

**KLASIFIKASI *TWEET* BERBAHASA INDONESIA BERISI UJARAN  
KEBENCIAN MENGGUNAKAN METODE *IMPROVED K-  
NEAREST NEIGHBOR* DENGAN PEMBOBOTAN BM25F**

**SKRIPSI**

Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun oleh:  
Nurdifa Febrianti  
NIM: 165150200111065



PROGRAM STUDI TEKNIK INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS BRAWIJAYA  
MALANG  
2020

## PENGESAHAN

KLASIFIKASI TWEET BERBAHASA INDONESIA BERISI UJARAN KEBENCIAN  
MENGUNAKAN METODE IMPROVED K-NEAREST NEIGHBOR DENGAN  
PEMBOBOTAN BM25F


### SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun Oleh :  
Nurdifa Febrianti  
NIM: 165150200111065

Skripsi ini telah diuji dan dinyatakan lulus pada  
7 Januari 2020  
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I



Indriati, S.T., M.Kom.  
NIP: 19831013 201504 2 002

Dosen Pembimbing II



M. Tanzil Furqon, S.Kom., M.ComSc.  
NIP: 19820930 200801 1 004

Mengetahui

Ketua Jurusan Teknik Informatika



Tri Astuti 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 referensi.

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, 7 Januari 2020



Nurdifa Febrianti

NIM: 165150200111065

## PRAKATA

Puji Syukur penulis panjatkan Kehadirat Allah SWT, atas limpahan rezeki, rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan dengan baik penelitian akhir atau skripsi yang berjudul “Klasifikasi Tweet Berbahasa Indonesia Berisi Ujaran Kebencian Menggunakan Metode *Improved K-Nearest Neighbor* dengan Pembobotan BM25F”.

Penelitian ini disusun dalam rangka memenuhi persyaratan kelulusan dalam kurikulum di Fakultas Ilmu Komputer (FILKOM) Universitas Brawijaya. Penulis ingin mengucapkan terima kasih kepada semua pihak yang telah memberi dukungan, bantuan dan panduannya untuk menyelesaikan skripsi ini, antara lain:

1. Ibu Indriati, S.T., M.Kom. dan Bapak M. Tanzil Furqon, S.Kom., M.Comp.Sc. sebagai dosen pembimbing I dan dosen pembimbing II penulis selama pelaksanaan skripsi.
2. Bapak Wayan Firdaus Mahmudi, S.Kom., M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Agus Wahyu Widodo, S.T., M.Cs., selaku Ketua Program Studi Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya dan selaku Dosen Pembimbing Akademik penulis selama masa perkuliahan.
5. Seluruh bapak dan ibu dosen pengajar yang telah mendidik penulis selama perkuliahan.
6. Kedua orang tua penulis, Firsta Nur Lifa dan Dwijanto Rahardjo, kedua saudara penulis, Alifia Afianti dan Adinda Rahmatanti, beserta segenap keluarga besar yang telah memberikan do’a dan dukungan kepada penulis.
7. Sahabat-sahabat penulis selama 4 tahun perkuliahan, diantaranya Etika Suzerein, Rumzil Innayatul Laily, Rizha A. A. S., Ira Meylan, Alifiya Rizki Laily, Shofya Rahman, Annisa Yasmintya, Dimas Candra Sugiarto, Fahrur Rachman, Moh. Anggit, Bayu Oktaviansyah, dan lainnya.
8. Teman-teman Teknik Informatika 2016 khususnya Puteri Aulia Indrasti, Candra Ardiansyah, Nabila Divanadia Luckyana, Nyoman Putra Utama, dan segenap keluarga Paduan Suara BIOS Fakultas Ilmu Komputer yang tidak bisa disebutkan satu persatu.
9. Kakak tingkat penulis yang sudah memberikan banyak saran bermanfaat bagi penulis selama perkuliahan maupun saat pengerjaan skripsi, Asa Kartika, Desy Andriani, Rahmat Faizal, Ardhimas Ilham Bagus P., Indri Dewi Onantya, Aang Muammar Zein, Delarta Tok Adin dan lainnya.

10. Seluruh pihak yang telah mendukung dan memberi bantuan saat penulisan skripsi ini baik secara langsung maupun tidak langsung.

Penulis sangat menyadari bahwa penelitian ini masih jauh dari kata sempurna dan memiliki banyak kekurangan, untuk itu penulis mengharapkan adanya kritik dan saran yang bersifat membangun. Semoga penelitian ini bermanfaat bagi pembaca maupun peneliti lainnya di masa yang akan datang.

Malang, 7 Januari 2020

Penulis

difadifa@student.ub.ac.id

## ABSTRAK

**Nurdifa Febrianti, Klasifikasi *Tweet* Berbahasa Indonesia Berisi Ujaran Kebencian Menggunakan Metode *Improved K-Nearest Neighbor* dengan Pembobotan BM25F**

**Pembimbing: Indriati, S.T., M.Kom. dan Muhammad Tanzil Furqon S.Kom., M.CompSc.**

Ujaran kebencian adalah tindakan kebencian verbal yang menargetkan sekelompok orang atau bagian dari komunitas tertentu berdasarkan ras, warna kulit, agama/keyakinan, jenis kelamin, kemampuan fisik, orientasi seksual, garis keturunan, negara dan suku asal, atau bahkan pandangan politik. Di Indonesia, ujaran kebencian semakin banyak ditemukan, terutama pada media sosial berbasis utama teks seperti Twitter. Sehingga menginspirasi ditulisnya penelitian ini, untuk mengidentifikasi ujaran kebencian di Twitter dengan klasifikasi *tweet*, khususnya yang berbahasa Indonesia. Penulis memilih menggunakan *Improved K-Nearest Neighbor* dengan menggunakan pembobotan kata BM25F, yaitu pembobotan yang mempertimbangkan *field/stream* dalam dokumen. Sehingga *tweet* yang dipilih sebagai dokumen latih dan dokumen uji penelitian, terdiri atas 2 *stream*, yaitu *tweet* dan *hashtag* atau tagar (tanda pagar). Dilakukan pengujian *K-Fold Cross Validation* (dengan  $K = 5$ ) terhadap parameter  $k$  untuk klasifikasi IKNN, serta parameter  $b_s$ ,  $v_s$ , dan  $k_1$  untuk pembobotan BM25F, dengan 400 dokumen latih dan 100 dokumen uji. Hasil pengujian menunjukkan bahwa penentuan nilai bobot *stream* pada BM25F cukup mempengaruhi hasil klasifikasi IKNN. Sedangkan hasil akhir terbaik untuk *F-Measure*, *Accuracy*, *Precision*, dan *Recall* dari rerata *5-Fold Cross Validation* yang didapatkan adalah 79,77%, 68,80%, 68,80%, dan 89,92% dengan  $k = 70$ ,  $b_s = 0,6$ ,  $v_1 = 2$ ,  $v_2 = 5$  dan  $k_1 = 2$  sebagai nilai terbaik untuk masing-masing parameternya.

Kata kunci: Ujaran Kebencian, *Tweet*, *Hashtag*, *Improved K-Nearest Neighbor*, BM25F

## ABSTRACT

**Nurdifa Febrianti, Classification of Indonesian Tweets Containing Hate Speech Using the *Improved K-Nearest Neighbor* Method with BM25F Weighting**

**Supervisors: Indriati, S.T., M.Kom. and Muhammad Tanzil Furqon, S.Kom., M.CompSc.**

*Hate speech is a verbal hatred act that targets a group of people or parts of a particular community based on race, color, religion/beliefs, gender, physical ability, sexual orientation, lineage, country and ethnic origin, or even political views. In Indonesia, hate speech is increasingly found, especially on text-based social media such as Twitter. So that inspired the writing of this research, to identify hate speech on Twitter with the classification of tweets, especially those in Indonesian. The author chooses to use Improved K-Nearest Neighbor by using the BM25F term weighting, which is a weighting that considers the fields / streams in the document. So the tweet chosen as a training and testing document, consists of 2 streams, the tweet and the hashtag. K-Fold Cross Validation testing (with  $K = 5$ ) was performed on the parameter  $k$  for IKNN classification,  $b_s$ ,  $v_s$ , and  $k_1$  for BM25F weighting, with 400 training documents and 100 test documents. The test results show that the determination of stream weight values on BM25F sufficiently influences the results of the IKNN classification. Meanwhile the best final results for the F-Measure, Accuracy, Precision, and Recall of the average 5-Fold Cross Validation obtained were 79.77%, 68.80%, 68.80%, and 89.92% with  $k = 70$ ,  $b_s = 0.6$ ,  $v_1 = 2$ ,  $v_2 = 5$  and  $k_1 = 2$  as the best value for each parameter.*

**Keywords:** Hate Speech, Tweet, Hashtag, Improved K-Nearest Neighbor, BM25F

## DAFTAR ISI

PENGESAHAN .....	ii
PERNYATAAN ORISINALITAS .....	iii
PRAKATA.....	iv
ABSTRAK.....	vi
ABSTRACT .....	vii
DAFTAR ISI .....	viii
DAFTAR TABEL.....	xii
DAFTAR GAMBAR .....	xiv
DAFTAR LAMPIRAN .....	xv
BAB 1 PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan masalah .....	3
1.3 Tujuan .....	3
1.4 Manfaat.....	3
1.5 Batasan masalah .....	3
1.6 Sistematika pembahasan .....	4
BAB 2 LANDASAN KEPUSTAKAAN .....	5
2.1 Kajian Pustaka .....	5
2.2 Ujaran Kebencian dan Penyebarannya di Media Sosial .....	7
2.3 <i>Preprocessing</i> Teks.....	8
2.4 <i>Cleaning</i> .....	8
2.5 <i>Case folding</i> .....	8
2.6 Tokenisasi.....	9
2.7 <i>Filtering</i> .....	9
2.8 <i>Stemming</i> .....	10
2.9 BM25F .....	10
2.10 Metode Klasifikasi <i>Improved K-Nearest Neighbor</i> (IKNN) .....	12
2.11 <i>K-Fold Cross Validation</i> .....	13
2.12 <i>Confusion Matrix</i> .....	14
2.13 <i>Accuracy</i> .....	14



2.14 <i>Precision</i> .....	15
2.15 <i>Recall</i> .....	15
2.16 <i>F-measure</i> .....	15
BAB 3 METODOLOGI .....	16
3.1 Tipe Penelitian .....	16
3.2 Strategi Penelitian .....	16
3.3 Partisipan Penelitian .....	16
3.4 Lokasi Penelitian .....	16
3.5 Teknik Pengumpulan Data .....	16
3.6 Peralatan Pendukung .....	17
3.7 Perancangan Sistem .....	18
3.8 Implementasi Sistem .....	18
3.9 Teknik Analisis Data .....	18
3.10 Pengambilan Kesimpulan .....	19
BAB 4 PERANCANGAN .....	20
4.1 Gambaran Umum Sistem .....	20
4.2 Diagram Alir Sistem .....	20
4.3 Diagram Alir Tahap Pelatihan .....	21
4.3.1 Diagram Alir Preprocessing .....	21
4.3.2 Diagram Alir <i>Cleaning</i> .....	25
4.3.3 Diagram Alir Tokenisasi .....	25
4.3.4 Diagram Alir <i>Case folding</i> .....	26
4.3.5 Diagram Alir <i>Split By Uppercase</i> .....	27
4.3.6 Diagram Alir Filtering .....	28
4.3.7 Diagram Alir <i>Stemming</i> .....	29
4.3.8 Diagram Alir <i>tf</i> Dokumen Latih .....	30
4.3.9 Diagram Alir <i>df</i> .....	32
4.4 Diagram Alir Tahap Pengujian .....	33
4.4.1 Diagram Alir TF Dokumen Uji .....	33
4.4.2 Diagram Alir <i>df</i> Dokumen Uji .....	35
4.4.3 Diagram Alir BM25F .....	35
4.4.4 Diagram Alir wIDF .....	36

4.4.5 Diagram Alir <i>sls</i> dan <i>avsls</i> .....	37
4.4.6 Diagram Alir <i>Bs</i> .....	38
4.4.7 Diagram Alir <i>tfi</i> .....	39
4.4.8 Diagram Alir BM25F .....	40
4.4.9 Diagram Alir <i>Improved K-Nearest Neighbor</i> .....	42
4.5 Manualisasi .....	45
4.5.1 <i>Preprocessing</i> .....	45
4.5.2 Perhitungan <i>tf</i> Dokumen Latih.....	52
4.5.3 Perhitungan <i>tf</i> Dokumen Uji .....	54
4.5.4 Perhitungan <i>df</i> dan <i>wIDF</i> .....	55
4.5.5 Perhitungan <i>sls</i> .....	56
4.5.6 Perhitungan <i>avsls</i> .....	56
4.5.7 Perhitungan <i>Bs</i> .....	56
4.5.8 Perhitungan <i>tfi</i> .....	57
4.5.9 Perhitungan BM25F Dokumen Uji .....	58
4.5.10 Perhitungan IKNN Dokumen Uji .....	59
4.6 Rancangan Pengujian.....	61
BAB 5 Implementasi .....	63
5.1 Lingkungan Penerapan .....	63
5.1.1 Lingkungan Penerapan Perangkat Lunak.....	63
5.1.2 Lingkungan Penerapan Perangkat Keras.....	63
5.2 Penerapan Algoritme .....	63
5.2.1 <i>Preprocessing</i> .....	64
5.2.2 <i>Cleaning</i> .....	65
5.2.3 Tokenisasi.....	65
5.2.4 <i>Case Folding</i> .....	65
5.2.5 <i>Split By Uppercase</i> .....	66
5.2.6 <i>Filtering</i> .....	66
5.2.7 <i>Stemming</i> .....	67
5.2.8 Perhitungan <i>tf</i> .....	67
5.2.9 Perhitungan <i>df</i> .....	68
5.2.10 Perhitungan <i>wIDF</i> .....	69

5.2.11 Perhitungan $s_{ls}$ .....	70
5.2.12 Perhitungan $B_s$ .....	70
5.2.13 Perhitungan $t_{fi}$ .....	71
5.2.14 Perhitungan BM25F .....	71
5.2.15 Perhitungan IKNN .....	72
BAB 6 PENGUJIAN DAN ANALISIS .....	74
6.1 Pengujian Sistem .....	74
6.1.1 Pengujian <i>5-Fold Cross Validation</i> pada Variabel $k$ .....	74
6.1.2 Pengujian <i>5-Fold Cross Validation</i> pada Variabel $bs$ .....	77
6.1.3 Pengujian <i>5-Fold Cross Validation</i> pada Variabel $vs$ .....	79
6.1.4 Pengujian <i>5-Fold Cross Validation</i> pada Variabel $k_1$ .....	82
6.2 Analisis Pengujian .....	85
6.2.1 Analisis Pengujian <i>5-Fold Cross Validation</i> Nilai $k$ .....	85
6.2.2 Analisis Pengujian <i>5-Fold Cross Validation</i> Nilai $bs$ .....	86
6.2.3 Analisis Pengujian <i>5-Fold Cross Validation</i> Nilai $vs$ .....	87
6.2.4 Analisis Pengujian <i>5-Fold Cross Validation</i> Nilai $k_1$ .....	88
BAB 7 Penutup .....	90
7.1 Kesimpulan .....	90
7.2 Saran .....	90
DAFTAR REFERENSI .....	91
LAMPIRAN A DATASET .....	94

## DAFTAR TABEL

Tabel 2.1 <i>Cleaning</i> .....	8
Tabel 2.2 <i>Case folding</i> .....	9
Tabel 2.3 <i>Tokenisasi</i> .....	9
Tabel 2.4 <i>Filtering</i> .....	10
Tabel 2.5 <i>Stemming</i> .....	10
Tabel 2.6 <i>Confusion Matrix</i> .....	14
Tabel 4.1 Data Latih .....	45
Tabel 4.2 Data Uji .....	45
Tabel 4.3 Hasil <i>Cleaning</i> Data Latih .....	46
Tabel 4.4 Hasil <i>Cleaning</i> Data Uji .....	46
Tabel 4.5 Hasil <i>Tokenisasi</i> Data Latih .....	47
Tabel 4.6 Hasil <i>Tokenisasi</i> Data Uji .....	48
Tabel 4.7 Hasil <i>Case folding</i> Data Latih .....	48
Tabel 4.8 Hasil <i>Case folding</i> Data Uji .....	49
Tabel 4.9 Hasil <i>Split By Uppercase</i> Data Latih .....	49
Tabel 4.10 Hasil <i>Split By Uppercase</i> Data Uji .....	50
Tabel 4.11 Hasil <i>Filtering</i> Data Latih .....	51
Tabel 4.12 Hasil <i>Filtering</i> Data Uji .....	51
Tabel 4.13 Hasil <i>Stemming</i> Data Latih .....	52
Tabel 4.14 Hasil <i>Stemming</i> Data Uji .....	52
Tabel 4.15 Hasil Perhitungan <i>tf</i> Dokumen Latih .....	52
Tabel 4.16 Hasil Perhitungan <i>tf</i> Dokumen Uji .....	54
Tabel 4.17 Hasil Perhitungan <i>df</i> dan <i>wIDF</i> .....	55
Tabel 4.18 Hasil Perhitungan <i>sl<sub>s</sub></i> .....	56
Tabel 4.19 Hasil Perhitungan <i>avsls</i> .....	56
Tabel 4.20 Hasil Perhitugan <i>B<sub>s</sub></i> .....	56
Tabel 4.21 Hasil Perhitungan <i>t<sub>fi</sub></i> .....	57
Tabel 4.22 Hasil Perhitungan BM25F .....	58
Tabel 4.23 Hasil Perhitungan <i>n</i> .....	59
Tabel 4.24 Probabilitas Kelas Dokumen Latih .....	60

Tabel 4.25 Tabel Rancangan Pengujian $k$ .....	61
Tabel 4.26 Rancangan Pengujian $bs$ .....	62
Tabel 4.27 Rancangan Pengujian $vs$ .....	62
Tabel 4.28 Rancangan Pengujian $k1$ .....	62
Tabel 6.1 <i>Fold</i> ke-1 Variabel $k$ .....	74
Tabel 6.2 <i>Fold</i> ke-2 Variabel $k$ .....	75
Tabel 6.3 <i>Fold</i> ke-3 Variabel $k$ .....	75
Tabel 6.4 <i>Fold</i> ke-4 Variabel $k$ .....	76
Tabel 6.5 <i>Fold</i> ke-5 Variabel $k$ .....	76
Tabel 6.6 Rata-rata Hasil <i>5-Fold Cross Validation</i> untuk nilai $k$ .....	77
Tabel 6.7 <i>Fold</i> ke-1 Variabel $bs$ .....	77
Tabel 6.8 <i>Fold</i> ke-2 Variabel $bs$ .....	78
Tabel 6.9 <i>Fold</i> ke-3 Variabel $bs$ .....	78
Tabel 6.10 <i>Fold</i> ke-4 Variabel $bs$ .....	78
Tabel 6.11 <i>Fold</i> ke-5 Variabel $bs$ .....	78
Tabel 6.12 Rata-rata Hasil <i>5-Fold Cross Validation</i> untuk nilai $bs$ .....	79
Tabel 6.13 <i>Fold</i> ke-1 Variabel $vs$ .....	79
Tabel 6.14 <i>Fold</i> ke-2 Variabel $vs$ .....	80
Tabel 6.15 <i>Fold</i> ke-3 Variabel $vs$ .....	80
Tabel 6.16 <i>Fold</i> ke-4 Variabel $vs$ .....	81
Tabel 6.17 <i>Fold</i> ke-5 Variabel $vs$ .....	81
Tabel 6.18 Rata-rata Hasil <i>5-Fold Cross Validation</i> untuk nilai $vs$ .....	81
Tabel 6.19 <i>Fold</i> ke-1 Variabel $k1$ .....	82
Tabel 6.20 <i>Fold</i> ke-2 Variabel $k1$ .....	83
Tabel 6.21 <i>Fold</i> ke-3 Variabel $k1$ .....	83
Tabel 6.22 <i>Fold</i> ke-4 Variabel $k1$ .....	83
Tabel 6.23 <i>Fold</i> ke-5 Variabel $k1$ .....	84
Tabel 6.24 Rata-rata Hasil <i>5-Fold Cross Validation</i> untuk nilai $k1$ .....	84
Tabel 6.25 Dokumen <i>tweet</i> bersifat ambigu .....	85

## DAFTAR GAMBAR

Gambar 4.1 Diagram Alir Sistem, (a) Diagram Tahap Pelatihan, (b) Diagram Tahap Pengujian.....	21
Gambar 4.2 Diagram Alir <i>Preprocessing</i> Dokumen Latih.....	22
Gambar 4.3 Diagram Alir <i>Preprocessing</i> Dokumen Uji .....	23
Gambar 4.4 Diagram Alir <i>Cleaning</i> .....	25
Gambar 4.5 Diagram Alir Tokenisasi .....	25
Gambar 4.6 Diagram Alir <i>Case Folding</i> .....	26
Gambar 4.7 Diagram Alir <i>Split By Uppercase</i> .....	27
Gambar 4.8 Diagram Alir <i>Filtering</i> .....	28
Gambar 4.9 Diagram Alir <i>Stemming</i> .....	29
Gambar 4.10 Diagram Alir <i>Stemming</i> .....	30
Gambar 4.11 Diagram Alir <i>tf</i> Dokumen Latih .....	30
Gambar 4.12 Diagram Alir <i>df</i> Dokumen Latih.....	32
Gambar 4.13 Diagram Alir <i>tf</i> Dokumen Uji .....	33
Gambar 4.14 Diagram Alir BM25F .....	36
Gambar 4.15 Diagram Alir wIDF.....	36
Gambar 4.16 Diagram Alir <i>avsls</i> dan <i>sls</i> .....	37
Gambar 4.17 Diagram Alir <i>Bs</i> .....	38
Gambar 4.18 Diagram Alir <i>tfi</i> .....	39
Gambar 4.19 Diagram Alir wBM25F .....	40
Gambar 4.20 Diagram Alir BM25F .....	42
Gambar 4.21 Diagram Alir Sorting berdasarkan nilai <i>k</i> baru ( <i>n</i> ) .....	43
Gambar 4.22 Diagram Alir Hitung IKNN.....	44
Gambar 6.1 Grafik Rata-rata 5-Fold Cross Validation Nilai <i>k</i> .....	86
Gambar 6.2 Grafik Rata-rata 5-Fold Cross Validation Nilai <i>bs</i> .....	87
Gambar 6.3 Grafik Rata-rata 5-Fold Cross Validation Nilai <i>vs</i> .....	88
Gambar 6.4 Grafik Rata-rata 5-Fold Cross Validation Nilai <i>k1</i> .....	89

## DAFTAR LAMPIRAN

LAMPIRAN A DATASET .....	94
--------------------------	----

## BAB 1 PENDAHULUAN

Pada bab Pendahuluan, penulis akan menjelaskan latar belakang dari dilakukannya penelitian tentang Klasifikasi *Tweet* Berbahasa Indonesia berisi Ujaran Kebencian Menggunakan Metode *Improved K-Nearest Neighbor* dengan BM25F, diikuti dengan perumusan masalah yang muncul, tujuan yang ingin dicapai, manfaat yang diperoleh dari penelitian, batasan masalah serta sistematika pembahasan.

### 1.1 Latar Belakang

*Hate Crime* atau kejahatan yang terdorong kebencian adalah tindakan komunikatif, sering disebabkan oleh serangan tertentu yang memicu adanya niat pembalasan dari kelompok yang merupakan target serangan tersebut, terhadap golongan yang memiliki karakteristik serupa dengan pelaku serangan (Sutton & King, 2013). *Hate crime* dapat berbentuk verbal, dan disebut sebagai *hate speech*, atau dalam bahasa Indonesia berarti ujaran kebencian. Menurut EU Framework Decision (2008), ujaran kebencian mencakup tindakan yang memicu kebencian dan diskriminasi yang menargetkan sekelompok orang atau bagian dari komunitas tertentu berdasarkan ras, warna kulit, agama/keyakinan, jenis kelamin, kemampuan fisik, orientasi seksual, garis keturunan dan negara atau suku asal.

Saat ini ujaran kebencian semakin banyak ditemukan seiring bertambahnya jumlah konten di dunia maya yang *user-generated*, seperti media sosial. Media sosial seperti Twitter, berpotensi menjadi sumber data *real-time* yang memungkinkan para peneliti untuk menganalisis respon masyarakat terhadap kondisi atau suasana sosial dan peristiwa-peristiwa berskala besar terbaru, terutama respon yang berupa kebencian (Williams & Burnap, 2015).

Di Indonesia, ribuan masyarakat mengunggah *tweet* berisi ujaran kebencian setiap harinya. Hal ini dapat memicu perseteruan antar golongan di Indonesia. Bahkan Badan Reserse Kriminal Kepolisian Negara Republik Indonesia (Bareskrim Polri) telah menangani 1.829 kasus ujaran kebencian pada 2016. Setelah dibentuk Direktorat Tindak Pidana Siber Bareskrim tahun 2017, kasus ujaran kebencian yang ditangani oleh Polri menjadi 3.325 kasus (Movanita, 2017).

Sudah ada penelitian terdahulu untuk perihal ujaran kebencian, misalnya penelitian klasifikasi *tweet* (Alfina et al., 2017). Data yang dipilih ialah *tweet* terkait Pemilihan Gubernur DKI Jakarta 2017. Beberapa kata kunci yang terkait dengan pemilihan tersebut misalnya frasa “Pilkada Jakarta 2017” dan *hashtag* seperti “#DebatPilkadaDKI”, “#SidangAhok”. Penelitian ini membandingkan performa metode *Naïve Bayes*, *Bayesian Logistic Regression*, dan *Random Forest Decision Tree* jika dikombinasikan dengan metode *Support Vector Machine* (SVM) menggunakan fitur *n-gram* dan *negative sentiment*. Penelitian tersebut menghasilkan nilai *F-measure*, diurutkan dari yang tertinggi ialah dengan



menggunakan metode *Random Forest Decision Tree* sebesar 93,5%, *Bayesian Logistic Regression* sebesar 91,5% dan *Naïve Bayes* sebesar 90,2%.

