\|\mathrm{si}) = \frac{(0+1)}{(5+17)} = 0{,}04545454545$ |
| kutil | $\mathrm{P}(Non\ Hate\ Speech\|\mathrm{kutil}) = \frac{(0+1)}{(5+17)} = 0{,}04545454545$ |
| babi | $\mathrm{P}(Non\ Hate\ Speech\|\mathrm{babi}) = \frac{(0+1)}{(5+17)} = 0{,}04545454545$ |

Setelah mendapatkan nilai *conditional probability* dari kelas *hate speech* dan nilai *conditional probability* dari kelas *non hate speech.* Tahapan selanjutnya adalah menghitung nilai *total conditional probability* dari kelas *hate speech* dan kelas *non hate speech* yang ditunjukkan pada Tabel 4.22.

**Tabel 4.22 Perhitungan *Total conditional probability***

| ***Total conditional probability (Hate Speech)*** |
|---|
| 0,06896551724 * 0,06896551724 * 0,1034482759<br>= $4{,}920250933 \times 10^{-4}$ |
| **Total conditional probability (Non Hate Speech)** |
| 0,04545454545 * 0,04545454545 * 0,04545454545<br>= $9{,}391435011 \times 10^{-5}$ |

Langkah terakhir yang dilakukan setelah mendapatkan *total conditional probability* dari kelas *hate speech* dan dari kelas *non hate speech* adalah melakukan perhitungan nilai *posterior*. Nilai *posterior* kelas *hate speech* didapatkan dengan mengalikan nilai *prior* dari kelas *hate speech * total conditional probability* dari kelas *hate speech*. Nilai *posterior* kelas *non hate speech* didapatkan dengan mengalikan nilai *prior* dari kelas *non hate speech * total conditional probability* dari kelas non *hate speech.* Nilai *posterior* pada kelas *hate speech* dan kelas *non hate speech* akan dibandingkan untuk mengetahui nilai *posterior* tertinggi. Nilai *posterior* yang memiliki nilai lebih tinggi akan membuat sebuah *tweet hate speech* terklasifikasi ke dalam kelas *hate speech* atau kelas *non hate speech*. Perhitungan *posterior* kelas *hate speech* dan kelas *non hate speech* dihitung menggunakan Persamaan 2.4 yang ditunjukkan pada Tabel 4.23.

**Tabel 4.23 Perhitungan *Posterior* kelas *Hate Speech* dan *Non Hate Speech***

| **Posterior (*Hate Speech*)** | **Posterior (*Non Hate Speech*)** |
|---|---|
| 0,5 * $4{,}920250933 \times 10^{-4}$<br>= $2{,}460125467 \times 10^{-4}$ | 0,5 * $9{,}391435011 \times 10^{-5}$<br>= $4{,}695717506 \times 10^{-5}$ |

Berdasarkan perhitungan nilai *posterior* dari kelas *hate speech* dan perhitungan nilai *posterior* dari kelas *non hate speech* Pada Tabel 4.23, maka nilai *posterior* pada kelas *hate speech* memiliki nilai yang lebih tinggi jika dibandingkan dengan nilai *posterior* pada kelas *non hate speech*. *Tweet* uji pada nomor 5 akan diklasifikasikan oleh sistem ke dalam kelas *hate speech* yaitu kelas pada label HS.

| No | Label | Tweet |
|---|---|---|
| 5 | HS | @tpk_ri itulah si ahok kutil babi |

## 4.4 Perancangan Pengujian

Rancangan pengujian pada penelitian normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain* dibuat untuk mengetahui apakah sistem mampu melakukan pengklasifikasian *hate speech* berbahasa Indonesia dengan tepat. Selain itu, rancangan pengujian juga dibuat untuk mengetahui hasil akurasi, *precison*, *recall*, dan *f-measure* yang dihasilkan pada normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain*. Dalam perancangan pengujian pada penelitian normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain* terdapat dua skenario pengujian, yaitu skenario ujian yang pertama adalah pengujian untuk mengetahui pengaruh ada dan tidak adanya proses normalisasi kata pada tahap *pre-processing* dan skenario ujian yang kedua adalah untuk mengetahui pengaruh penggunaan seleksi fitur *Information Gain* dengan beberapa variasi *threhold* yang berbeda.

### 4.4.1 Skenario pengujian pengaruh normalisasi kata

Skenario pengujian yang pertama ini dilakukan untuk mengetahui pengaruh ada dan tidak adanya proses normalisasi kata pada tahap *pre-processing.* Pengujian dilakukan pada tahapan *pre-processing* yaitu dengan melakukan *pre-processing* tanpa normalisasi kata dan melakukan *pre-processing* dengan normalisasi kata. Data yang digunakan berasal dari penelitian sebelumnya yang berisi *database tweet hate speech* dan *non hate speech* (Alfina, Mulia, Fanany, & Ekanata, 2017). Total data *tweet hate speech* yang akan digunakan adalah sebanyak 250 data, dengan total pembagian data *tweet hate speech* yaitu sebesar 80% untuk data latih dan 20% untuk data uji. Data latih yang akan digunakan sebanyak 100 data dengan label *hate speech* dan 100 data dengan label *non hate speech*. Adapun untuk data uji akan digunakan 25 data dengan label *hate speech* dan 25 data dengan label *non hate speech*. Skenario pengujian pengaruh normalisasi kata dapat dilihat pada Tabel 4.24.

**Tabel 4.24 Skenario pengujian pengaruh normalisasi kata**

| **Klasifikasi *Hate Speech* Tanpa Normalisasi dan Dengan Normalisasi** | | | | |
|---|---|---|---|---|
| **Jenis Pengujian** | ***accuracy*** | ***precison*** | ***recall*** | ***f-measure*** |
| *Pre-processing* tanpa normalisasi kata | | | | |
| *Pre-processing* dengan normalisasi kata | | | | |

### 4.4.2 Skenario pengujian pengaruh seleksi fitur *Information Gain*

Skenario pengujian ini menjelaskan tentang pengujian untuk mengetahui pengaruh proses seleksi fitur dengan *Information Gain*. Pada tahapan proses seleksi fitur dengan menggunakan *Information Gain, term* pada data uji yang tidak ada dalam daftar pengurutan *term Information Gain* akan dibuang dan tidak akan

dilakukan pembobotan kata, sebaliknya *term* pada data uji yang ada dalam daftar pengurutan term *Information Gain* akan disimpan dan akan dilakukan pembobotan kata yang akan dilanjutkan dengan perhitungan dengan Multinomial Naïve Bayes. *Threshold* yang akan digunakan untuk mengetahui pengaruh seleksi fitur *Information Gain* adalah sebesar 20%, 40%, 60%, 80%. Data yang digunakan berasal dari penelitian sebelumnya yang berisi *database tweet hate speech* dan *non hate speech* (Alfina, Mulia, Fanany, & Ekanata, 2017). Total data *tweet hate speech* yang akan digunakan adalah sebanyak 250 data, dengan total pembagian data *tweet hate speech* yaitu sebesar 80% untuk data latih dan 20% untuk data uji. Data latih yang akan digunakan sebanyak 100 data dengan label *hate speech* dan 100 data dengan label *non hate speech*. Adapun untuk data uji akan digunakan 25 data dengan label *hate speech* dan 25 data dengan label *non hate speech*. Skenario pengujian pengaruh seleksi fitur dengan *Information Gain* dapat dilihat pada Tabel 4.25.

**Tabel 4.25 Skenario pengujian pengaruh seleksi fitur *Information Gain***

| **Klasifikasi *Hate Speech* Tanpa dan Dengan *Information Gain*** | | | | | |
|---|---|---|---|---|---|
| Jenis Uji | *Threshold* | *Accuracy* | *Precision* | *Recall* | *F-Measure* |
| *Pre-processing* tanpa normalisasi dan Seleksi Fitur *Information Gain* | | | | | |
| *Pre-processing* dengan normalisasi dan Seleksi Fitur *Information Gain* | | | | | |

# BAB 5
# IMPLEMENTASI

## 5.1 Implementasi Algoritme

Implementasi algoritme adalah proses pembuatan dan penjelasan kode program untuk penelitian klasifikasi *hate speech* berbahasa Indonesia. Pada implementasi algoritme, setiap kode program akan dijelaskan dan akan mengacu pada perancangan proses pada Bab 4. Implementasi algoritme meliputi proses *pre-processing*, klasifikasi menggunakan *multinomial Naïve Bayes*, dan seleksi fitur menggunakan *Information Gain*.

### 5.1.1 Implementasi *Pre-processing*

Pada implementasi *pre-processing* dilakukan untuk mempersiapkan data sebelum diolah, data yang digunakan berupa teks. Data yang sudah dilakukan *pre-processing* nantinya akan diproses dengan menggunakan metode *Naive Bayes*. Pada tahap *preprocessing* ini terdapat beberapa proses, antara lain adalah *case folding, cleaning,* normalisasi, *filtering*, *stemming, dan tokenizing*. Proses pertama adalah *case folding*, proses ini mengubah setiap kata menjadi huruf kecil. Proses kedua adalah proses *cleaning*, proses ini berguna untuk menghilangkan karakter, simbol, ataupun pola karakter yang tidak diperlukan. Proses ketiga adalah proses normalisasi, proses ini berguna untuk mengubah kata-kata yang tidak baku menjadi baku. Proses keempat adalah proses *filtering*, proses ini berguna untuk menghilangkan kata-kata yang tidak penting pada suatu kalimat, seperti kata dan, atau, yang, dan masih banyak lagi. Proses *filtering* biasanya dikenal dengan *stopword removal*. Kemudian pada proses kelima adalah proses *stemming*, proses ini akan mengubah suatu kata menjadi kata dasar, seperti “memakan” menjadi “makan”, “tiduran” menjadi “tidur” dan masih banyak lagi. Proses terakhir pada *pre-processing* adalah proses *tokenizing*, proses ini berguna untuk memecah suatu kalimat *tweet* menjadi suatu kata per kata yang disebut *token*. Pada penelitian ini proses *pre-processing* dibuat dengan bantuan *library* Pujangga *Indonesian Natural Language Processing* REST API. Berikut proses *Pre-processing* yang disajikan pada potongan Source Code 5.1.

```
1   import re
2   import requests
3
4   def cleaning(t):
5       #m = re.sub(r'(rt)|(http?)(.*)|(@[A-Za-z0-
6   9_]+)|([#\t])|(\w+:\/\/\S+)|([0-9]+)', ' ', t)
7       m = re.sub(r'(rt )|(http?)(.*)|(@[A-Za-z0-9_]+)|(#[A-
8   Za-z0-9_]+)|([#\t])|(\w+:\/\/\S+)|([0-9]+)|(-)', ' ', t)
9       return m
10
11  def formalize(t):
12      r = requests.post('http://127.0.0.1:9000/formalizer',
13  json={'string': t})
14      return r.text[28:-2]
15
```

```
16  def filtering(t):
17      r = requests.post('http://127.0.0.1:9000/stopwords',
18  json={'string': t})
19      return r.text[28:-2]
20
21  def stemming(t):
22      r = requests.post('http://127.0.0.1:9000/stemmer',
23  json={'string': t})
24      return r.text[28:-2]
25
26  def tokenizing(t):
27      r =
28  requests.post('http://127.0.0.1:9000/sentence/tokenizer',
29  json={'string': t})
30      return r.text[28:-2]
31
32  def preprot(tx):
33      teks = tx.casefold()
34      teks = cleaning(teks)
35      teks = formalize(teks)
36      teks = filtering(teks)
37      teks = stemming(teks)
38      teks = teks.split()
39      return (teks)
```