Selain metode-metode yang digunakan dalam penelitian Alfina, dkk. (2017), terdapat pula metode klasifikasi lain yang umum digunakan, seperti *K-Nearest Neighbor* (KNN). Pada penelitian Wah, dkk (2016), kinerja KNN dibandingkan dengan metode *Support Vector Machine* dan *Logistic Regression* dalam klasifikasi *Imbalanced Dataset*, dan hasilnya menunjukkan bahwa KNN adalah metode yang menghasilkan akurasi terbaik bagi proses *training* maupun *testing* yaitu sebesar 96,7% dan 95,6%, dibandingkan dengan hasil *Support Vector Machine* sebesar 95,9% dan 95,5% serta hasil *Logistic Regression* sebesar 96,2% dan 95,6%.

Namun menurut penelitian Li, dkk. (2003), KNN memiliki kekurangan yang dipengaruhi oleh nilai  $k$  yang sama di semua kelas, padahal jumlah sampel dari tiap kelas belum tentu merata. Masalah ini diatasi dengan *Improved K-Nearest Neighbor* (IKNN), yaitu KNN dengan nilai  $k$  yang berbeda-beda untuk tiap kelas, dimana nilai  $k$  akan disesuaikan dengan proporsi jumlah data yang termasuk dalam masing-masing kelas pada kumpulan data latih. Sehingga mengurangi kemungkinan terjadinya salah klasifikasi data, dimana data yang seharusnya termasuk dalam kelas yang beranggota sedikit dalam kumpulan data latih, justru dimasukkan ke dalam kelas lain yang beranggota lebih banyak.

Klasifikasi IKNN memerlukan pembobotan untuk masing-masing kata yang ada dalam teks terkait, misalnya BM25 dan BM25F. BM25F adalah *upgrade* dari BM25 yang dapat diimplementasikan pada teks terstruktur dengan lebih dari satu *field/stream* (umumnya *head*, *body*, dan sebagainya), dimana jika terdapat kata yang sama dalam *stream* yang berbeda, signifikansi dari kata tersebut akan dibedakan dengan mengalikan bobot BM25 dengan konstanta unik sesuai *stream* masing-masing. Perez-Aguera, dkk. (2010) melakukan pengujian *Mean Average Precision* (MAP), *Precision* setelah pengambilan 5 dokumen, *Precision* setelah pengambilan 10 dokumen, *Geometric Mean Average Precision* (GMAP) dan *R-Precision* dalam penelitiannya untuk membandingkan kinerja BM25, BM25F, Lucene, dan LuceneF pada *Semantic Web Search*. BM25F terbukti memiliki kinerja lebih baik pada MAP, GMAP dan R-Prec apabila dibandingkan BM25 karena memperhitungkan perbedaan *stream*.

Hal ini mendorong penyusunan sebuah penelitian untuk mengidentifikasi ujaran kebencian melalui automasi, tepatnya dengan memanfaatkan *Improved K-Nearest Neighbor* dan pembobotan BM25F sehingga mungkin di masa yang akan datang mampu membantu pihak-pihak tertentu, seperti Kepolisian Republik Indonesia (POLRI) dalam menangani kasus tuduhan penyebaran kebencian di media sosial Twitter yang perlu secepatnya mengkonfirmasi ada atau tidaknya unsur ujaran kebencian dalam sebuah *tweet*.

## 1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan, berikut adalah masalah yang dirumuskan untuk penelitian ini.

1. Bagaimana pengaruh penentuan bobot *stream* pada BM25F terhadap klasifikasi *tweet* berisi ujaran kebencian menggunakan algoritme *Improved K-Nearest Neighbor*?
2. Bagaimana hasil *Accuracy*, *Precision*, *Recall*, *F-measure* dan *K-Fold Cross Validation* dari penggunaan BM25F dan *Improved K-Nearest Neighbor* (IKNN) pada klasifikasi *tweet* berisi ujaran kebencian?

## 1.3 Tujuan

Dari penjabaran latar belakang dan rumusan masalah yang telah ditentukan, maka tujuan dari penelitian ini dapat diuraikan sebagai berikut.

1. Mengetahui pengaruh penentuan bobot *stream* pada BM25F terhadap klasifikasi menggunakan metode *Improved K-Nearest Neighbor* (IKNN).
2. Mengetahui hasil pengujian dari penggunaan BM25F dan *Improved K-Nearest Neighbor* (IKNN) pada klasifikasi *tweet* berisi ujaran kebencian.

## 1.4 Manfaat

Manfaat yang diharapkan dari penelitian ini antara lain adalah sebagai berikut.

### BAGI PENGGUNA

Sistem ini diharapkan dapat membantu pihak berwajib seperti Kepolisian Republik Indonesia lebih mudah mengidentifikasi keberadaan ujaran kebencian dalam *tweet* berbahasa Indonesia sehingga jika diperlukan penindakan lanjut, prosesnya bisa berjalan lebih cepat.

### BAGI PEMBACA

Mengetahui pengaruh penggunaan pembobotan BM25F dalam klasifikasi *tweet* berbahasa Indonesia yang mengandung Ujaran Kebencian sebagai teks terstruktur sehingga dapat dijadikan bahan pertimbangan dalam menerapkan BM25F.

## 1.5 Batasan masalah

Adapun batasan masalah pada penelitian ini adalah kumpulan data latih dan data uji hanya mencakup 2 kelas *tweet*, yaitu kelas Ujaran Kebencian (UK) dan kelas Non Ujaran Kebencian (NUK), dimana jumlah data pada 2 kelas mengalami ketidakseimbangan yang disebabkan oleh lebih banyaknya jumlah data kelas UK (*imbalanced dataset*). Kumpulan data latih terdiri atas *tweet* yang diunggah dalam kurun waktu 10 bulan terhitung sejak Januari 2019 hingga Oktober 2019

selama periode kampanye politik Pemilihan Umum Presiden Indonesia tahun 2019/2024 hingga pelantikan Presiden Indonesia terpilih.

## **1.6 Sistematika pembahasan**

Guna memudahkan dan memahami penulisan tiap-tiap bab dalam pembuatan proposal skripsi ini, berikut dijabarkan secara singkat sistematika penulisan proposal yang terdiri dari beberapa subbab.

### **BAB I PENDAHULUAN**

Pendahuluan berisi latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, penelitian yang berkaitan, metodologi penelitian, definisi istilah, dan sistematika penulisan.

### **BAB II LANDASAN TEORI**

Landasan teori berisi tinjauan dari beberapa literatur, tentang teori-teori yang terkait dengan permasalahan yang diambil, sebagai acuan dalam analisis dan pemecahan masalah yang dibahas dan nantinya akan memudahkan penulis dalam menyelesaikan dan memecahkan masalah.

### **BAB III METODOLOGI**

Metodologi penelitian berisi langkah - langkah yang digunakan dalam penelitian ini agar terstruktur dengan baik. Dengan sistematika ini proses penelitian dapat dipahami dan diikuti oleh pihak lain.

### **BAB IV PERANCANGAN**

Perancangan berfungsi untuk menjabarkan rincian algoritme yang digunakan, misalnya diagram alir tiap proses yang ditempuh, perhitungan manual dengan algoritme yang diterapkan, rancangan antarmuka program, dan rencana pengujian.

### **BAB V IMPLEMENTASI**

Implementasi akan memuat hasil penerapan dasar teori dari algoritme yang telah dijelaskan sesuai dengan isi bab Landasan Kepustakaan dan analisis perancangan.

### **BAB VI PENGUJIAN DAN ANALISIS**

Pengujian berfungsi untuk menguraikan hasil dari proses pengujian sistem dan akan memuat analisis hasil implementasi.

### **BAB VII PENUTUP**

Penutup terdiri dari dua bagian, yaitu kesimpulan dan saran. Kesimpulan berisi rangkuman secara singkat dari hasil pembahasan masalah. Sedangkan saran berisi harapan dan kemungkinan lebih lanjut dari hasil pembahasan masalah yang diperoleh untuk menuju lebih baik.

## BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini, penulis akan menjabarkan penelitian-penelitian terdahulu yang memiliki keterkaitan dengan penelitian Klasifikasi *Tweet* Berbahasa Indonesia berisi Ujaran Kebencian Menggunakan Metode *Improved K-Nearest Neighbor* dengan BM25F, termasuk landasan teori mengenai pembobotan BM25F, klasifikasi *Improved K-Nearest Neighbor* (IKNN), dan metode-metode pengujian hasil klasifikasi yaitu *Accuracy*, *Precision*, *Recall*, *F-measure*, serta pengujian untuk nilai  $k$  dengan *K-Fold Cross Validation*.

### 2.1 Kajian Pustaka

Pada bagian ini akan dibahas tentang penelitian-penelitian yang sudah ada dan memiliki keterkaitan dengan judul skripsi Klasifikasi *Tweet* Berbahasa Indonesia Berisi Ujaran Kebencian Menggunakan Metode *Improved K-Nearest Neighbor* dengan Pembobotan BM25F.

Keluaran akhir dari penelitian ini berupa hasil klasifikasi apakah sebuah *tweet* termasuk ujaran kebencian atau bukan. Sehingga diharapkan di masa yang akan datang, dapat digunakan untuk membantu kepolisian dalam mempercepat suatu proses penetapan adanya unsur ujaran kebencian dalam *tweet* tertentu yang diperkarakan oleh pihak yang merasa ditargetkan oleh *tweet* tersebut.

Contoh salah satu penelitian terkait judul yang telah disebutkan adalah penelitian yang dilakukan oleh Antinasari, dkk. (2017), mengenai analisis sentimen terhadap kumpulan *tweet* seputar opini masyarakat terhadap film-film tertentu menggunakan metode *Naïve Bayes* dan perbaikan kata tidak baku. Penelitian ini menggunakan kamus kata tidak baku yang dibuat manual oleh penulis, dan normalisasi *Levenshtein Distance* untuk memperbaiki kata yang kemungkinan besar salah ketik (*typo*) menjadi kata baku dalam *tweet*, kemudian diklasifikasikan dengan *Naïve Bayes*. Hasil pengujian yang didapatkan adalah nilai *Accuracy*, *Precision*, *Recall*, dan *F-measure* berturut-turut sebesar 98.33%, 96.77%, 100%, dan 98.36%.

Salah satu metode klasifikasi yang umum digunakan selain *Naïve Bayes* adalah metode *K-Nearest Neighbor* (KNN). Pada penelitian Wah, dkk (2016), kinerja KNN dibandingkan dengan metode *Support Vector Machine* dan *Logistic Regression* dalam klasifikasi *Imbalanced Dataset*, dan hasilnya menunjukkan bahwa KNN adalah metode yang menghasilkan akurasi terbaik bagi proses training maupun testing yaitu sebesar 96,7% dan 95,6%, dibandingkan dengan hasil *Support Vector Machine* sebesar 95,9% dan 95,5% serta hasil *Logistic Regression* sebesar 96,2% dan 95,6%.

Contoh ketiga adalah penelitian Li, dkk. (2003), yang membahas tentang KNN, yang memiliki kekurangan yang dipengaruhi oleh nilai  $k$  yang sama di semua kelas, padahal jumlah data pada tiap kelas belum tentu merata. Masalah ini diatasi dengan *Improved K-Nearest Neighbor* (IKNN), yaitu KNN dengan nilai  $k$

yang berbeda-beda untuk tiap kelas, dimana nilai  $k$  akan disesuaikan dengan proporsi jumlah data yang termasuk dalam masing-masing kelas pada kumpulan data latih. Sehingga mengurangi kemungkinan terjadinya salah klasifikasi data, dimana data yang seharusnya termasuk dalam kelas yang beranggota sedikit dalam kumpulan data latih, justru dimasukkan ke dalam kelas lain yang beranggota lebih banyak. Hasil penelitian ini menunjukkan bahwa ketika nilai  $k$  semakin meningkat, nilai performa (*Precision*, *Recall* dan *F-measure*) KNN asli menurun lebih drastis dibandingkan dengan IKNN. Saat nilai  $k$  mencapai 60, nilai *F-measure* pada KNN asli menurun 13.32%. Selain itu, rata-rata performa IKNN juga dinyatakan lebih baik sebesar 1.4% dibandingkan rata-rata performa KNN.

Penelitian tersebut menginspirasi penelitian lain, yang dilakukan oleh Pranata, dkk. (2019), mengenai penggunaan *Improved K-Nearest Neighbor* dalam klasifikasi laporan kejahatan di Polres Kota Malang dengan BM25 sebagai metode pembobotannya. Pada penelitian ini, disebutkan bahwa *Improved K-Nearest Neighbor* lebih mampu menghasilkan klasifikasi yang stabil dari pada *K-Nearest Neighbor* biasa akibat penggunaan nilai  $k$  yang berbeda berdasarkan nilai jumlah data latih dari tiap kelas. Pengujian dengan *k-Fold Cross Validation* menghasilkan rata-rata nilai tertinggi pada  $k = 15$ , dimana *Precision*, *Recall*, *F-measure* dan *Accuracy* secara berurutan bernilai 0.953373, 0.931382, 0.938122, 0.956795.

Contoh selanjutnya yang menjadi landasan bagi penelitian ini ditulis oleh Perez-Aguera, dkk. (2010), mengenai penggunaan metode BM25F untuk pencarian semantik. Penelitian tersebut menganalisis cara menyesuaikan skema pembobotan untuk pengambilan dokumen semantik yang termasuk dokumen terstruktur, artinya dokumen terbagi menjadi beberapa bagian (*field/stream*). BM25F adalah perkembangan dari BM25 yang mampu memberikan bobot berbeda pada sebuah kata yang sama di masing-masing *stream*, sehingga dianggap lebih baik dalam mempertimbangkan signifikansi yang berbeda antara bagian-bagian dalam sebuah dokumen terstruktur. Terdapat tiga kali pengujian, pengujian ke-1 ialah pencarian dokumen *unstructured* berdasarkan kueri judul dan yang ke-2 ialah berdasarkan kueri yang panjang. Pengujian ke-3 ialah perbandingan BM25F dengan Lucene untuk pencarian dokumen *structured* (memiliki *field/stream*). Pengujian ke-1 menghasilkan nilai MAP, *Precision* setelah pengambilan 5 dokumen, *Precision* setelah pengambilan 10 dokumen, GMAP dan R-Prec dari BM25F secara berurutan sebesar 0.1743, 0.4412, 0.3765, 0.1030, dan 0.2172, dimana semua nilai tersebut lebih besar dibandingkan hasil pengujian BM25, Lucene, dan LuceneF. Pengujian ke-2 menghasilkan nilai MAP, P@5, P@10, GMAP dan R-Prec dari BM25F secara berurutan sebesar 0.1822, 0.4647, 0.3824, 0.1170, dan 0.2262, dimana nilai MAP, GMAP dan R-Prec lebih besar dibandingkan hasil pengujian BM25, Lucene, dan LuceneF. Pengujian ke-3 membuktikan bahwa BM25F memiliki nilai rata-rata MAP, GMAP, P@5 dan P@10 lebih baik dari Lucene.

Selain beberapa penelitian diatas, terdapat penelitian lain yang sudah diterbitkan dalam bentuk buku, ditulis oleh Stephen Robertson dan Hugo Zaragoza (2009) mengenai BM25 dan algoritme sejenisnya. Penelitian ini menjabarkan mengenai langkah-langkah lengkap perhitungan manual metode BM25F pada dokumen terstruktur yang memiliki 2 atau lebih *field*. Dalam buku tersebut, Robertson menyebut *field* dengan istilah *stream*. Ia juga menyatakan bahwa BM25F lebih unggul ketimbang BM25 dari segi fleksibilitasnya sehingga mampu diterapkan pada teks yang memiliki beberapa *stream* (berstruktur) maupun teks yang memiliki satu *stream* saja (tak berstruktur).

Dari penelitian-penelitian yang sudah disebutkan, belum ada di antaranya yang menggunakan pembobotan BM25F untuk diterapkan pada klasifikasi *tweet* seperti pada penelitian ini, dimana komponen dari *tweet* yang digunakan bukan hanya *body* atau kalimat konten dalam *tweet* tersebut, namun penulis juga memutuskan untuk menggunakan *hashtag* atau tagar dalam sebuah *tweet*, sehingga dapat dimanfaatkan sebagai *field/stream* kedua selain *body tweet* dalam menggunakan BM25F.

## 2.2 Ujaran Kebencian dan Penyebarannya di Media Sosial

*Hate speech* atau ujaran kebencian adalah ujaran, bahasa tubuh, perilaku, tulisan, maupun aksi publik yang dapat mendorong seseorang atau sekelompok orang untuk melakukan kekerasan dan tindakan akibat prasangka bertarget, yang mampu melukai orang lain dan merebut martabat mereka (Ezeibe, 2015).

Bentuk ujaran kebencian secara lisan maupun tulisan sangat sering dianggap *lumrah* atau sudah biasa oleh masyarakat Indonesia dalam kehidupan sehari-hari, terutama bagi oknum yang mengujarkannya. Masalah ini adalah akibat dari pemahaman yang salah mengenai terkait istilah kebebasan berpendapat.

Masyarakat menyalahgunakan asas kebebasan dalam berpendapat sebagai pembenaran mereka ketika ingin menjatuhkan maupun mencemarkan nama baik orang lain secara sengaja. Penyebaran kebencian melalui tulisan saat ini semakin 'didukung' dengan adanya media sosial, dimana siapapun hanya perlu mengetik beberapa kata sesuka hati mereka dan apapun yang tertera akan langsung dapat dibaca oleh jutaan pengguna lain. Salah satu media sosial yang tercatat paling banyak digunakan untuk menyebarkan ujaran kebencian adalah Twitter. Buku Saku Komisi Nasional Hak Asasi Manusia (KOMNASHAM RI, 2015) mengungkapkan bahwa penyebab bahayanya ujaran kebencian diantaranya adalah:

- 1) Merendahkan manusia lain: Manusia adalah ciptaan Tuhan dan tidak ada seorang pun yang berhak merendahkan manusia dan kemanusiaan seorang pun yang merupakan ciptaan Tuhan.
- 2) Menimbulkan kerugian materil dan korban manusia: Data penelitian menunjukkan jumlah kerugian material dan korban kekerasan berbasis identitas lebih besar daripada kekerasan lainnya.

- 3) Bisa berdampak pada konflik: Hasutan untuk memusuhi orang atau kelompok bisa menimbulkan konflik, konflik ini bisa antar individu dan meluas menjadi konflik komunal atau antar kelompok.
- 4) Bisa berdampak pada pemusnahan kelompok atau genosida: Hasutan kebencian ini bisa membuat *stereotyping*/pelabelan, stigma, pengucilan, diskriminasi, kekerasan. Pada tingkat yang paling mengerikan bisa menimbulkan kebencian kolektif pembantaian etnis, pembakaran kampung atau pemusnahan (genosida) terhadap kelompok yang menjadi sasaran ujaran kebencian.

## 2.3 Preprocessing Teks

*Preprocessing* adalah tahap pertama dalam mengelola teks. Sebelum melakukan klasifikasi dengan *IKNN*, tahap ini harus dilakukan untuk mendapatkan format teks yang siap untuk dijadikan data latih.

*Preprocessing* sangat penting dalam memahami makna penyampaian maksud dalam kalimat/teks, terutama dalam media sosial dimana komunikasi yang berlangsung sebagian besar menggunakan kata-kata yang tidak formal dan tidak terstruktur serta memiliki *noise* yang besar (Mujilawati, 2016). Tahapan *preprocessing* yang digunakan umumnya, dan juga akan digunakan dalam penelitian ini antara lain *Case folding*, Tokenisasi, *Filtering*, dan *Stemming*.

## 2.4 Cleaning

*Cleaning* merupakan tahap *preprocessing* untuk menghilangkan elemen yang tidak penting atau mengganggu tahap *preprocessing* selanjutnya. Untuk *cleaning* dengan objek tweet, secara umum, unsur-unsur yang dihilangkan adalah tagar (diawali dengan tanda '#'), *mention* (diawali dengan tanda '@') dan URL (Angiani et al., 2016). Namun penelitian ini memanfaatkan fitur tagar, sehingga tagar yang ditemukan dalam *tweet* tidak akan dihapus, melainkan hanya perlu dipisah berdasarkan huruf kapital. Tabel 2.2 menggambarkan prosesnya.

**Tabel 2.1 Cleaning**

Sebelum	Sesudah
Acara yang diselenggarakan selama bulan September tersebut dikunjungi oleh banyak turis asing, khususnya turis asal Perancis.	Acara yang diselenggarakan selama bulan September tersebut dikunjungi oleh banyak turis asing khususnya turis asal Perancis

## 2.5 Case folding

*Case folding* merupakan tahap selanjutnya dalam *preprocessing* yang mengubah semua alphabet "a" hingga "z" dalam dokumen terkait menjadi huruf kecil atau non-kapital (Puspitasari et al., 2018). Namun, *Case folding* terkadang menyebabkan kurang jelasnya makna dari sebuah kata. Contohnya adalah ISI,

yang merupakan akronim dari Institut Seni Indonesia, apabila dikenakan *case folding*, maka akan menjadi isi, yaitu kata yang dalam KBBI berarti *sesuatu yang ada (termuat, terkandung, dan sebagainya) di dalam suatu benda dan sebagainya*. Istilah tersebut yang sudah pasti memberi makna berbeda pada dokumen teks terkait. Untuk contoh *case folding*, berikut adalah Tabel 2.1 yang menggambarkan prosesnya.

**Tabel 2.2 Case folding**

Sebelum	Sesudah
Acara yang diselenggarakan selama bulan September tersebut dikunjungi oleh banyak turis asing khususnya turis asal Perancis	acara yang diselenggarakan selama bulan september tersebut dikunjungi oleh banyak turis asing khususnya turis asal perancis

## 2.6 Tokenisasi

Tokenisasi adalah tahap memisahkan kalimat/dokumen menjadi kata-kata yang menyusun kalimat/dokumen tersebut (Munir et al., 2018). Pemisahan ini dilakukan dengan cara memotong sebuah *string* menjadi sekumpulan kata tunggal yang dimasukkan, berdasarkan spasi yang digunakan sebagai pemisah (delimiter) antar kata. Berikut adalah Tabel 2.3 yang menggambarkan proses tokenisasi.

**Tabel 2.3 Tokenisasi**

Sebelum	Sesudah
acara yang diselenggarakan selama bulan september tersebut dikunjungi oleh banyak turis asing khususnya turis asal perancis	"acara", "yang", "diselenggarakan", "selama", "bulan", "september", "tersebut", "dikunjungi", "oleh", "banyak", "turis", "asing", "asal", "perancis"

## 2.7 Filtering

*Filtering* adalah tahap dimana kata-kata yang dianggap signifikan terhadap hasil klasifikasi dari hasil tokenisasi akan digunakan untuk mewakili isi dari suatu dokumen dan membedakannya dari dokumen lain dalam koleksi (Puspitasari et al., 2018). Filtering dilakukan dengan menghapus *stopword*, yaitu kata-kata yang dianggap tidak cukup penting untuk dijadikan sebagai kata kunci dalam klasifikasi dokumen, misalnya kata "yang".

Ada beberapa daftar *stopword* untuk bahasa Indonesia yang sudah umum digunakan, misalnya daftar *stopword* oleh Fadillah Z. Tala (Tala, 2003). Contoh *filtering* terdapat pada Tabel 2.4 berikut.



**Tabel 2.4 Filtering**

Sebelum	Sesudah
“acara”, “yang”, “diselenggarakan”, “selama”, “bulan”, “september”, “tersebut”, “dikunjungi”, “oleh”, “banyak”, “turis”, “asing”, “asal”, “perancis”	“acara”, “diselenggarakan”, “september”, “dikunjungi”, “turis”, “asing”, “perancis”

## 2.8 Stemming

*Stemming* merupakan tahap pengembalian sebuah kata menjadi kata dasar dari kata tersebut (Munir et al., 2018). Proses *stemming* dalam bahasa Indonesia melibatkan penghapusan awalan (prefiks), sisipan (infiks), imbuhan (sufiks), serta kombinasi awalan dan imbuhan (Puspitasari et al., 2018). Biasanya setelah melalui *stemming*, kata yang masih tersisa dari tahap *preprocessing* akan disebut sebagai *term*. Tabel 2.5 yang menggambarkan proses tokenisasi.

**Tabel 2.5 Stemming**

Sebelum	Sesudah
“acara”, “diselenggarakan”, “september”, “dikunjungi”, “turis”, “asing”, “perancis”	“acara”, “selenggara”, “september”, “kunjung”, “turis”, “asing”, “perancis”

## 2.9 BM25F

Pada penelitian ini, penulis memilih salah satu modifikasi dari BM25 (*Best Match 25*), yaitu BM25F, untuk memberi bobot pada *term* yang muncul nantinya. Pada BM25, semua jenis dokumen dianggap sebagai satu kesatuan teks, tidak terstruktur dan tiap bagian yang ada di dalam teks tidak dibedakan. Maka BM25 tidak bisa digunakan untuk mendapatkan bobot kata dari teks terstruktur, yaitu teks yang isinya terbagi menjadi beberapa bagian. Dari keterbatasan tersebut, algoritme BM25 dikembangkan menjadi BM25F, yang membagi teks terstruktur menjadi beberapa bagian (bisa disebut juga sebagai *field* atau *stream*). Contoh *stream* dalam teks yang paling umum adalah struktur *title/abstract/body* (Robertson & Zaragoza, 2009). Struktur ini dapat ditemukan di berbagai jenis teks seperti teks berita hingga artikel penelitian.

Menurut Robertson dan Zaragoza (2009), BM25F diawali dengan perhitungan BM25 tiap *term* pada masing-masing *stream*, kemudian hasil dari masing-masing *stream* akan dikombinasikan secara linear, yaitu mengalikan hasil tersebut dengan bobot unik untuk tiap *stream* yang disebut dengan *stream weight*. Sehingga *term* yang sama akan memiliki bobot yang berbeda jika kata tersebut terdapat pada *stream* yang berbeda. Misalkan terdapat kata “hilang” pada dua bagian dalam dokumen, yaitu bagian judul dokumen (*title*) dan badan dokumen

(*body*). Kata tersebut akan berbobot beda pada masing-masing bagian, karena keduanya dipengaruhi oleh *stream weight* masing-masing. *Stream weight* sendiri ditentukan berdasarkan prioritas penggunaan *stream* masing-masing dalam klasifikasi yang dilakukan. Semakin suatu *stream* dianggap signifikan dalam proses yang dilakukan, maka harusnya nilai *stream weight* yang digunakan semakin besar.

Berdasarkan panjang dokumennya, penulis/pengarang penelitian memiliki dua kecenderungan, yaitu *Verbosity* dan *Scope*. *Verbosity* adalah kecenderungan penulis menggunakan lebih banyak kata untuk mengatakan hal yang sama. Sedangkan *Scope* adalah kecenderungan menulis satu dokumen yang mencakup topic yang lebih luas. Dua kecenderungan inilah yang menyebabkan perlunya dilakukan normalisasi panjang dokumen dalam koleksi dokumen yang panjang tiap dokumennya bervariasi. Normalisasi panjang dokumen ini direpresentasikan dalam Persamaan 2.1.

$$B_s = \left( (1 - b_s) + b_s \frac{sl_s}{avsl_s} \right) \quad (2.1)$$

Dimana:

$b_s$  = parameter penentu normalisasi (umumnya  $0.5 \leq b_s \leq 0.8$ ). Bisa bernilai 0 jika tanpa normalisasi, bernilai 1 jika dengan normalisasi penuh.

$sl_s$  = *stream length*, panjang *stream s* (dihitung dari jumlah *term* di masing-masing *stream*)

$B_s$  = Komponen normalisasi panjang dokumen

$avsl_s$  = Rata-rata  $sl_s$  dari seluruh dokumen

Langkah kedua dalam perhitungan BM25F setelah memperoleh nilai Komponen normalisasi panjang suatu dokumen, total jumlah kemunculan *term* (*term frequency* atau *tf*) yang dinormalisasi (*normalized tf*) pada masing-masing *stream* dapat direpresentasikan dalam Persamaan 2.2.

$$\widetilde{tf}_i = \sum_{s=1}^S v_s \frac{tf_{si}}{B_s} \quad (2.2)$$

Dimana:

$\widetilde{tf}_i$  = Jumlah *normalized tf* dari seluruh *stream* dalam dokumen

$s$  = *stream*, misal  $s = 1$ , maksudnya adalah bagian/*field* pertama dalam dokumen.

$S$  = jumlah *stream* yang dimiliki suatu dokumen

$B_s$  = Komponen normalisasi panjang dokumen

$v_s$  = *stream weights* (bobot unik masing-masing *stream s* pada dokumen)

$tf_{si}$  = *tf* dari *term i* dalam *stream s* dalam dokumen

Menurut Robertson dan Zaragoza (2009), proses yang tidak melibatkan relevansi informasi atau koleksi dokumen relevan layaknya sistem temu kembali, harus menggunakan  $w^{idf}$ , yaitu bobot IDF yang telah dimodifikasi. Persamaan yang merepresentasikan perhitungan tersebut ialah Persamaan 2.3.

$$w_i^{IDF} = \log \frac{N - df_i + 0.5}{df_i + 0.5} \quad (2.3)$$

Dimana:

$N$  = Ukuran koleksi atau jumlah dokumen data latih

$df_i$  = Jumlah dokumen dalam data latih yang mengandung kata  $i$

$w_i^{IDF}$  = modifikasi bobot IDF dari kata  $i$

Langkah terakhir adalah perhitungan BM25F dari seluruh bagian dokumen, yang direpresentasikan dalam Persamaan 2.4.

$$w_d^{BM25F} = \sum_{i=1}^j \frac{\widetilde{tf}_i}{k_1 + \widetilde{tf}_i} w_i^{IDF} \quad (2.4)$$

Dimana:

$\widetilde{tf}_i$  = Jumlah *normalized tf* dari seluruh *stream* dalam dokumen

$k_1$  = konstanta saturasi, umumnya bernilai  $1.2 < k_1 \leq 2$

$w_i^{IDF}$  = modifikasi bobot IDF dari *term i*

$j$  = jumlah *term* dari kueri (dokumen uji)

$w_d^{BM25F}$  = Bobot BM25F semua *term* pada kueri dalam dokumen  $d$

## 2.10 Metode Klasifikasi *Improved K-Nearest Neighbor* (IKNN)

*Improved K-Nearest Neighbor* (IKNN), yaitu KNN dengan nilai  $k$  yang berbeda-beda untuk tiap kelas, berbeda dengan klasifikasi KNN yang menggunakan nilai  $k$  tetap. Secara umum, distribusi dokumen dari berbagai kelas dalam kumpulan data latih tidak merata. Beberapa kelas mungkin memiliki lebih banyak sampel daripada kelas yang lain. Oleh karena itu, sangat mungkin bahwa nilai  $k$  yang tetap akan menyebabkan kelas dengan jumlah sampel lebih banyak akan cenderung terpilih, walaupun sebenarnya tidak tepat (Li et al., 2003).

Menurut penelitian Li, dkk. (2003) dengan menggunakan tetangga terdekat sebanyak  $k$  awal, probabilitas bahwa satu dokumen termasuk dalam kelas tertentu dapat ditentukan dengan menggunakan tetangga terdekat sebanyak  $n$  untuk kelas tersebut, di mana  $n$  (nilai  $k$  baru) berasal dari  $k$  awal, sesuai dengan

ukuran kelas tersebut dalam kumpulan data latih. Perhitungan nilai  $n$  sebagai hasil modifikasi nilai  $k$  untuk tiap kelas ada pada Persamaan 2.5.

$$n = \left\lceil \frac{k \times N(C_m)}{\max\{N(C_j) | j = 1, \dots, N_c\}} \right\rceil \quad (2.5)$$

Dimana:

$n$	= nilai $k$ baru
$k$	= nilai $k$ awal
$N(C_m)$	= Jumlah dokumen yang termasuk kelas $m$ dalam kumpulan data latih
$\max\{N(C_j)   j = 1, \dots, N_c\}$	= Jumlah dokumen dalam kelas yang beranggota terbanyak pada kumpulan data latih

Berikutnya, akan dihitung peluang dokumen uji masuk ke masing-masing kelas yang tersedia, yang direpresentasikan oleh Persamaan 2.6.

$$P(x, C_m) = \operatorname{argsMax}_m \frac{\sum_{i=1}^n \operatorname{sim}(x, d_j) y(d_j, C_m)}{\sum_{i=1}^n \operatorname{sim}(x, d_j)} \quad (2.6)$$

Dimana:

$P(x, C_m)$	= Peluang dokumen $x$ masuk ke kelas $C_m$
$\operatorname{sim}(x, d_j)$	= similaritas/kemiripan dokumen $x$ dengan dokumen latih $d_j$
$y(d_j, C_m)$	= parameter indikasi kelas, akan bernilai 1 jika $d_j$ masuk ke kelas $C_m$ , bernilai 0 jika tidak masuk ke kelas $C_m$
$n$	= jumlah $k$ hasil modifikasi untuk IKNN
$\operatorname{argsMax}_m$	= menandakan dipilihnya $n$ data dengan nilai BM25F tertinggi dalam kelas $m$

## 2.11 K-Fold Cross Validation

*K-Fold Cross Validation* adalah metode pengujian kinerja klasifikasi dimana kumpulan data dipartisi sebanyak  $K$  segmen dengan ukuran yang sama (atau hampir sama). Pengujian *K-Fold Cross Validation* biasanya dilaksanakan dengan melalui iterasi sejumlah  $K$ . Pada setiap iterasi, kumpulan data dibagi menjadi 2 bagian, sebagian kumpulan data digunakan untuk pelatihan dan sebagian lainnya untuk pengujian (Singh & Shukla, 2016).

Sebagai contoh, terdapat sebuah kumpulan data berisi 5 buah data. Nilai  $K$  yang telah ditentukan adalah 2, berlangsung sebagai berikut.

1) Iterasi ke-1:

- Data ke-1 dan ke-2 menjadi Data Uji
- Data ke-3, ke-4 dan ke-5 menjadi Data Latih

2) Iterasi ke-2:

- Data ke-3, ke-4 dan ke-5 menjadi Data Uji
- Data ke-1 dan ke-2 menjadi Data Latih

## 2.12 Confusion Matrix

Pada tahapan ini akan dilakukan proses pengujian dan analisis terhadap hasil klasifikasi yang telah dilakukan terhadap kumpulan data uji. Pengujian akan dilakukan dengan menggunakan 4 jenis Evaluasi, yaitu *Accuracy*, *Precision*, *Recall*, dan *F-measure*. Keempat jenis evaluasi tersebut dapat dilakukan dengan berdasarkan Tabel kontingensi atau *confusion matrix*.

Penelitian ini menerapkan klasifikasi *binary-class* (dua kelas), dimana masing-masing kelas dapat dianggap kelas Positif dan kelas Negatif yang tersusun dalam Tabel kontingensi *binary-class* dalam langkah evaluasi. Kelas Positif disini digunakan untuk merujuk pada kelas data *tweet* yang mengandung ujaran kebencian, sementara kelas Negatif akan merujuk pada kelas data *tweet* yang tidak mengandung ujaran kebencian. Sehingga *confusion matrix* yang digunakan menurut Fawcett (2006) ditampilkan pada Tabel 2.6.

**Tabel 2.6 Confusion Matrix**

Prediksi	Aktual		
	Total Population	Kelas Positif	Kelas Negatif
	Kelas Positif	<i>True Positive (tp)</i>	<i>False Positive (fp)</i>
	Kelas Negatif	<i>False Negative (fn)</i>	<i>True Negtaive (tn)</i>

Dimana:

- True Positive (tp)* = Jumlah data kelas positif yang diprediksi secara benar sebagai kelas positif
- False Positive (fp)* = Jumlah data kelas negatif yang diprediksi secara salah sebagai kelas positif
- True Negtaive (tn)* = Jumlah data kelas negatif yang diprediksi secara benar sebagai kelas negatif
- False Negative (fn)* = Jumlah data kelas positif yang diprediksi secara salah sebagai kelas negatif

## 2.13 Accuracy

*Accuracy* atau *Accuracy* dalam Bahasa Indonesia adalah metrik evaluasi yang umum digunakan dalam klasifikasi. *Accuracy* berupa ukuran perbandingan/rasio hasil prediksi kelas data yang benar terhadap jumlah total data objek yang dievaluasi (Hossin & Sulaiman, 2015). *Accuracy* dapat dihitung dengan Persamaan 2.7.

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \quad (2.7)$$

### 2.14 Precision

*Precision* atau *Precision* dalam Bahasa Indonesia digunakan untuk mengukur perbandingan/rasio jumlah data kelas positif yang diprediksi dengan benar dari total jumlah data yang diprediksi sebagai kelas positif (Hossin & Sulaiman, 2015). *Precision* dapat dihitung dengan Persamaan 2.8.

$$Precision = \frac{tp}{tp + fp} \quad (2.8)$$

### 2.15 Recall

*Recall* digunakan untuk mengukur perbandingan/rasio jumlah data kelas positif yang diprediksi dengan benar dari total jumlah data kelas positif yang sebenarnya (termasuk jumlah data kelas positif yang diprediksi sebagai kelas negatif). Sehingga *Recall* dianggap sebagai model pengujian yang paling tepat pada kasus yang sangat dipengaruhi/dirugikan jika ada *False Negative*, misalnya diagnosa penyakit, dimana akan sangat berbahaya jika seseorang yang positif mengidap penyakit tertentu malah didiagnosis tidak mengidap penyakit apapun (Shung, 2018). *Recall* dapat dihitung dengan Persamaan 2.9.

$$Recall = \frac{tp}{tp + fn} \quad (2.9)$$

### 2.16 F-measure

Metrik ini merupakan representasi keseimbangan antara nilai *Recall* dan *Precision* dan ketika terdapat persebaran kelas hasil prediksi yang tidak merata, misalnya ketika jumlah data kelas negative yang sesungguhnya terlalu besar (Shung, 2018). *F-measure* dapat dihitung dengan Persamaan 2.10.

$$F - Measure = \frac{2 \times p \times r}{p + r} \quad (2.10)$$

## BAB 3 METODOLOGI

Bab Metodologi akan menguraikan rincian mengenai tipe penelitian yang sedang dilakukan, strategi penelitian, partisipan dari penelitian (jika ada), lokasi dimana penelitian dilakukan, peralatan yang digunakan, teknik pengumpulan data penelitian, gambaran perancangan sistem secara garis besar, pengujian, dan kesimpulan.

### 3.1 Tipe Penelitian

Tipe penelitian ini adalah penelitian Non-Implementatif Analitik. Non-Implementatif menitikberatkan pada pendalaman atau analisis terhadap situasi tertentu yang dipengaruhi faktor-faktor tertentu, dan menghasilkan hasil analisis ilmiah sebagai produk/artefak utamanya. Kemudian Analitik (Eksploratori) berarti penelitian ini bertujuan untuk menjelaskan derajat hubungan antar elemen dalam objek penelitian dengan situasi yang sedang diteliti.

### 3.2 Strategi Penelitian

Strategi penelitian ini adalah dengan studi kasus klasifikasi ujaran kebencian yang tersebar di Twitter. Objek dalam studi kasus ini adalah sejumlah *tweet* yang dipilih secara manual oleh penulis dari laman web <https://Twitter.com>. Cara yang ditempuh untuk mendapatkan kumpulan data dalam penelitian ini yang adalah memilih dan mengambil *tweet* secara manual berdasarkan tagar yang telah dipilih oleh penulis dengan memanfaatkan fitur/tab *Search* pada laman web Twitter (<http://Twitter.com>) maupun aplikasi Twitter melalui Android.

### 3.3 Partisipan Penelitian

Penelitian ini dilakukan dengan melibatkan partisipan langsung dan tidak langsung. Partisipan langsung yaitu pakar yang membantu penulis memberi label kelas pada 500 *tweet* sebagai data latih, dan partisipan tidak langsung yaitu pemilik akun Twitter yang *tweet*-nya digunakan oleh penulis sebagai bagian dari kumpulan data latih maupun kumpulan data uji dalam proses klasifikasi ujaran kebencian.

### 3.4 Lokasi Penelitian

Penelitian mengenai klasifikasi *tweet* berbahasa Indonesia berisi ujaran kebencian menggunakan metode *Improved K-Nearest Neighbor* dengan pembobotan BM25F dilakukan di Laboratorium Riset Fakultas Ilmu Komputer (FILKOM), atau di dalam lingkungan Universitas Brawijaya Malang.

### 3.5 Teknik Pengumpulan Data

Terdapat 2 kumpulan data yang digunakan, yaitu kumpulan data (dokumen) latih dan kumpulan data (dokumen) uji. Kumpulan dokumen Latih dan kumpulan

dokumen uji yang digunakan merupakan data primer berupa *tweet* yang disinyalir mengandung ujaran kebencian (kelas Ujaran Kebencian/UK) dan *tweet* yang tidak mengandung ujaran kebencian (kelas Non Ujaran Kebencian/NUK), agar selanjutnya dapat dimanfaatkan sebagai kumpulan dokumen latih dan set data uji dalam proses klasifikasi. Total data adalah sebanyak 500 *tweet*, dengan 324 *tweet* kelas Ujaran Kebencian dan 176 *tweet* kelas Non Ujaran Kebencian.

Teknik pengumpulan *tweet* ditempuh dengan 2 cara. Cara pertama adalah secara manual dengan memanfaatkan fitur/tab *Search* pada website Twitter maupun aplikasi Twitter android, untuk mengumpulllkan *tweet* berdasarkan beberapa kata kunci dan tagar yang telah dipilih, antara lain “#DebatPilpres2019”, “Prabohong” dan “#jokowiMUNDUR”. Cara kedua yaitu memanfaatkan aplikasi TAGS, yaitu template Google Sheet gratis yang dapat diatur untuk mengarsipkan *tweet* terbaru selama maksimal 7 hari terakhir berdasarkan tagar dan atau kata kunci, dimana penulis mencoba tagar “#Jokowi2Periode”, “#TetapMasihOposisi” dan kata “tolol”. Penulis akan menggunakan 500 data sebagai kumpulan dokumen latih dan uji. Pembagian antara dokumen latih dan data uji akan menggunakan *K-Fold Cross Validation*, dimana penulis berencana menggunakan 5 Fold.

Pengumpulan data dilakukan selama 3 bulan terhitung sejak Agustus 2019 sampai dengan November 2019, sedangkan untuk cakupan *tweet* yang diambil sebagai data latih adalah *tweet* yang diunggah dalam jangka waktu 10 bulan terhitung sejak Januari 2019 hingga Oktober 2019 selama periode kampanye politik Pemilihan Umum Presiden Indonesia tahun 2019/2024 hingga pelantikan Presiden Indonesia terpilih.

### 3.6 Peralatan Pendukung

Dalam melakukan klasifikasi *tweet* berbahasa Indonesia berisi ujaran kebencian menggunakan metode *Improved K-Nearest Neighbor* dengan pembobotan BM25F ini, beberapa perangkat yang mampu mendukung berjalannya proses penelitian ini diuraikan sebagai berikut.

1. Perangkat Lunak
  - 1) Sistem Operasi : *Windows 8.1*. (minimal)
  - 2) Bahasa Pemrograman : *Python 3.6* (minimal)
  - 3) *Editor* : Jupyter Notebook, Microsoft Excel 2010 (minimal)
  - 4) *Library* : Re, numpy, Math, csv, Sastrawi, collection
2. Perangkat Keras
  - 1) Processor AMD A4-6210 1.80 GHz
  - 2) AMD Radeon R3 Graphics
  - 3) RAM 4GB



### 3.7 Perancangan Sistem

Sistem ini akan menerima input/masukan berupa file dengan format csv yang berisi 500 data *tweet* yang sudah diberi label '1' untuk UK dan '0' untuk NUK oleh pakar. Kemudian dalam tahap *preprocessing*, pelatihan dan pengujian akan menggunakan file dan data yang berbeda sesuai dengan pembagian jumlah data latih dan data uji, setelah itu akan dilakukan *case folding*, tokenisasi, *filtering*, dan *stemming* pada data latih dan data uji tersebut. Setelah *preprocessing*, tahap selanjutnya adalah perhitungan *tf* dan *df* pada masing-masing dokumen, maupun itu dokumen latih atau dokumen uji. Tahap selanjutnya adalah pembobotan kata dengan BM25F.

Pembobotan kata dengan BM25F hanya akan dikenakan pada dokumen uji, sebab yang akan diklasifikasikan hanya dokumen uji saja. Proses perhitungan BM25F dimulai dengan menghitung panjang dokumen yang dinormalisasi ( $B_s$ ), sehingga selanjutnya nilai tersebut digunakan untuk menemukan nilai *tf* yg dinormalisasi ( $\widetilde{tf}_i$ ), yaitu *tf* setiap *term* di masing-masing *stream* yang dikalikan dengan bobot unik *stream* tersebut kemudian dibagi dengan nilai  $B_s$ . Setelah itu, perlu dihitung nilai IDF nya, dimana BM25F menggunakan IDF yang sudah dimodifikasi, yaitu  $w_i^{IDF}$ . Lalu bobot BM25F bisa mulai dihitung dengan menggunakan  $\widetilde{tf}_i$ ,  $w_i^{IDF}$ , serta sebuah konstanta  $k$  yang disebut sebagai konstanta saturasi. Setelah bobot didapatkan dengan BM25F, maka klasifikasi dapat dilakukan dengan menggunakan Improved KNN, dimana peran BM25F disini adalah sebagai jarak kedekatan atau *similarity*, yang standardnya secara umum menggunakan *Cosine Similarity* untuk data dalam bentuk teks.

Proses klasifikasi IKNN dilakukan dalam 2 tahap, yaitu menentukan nilai  $k$  sesuai dengan proporsi jumlah data yang masuk ke masing-masing kelas dalam klasifikasi, dan diikuti dengan proses perhitungan peluang sebuah data uji masuk ke masing-masing kelas yang ada.

### 3.8 Implementasi Sistem

Implementasi aplikasi dilakukan dengan menggunakan bahasa pemrograman *Python* dan perangkat lunak pendukung lainnya agar sesuai dengan perancangan yang telah diidentifikasi sebelumnya.

### 3.9 Teknik Analisis Data

Teknik Analisis Data atau pengujian dilakukan untuk mengetahui seberapa baik unjuk kerja sistem yang telah diimplementasikan sesuai dengan algoritme dan landasan teori. Ukuran unjuk kerja sistem diperoleh menggunakan perhitungan matematis. Perhitungan yang digunakan pada penelitian ini adalah dengan *K-Fold Cross Validation* dan *Confusion Matrix*.

### **3.10 Pengambilan Kesimpulan**

Kesimpulan didapatkan setelah semua tahapan proses telah selesai dilakukan sampai ke tahap evaluasi/pengujian sistem. Hasil klasifikasi dan pengujian akan dikombinasikan untuk menyusun sebuah kesimpulan tentang kinerja metode yang digunakan. Setelah kesimpulan didapatkan, penulis dapat menyusun saran untuk penelitian lanjutan.

## BAB 4 PERANCANGAN

Pada bab ini dijelaskan tentang perancangan secara menyeluruh, menyertakan diagram alir setiap algoritme yang digunakan oleh penulis dalam sistem, serta rincian mengenai implementasi program yang dilakukan. Bab ini menjabarkan diagram alir *preprocessing* teks *tweet*, BM25F dan IKNN.

### 4.1 Gambaran Umum Sistem

Alur kerja secara general dari sistem yang akan dibangun oleh penulis dijabarkan pada bagian ini. Penelitian ini akan diterapkan dengan menggunakan bahasa pemrograman Python dengan menggunakan Jupyter Notebook sebagai *executable environment*, yaitu *environment* untuk mengeksekusi kode Python, dimana apabila kode terkait sudah selesai, dapat disimpan dalam format *script* python berekstensi .py.

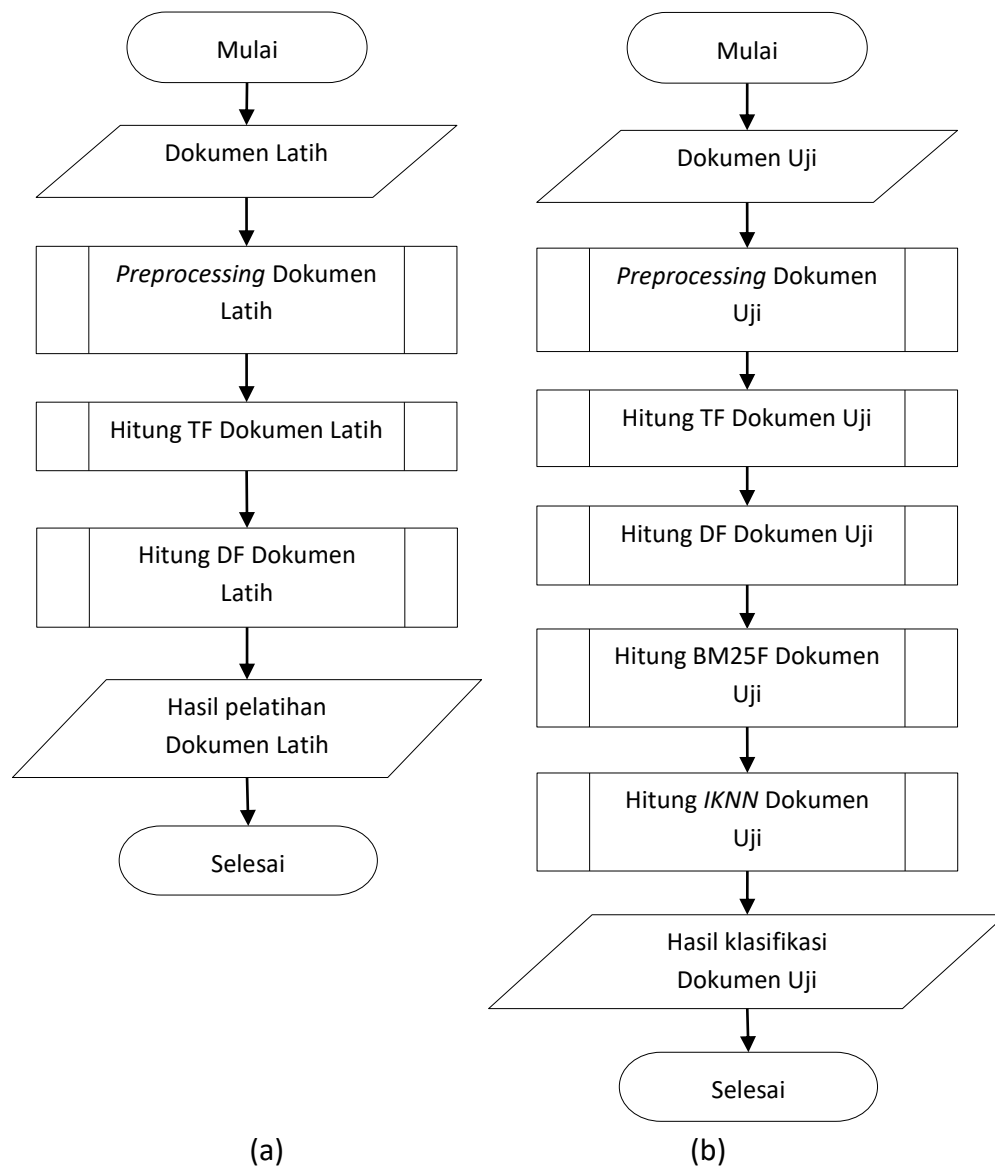
Sistem akan menerima masukan berupa dua dokumen berekstensi .csv, di antaranya ialah dokumen yang berisi 400 *tweet* dokumen latih dan 100 *tweet* data uji. Kedua dokumen memuat 2 kolom, yaitu kolom 'tweet' yang berisi kalimat *tweet* beserta tagar, dan kolom 'label' yang berisi label kelas masing-masing *tweet* yang telah ditentukan secara manual oleh penulis dengan bantuan pakar. Semua *tweet* akan dimasukkan ke dalam sistem untuk melalui *preprocessing*.

Pada penelitian ini, *preprocessing* yang dilakukan terdapat 5 tahap yaitu *cleaning*, *case folding*, tokenisasi, *split by uppercase*, *filtering* dan *stemming*. Keluaran setelah *preprocessing* berupa *term-term* pada masing-masing dokumen latih dan data uji. Proses dilanjutkan dengan BM25F, yang terdiri atas beberapa tahap, yakni perhitungan *tf*, *df*, panjang dokumen yang dinormalisasi ( $B_s$ ), *tf* yg dinormalisasi ( $\widetilde{tf}_i$ ), yaitu *tf* setiap *term* di masing-masing *stream* yang dikalikan dengan bobot unik *stream* tersebut kemudian dibagi dengan nilai  $B_s$ , kemudian perhitungan nilai IDF yang sudah dimodifikasi (*wIDF*) dan bobot akhir BM25F masing-masing dokumen latih.