**Source Code 5.1 Proses *Pre-processing***

Penjelasan Source Code 5.1 proses *pre-processing*, baris ke:

1. Mengimpor *library re* yang berguna sebagai *regex*.
2. Mengimpor *library request* yang berfungsi untuk memanggil *API*.

4-9. Fungsi untuk proses *cleaning*, dengan tujuan untuk menghilangkan simbol, URL, *hashtag*, kata-kata RT, *username*, dan pola karakter yang tidak diperlukan. Fungsi ini akan menerima parameter berupa *string.*

11-14. Fungsi untuk proses normalisasi, dengan tujuan untuk mengubah kata-kata yang tidak baku menjadi baku sesuai dengan *formalizationDict*. Fungsi ini akan menerima parameter berupa *string*.

16-19. Fungsi untuk proses *filtering*, dengan tujuan untuk menghilangkan kata-kata yang tidak penting atau yang biasanya dikenal sebagai *stopword.* Fungsi ini akan menerima parameter berupa *string*.

21-24. Fungsi untuk proses *stemming*, dengan tujuan untuk mengubah setiap kata menjadi bentuk dasar. Fungsi ini menerima parameter berupa *string*.

26-30. Fungsi untuk proses *tokenizing*, dengan tujuan untuk memecah kalimat menjadi kata demi kata atau yang dikenal dengan istilah *token*. Fungsi ini akan menerima parameter berupa *string*.

32-39. Fungsi untuk melakukan tahap *pre-processing* dengan parameter berupa *string* dengan memanggil setiap fungsi yang akan digunakan pada tahap *pre-processing*.

### 5.1.2 Implementasi Proses Pembobotan Kata

Proses pembobotan kata berguna untuk mengetahui bobot setiap kata, metode yang digunakan adalah *Term Frequency* (TF). Metode ini akan menghitung bobot setiap kata dengan cara menghitung jumlah kemunculan sebuah kata pada setiap dokumen. Proses pembobotan kata diperlukan untuk dilanjutkan pada proses klasifikasi menggunakan *Multinomial Naïve Bayes*. Berikut implementasi proses pembobotan kata yang disajikan pada potongan Source Code 5.2.

```
import openpyxl as op
import preprocessing as pre

from pymongo import MongoClient
client = MongoClient('localhost', 27017)
db = client.Bayes.TFFormalize

excel = op.load_workbook("Data Latih.xlsx")
sheet= excel["DataLatih"]

kelas =[]
for x in range(200):
    if(x<100):
        doc =sheet.cell(row=x+1,column=2).value
        kelas.append([pre.preprot(doc),0])
    else:
        doc =sheet.cell(row=x+1,column=2).value
        kelas.append([pre.preprot(doc),1])
print(kelas)

dictHS ={}
dictNONHS ={}
TF = []
indeks = 0
for x in kelas:
    for y in x[0]:
        if(x[1]==0):
            if(str(dictHS.get(y))=='None'):
                dictHS.update({y:1})
            else:
                dictHS[y]=dictHS.get(y)+1
        elif(x[1]==1):
            if(str(dictNONHS.get(y))=='None'):
                dictNONHS.update({y:1})
            else:
                dictNONHS[y]=dictNONHS.get(y)+1
TF.append(dictHS)
TF.append(dictNONHS)

indeks=0
for x in TF:
    for y in x:
        if(indeks==0):
            print(y,x.get(y),"HS")
            dok = {"kata":y,"jumlah" :
x.get(y),"kelas":"HS"}
            #db.insert_one(dok)
        elif(indeks==1):
```

```
49              print(y,x.get(y),"NONHS")
                dok = {"kata":y,"jumlah" :
50  x.get(y),"kelas":"NONHS"}
51              #db.insert_one(dok)
52      indeks+=1
53
```

**Source Code 5.2 Proses Pembobotan Kata**

Penjelasan Source Code 5.2 proses pembobotan kata, baris ke:

1. Mengimpor *library openpyxl* dengan nama *op* yang berguna untuk memproses data pada excel.

2. Mengimpor kelas *pre-processing* dengan nama *pre* agar bisa dilakukan proses *pre-processing*.

4-6. Mengimpor *MongoClient* dari *pymongo* untuk mengakses *database* MongoDB yang telah diberi nama *Bayes* dan di dalamnya terdapat tabel *TFFormalize*.

8-9. Membuka data latih pada excel dengan nama *sheet* DataLatih.

11-19. Melakukan inisialisasi variabel kelas yang kemudian akan dilakukan perulangan pada Data Latih yang berjumlah 200 Data. Jika data berada pada rentang 1-100 maka akan masuk ke dalam kelas HS dan jika data pada rentang 101-200 akan masuk ke dalam kelas NONHS. Setelah dilakukan perulangan maka variabel kelas akan di cetak.

21-38. Melakukan inisialisasi variabel *dictHS*, *dictNONHS*, dan juga variabel *TF*. Kemudian akan dilakukan perulangan untuk menghitung jumlah masing-masing kata pada masing-masing *dictionary* yaitu *dictHS* dan *dictNONHS*. Kemudian setiap masing-masing kata yang telah dihitung akan dimasukan

40-53. Melakukan inisialisasi indeks sama dengan 0. Kemudian melakukan perulangan untuk memasukkan masing-masing kata ke dalam *database* MongoDB sesuai dengan kelasnya masing-masing yaitu HS dan NONHS.

### 5.1.3 Implementasi Algoritme Seleksi Fitur *Information Gain*

Algoritme seleksi fitur digunakan adalah *Information Gain*. Seleksi fitur ini akan menghitung kata/term dengan nilai *Information Gain* tertinggi yang akan diurutkan dari nilai terbesar hingga nilai terkecil. Kemudian akan disimpan dalam format notepad.txt yang selanjutnya akan diseleksi dengan jumlah tertentu. Kata yang terpilih adalah kata dengan nilai *Information Gain* teratas karena nilai *Information Gain* yang tinggi memiliki arti bahwa sebuah kata dapat merepresentasikan suatu kelas dengan baik yang mana dalam kasus ini adalah kelas *Hate Speech* dan juga *Non Hate Speech*. Berikut implementasi proses seleksi fitur *Information Gain* yang disajikan pada potongan Source Code 5.3.

```
1   #Menghitung nilai pt
2   def hitungpt(kelas):
3       nilai = {}
4       #IG = []
5       for x in dokumenall:
```

```
        if(x[1]==kelas):
            for y in x[0]:
                for z in term:
                    if(str(y)==str(z) and
str(nilai.get(z))=='None'):
                        nilai.update({z:1})
                    elif(str(y)==str(z)):
                        nilai[z]=nilai.get(z)+1
    return(nilai)

for x in range(2):
    print("\n============================")
    for y in term:
        ppt = term.get(y)/len(dokumen)
        ppnt = (len(dokumen)-term.get(y))/len(dokumen)
        if(str(IG[x][0].get(y))=='None'):
            pt = 0
            pnt = ((100-0)/(len(dokumenall)-
term.get(y)))*(math.log10(((100-0)/(len(dokumenall)-
term.get(y)))))
            print(y+"\t"+str(pt)+" "+str(pnt))
            #print(y+"\t"+str(ppt)+" "+str(ppnt))
            dictpt.update({y:pt})
            dictpnt.update({y:pnt})
            totalpt.update({y:ppt})
            totalpnt.update({y:ppnt})
        else:
            pt =
(float(IG[x][0].get(y))/term.get(y))*(math.log10((float(IG[
x][0].get(y))/term.get(y))))
            if((100-float(IG[x][0].get(y)))==0):
                pnt = 0
            else:
                pnt = ((100-
float(IG[x][0].get(y)))/(len(dokumenall)-
term.get(y)))*(math.log10(((100-
float(IG[x][0].get(y)))/(len(dokumenall)-term.get(y)))))
            print(y+"\t"+str(pt)+" "+str(pnt))
            #print(y+"\t"+str(ppt)+" "+str(ppnt)+"= - = - =
- = - = - = - = - = - =")
            dictpt.update({y:pt})
            dictpnt.update({y:pnt})
            totalpt.update({y:ppt})
            totalpnt.update({y:ppnt})
    totalIGpt.append([dictpt,x])
    totalIGpnt.append([dictpnt,x])
    dictpt={}
    dictpnt={}

pc = 0
#menghitung p|ci
for x in range(2):
    pc +=
((100/len(dokumen))*(math.log10(100/len(dokumen))))
pc = -(pc)

nilaigain = {}
totalgain = 0
sortIG = []
```

```
for x in term:
    for y in range(2):
        totalt += (totalIGpt[y][0].get(x))
        totalnt += (totalIGpnt[y][0].get(x))
    totalgain =
(pc)+(totalpt.get(x)*totalt)+(totalpnt.get(x)*totalnt)
    print(x+" "+str(totalgain))

print(x,pc,(totalpt.get(x)*totalt),(totalpnt.get(x)*totalnt
))
    nilaigain.update({x:totalgain})
    totalt = 0
    totalnt = 0

print("\nHasil Perhitungan Nilai Information Gain Masing-
Masing Kata:")
for x in nilaigain:
    print(str(x)+" "+str(nilaigain.get(x)))

for x in nilaigain:
    if nilaigain.get(x) not in sortIG:
        sortIG.append(nilaigain.get(x))
sortIG.sort(reverse=True)
print("\nNilai Information Gain Setelah diurutkan:")
print(sortIG)
selectedterm = {}
for x in sortIG:
    for y in nilaigain:
        if(x==nilaigain.get(y)):
            selectedterm.update({y:nilaigain.get(y)})

file_bio = open("IG.txt", "w")

indeks = 0
jumlahkata = len(selectedterm)*0.8
for x in selectedterm:
    if(indeks<(jumlahkata-1)):
        print(x,selectedterm.get(x))
        file_bio.write(x+"|")
    elif(indeks==jumlahkata-1):
        print(x,selectedterm.get(x))
        file_bio.write(x)
    indeks+=1

file_bio.close()
```

**Source Code 5.3 Seleksi Fitur *Information Gain***

Penjelasan Source Code 5.3 seleksi fitur *Information Gain*, baris ke:

1-14. Menginisialisasi sebuah fungsi *hitungpt* dan variabel nilai yang kemudian akan dilakukan perulangan untuk menghitung kemunculan umlah sebuah kata pada kelas tertentu.