Selanjutnya terdapat klasifikasi IKNN, yang dilakukan dalam 2 tahap, yaitu menentukan nilai  $k$  sesuai dengan proporsi jumlah data yang masuk ke masing-masing kelas dalam klasifikasi, dan diikuti dengan proses perhitungan peluang sebuah data uji masuk ke masing-masing kelas yang ada.

### 4.2 Diagram Alir Sistem

Pada bagian ini akan dijabarkan langkah-langkah secara umum yang dilakukan pada penelitian ini. Pada Gambar 4.1, terdapat diagram alir atau *flowchart* sistem secara umum. Bagian berikutnya diikuti oleh penjelasan beserta diagram alir dari langkah-langkah detail *preprocessing*, BM25F, dan *Improved K-Nearest Neighbor* (IKNN).



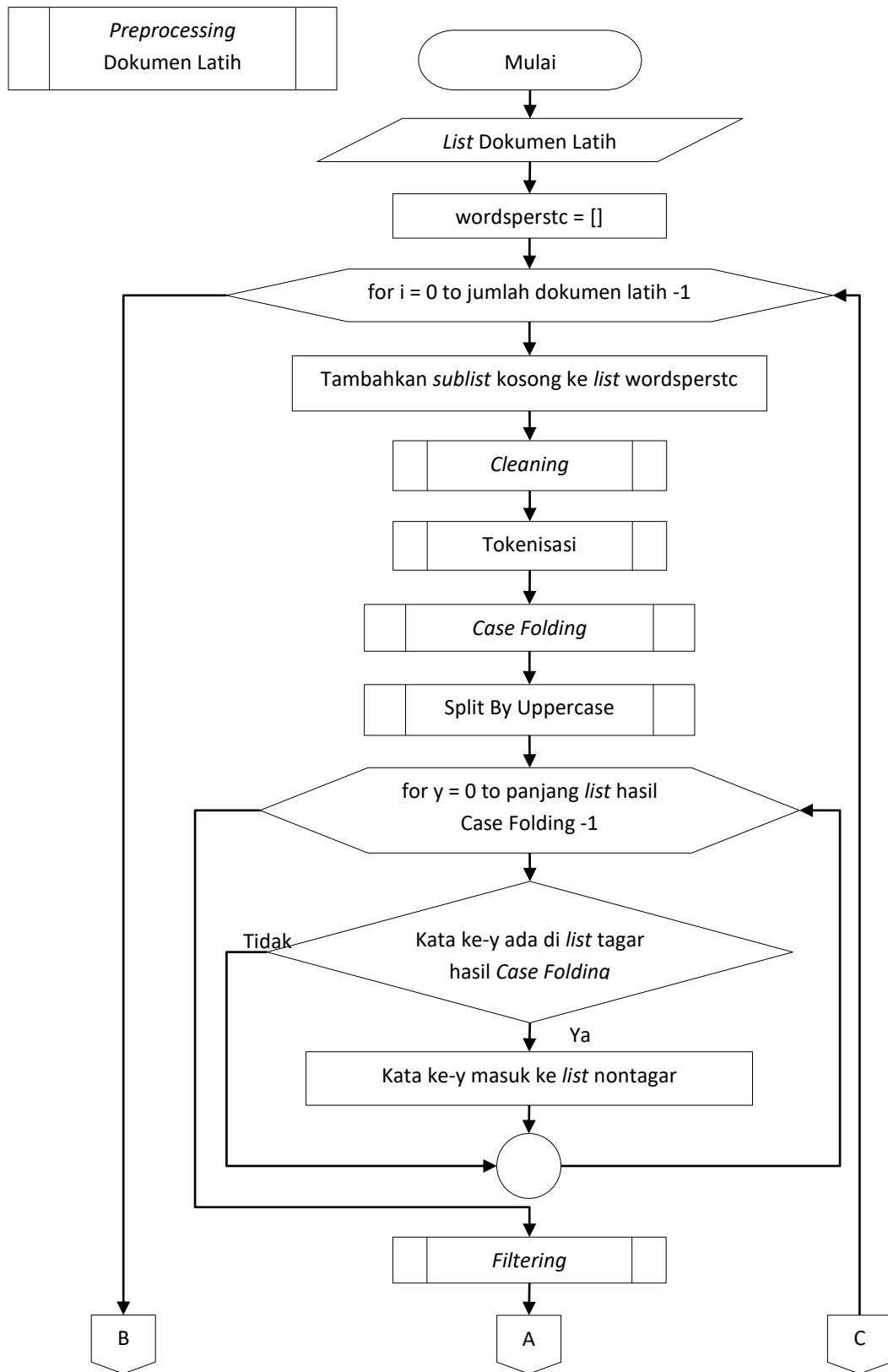
**Gambar 4.1 Diagram Alir Sistem, (a) Diagram Tahap Pelatihan, (b) Diagram Tahap Pengujian**

### 4.3 Diagram Alir Tahap Pelatihan

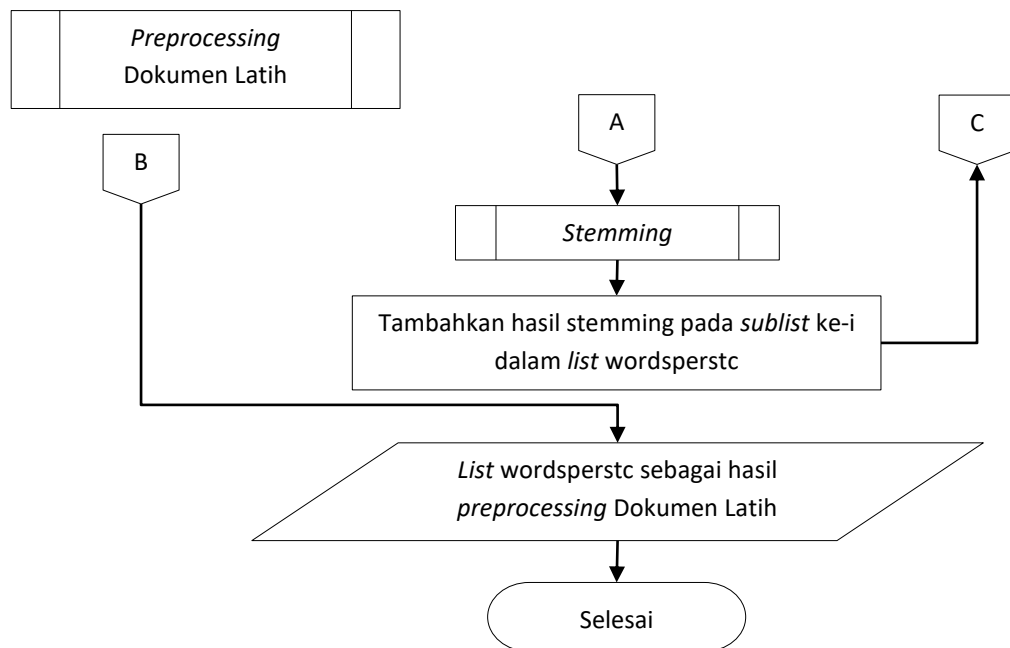
Penelitian ini melalui pelatihan terlebih dahulu sebelum masuk ke tahap pengujian, dimana pelatihan terbagi menjadi 3 tahap yaitu *preprocessing*, perhitungan TF dan perhitungan DF untuk dokumen latih.

#### 4.3.1 Diagram Alir Preprocessing

Pada bagian *preprocessing* terdapat beberapa tahap yang akan dilalui oleh dokumen masukan. Tahap pertama adalah *cleaning*, tokenisasi, *case folding*, *filtering*, dan *stemming*, menghapus awalan dan imbuhan kata. Alur kerja *preprocessing* untuk Dokumen Uji dan Dokumen Latih memiliki sedikit perbedaan, dan hal tersebut tampak pada Gambar 4.2 dan Gambar 4.3.

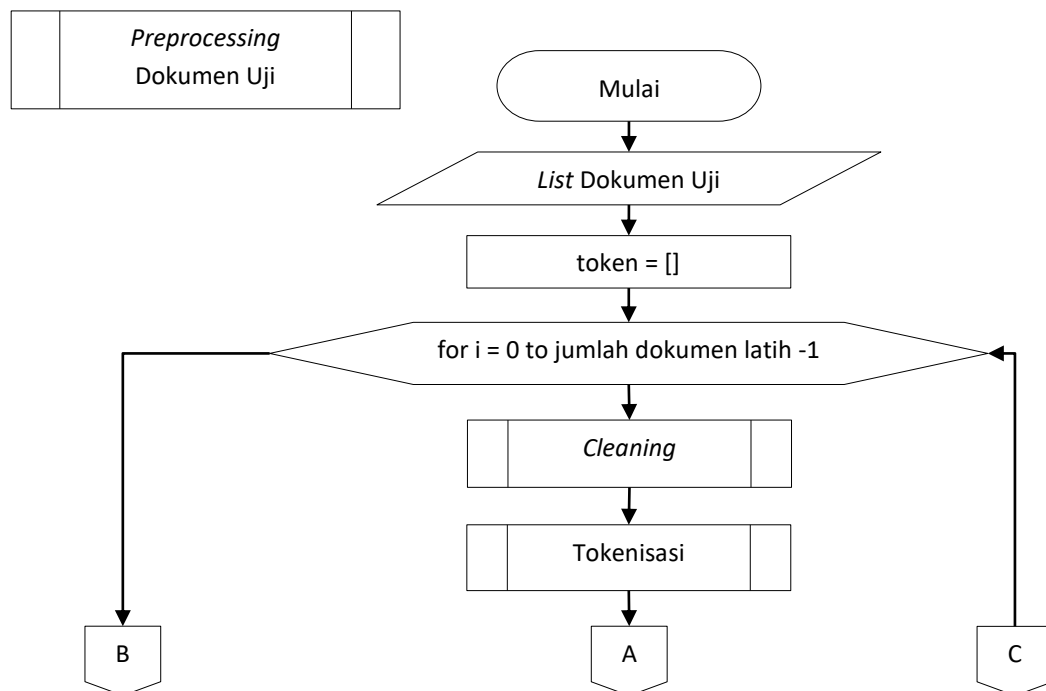


**Gambar 4.2 Diagram Alir *Preprocessing* Dokumen Latih**

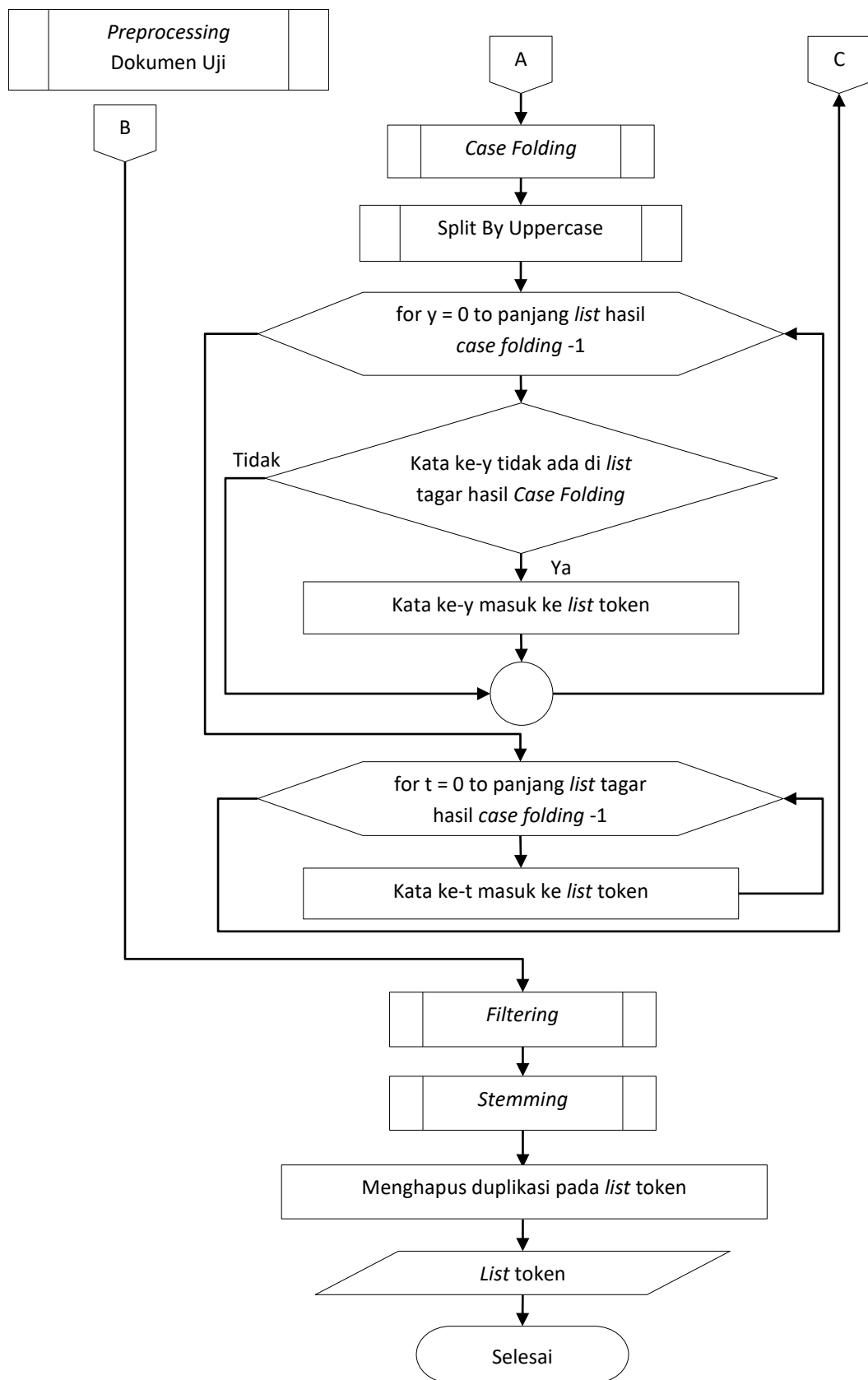


**Gambar 4.2 Diagram Alir *Preprocessing* Dokumen Latih(lanjutan)**

Pada Gambar 4.2, terdapat *sublist* sebanyak jumlah dokumen latih yang disimpan ke dalam *list wordsperstc* sebagai hasil akhir *preprocessing* dokumen latih. Setiap *sublist* tersebut berisi hasil *preprocessing* untuk setiap dokumen latih secara terpisah. Hal tersebut adalah yang membedakan *preprocessing* antara dokumen latih dengan dokumen uji.



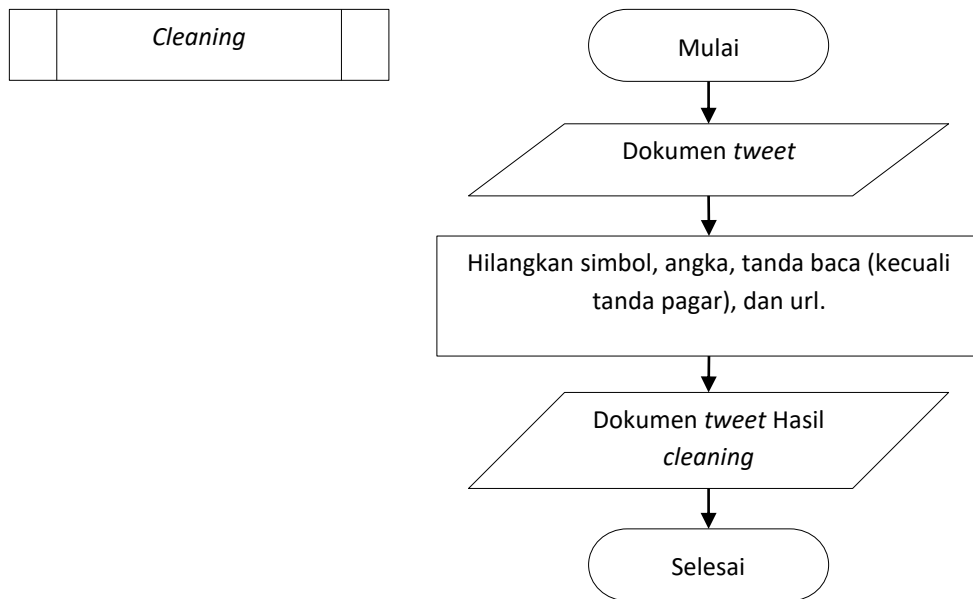
**Gambar 4.3 Diagram Alir *Preprocessing* Dokumen Uji**



**Gambar 4.3 Diagram Alir Preprocessing Dokumen Uji (lanjutan)**

### 4.3.2 Diagram Alir *Cleaning*

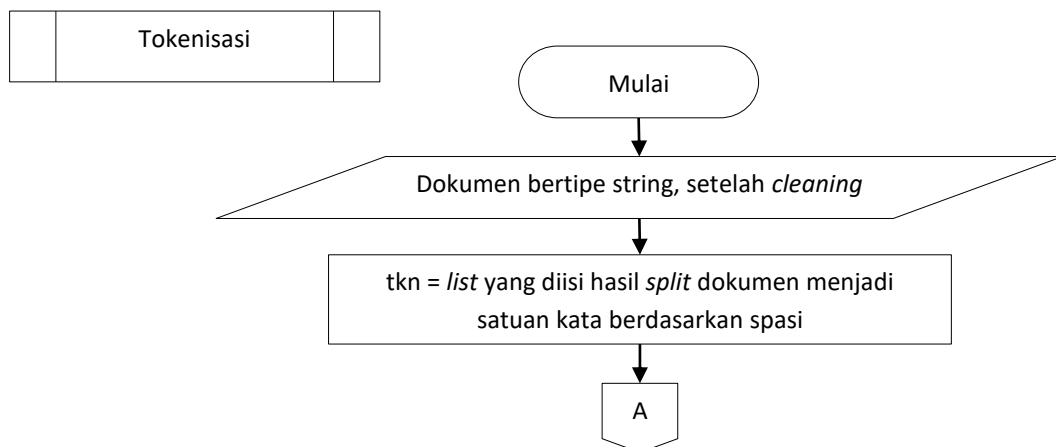
Tahap *preprocessing* yang pertama dilakukan dalam penelitian ini adalah *cleaning*. *Cleaning* merupakan sebuah proses penghilangan karakter tanda baca dan URL yang dilakukan pada dokumen latih dan maupun dokumen uji yang diinputkan. Tahap-tahap dalam proses *cleaning* dijabarkan dalam bentuk diagram alir yang pada Gambar 4.4.



Gambar 4.4 Diagram Alir *Cleaning*

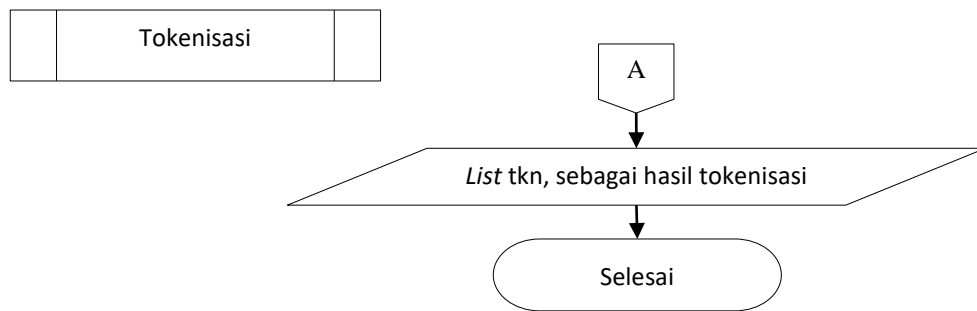
### 4.3.3 Diagram Alir Tokenisasi

Tahap *preprocessing* yang selanjutnya dilakukan dalam penelitian ini adalah tokenisasi. Tokenisasi adalah pemisahan (*split*) sebuah kalimat menjadi satuan kata. Umumnya pemisahan didasari oleh keberadaan spasi. Alur dari proses Tokenisasi dapat dilihat pada Gambar 4.5.



Gambar 4.5 Diagram Alir Tokenisasi

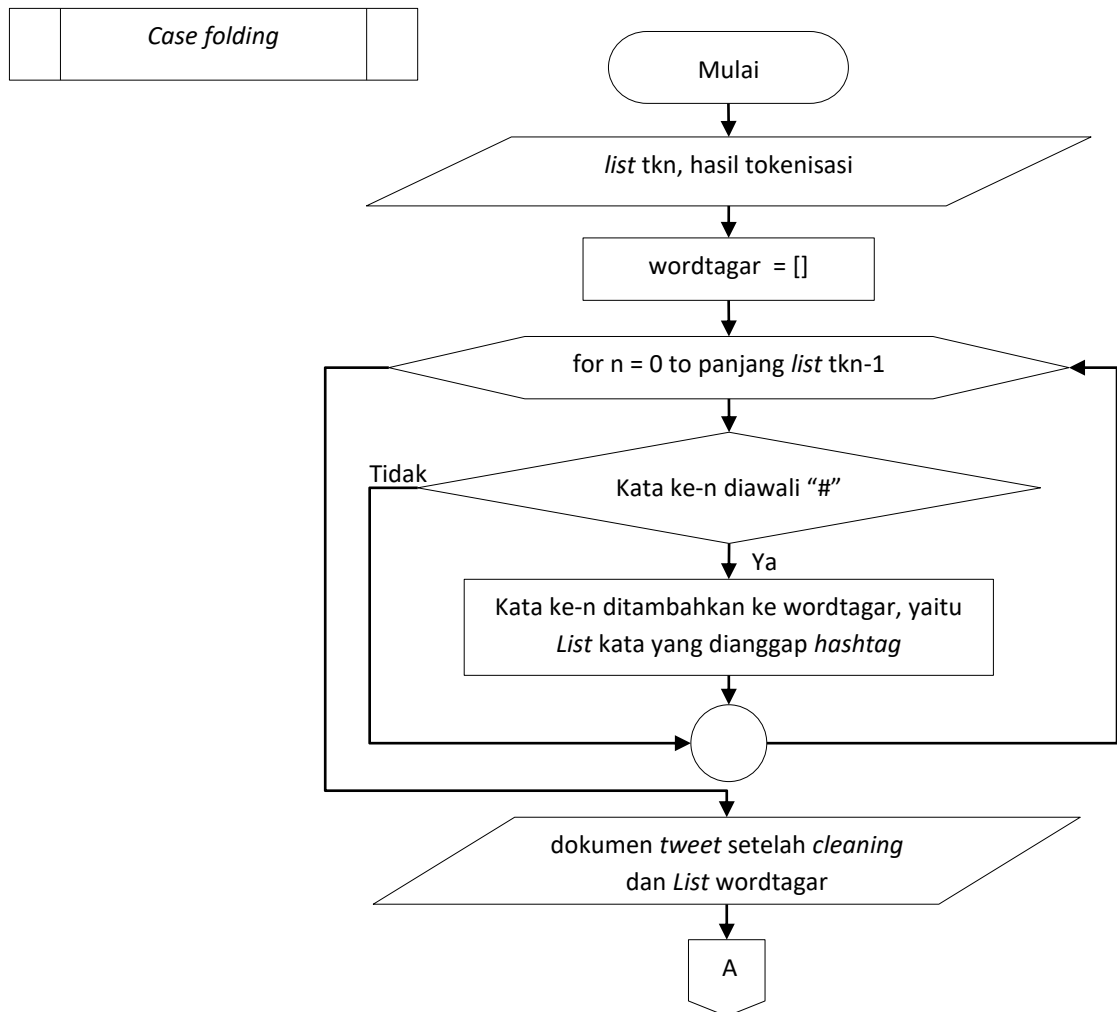




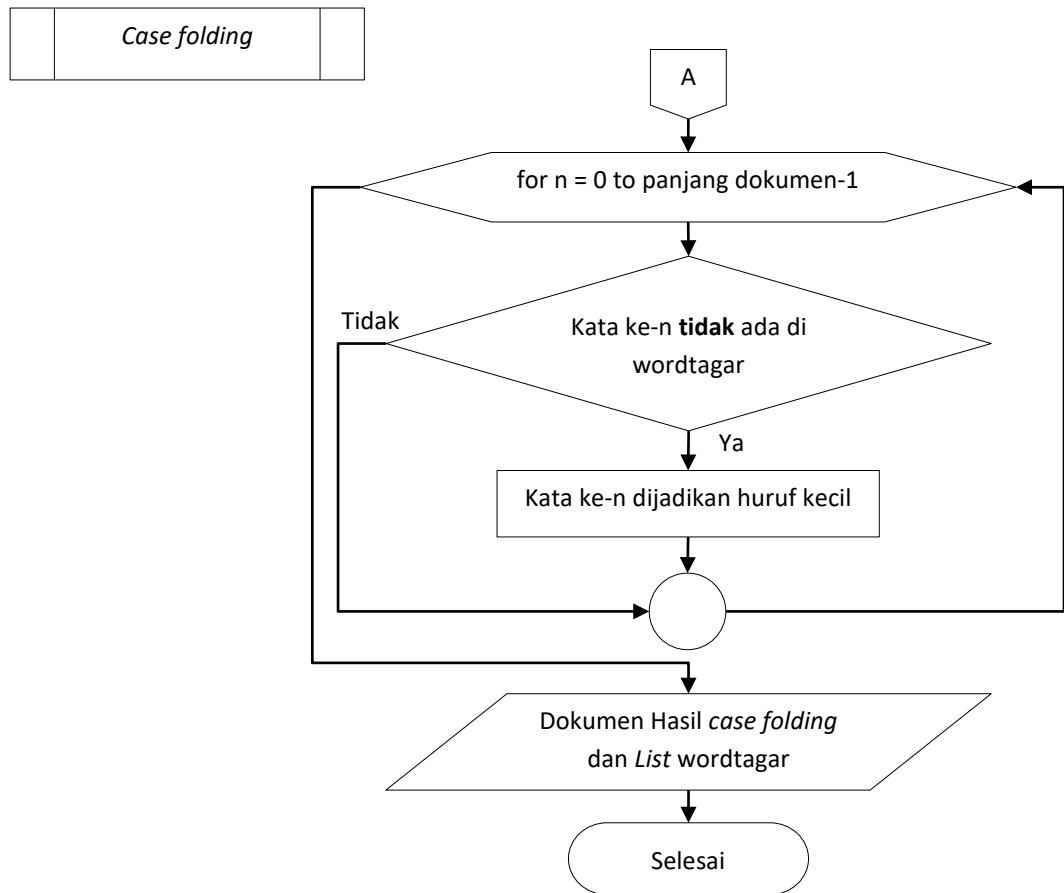
**Gambar 4.5 Diagram Alir Tokenisasi (lanjutan)**

#### 4.3.4 Diagram Alir *Case folding*

Tahap *preprocessing* yang selanjutnya dilakukan dalam penelitian ini adalah *case folding*. *Case folding* adalah proses mengubah karakter dalam dokumen yang sebelumnya adalah *uppercase* menjadi huruf kecil atau *lowercase*. Diagram Alir dari proses *case folding* terdapat pada Gambar 4.6.