16-20. Melakukan proses perulangan untuk menghitung peluang kemunculan suatu kata pada dokumen setiap kelas dan ketidak munculan kata pada dokumen setiap kelas dan melakukan proses perulangan untuk menghitung kemunculan kata dalam setiap dokumen.

21-31. Melakukan proses seleksi kondisi ketika nilai peluang term *t* yang muncul dalam dokumen atau $P_Y(c_i|t)$ bernilai 0.

32-35. Melakukan proses seleksi kondisi ketika nilai peluang term *t* yang muncul dalam dokumen atau $P_Y(c_i|t)$ tidak bernilai 0.

36-37. Melakukan seleksi kondisi ketika nilai peluang term *t* yang tidak muncul dalam dokumen atau $P_Y(c_i|\bar{t})$ bernilai 0.

38-43. Melakukan seleksi kondisi ketika nilai peluang term *t* yang tidak muncul dalam dokumen atau $P_Y(c_i|\bar{t})$ tidak bernilai 0.

46-51. Menyimpan hasil perhitungan peluang kemunculan kata dari nilai peluang term *t* yang muncul dan tidak muncul.

52-53. Mengosongkan *dictionary pt* dan *pnt*.

55-60. Melakukan proses perulangan untuk menghitung peluang dari kelas data.

65. Melakukan proses perulangan untuk menghitung nilai *Information Gain* setiap *term.*

66. Melakukan proses perulangan untuk menghitung *nilai Information Gain* setiap kelas.

67-74. Melakukan proses perhitungan untuk menghitung total nilai *Information Gain* dari setiap kata.

75. Melakukan proses untuk menyimpan nilai *Information Gain* dari masing-masing kata.

76-77. Mengosongkan nilai *totalt* dan *totalnt*.

79-82. Menampilkan proses perhitungan *Information Gain* dari masing-masing kata.

84-94. Melakukan proses perulangan untuk mengurutkan nilai *Information Gain* dari masing-masing kata dan mengurutkan seluruh *term* serta menyimpan term berdasarkan nilai *Information Gain* mulai dari nilai tertinggi.

96. Membuka sebuah file dengan nama IG.txt untuk memasukkan nilai *Information Gain* dari masing-masing kata.

98. Melakukan deklarasi variabel indeks sama dengan 0.

99. Deklarasi variabel jumlah kata untuk menentukan *threshold* dari jumlah *term* yang akan digunakan sebagai seleksi fitur pada metode Multinomial Naïve Bayes.

100. Melakukan proses perulangan untuk menyimpan *term* yang akan dilakukan proses seleksi.

101-107. Melakukan proses seleksi kondisi untuk menyimpan *term.*

108. Menutup file IG.txt yang telah berhasil diisikan dengan nilai *Information Gain* yang telah diurutkan berdasarkan nilai tertinggi.

### 5.1.4 Implementasi Algoritme *Naive Bayes Classifier*

Algoritme yang digunakan untuk melakukan klasifikasi adalah algoritme *Naive Bayes Classifier*. Dalam penelitian ini algoritme *Naive Bayes Classifier* yang digunakan adalah *Multinomial Naïve Bayes*. Hal ini dikarenakan *Multinomial Naive Bayes* merupakan algoritme yang paling cocok dan memiliki akurasi yang paling baik jika diterapkan pada pemrosesan teks. Pada proses perhitungan *Multinomial Naive Bayes* diperlukan beberapa tahapan perhitungan di antaranya adalah perhitungan *prior, conditional probability, total conditional probability, posterior,* dan hasil identifikasi *hate speech*. Implementasi algoritme *Naive Bayes Classifier* dapat dilihat dalam kode program yang disajikan pada potongan Source Code 5.4.

```
1   from pymongo import MongoClient
2   client = MongoClient('localhost', 27017)
3   db = client.Bayes.TFFormalize
4   db1 = client.Bayes.DataLatih
5
6   #Deklarasi Prior
7   priorHS = 0
8   priorNONHS = 0
9   total = 0
10  data = db1.find()
11
12  #Menghitung Prior
13  for x in data:
14      if(x['kelas']=="HS"):
15          priorHS+=1
16          total+=1
17      else:
18          priorNONHS+=1
19          total+=1
20
21  #Memasukkan Input Tweet
22  nilai = []
23  term = input("Masukkan Tweet: ")
24  term = term.split()
25  text_file = open("IG.txt", "r")
26  kata = text_file.read().split('|')
27  proses = []
28  #print(kata)
29  for x in term:
30      if(x in kata):
31          proses.append(x)
32  print(proses)
33  text_file.close()
34
35  #Menghitung Conditional Probability (likelihood)
36  kategori= ""
37  likelihood = {}
38  for z in range(2):
39      for x in proses:
40          if(z==0):
41              kategori="NONHS"
```

```
42              kataunik = len(dictnonhs)
43          else:
44              kategori="HS"
45              kataunik = len(dicths)
46          jumlah=(db.find_one({"kata":x,"kelas":kategori}))
47          if(str(jumlah)=="None"):
48              total = (0+1)/(kataunik+len(dictall))
49              print(x,0,total)
50          else:
51              total =
52  (jumlah['jumlah']+1)/(kataunik+len(dictall))
53              print(x,jumlah['jumlah'],total)
54          likelihood.update({x:total})
55      nilai.append(likelihood)
56      likelihood={}
57  print("\n[Conditional Probability NONHS, Conditional
58  Probability HS]")
59  print(nilai,"\n")
60
61  #Menghitung Posterior
62  posterior = 0
63  total = 1
64  totalpos=[]
65  for x in range(2):
66      for y in nilai[x]:
67          total*=nilai[x].get(y)
68      if(x==0):
69
70  totalpos.append((priorNONHS/(priorHS+priorNONHS))*total)
71          print("Total Conditional Probability NON HS:",total)
72      elif(x==1):
73
74  totalpos.append((priorHS/(priorHS+priorNONHS))*total)
75          print("Total Conditional Probability
76  HS:",total,"\n")
77      total=1
78
79  print("[Posterior NON HS, Posterior HS]")
80  print(totalpos,"\n")
81  maks =max(totalpos[0],totalpos[1])
82
83  if(maks==totalpos[0]):
84      print("Dengan nilai posterior",maks,"maka tweet masuk
85  pada kelas NONHS")
86  elif(maks==totalpos[1]):
87      print("Dengan nilai posterior",maks,"maka tweet masuk
88  pada kelas HS")
```

**Source Code 5.4 Algoritme *Naïve Bayes Classifier***

Penjelasan Source Code 5.4 Algoritme *Naïve Bayes Classifier*, baris ke:

1-4. Mengimpor *MongoClient* dari *pymongo* untuk mengakses *database* MongoDB yang telah diberi nama Bayes dan di dalamnya terdapat tabel perhitungan *TFFormalize* dan juga tabel *DataLatih*.

6-9. Melakukan inisialisasi awal untuk menghitung nilai prior dari kelas *hate speech* dan nilai prior dari kelas *non hate speech* yang diberi nama *priorHS* dan *priorNONHS*.

10. Melakukan proses pencarian pada *database* berapa jumlah *tweet* yang masuk dalam kelas HS dan juga kelas NONHS.

12-19. Melakukan perulangan dan melakukan seleksi kondisi untuk menghitung jumlah *prior* dari masing-masing kelas, yaitu kelas HS dan juga kelas NONHS.

21-33. Melakukan inisialisasi variabel nilai dan memberikan input untuk *user* agar bisa menuliskan sebuah *tweet*. Kemudian *tweet* yang telah ditulis oleh *user* akan dilakukan proses *split* dan akan dilakukan pencocokan kata atau term dengan seleksi fitur *Information Gain* yang sudah disimpan dengan file IG.txt. Selanjutnya kata atau term yang berada pada file IG.txt akan dilanjutkan untuk dilakukan proses perhitungan *conditional probability.*

35-59. Melakukan inisialisasi variabel kategori dan *likelihood.* Kemudian melakukan proses perhitungan *conditional probability* dengan melakukan seleksi kondisi untuk mencari setiap kata pada masing-masing kelas yang telah melewati proses *Information Gain* dan menghitung jumlah kemunculan masing-masing kata pada setiap kelas kemudian dibagi dengan jumlah kata unik pada masing-masing kelas yang ditambahkan dengan jumlah kata unik pada keseluruan kelas.

61-88. Melakukan inisialisasi variabel *posterior*, *total*, dan *totalpos*. Kemudian melakukan perulangan untuk menghitung *posterior* dari masing-masing kelas. Perhitungan *posterior* didapatkan dengan cara mengalikan nilai *prior* dari masing-masing kelas dengan nilai *conditional probability* dari setiap kata pada masing-masing kelas. Selanjutkan masing-masing nilai posterior dari kelas HS dan NONHS akan dibandingkan dan kelas dengan nilai *posterior* tertinggi akan mewakili bahwa *tweet* yang ditulis oleh *user* akan diklasifikasikan ke dalam kelas tersebut.