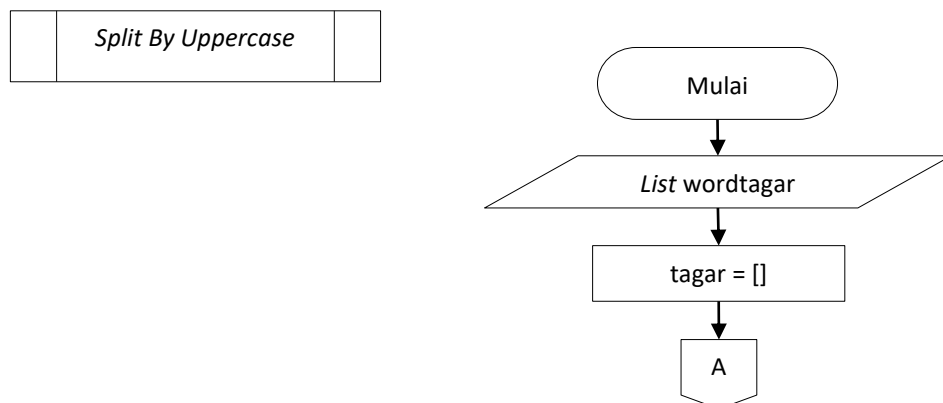
**Gambar 4.6 Diagram Alir Case Folding**



**Gambar 4.6 Diagram Alir Case folding (lanjutan)**

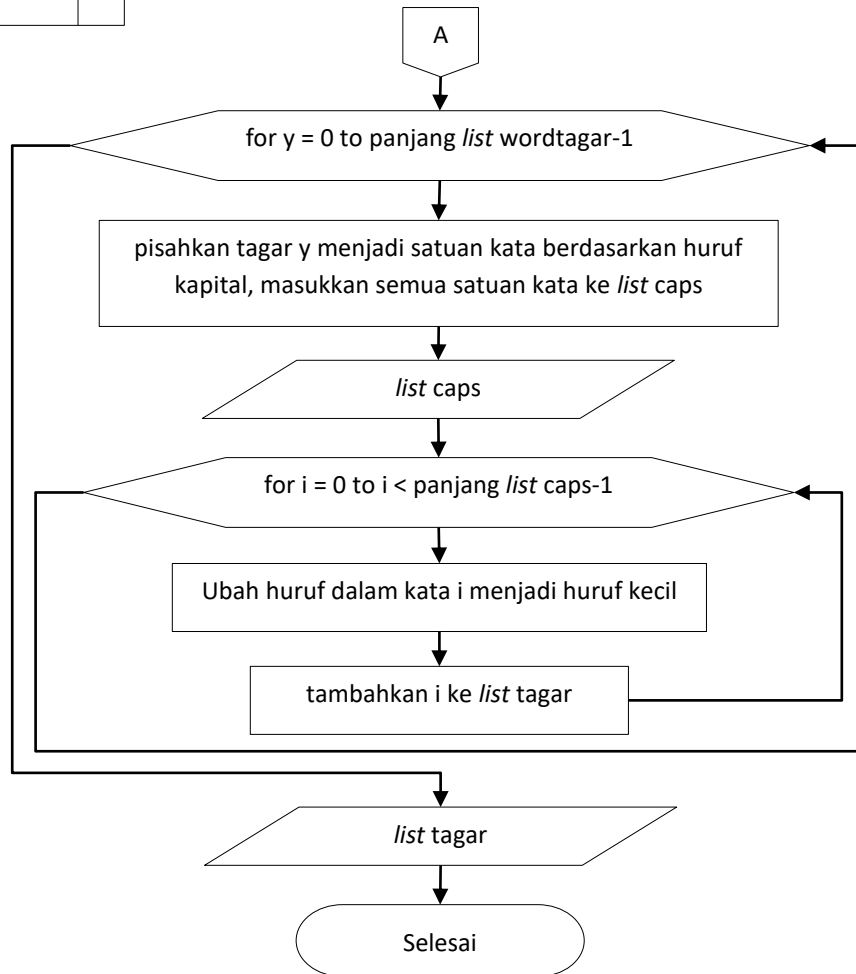
#### 4.3.5 Diagram Alir *Split By Uppercase*

Tahap *preprocessing* yang selanjutnya dilakukan dalam penelitian ini adalah *Split By Uppercase*. *Split By Uppercase* adalah pemisahan sebuah satuan kata menjadi beberapa kata yang lebih kecil berdasarkan keberadaan huruf kapitalnya (*uppercase*). Alur dari proses *Split By Uppercase* dapat dilihat pada Gambar 4.8.



**Gambar 4.7 Diagram Alir *Split By Uppercase***

	<i>Split By Uppercase</i>	
--	---------------------------	--

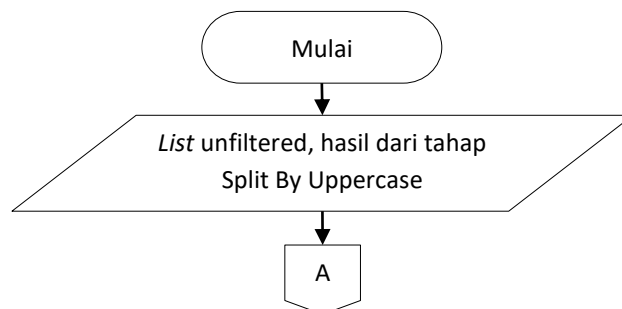


**Gambar 4.6 Diagram Alir *Split By Uppercase* (lanjutan)**

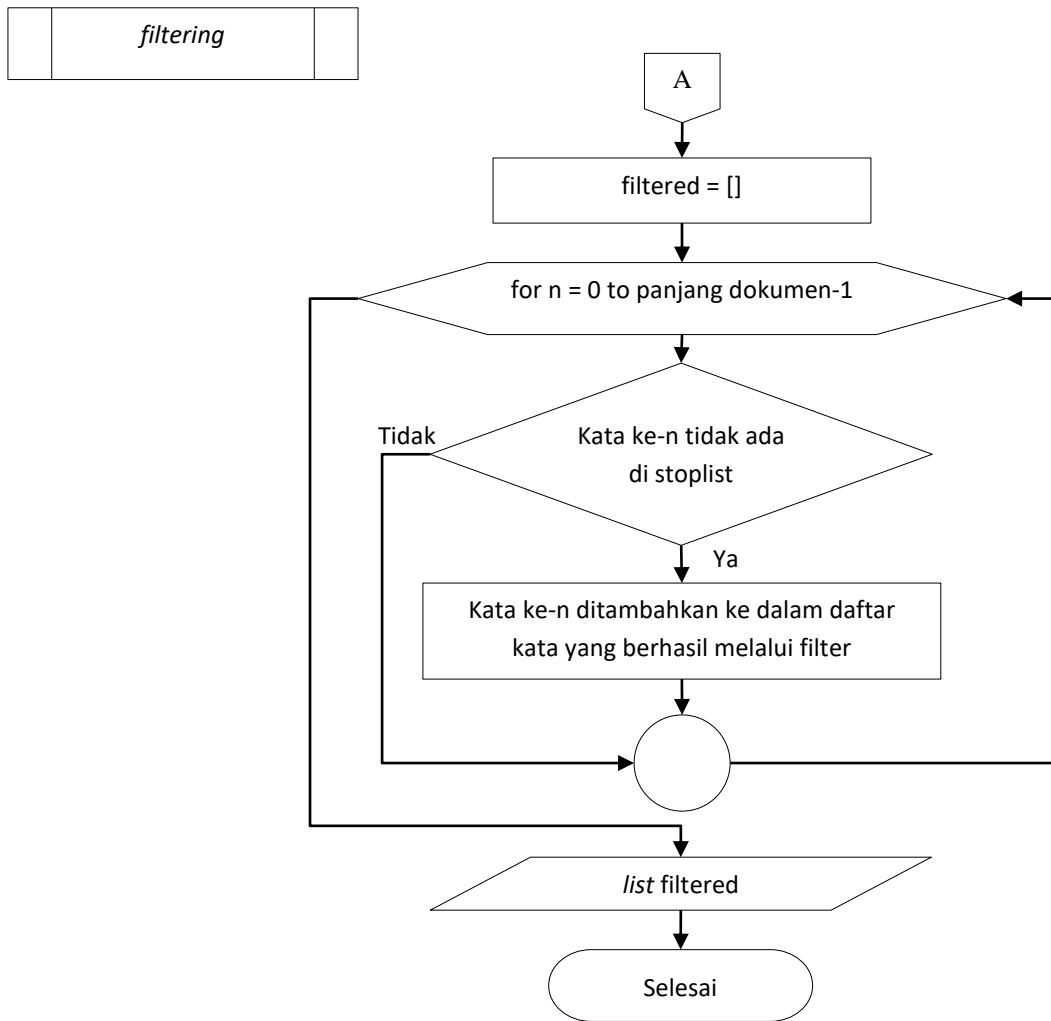
#### 4.3.6 Diagram Alir Filtering

*Filtering* adalah proses yang dilalui setelah *Case Folding* dan *Split By Uppercase*. *Filtering* merupakan proses menghapus kata *stopword* atau kata yang terdapat di dalam *stoplist*. Alur *filtering* ditunjukkan pada Gambar 4.7.

	<i>filtering</i>	
--	------------------	--



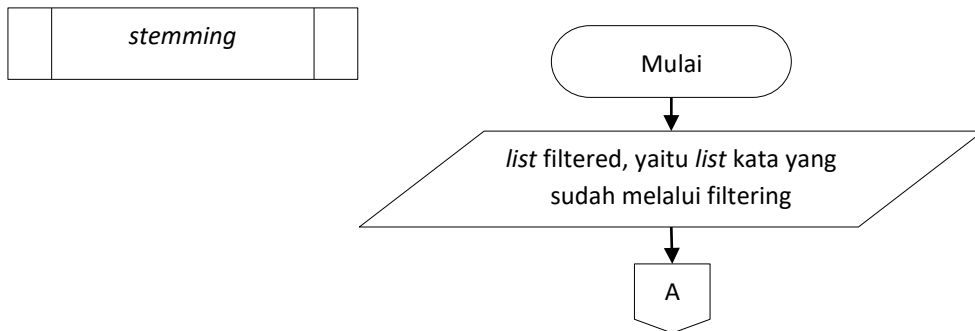
**Gambar 4.8 Diagram Alir *Filtering***



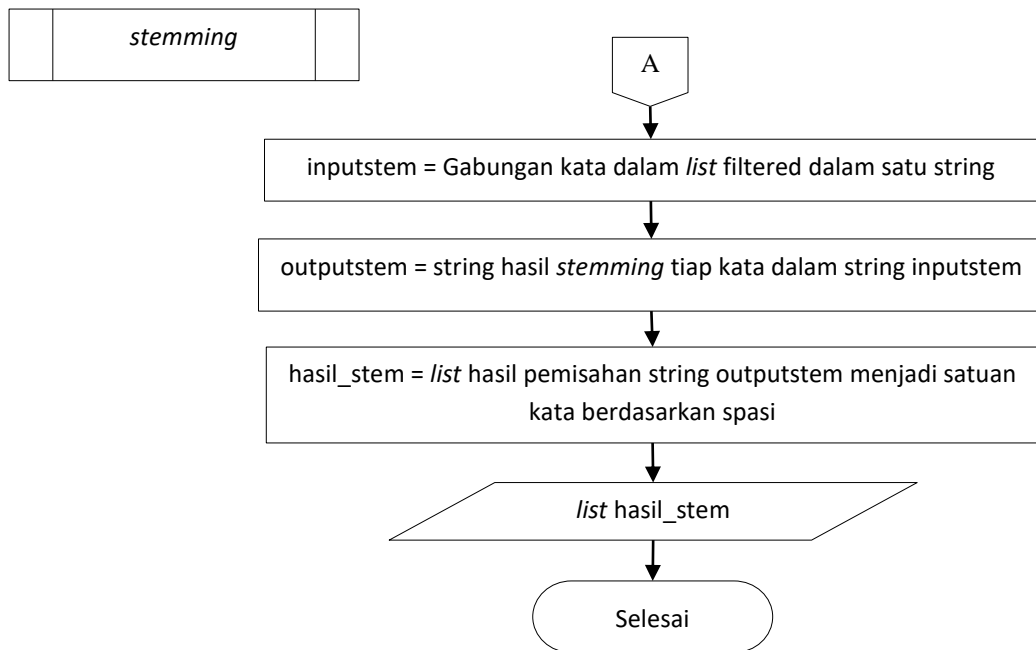
**Gambar 4.8 Diagram Alir *Filtering* (lanjutan)**

#### 4.3.7 Diagram Alir *Stemming*

*Stemming* adalah proses yang dilalui setelah *filtering*, yaitu proses mengubah kata yang berimbuhan maupun berawalan dalam sebuah kalimat kembali menjadi kata dasarnya. Alur *stemming* ditunjukkan pada Gambar 4.9.



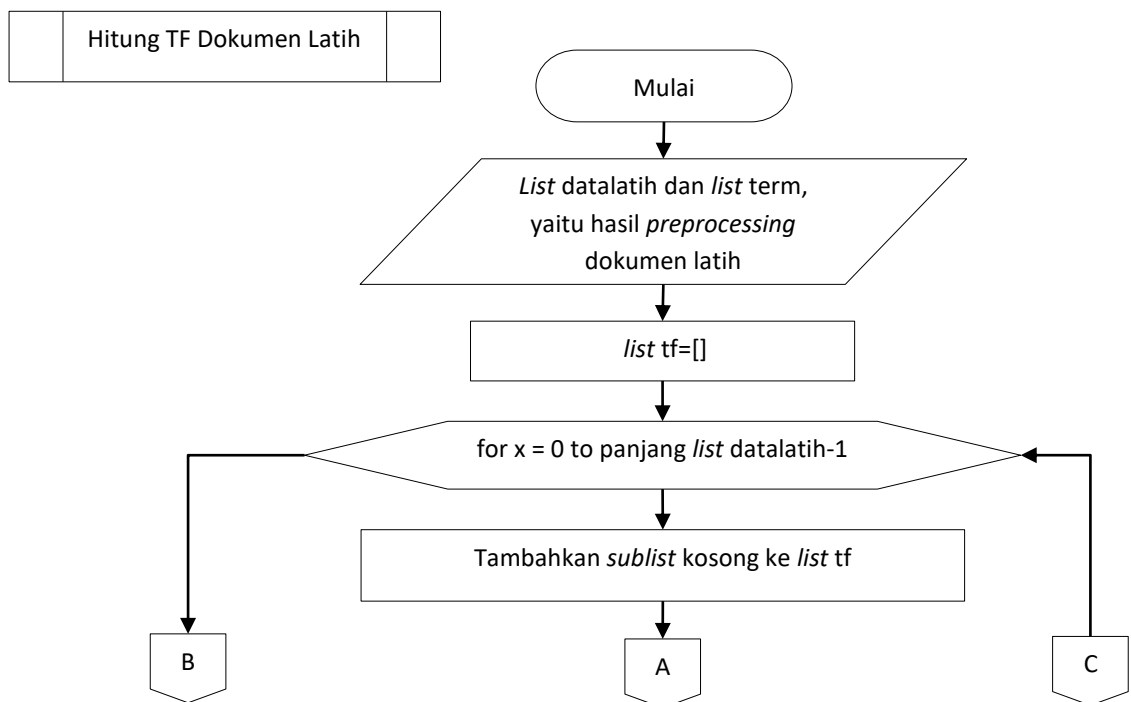
**Gambar 4.9 Diagram Alir *Stemming***



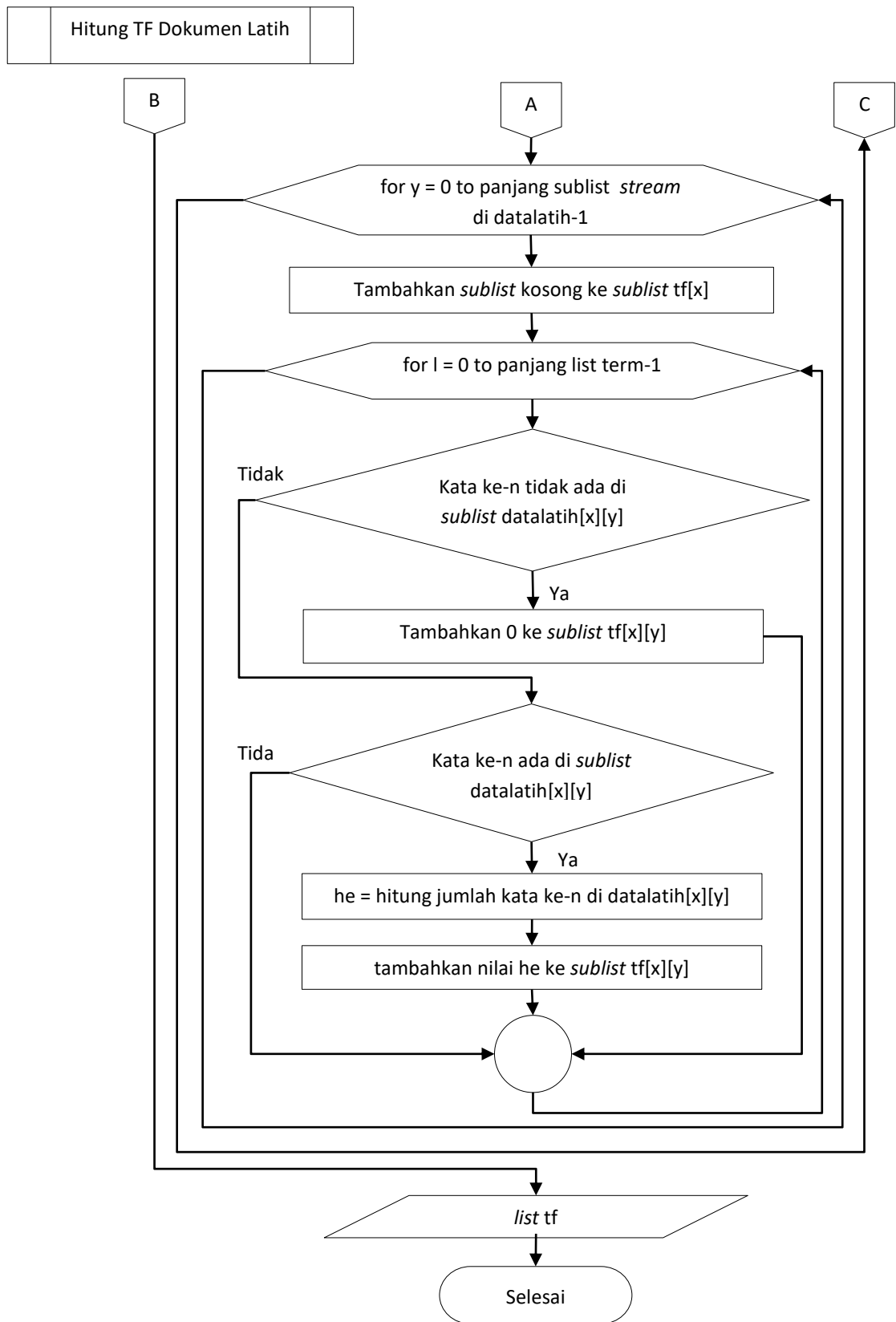
**Gambar 4.10 Diagram Alir Stemming**

#### 4.3.8 Diagram Alir *tf* Dokumen Latih

Setelah melalui *preprocessing*, dokumen latih akan melalui proses perhitungan *tf* (*term frequency*), yaitu menghitung munculnya sebuah *term* atau kata pada sebuah dokumen. Alur *tf* ditunjukkan pada Gambar 4.11.



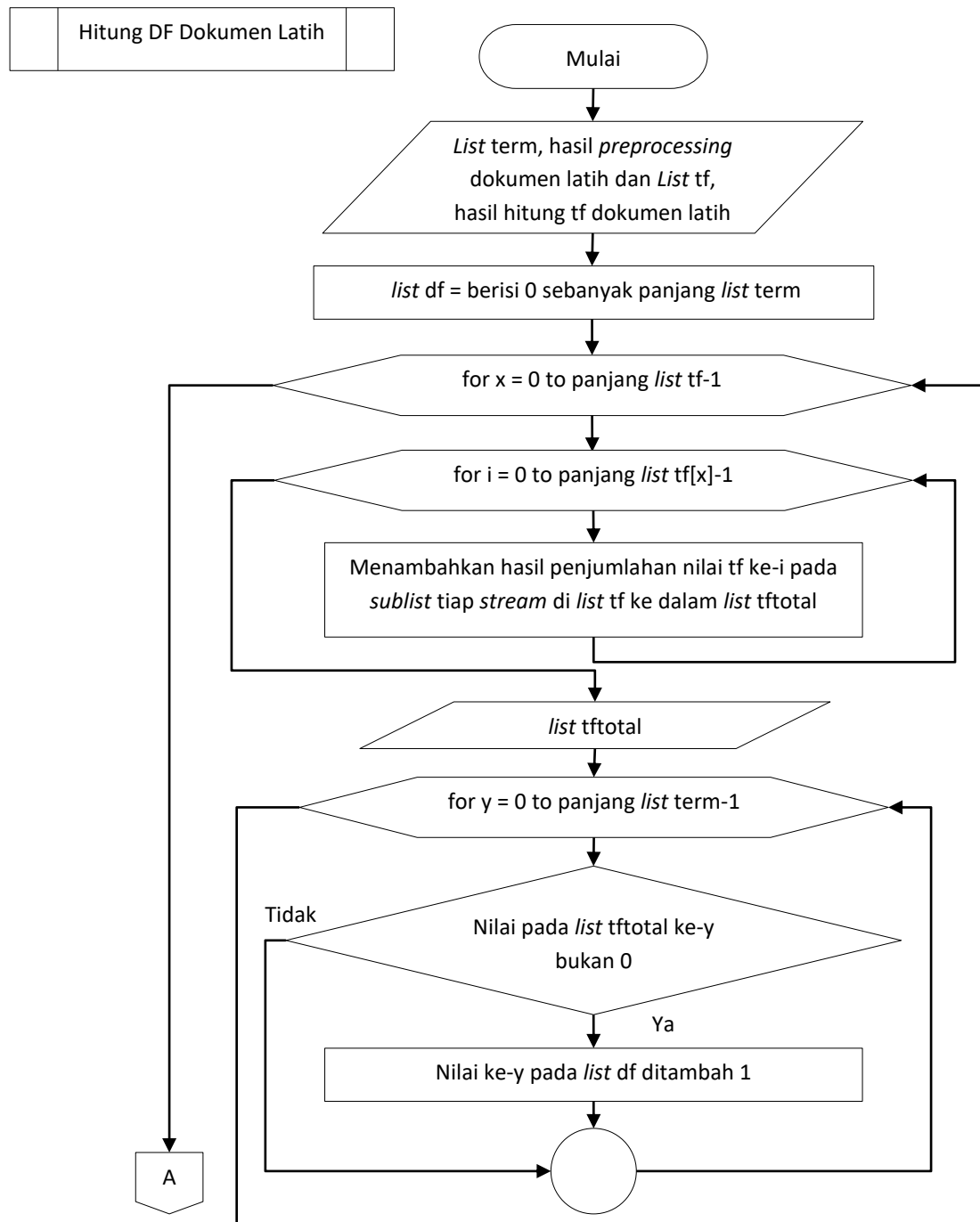
**Gambar 4.11 Diagram Alir *tf* Dokumen Latih**



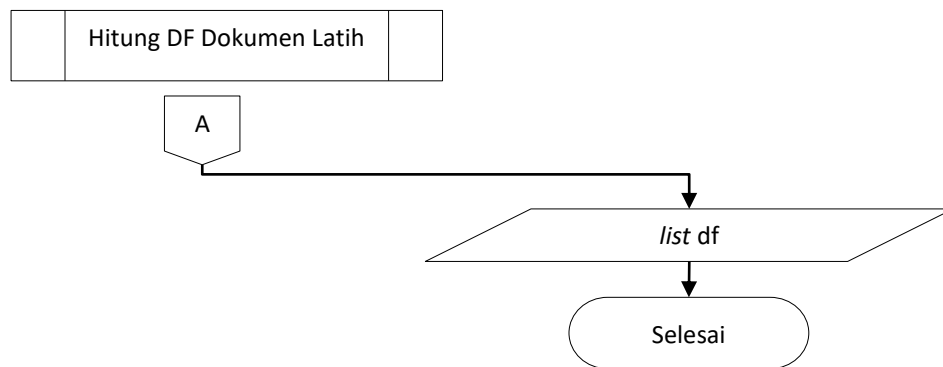
**Gambar 4.11 Diagram Alir *tf* Dokumen Latih (lanjutan)**

#### 4.3.9 Diagram Alir *df*

Setelah melalui perhitungan *tf*, dokumen latih akan melalui proses perhitungan *df* (*document frequency*), yaitu menghitung munculnya *term* atau kata pada berapa banyak dokumen latih. Alur *df* ditunjukkan pada Gambar 4.12.