## 5.2 Tampilan Program

Subbab ini menjelaskan tentang tampilan hasil program dari tahap *pre-processing* sampai dengan perhitungan klasifikasi menggunakan *Multinomial Naive Bayes* pada kelas data uji. Penjelasan masing-masing tampilan terdapat pada Subbab 5.2.1 hingga Subbab 5.2.2.

### 5.2.1 Tampilan dari Tahap *Pre-processing*

Tampilan dari tahap *pre-processing* merupakan tampilan hasil dari proses *pre-processing* mulai dari proses *case folding*, *cleaning*, normalisasi, *filtering*, *stemming*, dan *tokenizing*. Contoh tampilan dari tahap *pre-processing* terdapat pada Gambar 5.1.

```
Masukkan Tweet: #MataNajwaDebatJakarta pak anies banyak omdo dulu waktu jd menteri
terobosannya apa?? skrg ngomong ky sok Jago bgt

#Pre-processing

1. case folding: #matanajwadebatjakarta pak anies banyak omdo dulu waktu jd menteri
terobosannya apa?? skrg ngomong ky sok jago bgt

2. cleaning:  pak anies banyak omdo dulu waktu jd menteri terobosannya apa?? skrg
ngomong ky sok jago bgt

3. normalisasi: pak anies banyak omong doang dulu waktu jadi menteri terobosannya
apa? ? sekarang bicara seperti sok jago banget

4. filtering: anies omong doang menteri terobosannya ? ? bicara sok jago banget

5. stemming: anies omong doang menteri terobos bicara sok jago banget

6. tokenizing: ['anies', 'omong', 'doang', 'menteri', 'terobos', 'bicara', 'sok',
'jago', 'banget']
```

**Gambar 5.1 Tampilan dari Tahap *Pre-processing***

### 5.2.2 Tampilan Kelas dari Data Uji

Tampilan kelas dari data uji merupakan tampilan hasil dari perhitungan klasifikasi menggunakan *Multinomial Naive Bayes* antara kelas *hate speech* (HS) dan kelas *non hate speech* (NON HS) pada data uji. Contoh tampilan kelas dari data uji terdapat pada Gambar 5.2.

```
Masukkan Tweet: anies omong doang menteri terobos bicara sok jago banget
['sok', 'jago']
sok 0 0.000819000819000819
jago 0 0.000819000819000819
sok 3 0.0030745580322828594
jago 1 0.0015372790161414297

[Conditional Probability NONHS, Conditional Probability HS]
[{'sok': 0.000819000819000819, 'jago': 0.000819000819000819}, {'sok':
0.0030745580322828594, 'jago': 0.0015372790161414297}]

Total Conditional Probability NON HS: 6.707623415240124e-07
Total Conditional Probability HS: 4.726453546937524e-06

[Posterior NON HS, Posterior HS]
[3.353811707620062e-07, 2.363226773468762e-06]

Dengan nilai posterior 2.363226773468762e-06 maka tweet masuk pada kelas HS
```

**Gambar 5.2 Tampilan Kelas dari Data Uji**

# BAB 6
# PENGUJIAN DAN ANALISIS

## 6.1 Skenario Pengujian

Pada skenario pengujian yang akan dilakukan, terdapat 2 macam skenario pengujian. Skenario pengujian yang pertama adalah skenario pengujian untuk mengetahui pengaruh proses normalisasi kata dan tidak adanya proses normalisasi kata pada tahap *pre-processing.* Skenario pengujian yang kedua adalah skenario pengujian untuk mengetahui pengaruh variasi *threshold* yang akan digunakan pada proses seleksi fitur *Information Gain*.

### 6.1.1 Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi

Skenario pengujian ini dilakukan untuk mengetahui pengaruh proses normalisasi kata dan tidak adanya proses normalisasi kata pada tahap *pre-processing*. Pada proses pengujian dengan menggunakan normalisasi dilakukan proses pencocokan kata-kata yang tidak sesuai dengan Kamus Besar Bahasa Indonesia (kata tidak baku) pada *formalizationDict*. Proses normalisasi kata yang dilakukan ini menggunakan Pujangga *Indonesian Natural Language Processing* REST API. Pada proses *pre-processing,* seluruh data uji dan data latih akan melewati seluruh tahapan *pre-processing*, namun nantinya hanya dilakukan perbedaan pada proses normalisasi kata dan tidak adanya normalisasi kata. Pada pengujian ini data yang akan digunakan adalah data sekunder yang berisi *tweet hate speech* dan *non hate speech* yang didapatkan dari dari penelitian sebelumnya (Alfina, Mulia, Fanany, & Ekanata, 2017). Data latih yang diambil sebanyak 200 *tweet*, yang terdiri dari 100 *tweet hate speech* dan 100 *tweet non hate speech*. Data uji yang akan digunakan sebanyak 50 data, yang terdiri dari 25 *tweet hate speech* dan 25 *tweet non hate speech.* Hasil skenario pengujian pengaruh normalisasi kata pada tahap *pre-processing* dapat dilihat pada Tabel 6.1.

**Tabel 6.1 Hasil Pengujian Normalisasi dan Tanpa Normalisasi**

| **Klasifikasi *Hate Speech* Tanpa dan Dengan Normalisasi** | | | | |
|---|---|---|---|---|
| **Jenis Uji** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F-Measure*** |
| *Pre-processing* tanpa normalisasi | 92% | 92% | 92% | 92% |
| *Pre-processing* dengan normalisasi | 96% | 92% | 100% | 95,83% |

### 6.1.2 Pengujian Pengaruh Seleksi Fitur *Information Gain*

Pengujian ini menjelaskan tentang pengujian untuk mengetahui pengaruh proses seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* tanpa normalisasi dan dengan normalisasi. Pada proses pengujian pengaruh seleksi fitur *Information Gain*, digunakan beberapa *threshold* untuk mengambil fitur yang akan

digunakan pada proses perhitungan klasifikasi menggunakan *Multinomial Naïve Bayes*, yaitu 20%, 40%, 60%, 80%, dan 90%. Pada proses *pre-processing,* seluruh data uji dan data latih akan melewati seluruh tahapan *pre-processing*, namun nantinya hanya dilakukan perbedaan pada proses normalisasi kata dan tidak adanya normalisasi kata. Pada pengujian ini data yang akan digunakan adalah data sekunder yang berisi *tweet hate speech* dan *non hate speech* yang didapatkan dari dari penelitian sebelumnya (Alfina, Mulia, Fanany, & Ekanata, 2017). Data latih yang diambil sebanyak 200 *tweet*, yang terdiri dari 100 *tweet hate speech* dan 100 tweet *non hate speech*. Untuk data uji yang akan digunakan sebanyak 50 data, yang terdiri dari 25 *tweet hate speech* dan 25 tweet *non hate speech.* Hasil skenario pengujian terhadap variasi *threshold* yang akan digunakan pada proses seleksi fitur *Information Gain* dapat dilihat pada Tabel 6.2.

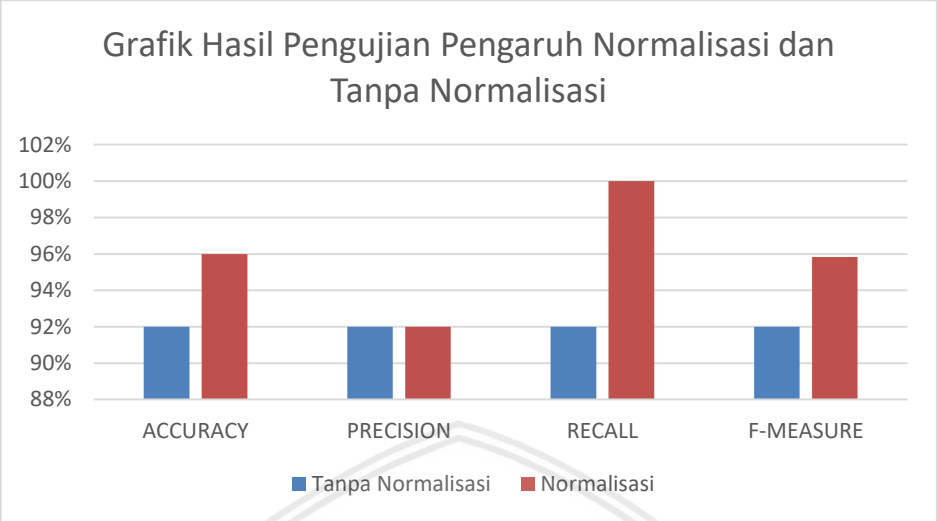
**Tabel 6.2 Hasil Pengujian Pengaruh Seleksi Fitur *Information Gain***

| **Klasifikasi *Hate Speech* Tanpa dan Dengan *Information Gain*** | | | | | |
|---|---|---|---|---|---|
| **Jenis Uji** | ***Threshold*** | ***Accuracy*** | ***Precision*** | ***Recall*** | ***F-Measure*** |
| Seleksi Fitur *Information Gain* Tanpa Normalisasi | 20% | 88% | 88% | 88% | 88% |
| | 40% | 88% | 92% | 85,18% | 88,46% |
| | 60% | 92% | 100% | 86,20% | 92,59% |
| | 80% | 94% | 100% | 89,28% | 94,33% |
| | 90% | 92% | 96% | 88,89% | 92,30% |
| Seleksi Fitur *Information Gain Dengan* Normalisasi | 20% | 92% | 88% | 95,65% | 91,66% |
| | 40% | 92% | 92% | 92% | 92% |
| | 60% | 96% | 100% | 92,59% | 96,15% |
| | 80% | 98% | 100% | 96,15% | 98,03% |
| | 90% | 96% | 96% | 96% | 96% |

## 6.2 Hasil Pengujian

Hasil pengujian pada penelitian ini akan menjelaskan tentang analisis dari skenario pengujian yang sudah dilakukan. Terdapat 2 hasil pengujian pada penelitian ini. Pertama, yaitu hasil pengujian penggunaan normalisasi kata dan tidak adanya normalisasi kata pada tahapan proses *pre-processing*. Kedua, yaitu hasil pengujian pengaruh penggunaan proses seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* dengan normalisasi dan tanpa normalisasi.