Gambar 4.12 Diagram Alir *df* Dokumen Latih



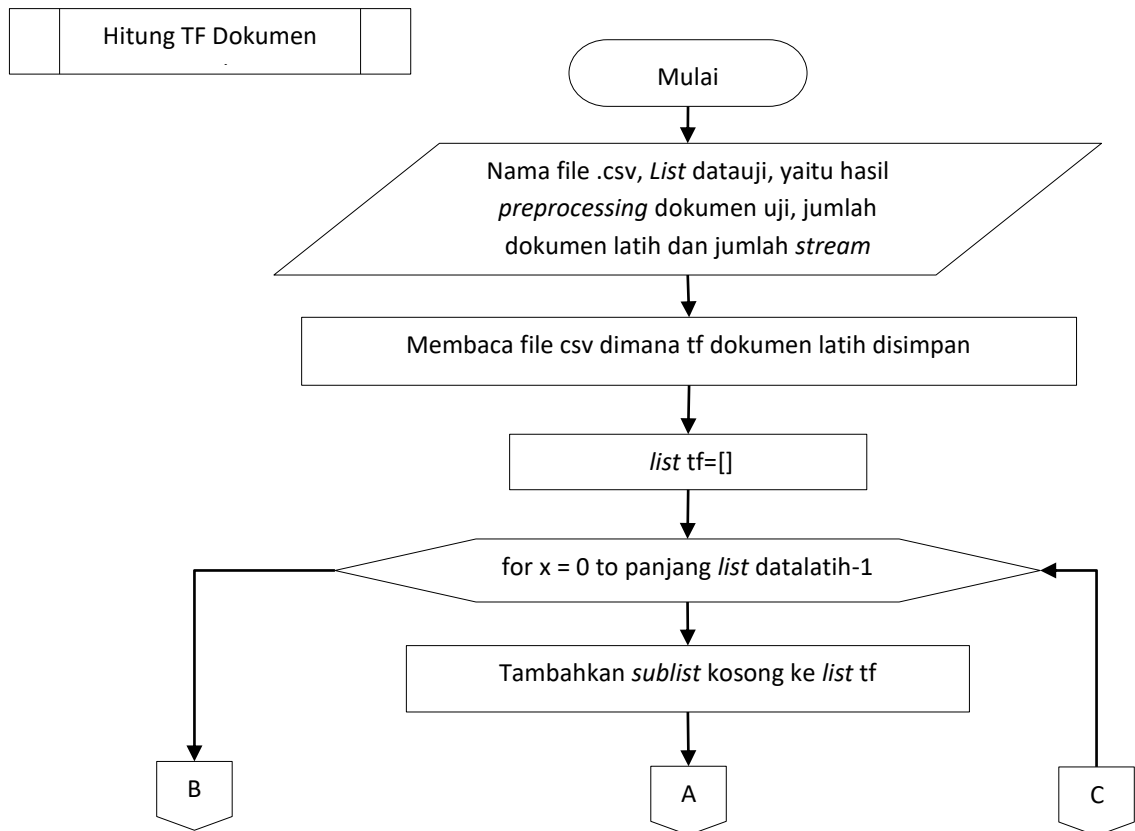
**Gambar 4.12 Diagram Alir *df* Dokumen Latih**

#### 4.4 Diagram Alir Tahap Pengujian

Setelah pelatihan, penelitian ini akan masuk ke tahap pengujian yang terbagi menjadi perhitungan BM25F dan klasifikasi IKNN untuk dokumen uji.

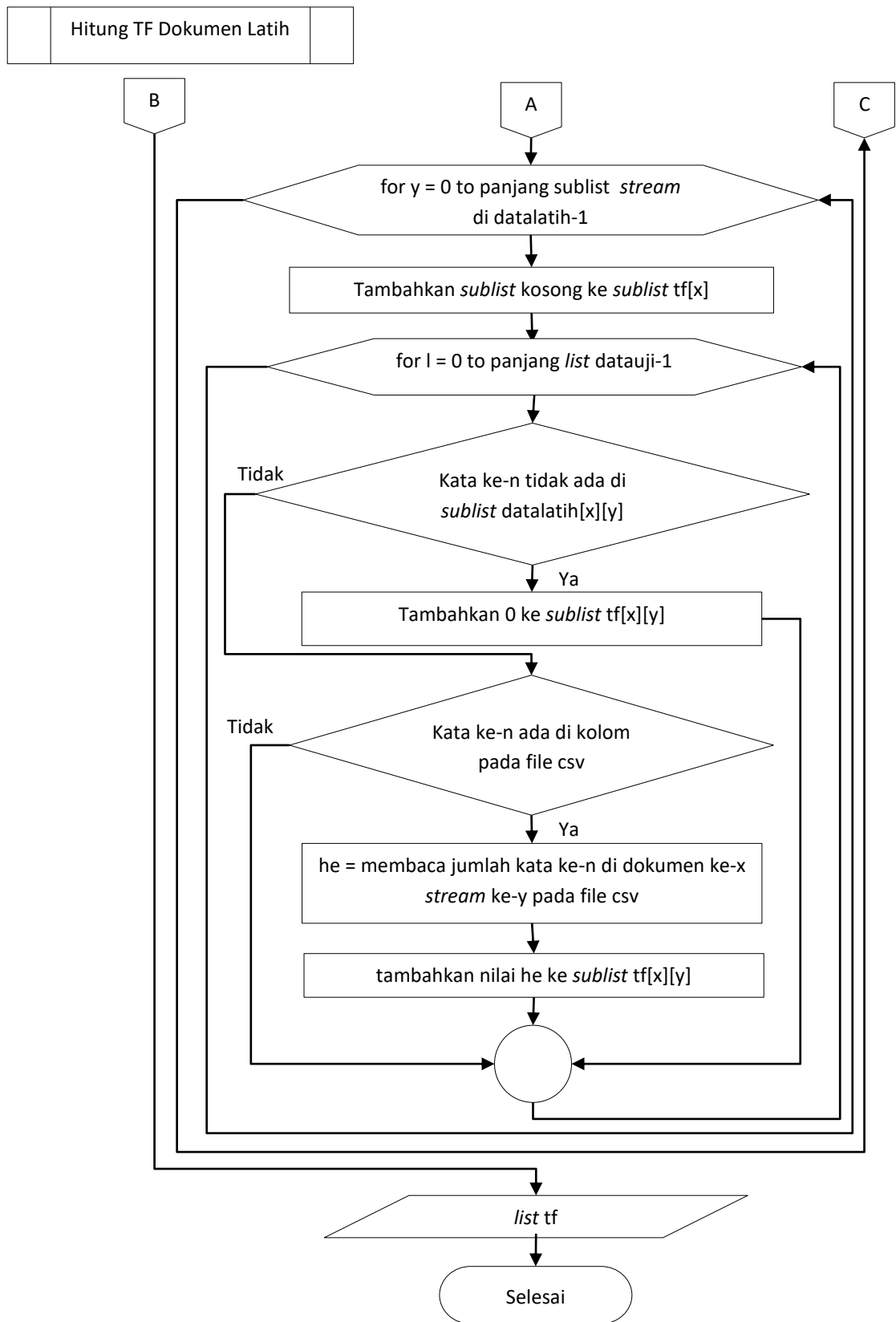
##### 4.4.1 Diagram Alir TF Dokumen Uji

Dokumen Uji akan melalui proses perhitungan *tf* (*term frequency*), yaitu menghitung munculnya *term* atau kata dari dokumen uji, di dalam dokumen latih. Alur *tf* ditunjukkan pada Gambar 4.13.



**Gambar 4.13 Diagram Alir *tf* Dkumen Uji**

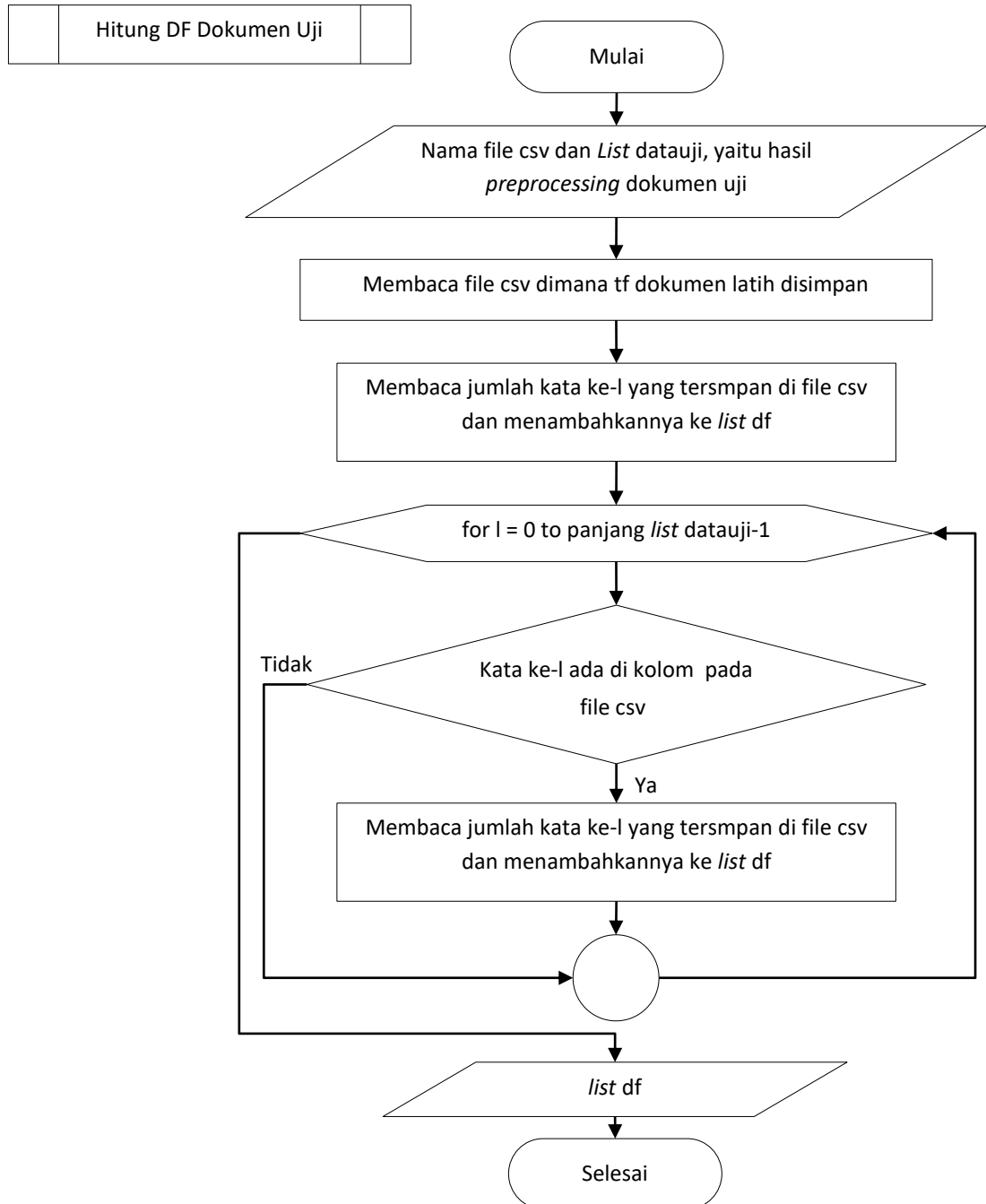




**Gambar 4.13 Diagram Alir  $tf$  Dokumen Uji (lanjutan)**

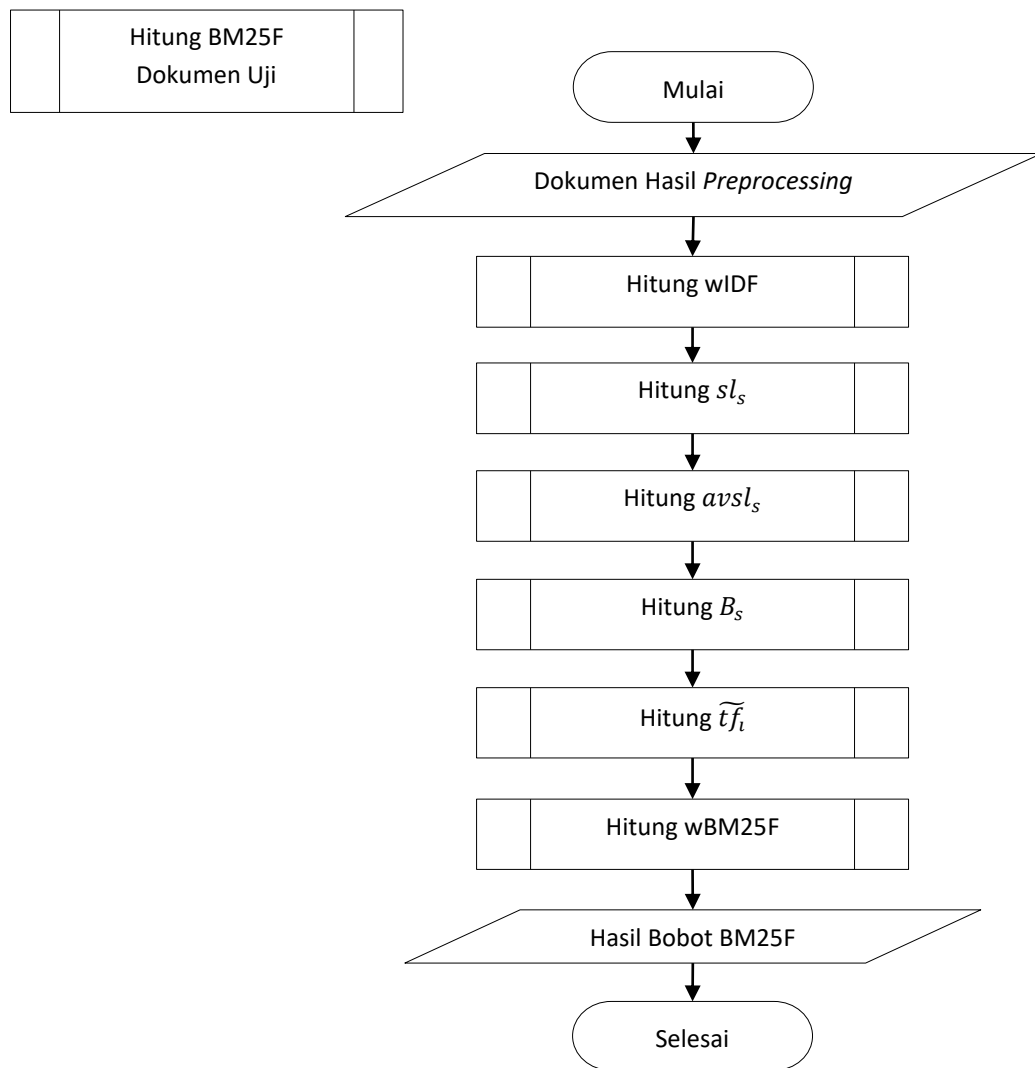
#### 4.4.2 Diagram Alir *df* Dokumen Uji

Dokumen Uji akan melalui proses perhitungan *tf* (*term frequency*), yaitu menghitung munculnya *term* atau kata dari dokumen uji, di dalam dokumen latih. Alur *tf* ditunjukkan pada Gambar 4.13.



#### 4.4.3 Diagram Alir BM25F

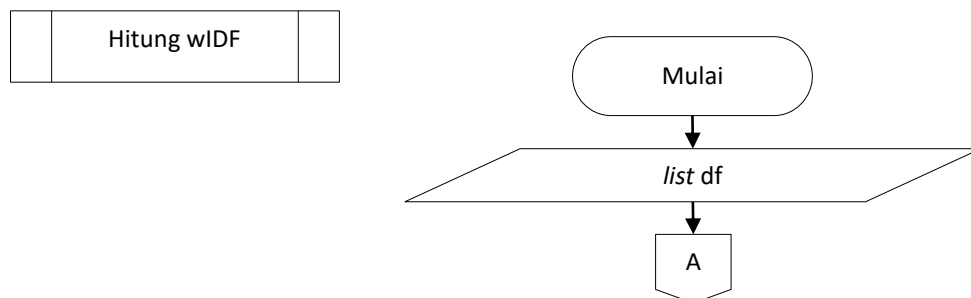
Setelah melalui *preprocessing*, kata yang sudah didapatkan akan dihitung bobotnya dengan BM25F. Dalam metode ini, terdapat perhitungan *tf*, *idf*, dan sebagainya. Alur dari metode BM25F ditunjukkan pada Gambar 4.10.



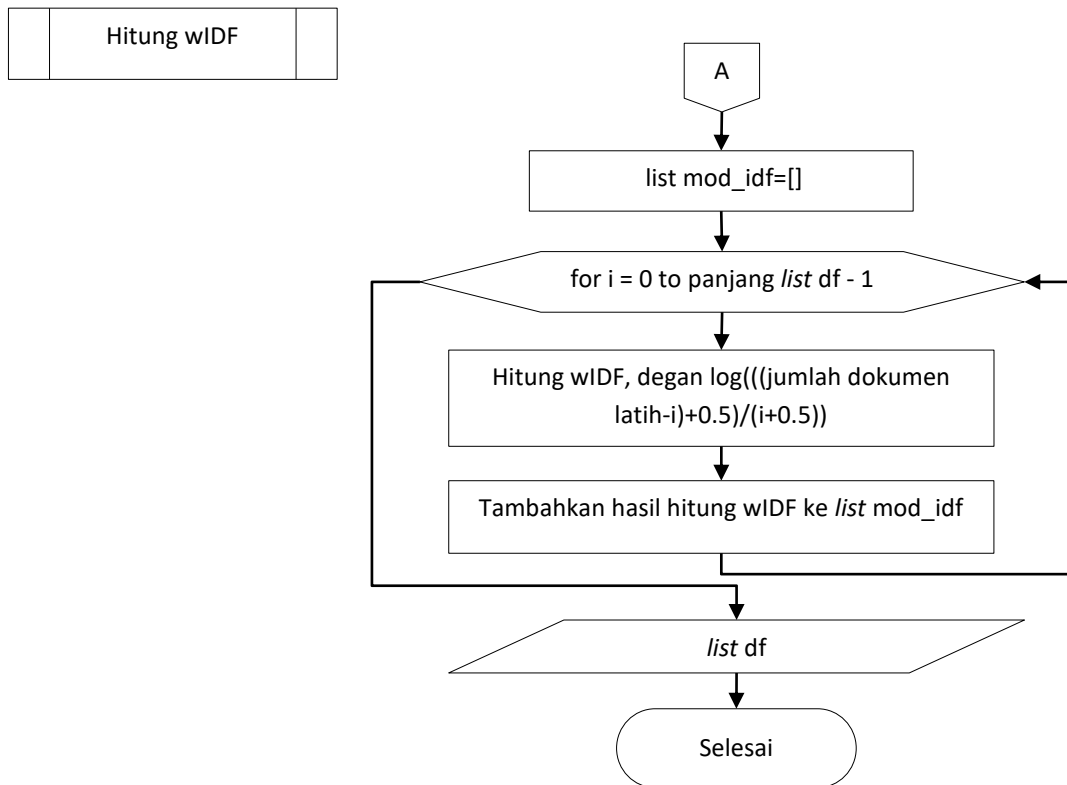
**Gambar 4.14 Diagram Alir BM25F**

#### 4.4.4 Diagram Alir wIDF

Tahap pertama dari BM25F adalah menghitung nilai wIDF, yaitu perhitungan IDF yang telah dimodifikasi setelah melalui *preprocessing*. Alur perhitungan wIDF ditunjukkan pada Gambar 4.15.



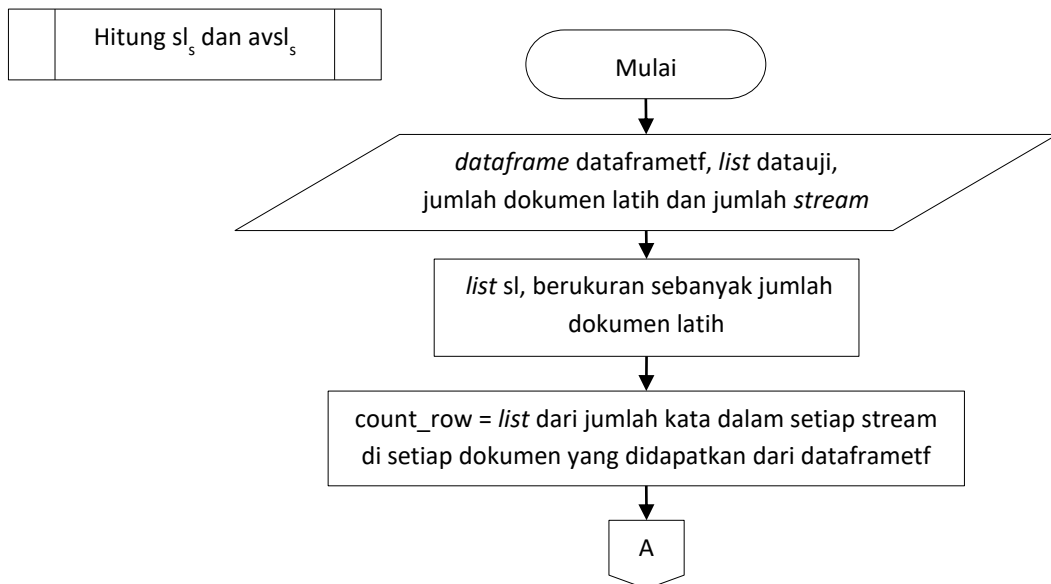
**Gambar 4.15 Diagram Alir wIDF**



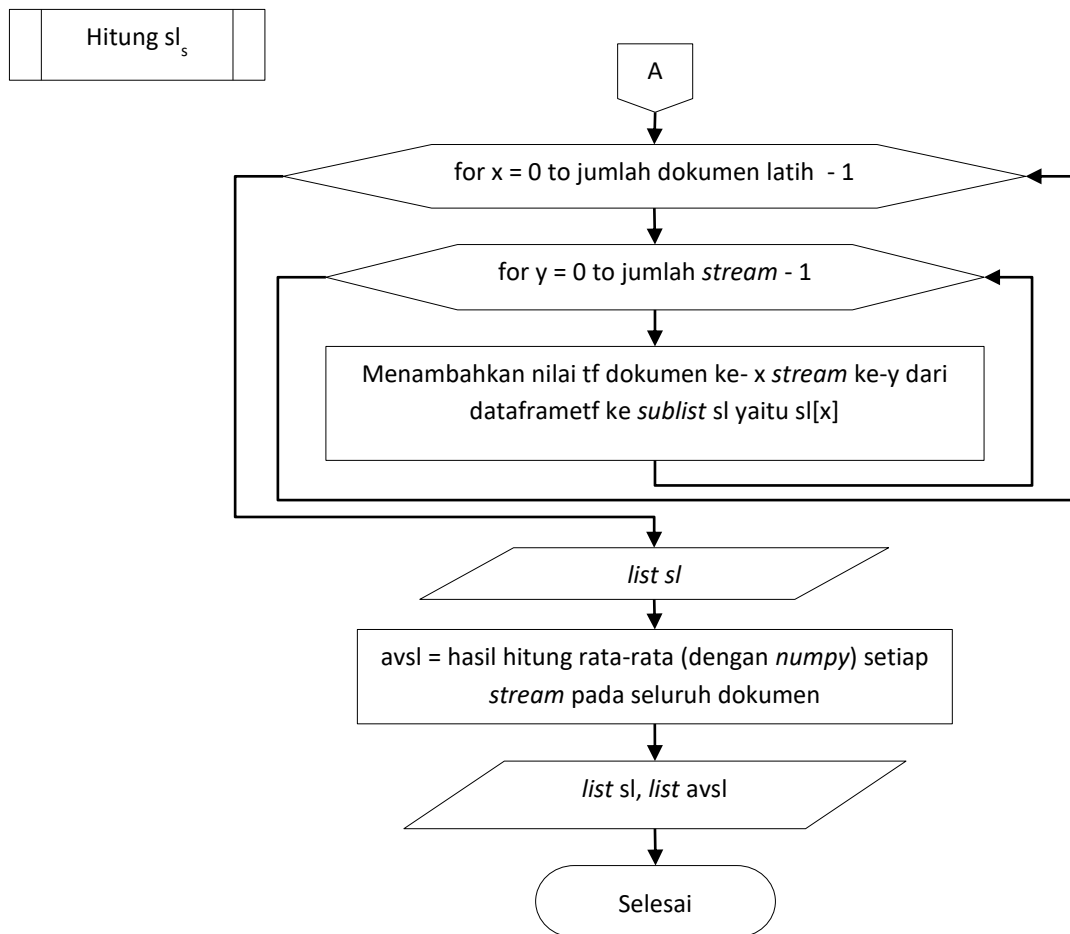
**Gambar 4.15 Diagram Alir wIDF (lanjutan)**

#### 4.4.5 Diagram Alir $sl_s$ dan $avsl_s$

Tahap selanjutnya dari BM25F adalah menghitung nilai  $sl_s$ , yaitu perhitungan jumlah *term* di masing-masing *stream* pada tiap dokumen latih setelah melalui *preprocessing*. Alur perhitungan  $sl_s$  ditunjukkan pada Gambar 4.16.



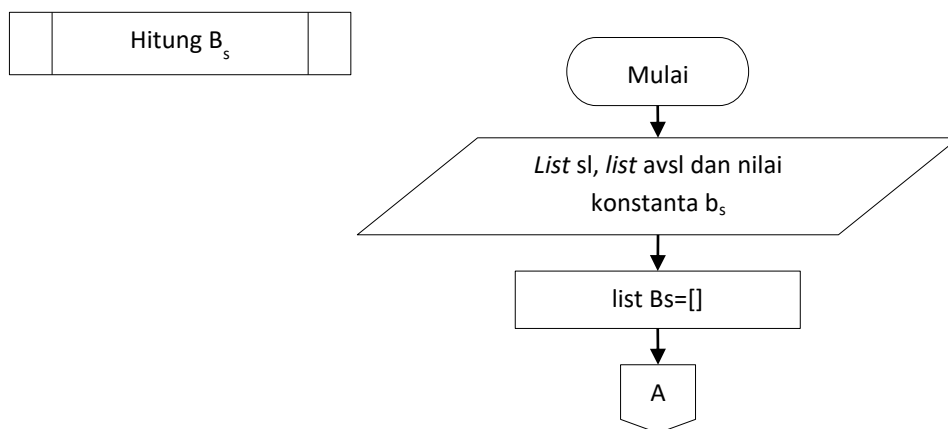
**Gambar 4.16 Diagram Alir  $avsl_s$  dan  $sl_s$**



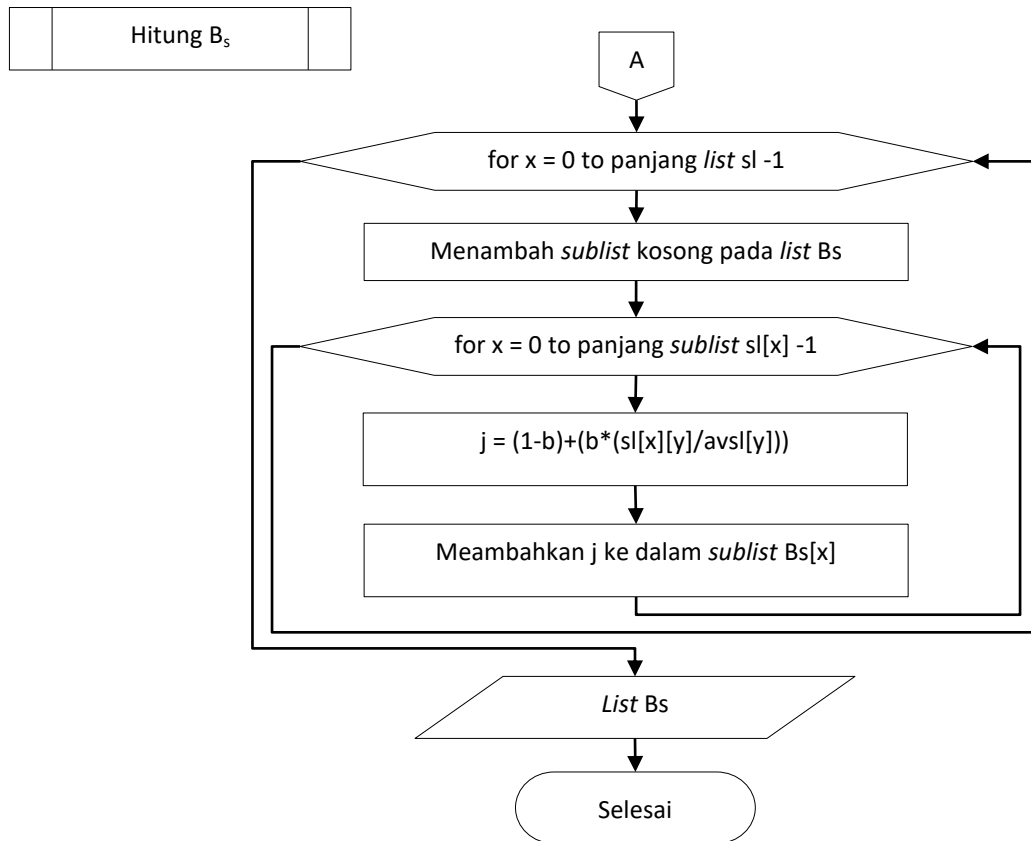
**Gambar 4.16 Diagram Alir  $sl_s$  dan  $avsl_s$  (lanjutan)**

#### 4.4.6 Diagram Alir $B_s$

Tahap selanjutnya dari BM25F adalah menghitung nilai  $B_s$ , yaitu perhitungan panjang tiap *stream* dari tiap dokumen latih setelah melalui *preprocessing*. Alur perhitungan  $B_s$  ditunjukkan pada Gambar 4.17.