### 6.2.1 Hasil Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi




**Gambar 6.1 Hasil Pengujian Pengaruh Normalisasi dan Tanpa Normalisasi**

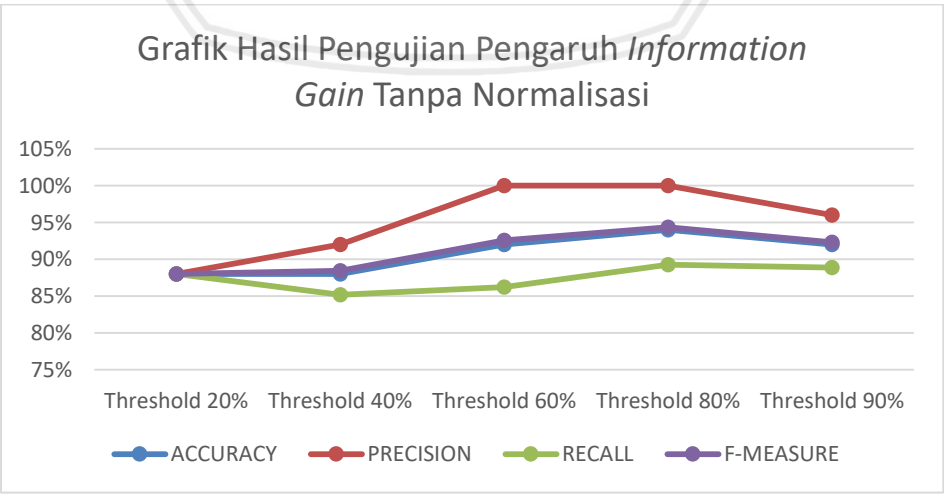
Pada pengujian pengaruh penggunaan *pre-processing* dengan normalisasi dan tanpa normalisasi dilakukan dua proses pengujian. Proses pengujian pertama *tweet* data uji *hate speech* tidak dilakukan proses normalisasi kata pada tahapan proses *pre-processing*. Kemudian pada proses pengujian kedua, *tweet* data uji *hate speech* dilakukan proses normalisasi kata pada tahapan proses *pre-processing* menggunakan kamus *formalizationDict*. Dari kedua proses pengujian yang telah dilakukan, diperoleh hasil dari tahap pengujian pertama pada saat *pre-processing* dilakukan tanpa menggunakan normalisasi dan mendapatkan nilai *accuracy*, *precision, recall,* dan *f-measure* sebesar 92%, 92%, 92%, dan 92%. Hasil tersebut sudah bisa dikatakan cukup baik karena proses klasifikasi *Multinomial Naïve Bayes* dengan *pre-processing* tanpa menggunakan normalisasi mampu menghasilkan akurasi sebesar 92% dan *f-measure* sebesar 92%. Hal ini menunjukkan bahwa setiap *tweet* sudah bisa diklasifikasikan tepat sesuai dengan kelas aslinya. Namun, *pre-processing* tanpa menggunakan normalisasi masih memiliki banyak kekurangan, antara lain pada saat *tweet* pada data uji memiliki kata yang sama tetapi berbeda dalam penulisannya seperti kata “sm”, “sma”, “sama”, “samaa”, “samma” yang seharusnya maknanya adalah “sama”, maka akan dibedakan setiap kata dan membuat data menjadi semakin lebih banyak padahal kata tersebut tidak penting dan kurang bisa mendefinisikan secara jelas suatu kelas tertentu. Hal itu akan menyebabkan proses klasifikasi *hate speech* menjadi kurang maksimal dan memberikan hasil akurasi yang belum optimal.

Kemudian pada tahapan pengujian yang kedua, *tweet* data uji *hate speech* dilakukan proses normalisasi kata pada tahapan proses *pre-processing* dengan menggunakan kamus *formalizationDict*. Kemudian dari hasil pengujian, dihasilkan nilai *accuracy* sebesar 96%, nilai *precision* sebesar 92%, nilai *recall* sebesar 100%, dan nilai *f-measure* sebesar 95.83%. Hasil ini membuktikan bahwa klasifikasi *hate speech* mendapatkan hasil yang lebih baik dan lebih optimal jika menggunakan proses normalisasi kata pada tahapan proses *pre-processing*. Hal itu dikarenakan

semua kata yang berada pada *tweet* data uji *hate speech* dilakukan normalisasi kata dengan menggunakan kamus *formalizationDict*. Contohnya kata "sm" menjadi "sama", kata "samaa" menjadi "sama", kata "s4ma" menjadi "sama", kata "yg" menjadi "yang", dan masih banyak lagi. Hal itu menyebabkan kata yang berada pada *tweet* data uji *hate speech* akan lebih banyak muncul pada *tweet* data latih *hate speech* dibandingkan *pre-processing* tanpa normalisasi kata. Data yang disimpan juga tidak akan sebanyak dengan data yang disimpan pada *pre-processing* tanpa normalisasi, dengan begitu kata dengan makna yang sama namun penulisannya berbeda akan diubah menjadi satu kata dan kata yang seharusnya dibuang karena merupakan *stopword* seperti kata "yg" kemudian karena dilakukan proses normalisasi kata maka akan menjadi kata "yang" dan akan dibuang karena merupakan *stopword* pada proses *filtering*. Namun, penggunaaan *pre-processing* dengan normalisasi masih juga mempunyai salah satu kekurangan yaitu gagalnya kamus *formalizationDict* dalam mengubah kata yang tidak baku menjadi baku seperti kata "agama!" menjadi "agamai".

### 6.2.2 Hasil Pengujian Pengaruh Seleksi Fitur *Information Gain*

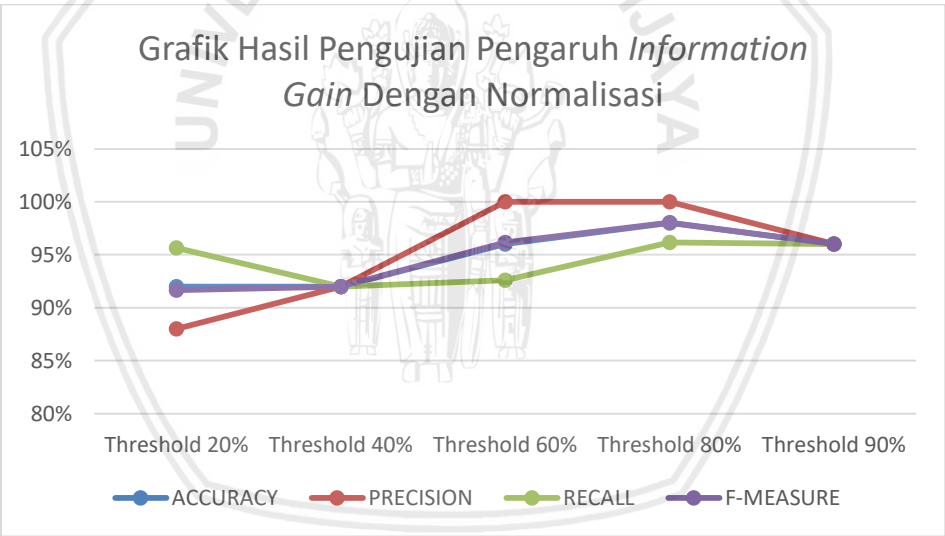
Pada pengujian ini dilakukan untuk mengetahui pengaruh proses seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* tanpa normalisasi dan dengan normalisasi. Pada proses pengujian pengaruh seleksi fitur *Information Gain*, digunakan beberapa *threshold* untuk mengambil fitur yang akan digunakan pada proses perhitungan klasifikasi menggunakan *Multinomial Naïve Bayes*, yaitu 20%, 40%, 60%, 80%, dan 90%. Pada pengujian pengaruh seleksi fitur *Information Gain* dilakukan dua proses. Proses pertama dilakukan dengan seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* tanpa normalisasi dan yang kedua adalah proses seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* dengan normalisasi. Hasil dari pengujian tahap pertama dilakukan dengan seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* tanpa normalisasi dan mendapatkan hasil *accuracy,* hasil *precision,* hasil *recall,* dan hasil *f-measure* yang terdapat pada Gambar 6.2.




**Gambar 6.2 Hasil Pengujian Pengaruh *Information Gain* Tanpa Normalisasi**

Hasil ini menghasilkan akurasi yang baik dari setiap variasi *threshold* seleksi fitur *Information Gain* yang digunakan, walaupun tidak dilakukan proses normalisasi pada tahap *pre-processing*. Jika dilihat pada Gambar 6.2 hasil terbaik didapatkan ketika menggunakan *Information Gain* dengan *threshold* sebesar 80% dan mendapatkan nilai *accuracy, precision, recall, dan f-measure* sebesar 94%, 100%, 89%, dan 94%. Hal ini dikarenakan dengan bantuan seleksi fitur *Information Gain,* suatu kata yang mencerminkan suatu kelas akan mendapatkan nilai tertinggi dan akan menjadi kata yang akan dihitung pada proses klasifikasi *Multinomial Naive Bayes* dan kata-kata yang tidak penting dan tidak mencerminkan suatu kelas akan diabaikan dan tidak dihitung sehingga akan meningkatkan akurasi. Pemilihan variasi *threshold* yang optimal dan pas pada seleksi fitur *Information Gain* juga akan mempengaruhi nilai akurasi dari klasifikasi *Multinomial Naïve Bayes*.

Kemudian pada tahap kedua adalah dilakukan pengujian dengan seleksi fitur *Information Gain* yang dipadukan dengan *pre-processing* dengan normalisasi dan nilai *threshold* dari *Information Gain* sebesar 20%, 40%, 60%, 80%, dan 90%. Hasil yang didapatkan dari pengujian yang telah dilakukan mendapatkan hasil *accuracy,* hasil *precision,* hasil *recall, dan* hasil *f-measure* yang dapat dilihat pada Gambar 6.3.




**Gambar 6.3 Hasil Pengujian Pengaruh *Information Gain* Dengan Normalisasi**

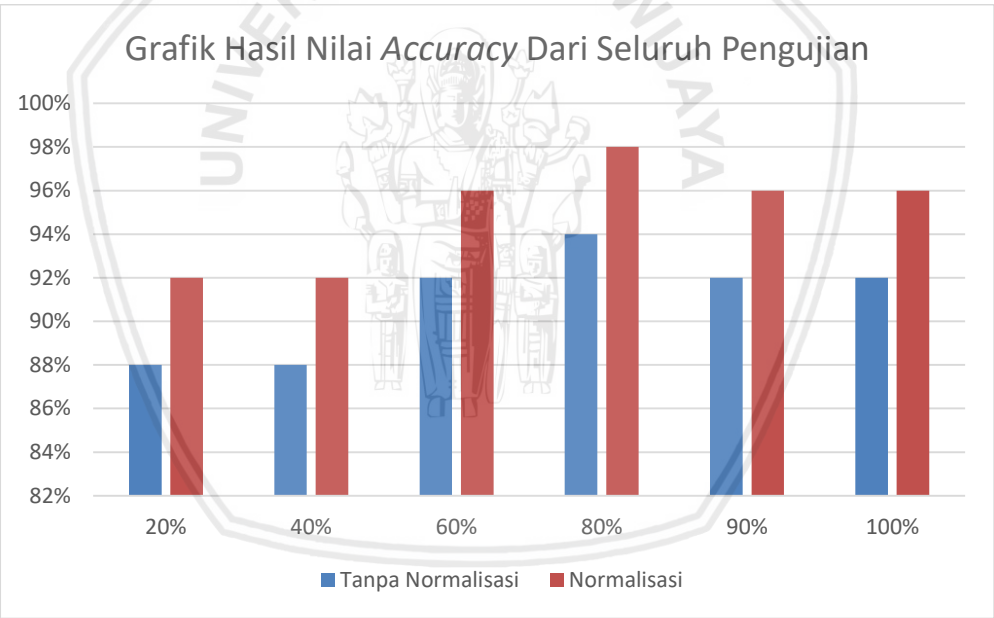
Hasil ini menghasilkan akurasi terbaik dari seluruh pengujian yang dilakukan. Hasil terbaik didapatkan ketika menggunakan seleksi fitur *Information Gain* dengan nilai *threshold* sebesar 80% dengan proses normalisasi pada tahap *pre-processing*. Hal itu dikarenakan pada pengujian pengaruh *Information Gain* dengan normalisasi dilakukan proses normalisasi kata pada proses *pre-processing* yang mengubah kata-kata yang tidak sesuai dengan Kamus Besar Bahasa Indonesia (kata tidak baku) menjadi kata yang baku dengan menggunakan kamus *formalizationDict*. Kemudian melakukan seleksi fitur *Information Gain* dengan pemilihan *threshold* yang optimal akan berguna untuk menyeleksi kata atau fitur yang mencerminkan suatu kelas dengan jelas dan akan membuang kata-kata yang tidak penting atau yang tidak berpengaruh pada suatu kelas, sehingga kata-kata

yang dihasilkan pada *tweet* data uji *hate speech* memiliki banyak kesamaan kemunculan kata dengan *tweet* data latih *hate speech* yang akan memberikan banyak pengaruh pada proses klasifikasi *Multinomial Naïve Bayes* dan akan meningkatkan akurasi secara optimal. Banyaknya kata yang muncul pada data latih dan menghitung kata-kata yang mencerminkan suatu kelas akan membuat hasil akurasi menjadi lebih optimal.

### 6.2.3 Hasil dari seluruh pengujian

Hasil ini bertujuan untuk menggambarkan seluruh hasil pengujian yang telah dilakukan mulai dari skenario pengujian untuk mengetahui pengaruh proses normalisasi kata dan tidak adanya proses normalisasi kata pada tahap *pre-processing* dan juga skenario pengujian untuk mengetahui pengaruh variasi *threshold* yang akan digunakan pada proses seleksi fitur *Information Gain*. Hasil seluruh pengujian akan menampilkan grafik masing-masing dari hasil *accuracy,* hasil *precision,* hasil *recall, dan* hasil *f-measure* yang didapatkan dari seluruh pengujian.

1. Hasil *accuracy*

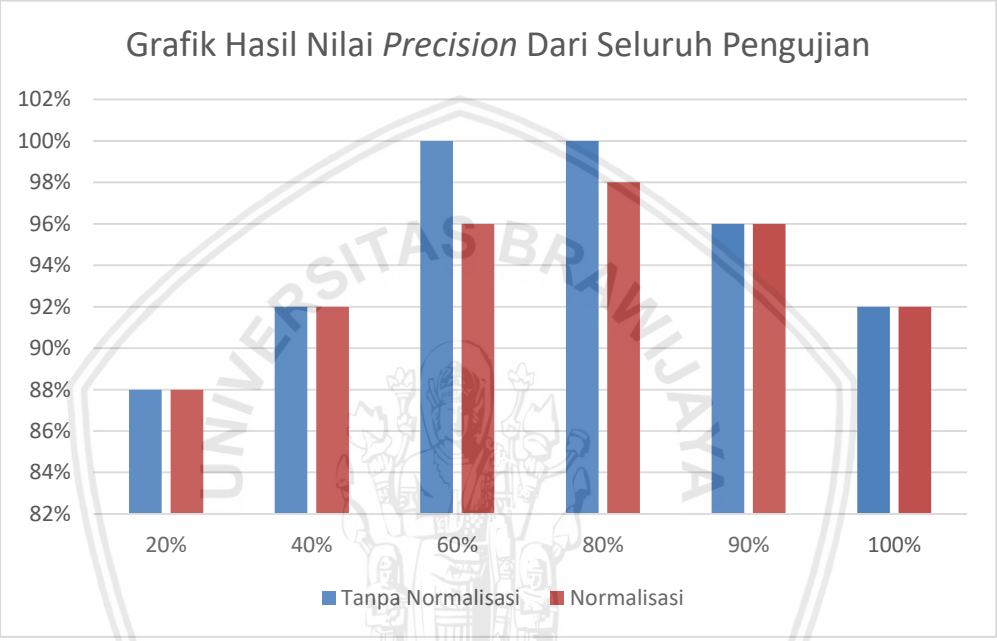



**Gambar 6.4 Hasil Nilai *Accuracy* Dari Seluruh Pengujian**

Gambar 6.4 menunjukkan semua hasil akurasi yang didapatkan dari seluruh proses pengujian. Akurasi menggambarkan nilai kedekatan antara *actual class* dengan *prediction class*. Pada Gambar 6.4 dapat disimpulkan bahwa *pre-processing* dengan normalisasi kata yang dipadukan dengan seleksi fitur *Information Gain* dengan *threshold* 80% menghasilkan akurasi tertinggi yaitu sebesar 98%. Hal itu dikarenakan pada pengujian tersebut pada tahapan proses *pre-processing* dilakukan satu tahapan tambahan yaitu normalisasi kata menggunakan kamus *formalizationDict* dan dilakukan proses seleksi fitur *Information Gain* yang berguna untuk menghitung kata-kata yang mencerminkan suatu kelas dengan nilai *Information Gain* tertinggi dan

mengabaikan kata yang tidak penting. Sementara pada proses pengujian yang lain, hasil akurasi yang didapatkan menghasilkan hasil akurasi yang lebih rendah karena dipengaruhi oleh salah satu faktor yaitu tidak adanya proses normalisasi kata pada *tweet* yang diuji sehingga membuat klasifikasi *Multinomial Naïve Bayes* tidak berjalan secara maksimal. Pada grafik *accuracy,* proses normalisasi yang dilakukan terbukti mampu meningkatkan nilai akurasi dan variasi pemilihan *threshold Information Gain* yang tepat juga akan memengaruhi tingkat akurasi yang semakin baik atau meningkat.

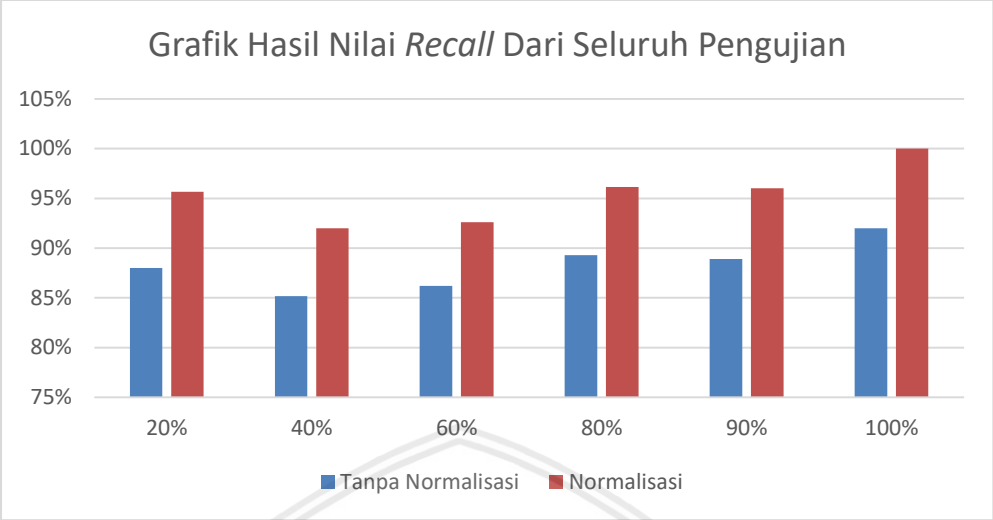
2. Hasil *precision*




**Gambar 6.5 Hasil Nilai *Precision* Dari Seluruh Pengujian**

Gambar 6.5 menunjukkan semua hasil *precision* yang didapatkan dari seluruh proses pengujian. *Precision* adalah tingkat ketepatan hasil klasifikasi yang dilakukan oleh sistem dari seluruh dokumen *tweet hate speech* yang ada. *Precision* dihitung dengan cara membagi kelas *hate speech* yang diklasifikasikan secara benar oleh sistem pada kelas *hate speech* dengan total data uji yang diklasifikasikan sebagai label kelas *hate speech*. Hasil terbaik yang didapatkan dari nilai *precision* pada pegujian ini adalah *Information Gain* tanpa normalisasi dengan *threshold* 60% dan 80% serta *Information Gain* dengan normalisasi dengan *threshold* sebesar 60% dan 80%. Hasil dari nilai *precision* yang didapatkan membuktikan bahwa seleksi fitur *Information Gain* dengan *threshold* yang optimal mampu membuat hasil *precision* menjadi optimal. Hal itu dikarenakan seleksi fitur *Information Gain* akan membantu untuk menyeleksi fitur-fitur atau kata-kata yang tidak mencerminkan suatu kelas dan akan menyimpan suatu kata yang dapat membantu sistem untuk melakukan prediksi label kelas yang benar karena memilih fitur atau kata dengan nilai yang tertinggi.