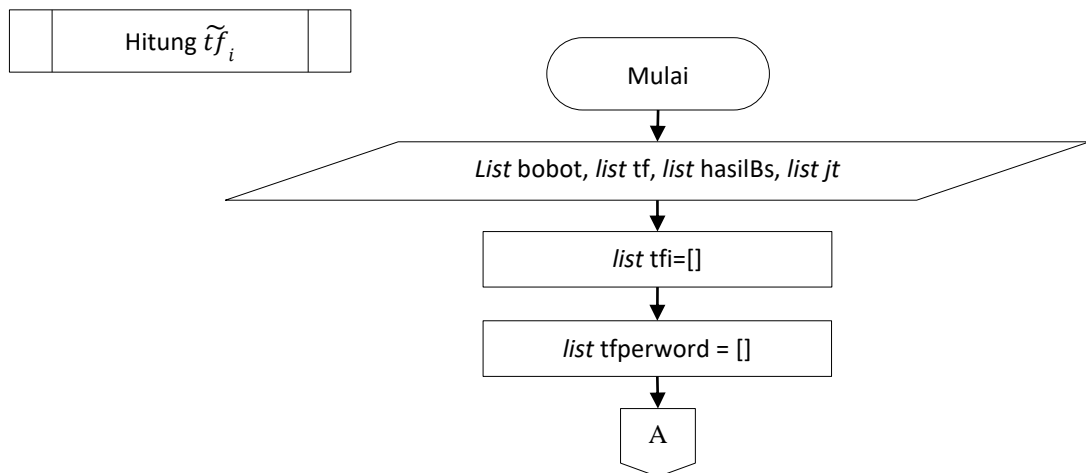
**Gambar 4.17 Diagram Alir  $B_s$**



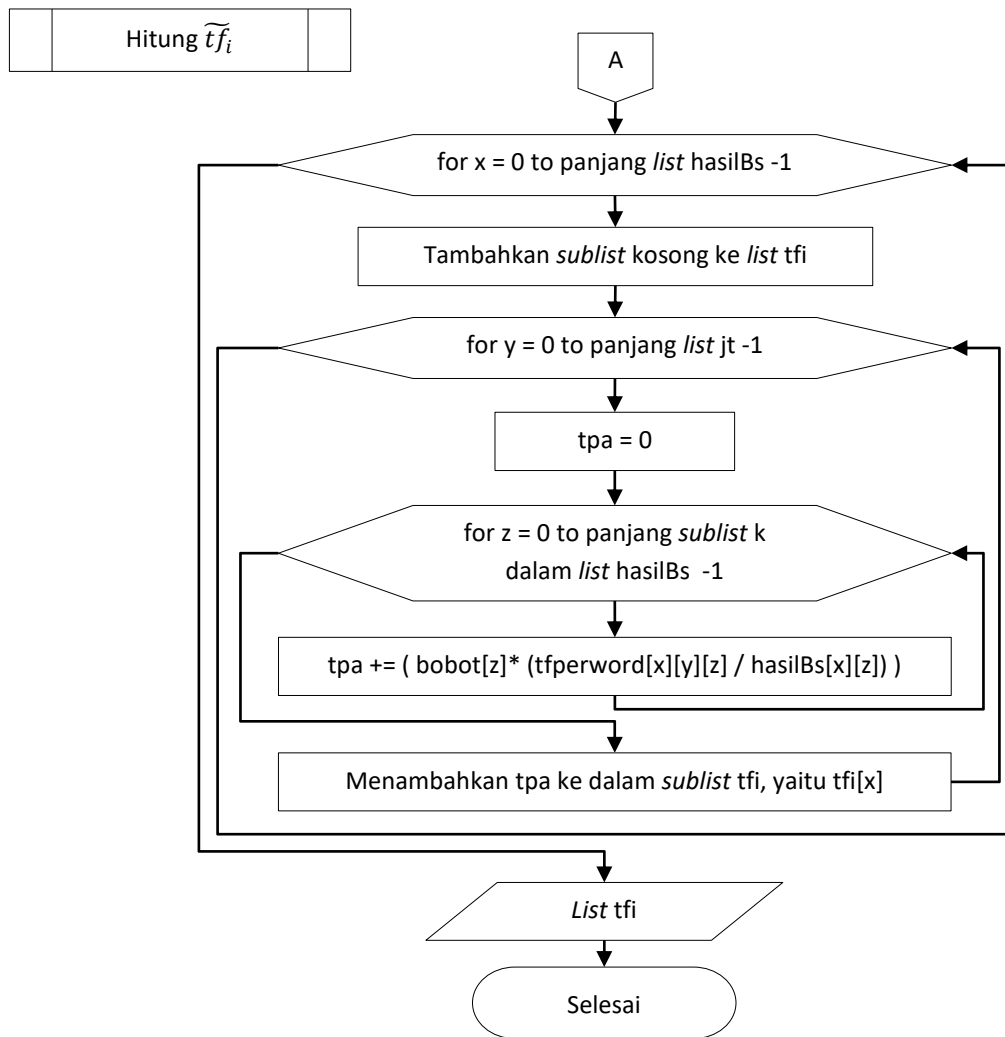
**Gambar 4.17 Diagram Alir  $B_s$  (lanjutan)**

#### 4.4.7 Diagram Alir $\widetilde{tf}_i$

Tahap selanjutnya dari BM25F adalah menghitung nilai  $\widetilde{tf}_i$ , yaitu perhitungan total jumlah kemunculan *term* (*term frequency* atau *tf*) yang dinormalisasi (*normalized tf*) pada masing-masing *stream* dokumen latih setelah melalui *preprocessing*. Alur perhitungan  $\widetilde{tf}_i$  ditunjukkan sebagai ntf pada Gambar 4.18.



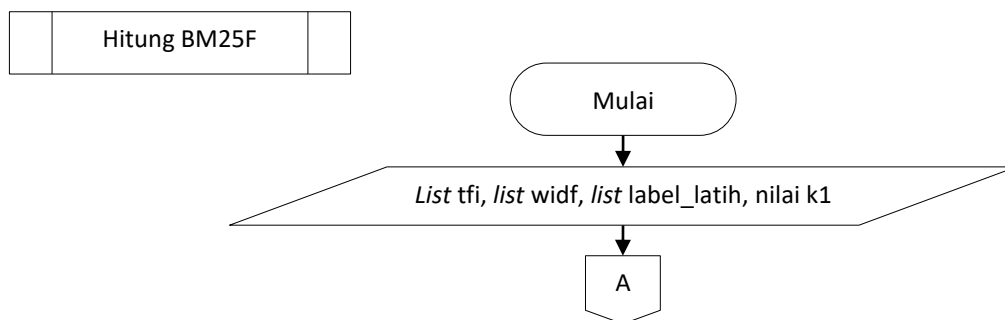
**Gambar 4.18 Diagram Alir  $\widetilde{tf}_i$**



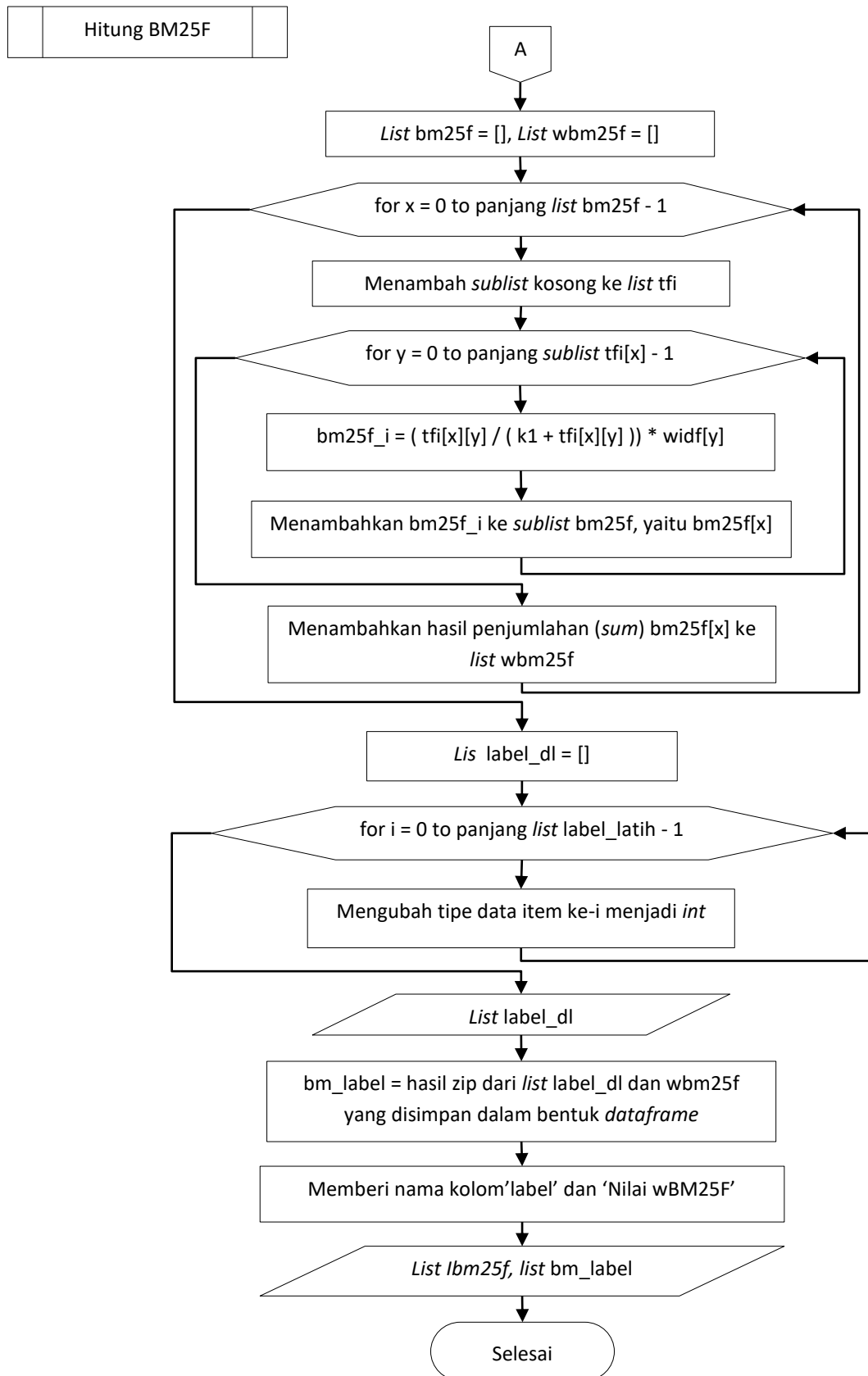
**Gambar 4.18 Diagram Alir  $\widetilde{tf}_i$  (lanjutan)**

#### 4.4.8 Diagram Alir BM25F

Tahap selanjutnya dari BM25F adalah tahap menghitung total BM25F tiap dokumen latih. Alur perhitungan BM25F ditunjukkan pada Gambar 4.19.



**Gambar 4.19 Diagram Alir wBM25F**

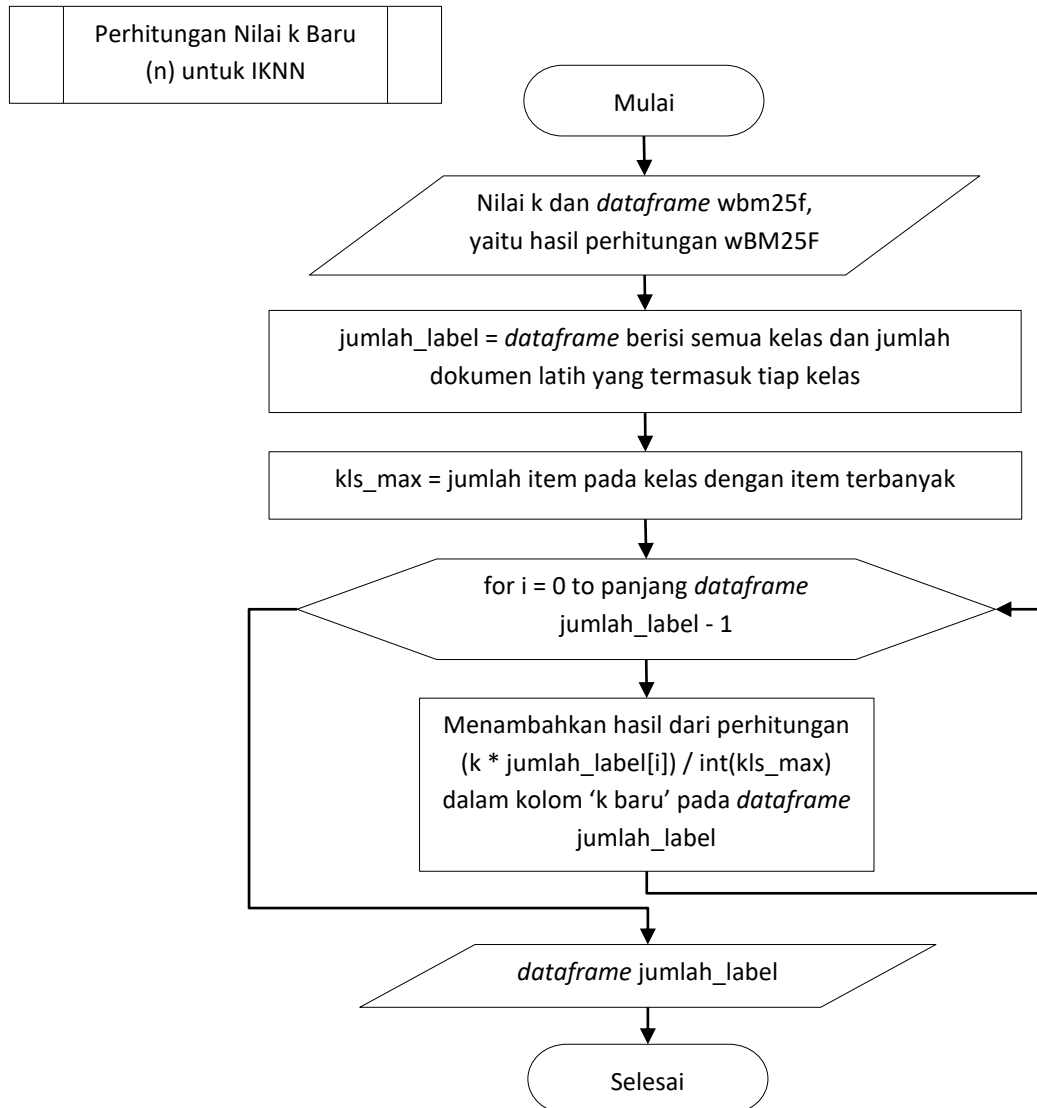


**Gambar 4.19 Diagram Alir wBM25F (lanjutan)**



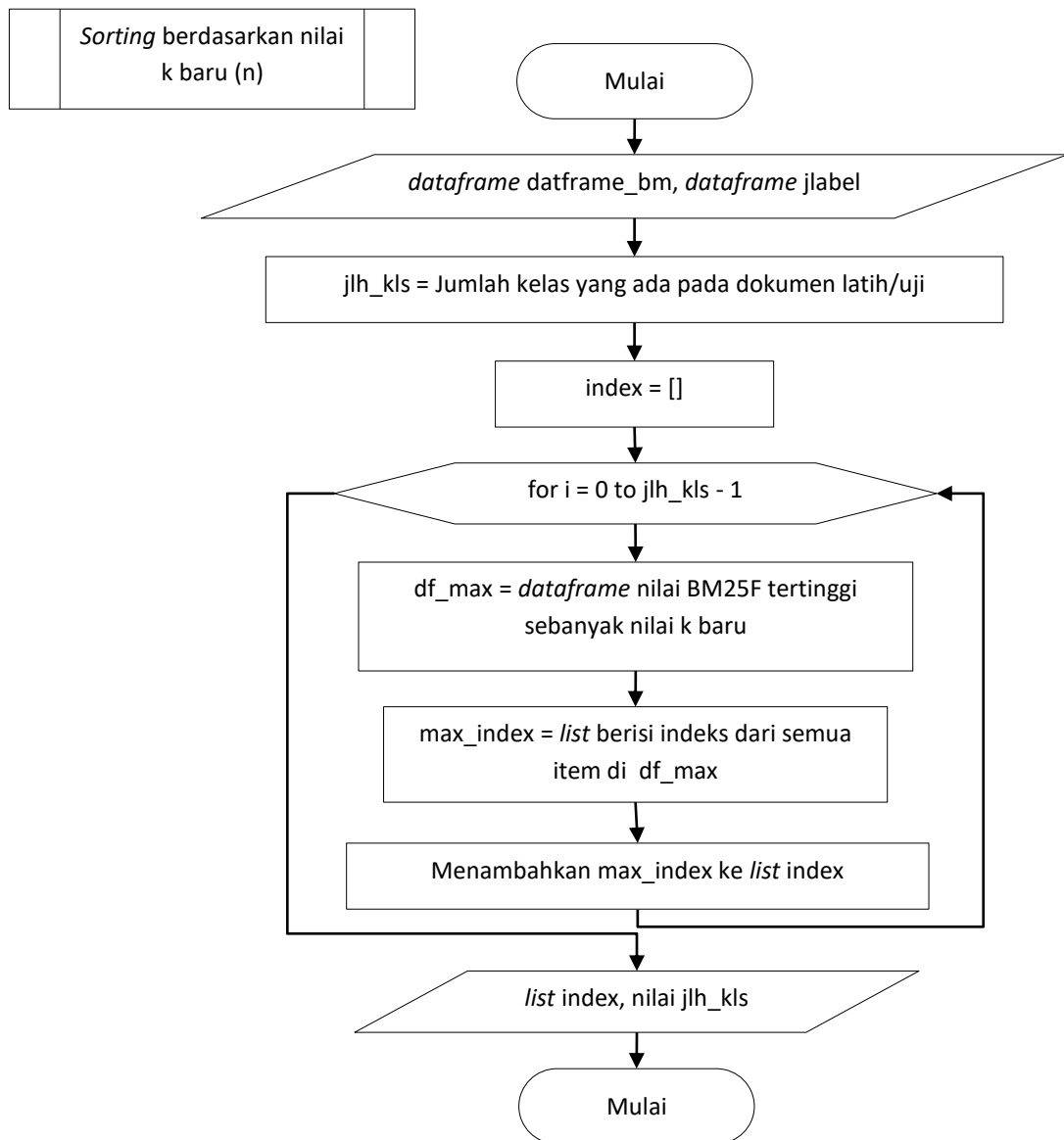
#### 4.4.9 Diagram Alir *Improved K-Nearest Neighbor*

Tahap selanjutnya setelah BM25F adalah Klasifikasi *Improved K-Nearest Neighbor* (IKNN). Dalam IKNN terdapat beberapa tahap, yaitu menghitung nilai  $k$  yang baru ( $n$ ) untuk IKNN, mengurutkan dokumen latih dengan BM25F tertinggi (*sorting*) berdasarkan nilai  $k$  baru, dan menentukan peluang tiap kelas dengan IKNN. Alur perhitungan nilai  $k$  yang baru ( $n$ ) ditunjukkan pada Gambar 4.20.



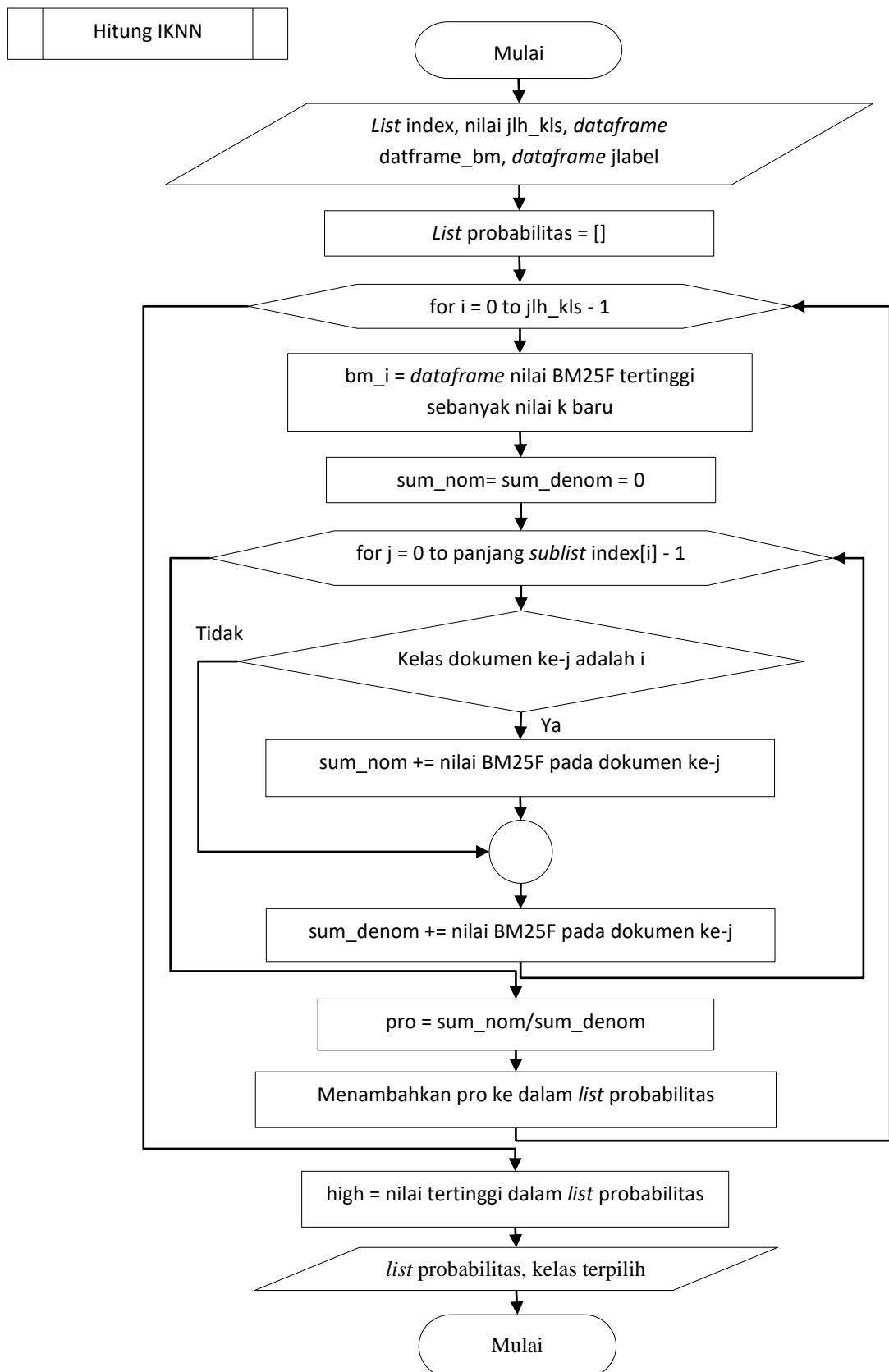
**Gambar 4.20 Diagram Alir BM25F**

Setelah nilai  $k$  baru sudah didapatkan untuk masing-masing kelas, maka saatnya mengurutkan dokumen latih dengan BM25F tertinggi berdasarkan nilai  $k$  baru. Berikut adalah penjabaran alurnya pada Gambar 4.21.



**Gambar 4.21 Diagram Alir Sorting berdasarkan nilai  $k$  baru ( $n$ )**

Setelah mengurutkan dokumen latih dengan BM25F tertinggi berdasarkan nilai  $k$  baru, maka langkah akhir adalah menghitung peluang tiap kelas dengan IKNN. Berikut adalah penjabaran alurnya pada Gambar 4.22.



**Gambar 4.22 Diagram Alir Hitung IKNN**

## 4.5 Manualisasi

Manualisasi akan menunjukkan rincian tahap *preprocessing* dokumen *tweet* latih dan uji, perhitungan BM25F masing-masing data latih, dan Klasifikasi IKNN. Dokumen latih dan dokumen uji ditunjukkan pada Tabel 4.1 dan Tabel 4.2.