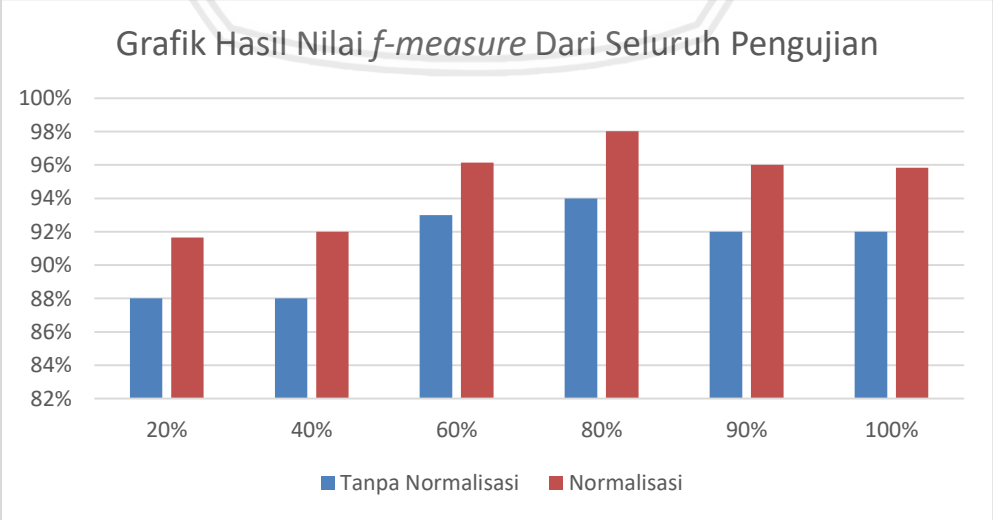
3. Hasil *recall*




**Gambar 6.6 Hasil Nilai *Recall* Dari Seluruh Pengujian**

Gambar 6.6 menunjukkan semua hasil *recall* yang didapatkan dari seluruh proses pengujian. *Recall* adalah nilai yang menggambarkan suatu sistem ketika berhasil menemukan sebuah informasi. Hasil yang dapat disimpulkan yaitu proses normalisasi pada *pre-processing* yang dipadukan dengan seleksi fitur *Information Gain* akan memiliki nilai *recall* yang lebih baik dibandingkan dengan proses tanpa normalisasi pada *pre-processing* yang dipadukan dengan seleksi fitur *Information Gain.* Nilai *recall* tertinggi yaitu sebesar 100% didapatkan pada saat melakukan klasifikasi menggunakan *Multinomial Naïve Bayes* dengan proses normalisasi pada tahap *pre-processing* yang dipadukan dengan seleksi fitur *Information Gain* dengan *threshold* 100% atau yang artinya seluruh fitur tidak ada yang diseleksi dan seluruh kata akan digunakan pada proses klasifikasi.

4. Hasil *f-measure*

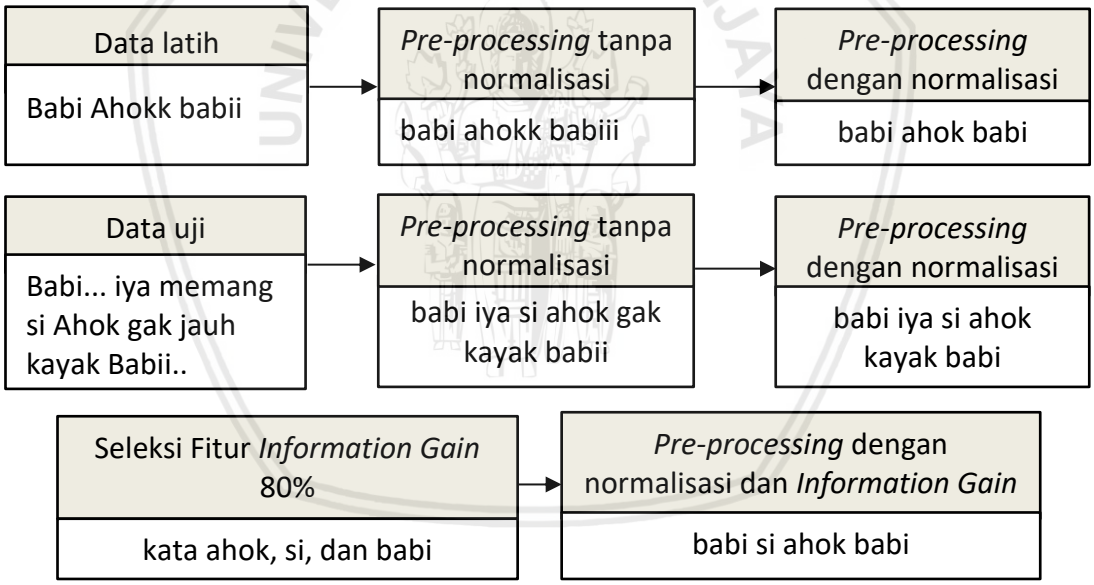



**Gambar 6.7 Hasil Nilai *f-measure* Dari Seluruh Pengujian**

Gambar 6.7 menunjukkan semua hasil *f-measure* yang didapatkan dari seluruh proses pengujian. *F-measure* adalah nilai yang menggambarkan seluruh kinerja sistem dalam melakukan proses klasifikasi. Nilai *f-measure* tertinggi didapatkan pada proses pengujian dengan menambahkan normalisasi kata pada tahap *pre-processing* yang dipadukan dengan seleksi fitur *Information Gain* dengan nilai *threshold* sebesar 80% yaitu mendapatkan nilai *f-measure* sebesar 98%. Hasil dari *f-measure* menunjukkan bahwa proses normalisasi yang dilakukan terbukti mampu meningkatkan nilai *f-measure* dan variasi pemilihan *threshold Information Gain* yang tepat juga akan memengaruhi tingkat nilai *f-measure* yang semakin baik atau meningkat.

## 6.3 Analisis Hasil

Berdasarkan hasil dari seluruh pengujian yang telah dilakukan, dapat disimpulkan bahwa hasil terbaik yang diperoleh adalah dengan melakukan seluruh kombinasi dari setiap proses yang ada, yaitu proses normalisasi pada tahap *pre-processing* dan dengan menggunakan seleksi fitur *Information Gain* dengan *threshold* sebesar 80%. Untuk dapat menjelaskannya lebih lanjut dapat dilihat pada Gambar 6.8.




**Gambar 6.8 Ilustrasi Proses Pengujian**

Dari Gambar 6.8 ilustrasi dibuat sebuah analisis antara lain:

1. Pada pengujian tanpa menambahkan proses normalisasi kata pada tahap *pre-processing*, data yang diklasifikasi memiliki hasil yang belum optimal dengan nilai *accuracy* sebesar 92%, nilai *precision* sebesar 92%, nilai *recall* sebesar 92%, dan nilai *f-measure* sebesar 92%. Pada kata yang tidak dilakukan proses normalisasi kata pada tahap *pre-processing* dalam melakukan proses klasifikasi masih memiliki kata-kata yang berbeda penulisannya namun maknanya sama seperti “babi”, “babiiiiiii”, “babiii”. Padahal seharusnya kata tersebut bisa diwakili dengan satu kata yaitu babi. Akibat tidak adanya proses

normalisasi kata pada tahap *pre-processing* membuat kata yang tidak penting menjadi lebih banyak dan kata yang memiliki makna sama tapi penulisannya berbeda membuat *tweet* pada data uji *hate speech* menjadi tidak banyak muncul pada *tweet* data latih *hate speech*. Pada pengujian tanpa adanya normalisasi kata pada tahap *pre-processing* menghasilkan akurasi yang belum optimal walaupun sudah mampu mengklasifikasikan kelas dengan cukup baik. Kemudian setelah dilakukan proses normalisasi kata pada tahap *pre-processing* mampu menghasilkan hasil yang lebih baik dibandingkan dengan pengujian tanpa menambahkan proses normalisasi kata pada tahap *pre-processing*. Hal itu dikarenakan setiap kata yang ada pada data uji akan dilakukan normalisasi kata dengan menggunakan kamus *formalizationDict*. Contohnya pada kata "babii" dinormalisasi menjadi "babi". Sehingga setiap kata pada *tweet* data uji *hate speech* akan lebih sering muncul pada *tweet* data latih *hate speech* dibandingkan proses pengujian tanpa normalisasi kata. Namun pada proses normalisasi masih memiliki banyak kekurangan antara lain adalah kesalahan kamus dalam melakukan normalisasi kata seperti kata "agama!" menjadi "agamai" dan masih banyak kata-kata yang tidak penting atau kata yang tidak mencerminkan kelas *hate speech* ataupun kelas *non hate speech* yang masih dihitung pada proses klasifikasi *Multinomial Naive Bayes* karena belum adanya seleksi fitur. Kesimpulan yang dapat diambil pada pengujian ini bahwa normalisasi berpengaruh untuk meningkatkan akurasi dalam melakukan pengklasifikasian *hate speech,* karena pengujian yang dilakukan proses normalisasi pada tahap *pre-processing*, data yang diklasifikasi memiliki hasil akurasi yang meningkat atau lebih baik dengan nilai *accuracy* sebesar 96%, nilai *precision* sebesar 92%*,* nilai *recall* sebesar 100%*,* dan nilai *f-measure* sebesar 96%.

2. Pada pengujian dengan adanya proses normalisasi kata pada tahap *pre-processing* yang dipadukan dengan menggunakan seleksi fitur *Information Gain* dengan *threshold* 80% mampu mendapatkan hasil terbaik dengan nilai *accuracy* sebesar 98%, nilai *precision* sebesar 100%*,* nilai *recall* sebesar 96%*,* dan nilai *f-measure* sebesar 98%. Hasil ini mampu memberikan nilai akurasi terbaik dari seluruh pengujian yang telah dilakukan. Pada pengujian ini dilakukan seluruh kombinasi dari setiap proses yang ada mulai dari proses *pre-processing* dengan normalisasi yaitu perbaikan kata tidak baku menjadi baku menggunakan kamus *formalizationDict*. Kemudian dilanjutkan dengan melakukan seleksi fitur *Information Gain* dengan mencari *threshold* terbaik yaitu 80%. Proses seleksi fitur *Information Gain* mampu membantu sistem untuk menyeleksi fitur-fitur atau kata-kata yang akan digunakan untuk perhitungan algoritme *Multinomial Naïve Bayes* yang mana setiap fitur pasti memiliki nilai *Information Gain* yang tinggi dan mampu mencerminkan suatu kelas tertentu. Setiap kata yang ada pada *tweet* data uji *hate speech* akan lebih sering muncul pada *tweet* data latih *hate speech* yang akan memengaruhi proses klasifikasi menjadi lebih baik dan mendapatkan hasil akurasi yang lebih optimal.

# BAB 7
# PENUTUP

## 7.1 Kesimpulan

Berdasarkan analisis dan hasil pengujian yang telah dilakukan, maka dapat disimpulkan sebagai berikut:

1. Pada tahap *pre-processing* dengan normalisasi kata atau perbaikan kata tidak baku terhadap hasil pengklasifikasian *hate speech* akan memberikan hasil akurasi yang lebih optimal daripada proses *pre-processing* tanpa normalisasi, yaitu yang semula nilai akurasi tanpa normalisasi sebesar 92% mengalami peningkatan menjadi sebesar 96%. Kemudian jika ditambahkan dengan seleksi fitur menggunakan *Information Gain* dengan nilai *threshold* 80% maka nilai akurasi akan mencapai angka maksimal yaitu sebesar 94% untuk *pre-processing* tanpa normalisasi dengan seleksi fitur *Information Gain* dan sebesar 98% untuk *pre-processing* dengan normalisasi yang dipadukan dengan seleksi fitur *Information Gain.*
2. Berdasarkan pengujian yang telah dilakukan, diperoleh hasil akurasi terbaik dari normalisasi *pre-processing* data Twitter menggunakan metode Naive Bayes dan seleksi fitur *Information Gain* untuk klasifikasi *hate speech* berbahasa Indonesia adalah sebesar 98%. Hasil terbaik dari seluruh pengujian diperoleh dari pengujian normalisasi kata pada tahap *pre-processing* yang dipadukan dengan seleksi fitur *Information Gain* dengan *threshold* 80%. Pada pengujian ini didapatkan hasil *accuracy* sebesar 98%, nilai *precision* sebesar 100%*,* nilai *recall* sebesar 96%*,* dan nilai *f-measure* sebesar 98%. Hal ini dikarenakan proses normalisasi kata pada tahap *pre-processing* membuat *tweet* data uji *hate speech* menjadi lebih sering muncul pada *tweet* data uji *hate speech* karena telah dilakukan perbaikan kata tidak baku menjadi baku, seperti kata "*sm*", "*sma*", "*samaa", "samaaa"* diperbaiki menjadi satu kata yaitu "sama". Selain itu seleksi fitur *Information Gain* akan mengukur berapa banyak informasi kehadiran dan ketidakhadiran dari suatu kata yang berperan untuk membuat keputusan klasifikasi yang benar dalam suatu kelas. *Information Gain* akan memberi peringkat pada kata-kata penting dari hasil reduksi fitur. Hasil reduksi fitur dari proses *Information Gain* adalah kata-kata penting yang bersifat informatif dan memiliki nilai yang tinggi untuk mencerminkan suatu kelas *hate speech* maupun *non hate speech.* Seleksi fitur *Information Gain* akan mengoptimalkan hasil klasifikasi dalam memprediksi label kelas yang benar yaitu *hate speech* dan *non hate speech*.

## 7.2 Saran

Berdasarkan analisis dan hasil pengujian yang sudah dilakukan sebelumnya, tentu saja masih memiliki banyak kekurangan yang perlu diperbaiki untuk perkembangan penelitian selanjutnya yang berhubungan dengan topik *text*

*mining*, klasifikasi *hate speech* atau penggunaan metode Naive Bayes dengan normalisasi kata dan juga seleksi fitur menggunakan *Information Gain,* yaitu:

1. Pada normalisasi kata, penulis masih menggunakan kamus dari Pujangga *Indonesian Natural Language Processing* REST API yaitu *formalizationDict* yang didasarkan pada perbaikan kata-kata yang alay ataupun kata yang disingkat-singkat. Karena formalizationDict pada Pujangga *Indonesian Natural Language Processing* REST API tidak bisa ditambahkan atau dikurangi, maka masih banyak kekurangan yang ada pada proses normalisasi pada formalizationDict, sehingga mungkin dapat ditambahkan kosakata lain yang tidak ada pada formalizationDict sehingga kamus kata baku bisa dibuat sendiri secara manual.
2. Pada penelitian selanjutnya pada saat melakukan evaluasi sistem dapat menggunakan *cross validation*. *Cross validation* adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritme dimana data dipisahkan menjadi dua *subset* yaitu data proses pembelajaran dan data validasi atau evaluasi. *Cross validation* berguna untuk menghindari kasus *overfitting* dan *underfitting* dalam *machine learning*. Selain itu pada penelitian selanjutnya juga dapat menggunakan fitur lain, selain fitur *bag of words* yang digunakan oleh penulis, seperti fitur berbasis *lexicon*, pembobotan nilai pada *emoticon*, atau membuat kamus kata *hate speech* dan *non hate speech*. Kemudian untuk proses seleksi fitur dapat ditambahkan seleksi fitur lain yang dapat dipadukan dengan *Information Gain* dalam kasus klasifikasi *hate speech* berbahasa Indonesia pada Twitter. Selain itu pada penelitian selanjutnya juga bisa memakai metode lain seperti *neural network, Support Vector Machine* untuk membandingkan nilai *accuracy*, *precision, recall,* dan *f-measure* dengan metode Naïve Bayes yang dilakukan oleh penulis.

# DAFTAR REFERENSI


Aggarwal, C. C., 2015. *Data Classification Algorithms and Applications.* New York: CRC Press.

Agarwal, A. et al., 2014. *Sentiment Analysis of Twitter Data. Department of Computer Science Columbia University*. Available at: http://www.cs.columbia.edu/~julia/papers/Agarwaletal11.pdf.

Alfina, I., Mulia, R., Fanany, M. I., & Ekanata, Y., 2017. *Hate Speech Detection in the Indonesian Language: A Dataset and Preliminary Study*. 9th International Conference on Advanced Computer Science and Information Systems 2017 (ICACSIS).

Andilala, 2016. *Movie Review Sentimen Analisis dengan Metode Naïve Bayes Base on Feature Selection*. Jurnal Pseudocode, III, hal.1–9.

Aziz, A.T.A., 2013. *Sistem Pengklasifikasian Entitas Pada Pesan Twitter Menggunakan Ekspresi Regular dan Naïve Bayes.*

Benevenuto, F., Magno, G., Rodrigues, T. & Almeida, V., 2010. *Detecting Spammers on Twitter*. *In Collaboration, electronic messaging, anti-abuse and spam conference*, 6, p.pp.1-9.

Buntoro. A., Adji, T. B., & Purnamasari, A. E., 2014. *Sentiment Analysis Twitter dengan Kombinasi Lexicon Based dan Double Propagation*. CTIEE. Halaman 39-43.

Burnap, P., & Williams, M. L., 2014. Hate Speech, *Machine Classification and Statistical Modelling of Information Flows on Twitter*: *Interpretation and Communication for Policy Decision Making*.

Davidson, T., Warmsley, D., Macy, M., & Weber, I., 2017. *Automated Hate Speech Detection and the Problem of Offensive Language*. Proceedings of the Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017*)*.

Destuardi dan Surya, S., 2009. *Klasifikasi Emosi Untuk Teks Bahasa Indonesia Menggunakan Metode Naive Bayes.* Teknik, Institut Teknologi Sepuluh Nopember, Surabaya.

Direktorat Tindak Pidana Siber Bareskrim Polri, 2017.

Feldman, R., & Sanger, J., 2007. *The Text Mining Handbook Advanced Approaches in Analyzing Unstructured Data.* USA: Cambridge University Press.

Fitri Cahyanti, A., Saptono, R. and Widya Sihwi, S., 2016. *Penentuan Model Terbaik pada Metode Naive Bayes Classifier dalam Menentukan Status Gizi Balita dengan Mempertimbangkan Independensi Parameter*. Jurnal Teknologi & Informasi ITSmart, [online] 4(1), p.28. Available at:

<https://jurnal.uns.ac.id/index.php?journal=itsmart&page=article&op=view&path[]=1754>.

Hadna, N. et al., 2016. *Studi Literatur tentang perbandingan metode proses analisis sentimen di twitter.* Fakultas Teknik, Universitas Gadjah Mada.

Hall, M., 2006. *A Decision Tree-Based Attribute Weighting Filter for Naive Bayes*. Research and Development in Intelligent Systems XXIII (hal. pp 59-70). New Zealand: Department of Computer Science, University of Waikato.

Ian H. Witten, E. F., 2011. *Data Mining: Practical Machine Learning Tools and Techniques.*

Kiilu, K.K., Okeyo, G., Rimiru, R. dan Ogada, K., 2018. *Using Naïve Bayes Algorithm in detection of Hate Tweets*. International Journal of Scientific and Research Publications (IJSRP), [daring] 8(3), hal.99–107. Tersedia pada: <http://www.ijsrp.org/research-paper-0318.php?rp=P757259>.

Nockleby, J.T., 2000. *Hate Speech. In Encyclopedia of the American Constitution*. Second ed. New York: Macmillan, 2000.

Rozzaqi, A.R., 2015. *Naive Bayes dan Filtering Feature Selection Information Gain untuk Prediksi Ketepatan Kelulusan Mahasiswa*. Jurnal Informatika UPGRIS, 1, hal.30–41.

Saputra, N., Adji, T.B. and Permanasari, A.E., 2015. *Analisis Sentimen Data Presiden Jokowi Dengan Preprocessing Normalisasi dan Stemming Menggunakan Metode Naive Bayes dan SVM*. Jurnal Dinamika Informatika, Volume 5 N, pp.1–12.

Sona Taheri, M. M., 2013. L*earning The Naive Bayes Classifier With Optimization Models*. International Journal of Applied Mathematics and Computer Science, (hal. 787-795 ).

Triawati, C., 2009. *Metode Pembobotan Statistical Concept Based untuk Klastering dan Kategorisasi Dokumen Berbahasa Indonesia.* Bandung: Institut Teknologi Bandung.

Uysal, A.K. and Gunal, S., 2012. *A novel probabilistic feature selection method for text classification*. Knowledge-Based Systems, [online] 36, pp.226–235. Available at: <http://dx.doi.org/10.1016/j.knosys.2012.06.005>.

Wu, X., Kumar, V., Ross, Q.J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand, D.J. and Steinberg, D., 2008. *Top 10 algorithms in data mining*. Knowledge and Information Systems.