**Tabel 4.1 Data Latih**

Dokumen	Data Latih	Label
1	Entah apa yang merasukimu pemerintah. Dan.....!!!! Demokrasi di kebiri Kpk di lemahkan Buruh di peras Dan MANIPULASI PUBLIK NEGARA INI!!!! Nikmat apa lagi yang engkau akan ciptakan demi kepentingan sendiri wahai pimpinan ELIT!!!! #NegaraGagalLindungiRakyat #GejayanMemanggil	UK
2	Kurang anjing bagaimana coba.. dan ambulance pun ditembak. Ya tuhan tolonglah kami dari org2 yang dzalim.. #KedaulatanMutlakMilikRakyat	UK
3	Rakyat pasti menang melawan penindasan Rakyat kita pasti akan menang! #MosiTidakPercaya #ReformasiDikorupsi	NUK
4	Di saat aparat Kepolisian yang seharusnya merangkul rakyat jadi arogan, kini @tni_ad adalah satu aparat yang tidak pernah mengingkari sumpahnya bagi NKRI. Tuhan, Tegarkanlah negaraku. #HidupMahasiswa #MosiTidakPercaya <a href="https://t.co/r0DONxHoQ9">https://t.co/r0DONxHoQ9</a>	NUK
5	MANA BONEKA PINOKIO? SEBAGIAN BESAR RAKYAT SUDAH TIDAK MAU LAGI KAU JADI PEMIMPIN MEREKA SUDAH MUAK DENGAN MU KOK KENAPA TIDAK MAU TURUN tahta JUGA? TIDAK tahu MALU!! #KedaulatanMutlakMilikRakyat	UK

**Tabel 4.2 Data Uji**

Dokumen	Data Uji	Label
1	Ini sekarang saya mengetik serius ya... Tolong pak presiden tegakkan keadilan, jangan mau jadi boneka mulu pak .. rakyat indonesia butuh pemimpin yang tegas. Terima kasih #HidupMahasiswa #TolakRUUKUHP #SaveKPK	?

### 4.5.1 Preprocessing

Tahap pertama dari *preprocessing* pada penelitian ini adalah *cleaning*. *Cleaning* adalah proses penghapusan karakter tanda baca dan URL dari dokumen data latih maupun uji. Tabel 4.3 dan 4.4 menunjukkan proses *cleaning*.

**Tabel 4.3 Hasil *Cleaning* Data Latih**

Dokumen	Hasil <i>cleaning</i>	
	<i>tweet</i>	tagar
1	Entah apa yang merasukimu pemerintah Dan Demokrasi di kebiri Kpk di lemahkan Buruh di peras Dan MANIPULASI PUPLIK NEGARA INI Nikmat apa lagi yang engkau akan ciptakan demi kepentingan sendiri wahai pimpinan ELIT	NegaraGagalLindungiRakyat GejayanMemanggil
2	Kurang anjing bagaimana coba dan ambulance pun ditembak Ya tuhan tolonglah kami dari org yang dzalim	KedaulatanMutlakMilikRakyat
3	Rakyat pasti menang melawan penindasan Rakyat kita pasti akan menang	MosiTidakPercaya ReformasiDikorupsi
4	Di saat aparat Kepolisian yang seharusnya merangkul rakyat jadi arogan kini tni_ad adalah satu aparat yang tidak pernah mengingkari sumpahnya bagi NKRI Tuhan Tegarkanlah negaraku	HidupMahasiswa MosiTidakPercaya
5	MANA BONEKA PINOKIO SEBAGIAN BESAR RAKYAT SUDAH TIDAK MAU LAGI KAU JADI PEMIMPIN MEREKA SUDAH MUAK DENGAN MU KOK KENAPA TIDAK MAU TURUN tahta JUGA TIDAK tahu MALU	KedaulatanMutlakMilikRakyat

**Tabel 4.4 Hasil *Cleaning* Data Uji**

Dokumen	Hasil <i>cleaning</i>	
	<i>tweet</i>	Tagar
1	Ini sekarang saya mengetik serius ya Tolong pak presiden tegakkan keadilan jangan mau jadi boneka mulu pak rakyat indonesia butuh pemimpin yang tegas Terima kasih	HidupMahasiswa TolakRUUKUHP SaveKPK

Selanjutnya adalah tokenisasi. Tokenisasi adalah proses pemisahan kalimat menjadi satuan kata berdasarkan *delimiter* yang ditentukan. Pada penelitian ini penulis menggunakan spasi sebagai *delimiter* tokenisasi. Tabel 4.5 dan Tabel 4.6 menunjukkan proses tokenisasi.

**Tabel 4.5 Hasil Tokenisasi Data Latih**

Dokumen	Hasil Tokenisasi	
	<i>tweet</i>	tagar
1	Entah, apa, yang, merasukimu, pemerintah, Dan, Demokrasi, di, kebiri, Kpk, di, lemahkan, Buruh, di, peras, Dan, MANIPULASI, PUPLIK, NEGARA, INI, Nikmat, apa, lagi, yang, engkau, akan, ciptakan, demi, kepentingan, sendiri, wahai, pimpinan, ELIT	NegaraGagalLindungiRakyat, GejayanMemanggil
2	Kurang, anjing, bagaimana, coba, dan, ambulance, pun, ditembak, Ya, tuhan, tolonglah, kami, dari, org, yang, dzalim	KedaulatanMutlakMilikRakyat
3	Rakyat, pasti, menang, melawan, penindasan, Rakyat, kita, pasti, akan, menang	MosiTidakPercaya, ReformasiDikorupsi
4	Di, saat, aparat, Kepolisian, yang, seharusnya, merangkul, rakyat, jadi, arogan, kini, tni_ad, adalah, satu, aparat, yang, tidak, pernah, mengingkari, sumpahnya, bagi, NKRI, Tuhan, Tegarkanlah, negaraku	HidupMahasiswa, MosiTidakPercaya
5	MANA, BONEKA, PINOKIO, SEBAGIAN, BESAR, RAKYAT, SUDAH, TIDAK, MAU, LAGI, KAU, JADI, PEMIMPIN, MEREKA, SUDAH, MUAK, DENGAN, MU, KOK, KENAPA, TIDAK, MAU, TURUN, tahta, JUGA, TIDAK, tahu, MALU	KedaulatanMutlakMilikRakyat

**Tabel 4.6 Hasil Tokenisasi Data Uji**

Dokumen	Hasil Tokenisasi	
	<i>tweet</i>	Tagar
1	Ini, sekarang, saya, mengetik, serius, ya, Tolong, pak, presiden, tegakkan, keadilan, jangan, mau, jadi, boneka, mulu, pak, rakyat, indonesia, butuh, pemimpin, yang, tegas, Terima, kasih	HidupMahasiswa, TolakRUUKUHP, SaveKPK

Selanjutnya adalah *case folding*. *Case folding* ialah proses mengubah semua huruf dalam kalimat menjadi huruf kecil (untuk penelitian ini, tagar/*hashtag* melalui *case folding* terpisah). Proses ditunjukkan pada Tabel 4.7 dan Tabel 4.8.

**Tabel 4.7 Hasil *Case folding* Data Latih**

Dokumen	Hasil <i>Case folding</i>	
	<i>tweet</i>	Tagar
1	entah, apa, yang, merasukimu, pemerintah, dan, demokrasi, di, kebiri, kpk, di, lemahkan, buruh, di, peras, dan, 'manipulasi, puplik, negara, ini, nikmat, apa, lagi, yang, engkau, akan, ciptakan, demi, kepentingan, sendiri, wahai, pimpinan, elit	NegaraGagalLindungiRakyat, GejayanMemanggil
2	kurang, anjing, bagaimana, coba, dan, ambulance, pun, ditembak, ya, tuhan, tolonglah, kami, dari, org, yang, dzalim	KedaulatanMutlakMilikRakyat
3	rakyat, pasti, menang, melawan, penindasan, rakyat, kita, pasti, akan, menang	MosiTidakPercaya, ReformasiDikorupsi
4	di, saat, aparat, kepolisian, yang, seharusnya, merangkul, rakyat, jadi, arogan, kini, tni_ad, adalah, satu, aparat, yang, tidak, pernah, mengingkari, sumpahnya, bagi, nkri, tuhan, tegarkanlah, negaraku	HidupMahasiswa, MosiTidakPercaya

**Tabel 4.7 Hasil *Case folding* Data Latih (lanjutan)**

Dokumen	Hasil <i>Case folding</i>	
	<i>tweet</i>	tagar
5	mana, boneka, pinokio, sebagian, besar, rakyat, sudah, tidak, mau, lagi, kau, jadi, pemimpin, mereka, sudah, muak, dengan, mu, kok, kenapa, tidak, mau, turun, tahta, juga, tidak, tahu, malu	KedaulatanMutlakMilikRakyat

**Tabel 4.8 Hasil *Case folding* Data Uji**

Dokumen	Hasil <i>Case folding</i>	
	<i>tweet</i>	tagar
1	ini, sekarang, saya, mengetik, serius, ya, tolong, pak, presiden, tegakkan, keadilan, jangan, mau, jadi, boneka, mulu, pak, rakyat, indonesia, butuh, pemimpin, yang, tegas, terima, kasih	HidupMahasiswa, TolakRUUKUHP, SaveKPK

Selanjutnya adalah *Split By Uppercase*, yaitu pemisahan kata berdasarkan keberadaan huruf kapital, ditemukan pada *hashtag* atau tagar dalam Bahasa Indonesi, lalu diubah menjadi huruf kecil. Proses ada di Tabel 4.9 dan Tabel 4.10.

**Tabel 4.9 Hasil *Split By Uppercase* Data Latih**

Dokumen	Hasil <i>Split By Uppercase</i>	
	<i>tweet</i>	tagar
1	entah, apa, yang, merasukimu, pemerintah, dan, demokrasi, di, kebiri, kpk, di, lemahkan, buruh, di, peras, dan, 'manipulasi, puplik, negara, ini, nikmat, apa, lagi, yang, engkau, akan, ciptakan, demi, kepentingan, sendiri, wahai, pimpinan, elit	negara, gagal, lindungi, rakyat gejayan, memanggil
2	kurang, anjing, bagaimana, coba, dan, ambulance, pun, ditembak, ya, tuhan, tolonglah, kami, dari, org, yang, dzalim	kedaulatan, mutlak, milik, rakyat



**Tabel 4.9 Hasil *Split By Uppercase* Data Latih (lanjutan)**

Dokumen	Hasil <i>Split By Uppercase</i>	
	<i>tweet</i>	tagar
3	rakyat, pasti, menang, melawan, penindasan, rakyat, kita, pasti, akan, menang	mosi, tidak, percaya reformasi, dikorupsi
4	di, saat, aparat, kepolisian, yang, seharusnya, merangkul, rakyat, jadi, arogan, kini, tni_ad, adalah, satu, aparat, yang, tidak, pernah, mengingkari, sumpahnya, bagi, nkri, tuhan, tegarkanlah, negaraku	hidup, mahasiswa mosi, tidak, percaya
5	mana, boneka, pinokio, sebagian, besar, rakyat, sudah, tidak, mau, lagi, kau, jadi, pemimpin, mereka, sudah, muak, dengan, mu, kok, kenapa, tidak, mau, turun, tahta, juga, tidak, tahu, malu	kedaulatan, mutlak, milik, rakyat

**Tabel 4.10 Hasil *Split By Uppercase* Data Uji**

Dokumen	Hasil <i>Split By Uppercase</i>	
	<i>tweet</i>	tagar
1	ini, sekarang, saya, mengetik, serius, ya, tolong, pak, presiden, tegakkan, keadilan, jangan, mau, jadi, boneka, mulu, pak, rakyat, indonesia, butuh, pemimpin, yang, tegas, terima, kasih	hidup, mahasiswa tolak, ruukuhp save, kpk

Selanjutnya adalah *Filtering*, yaitu penyaringan untuk menghilangkan kata yang merupakan *stopword*. Proses ini ditunjukkan pada Tabel 4.11 dan 4.12.

**Tabel 4.11 Hasil *Filtering* Data Latih**

Dokumen	Hasil <i>Filtering</i>	
	<i>Tweet</i>	Tagar
1	merasukimu, pemerintah, demokrasi, kebiri, kpk, lemahkan, buruh, peras, manipulasi, puplik, negara, nikmat, engkau, ciptakan, kepentingan, pimpinan, elit	negara, gagal, lindungi, rakyat gejayan, memanggil
2	anjing, coba, ambulance, ditembak, tuhan, tolonglah, org, dzalim	kedaulatan, mutlak, milik, rakyat
3	rakyat, menang, melawan, penindasan, rakyat, menang	mosi, tidak, percaya reformasi, dikorupsi
4	aparatus, kepolisian, merangkul, rakyat, arogan, tni_ad, aparat, mengingkari, sumpahnya, nkri, tuhan, tegarkanlah, negaraku	hidup, mahasiswa mosi, tidak, percaya
5	boneka, pinokio, rakyat, kau, pemimpin, muak, mu, turun, tahta, malu	kedaulatan, mutlak, milik, rakyat

**Tabel 4.12 Hasil *Filtering* Data Uji**

Dokumen	Hasil <i>Filtering</i>	
	<i>tweet</i>	tagar
1	mengketik, serius, tolong, presiden, tegakkan, keadilan, boneka, mulu, rakyat, indonesia, butuh, pemimpin, terima, kasih	hidup, mahasiswa tolak, ruukuhp save, kpk

Selanjutnya adalah *Stemming*, adalah tahap akhir *preprocessing*, dimana daftar kata/*term* sudah disaring akan melalui proses penghilangan imbuhan, sisipan dan awalan. Proses ini ditunjukkan pada Tabel 4.13 dan Tabel 4.14.

**Tabel 4.13 Hasil Stemming Data Latih**

Dokumen	Hasil Stemming	
	<i>tweet</i>	tagar
1	rasuk, perintah, demokrasi, kebiri, kpk, lemah, buruh, peras, manipulasi, puplik, negara, nikmat, engkau, cipta, penting, pimpin, elit	negara, gagal, lindungi, rakyat gejayan, memanggil
2	anjing, coba, ambulance, tembak, tuhan, tolong, org, dzalim	kedaulatan, mutlak, milik, rakyat
3	rakyat, menang, lawan, tindas, rakyat, menang	mosi, tidak, percaya reformasi, dikorupsi
4	aparatus, polisi, rangkul, rakyat, arogan, tni, ad, aparat, ingkar, sumpah, nkri, tuhan, tegar, negara	hidup, mahasiswa mosi, tidak, percaya
5	boneka, pinokio, rakyat, kau, pimpin, muak, mu, turun, tahta, malu	kedaulatan, mutlak, milik, rakyat

**Tabel 4.14 Hasil Stemming Data Uji**

Dokumen	Hasil Stemming	
	<i>tweet</i>	tagar
1	ketik, serius, tolong, presiden, tegak, adil, boneka, mulu, rakyat, indonesia, butuh, pimpin, terima, kasih	hidup, mahasiswa tolak, ruukuhp save, kpk

#### 4.5.2 Perhitungan $tf$ Dokumen Latih

Nilai  $tf$  adalah jumlah *term* setelah *preprocessing* yang muncul pada dokumen latih. Penelitian ini menggunakan 2 *stream* yaitu isi *tweet* (s1) dan *hashtag* atau tagar (s2). Nilai  $tf$  dapat dilihat pada Tabel 4.15.

**Tabel 4.15 Hasil Perhitungan  $tf$  Dokumen Latih**

<i>term</i>	D1		D2		D3		D4		D5	
	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
rasuk	1	0	0	0	0	0	0	0	0	0
perintah	1	0	0	0	0	0	0	0	0	0
demokrasi	1	0	0	0	0	0	0	0	0	0
kebiri	1	0	0	0	0	0	0	0	0	0

**Tabel 4.15 Hasil Perhitungan tf Dokumen Latih (lanjutan)**

<i>term</i>	D1		D2		D3		D4		D5	
	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
kpk	1	0	0	0	0	0	0	0	0	0
lemah	1	0	0	0	0	0	0	0	0	0
buruh	1	0	0	0	0	0	0	0	0	0
peras	1	0	0	0	0	0	0	0	0	0
manipulasi	1	0	0	0	0	0	0	0	0	0
puplik	1	0	0	0	0	0	0	0	0	0
negara	1	1	0	0	0	0	1	0	0	0
nikmat	1	0	0	0	0	0	0	0	0	0
engkau	1	0	0	0	0	0	0	0	0	0
cipta	1	0	0	0	0	0	0	0	0	0
penting	1	0	0	0	0	0	0	0	0	0
pimpin	1	0	0	0	0	0	0	0	1	0
elit	1	0	0	0	0	0	0	0	0	0
gagal	0	1	0	0	0	0	0	0	0	0
lindung	0	1	0	0	0	0	0	0	0	0
rakyat	0	1	0	0	2	0	1	0	1	1
gejayan	0	1	0	0	0	0	0	0	0	0
panggil	0	1	0	0	0	0	0	0	0	0
anjing	0	0	1	1	0	0	0	0	0	0
coba	0	0	1	1	0	0	0	0	0	0
ambulance	0	0	1	1	0	0	0	0	0	0
tembak	0	0	1	1	0	0	0	0	0	0
tuhan	0	0	1	1	0	0	1	0	0	0
tolong	0	0	1	1	0	0	0	0	0	0
org	0	0	1	1	0	0	0	0	0	0
dzalim	0	0	1	1	0	0	0	0	0	0
daulat	0	0	0	0	0	0	0	0	0	1
mutlak	0	0	0	0	0	0	0	0	0	1
milik	0	0	0	0	0	0	0	0	0	1
menang	0	0	0	0	2	0	0	0	0	0
lawan	0	0	0	0	1	0	0	0	0	0
tindas	0	0	0	0	1	0	0	0	0	0
mosi	0	0	0	0	0	1	0	1	0	0
percaya	0	0	0	0	0	1	0	1	0	0
reformasi	0	0	0	0	0	1	0	0	0	0
korupsi	0	0	0	0	0	1	0	0	0	0
aparatus	0	0	0	0	0	0	2	0	0	0
polisi	0	0	0	0	0	0	1	0	0	0
rangkul	0	0	0	0	0	0	1	0	0	0

**Tabel 4.15 Hasil Perhitungan *tf* Dokumen Latih (lanjutan)**

<i>term</i>	D1		D2		D3		D4		D5	
	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
arogan	0	0	0	0	0	0	1	0	0	0
tni	0	0	0	0	0	0	1	0	0	0
ad	0	0	0	0	0	0	1	0	0	0
ingkar	0	0	0	0	0	0	1	0	0	0
sumpah	0	0	0	0	0	0	1	0	0	0
nkri	0	0	0	0	0	0	1	0	0	0
tegar	0	0	0	0	0	0	1	0	0	0
hidup	0	0	0	0	0	0	0	1	0	0
mahasiswa	0	0	0	0	0	0	0	1	0	0
boneka	0	0	0	0	0	0	0	0	1	0
pinokio	0	0	0	0	0	0	0	0	1	0
kau	0	0	0	0	0	0	0	0	1	0
muak	0	0	0	0	0	0	0	0	1	0
mu	0	0	0	0	0	0	0	0	1	0
turun	0	0	0	0	0	0	0	0	1	0
tahta	0	0	0	0	0	0	0	0	1	0
malu	0	0	0	0	0	0	0	0	1	0

#### 4.5.3 Perhitungan *tf* Dokumen Uji

Nilai *tf* Dokumen Uji adalah jumlah *term* pada dokumen uji setelah *preprocessing* yang muncul pada tiap dokumen latih. Penelitian ini menggunakan 2 *stream* yaitu isi *tweet* (s1) dan *hashtag* atau tagar (s2). Nilai *tf* dapat dilihat pada Tabel 4.16.

**Tabel 4.16 Hasil Perhitungan *tf* Dokumen Uji**

<i>term</i>	D1		D2		D3		D4		D5	
	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
ketik	0	0	0	0	0	0	0	0	0	0
serius	0	0	0	0	0	0	0	0	0	0
tolong	0	0	1	0	0	0	0	0	0	0
presiden	0	0	0	0	0	0	0	0	0	0
tegak	0	0	0	0	0	0	0	0	0	0
adil	0	0	0	0	0	0	0	0	0	0
boneka	0	0	0	0	0	0	0	0	1	0
mulu	0	0	0	0	0	0	0	0	0	0
rakyat	0	1	1	0	2	0	1	0	1	1
indonesia	0	0	0	0	0	0	0	0	0	0
butuh	0	0	0	0	0	0	0	0	0	0

**Tabel 4.16 Hasil Perhitungan  $tf$  Dokumen Uji (lanjutan)**

<i>term</i>	D1		D2		D3		D4		D5	
	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
pimpin	1	0	0	0	0	0	0	0	1	0
terima	0	0	0	0	0	0	0	0	0	0
kasih	0	0	0	0	0	0	0	0	0	0
hidup	0	0	0	0	0	0	0	1	0	0
mahasiswa	0	0	0	0	0	0	0	1	0	0
tolak	0	0	0	0	0	0	0	0	0	0
ruukuhp	0	0	0	0	0	0	0	0	0	0
save	0	0	0	0	0	0	0	0	0	0
kpk	1	0	0	0	0	0	0	0	0	0

#### 4.5.4 Perhitungan $df$ dan $wIDF$

Nilai  $df$  (*Document Frequency*) merupakan jumlah dokumen latih dimana suatu *term* pada dokumen uji ditemukan. Sedangkan nilai  $wIDF$  adalah modifikasi IDF. Perhitungan nilai  $df$  dan  $wIDF$  terdapat pada Tabel 4.17.

**Tabel 4.17 Hasil Perhitungan  $df$  dan  $wIDF$**

term	$df$	$wIDF$
ketik	0	1.041393
serius	0	1.041393
tolong	1	0.477121
presiden	0	1.041393
tegak	0	1.041393
adil	0	1.041393
boneka	1	0.477121
mulu	0	1.041393
rakyat	5	-1.04139
indonesia	0	1.041393
butuh	0	1.041393
pimpin	2	0.146128
terima	0	1.041393
kasih	0	1.041393
hidup	1	0.477121
mahasiswa	1	0.477121
tolak	0	1.041393
ruukuhp	0	1.041393
save	0	1.041393
kpk	1	0.477121

Berikut adalah contoh perhitungan manual untuk wIDF dari *term* “pimpin”, dengan menggunakan Persamaan 2.3.

$$\begin{aligned}
 N &= 5 \\
 w_i^{IDF} &= \log \frac{N - df + 0,5}{df + 0,5} \\
 w_{pimpin}^{IDF} &= \log \frac{5 - 2 + 0,5}{2 + 0,5} \\
 &= 0,146128
 \end{aligned}$$

#### 4.5.5 Perhitungan $sl_s$

*Stream Length*, disebut juga sebagai  $sl_s$  adalah panjang tiap dokumen latih yang didapatkan dengan cara menghitung jumlah kata dalam masing-masing *stream* di setiap dokumen setelah *preprocessing*. Nilai  $sl_s$  untuk *stream* isi *tweet* ( $s=1$ ) dan  $sl_s$  untuk *stream* tagar ( $s=2$ ) ditunjukkan pada Tabel 4.18.

**Tabel 4.18 Hasil Perhitungan  $sl_s$**

s	Nilai $sl_s$				
	D1	D2	D3	D4	D5
1	17	8	6	14	10
2	6	4	4	4	4

#### 4.5.6 Perhitungan $avsl_s$

Nilai  $avsl_s$  merupakan perhitungan rata-rata nilai  $sl_s$  dari seluruh dokumen latih pada masing-masing *stream* yang ada. Nilai  $avsl_s$  untuk *stream* isi *tweet* ( $s=1$ ) dan  $avsl_s$  untuk *stream* tagar ( $s=2$ ) ditunjukkan pada Tabel 4.19.

**Tabel 4.19 Hasil Perhitungan  $avsl_s$**

s	Nilai $avsl_s$
1	11
2	4,4

#### 4.5.7 Perhitungan $B_s$

$B_s$  adalah normalisasi panjang tiap dokumen latih yang didapatkan setelah nilai  $sl_s$  dan  $avsl_s$  sudah didapatkan. Pada penelitian ini, nilai  $b_s$ , yaitu parameter normalisasi, akan ditetapkan sebesar 0.75. Nilai  $B_s$  ditunjukkan pada Tabel 4.20.

**Tabel 4.20 Hasil Perhitungan  $B_s$**

s	$B_s$				
	D1	D2	D3	D4	D5
1	1,409091	0,795454	0,659090	1,204545	0,931818
2	1,272727	0,931818	0,931818	0,931818	0,931818

Berikut adalah contoh perhitungan manual untuk  $B_s$  dari Dokumen 1 *stream* 1 (D1, s=1), dengan menggunakan Persamaan 2.1.

$$\begin{aligned}
 b_s &= 0,75 \\
 B_s &= \left( (1 - b_s) + b_s \frac{sl_s}{avsl_s} \right) \\
 B_1 &= \left( (1 - 0,75) + 0,75 \frac{17}{11} \right) \\
 &= 1,409091
 \end{aligned}$$

#### 4.5.8 Perhitungan $\widetilde{tf}_i$

Perhitungan  $\widetilde{tf}_i$  ialah total jumlah kemunculan *term* yang dinormalisasi (disebut juga *normalized tf*) pada masing-masing *stream* dokumen latih setelah melalui *preprocessing*. Perhitungan ini memerlukan nilai  $v_s$ , yaitu bobot unik untuk masing-masing *stream*. Untuk *tweet* (s=1) akan ditetapkan sebesar 5, dan untuk tagar (s=2) akan ditetapkan sebesar 3. Nilai  $\widetilde{tf}_i$  terdapat pada Tabel 4.21.

**Tabel 4.21 Hasil Perhitungan  $\widetilde{tf}_i$**

<i>term</i>	$\widetilde{tf}$				
	D1	D2	D3	D4	D5
ketik	0	0	0	0	0
serius	0	0	0	0	0
tolong	0	6,2857143	0	0	0
presiden	0	0	0	0	0
tegak	0	0	0	0	0
adil	0	0	0	0	0
boneka	0	0	0	0	5,365854
mulu	0	0	0	0	0
rakyat	2,357143	3,2195122	15,172414	4,1509434	8,585366
indonesia	0	0	0	0	0
butuh	0	0	0	0	0
pimpin	3,548387	0	0	0	5,365854
terima	0	0	0	0	0
kasih	0	0	0	0	0
hidup	0	0	0	3,2195122	0
mahasiswa	0	0	0	3,2195122	0
tolak	0	0	0	0	0
rukuhp	0	0	0	0	0
save	0	0	0	0	0
kpk	3,548387	0	0	0	0



Berikut adalah contoh perhitungan manual untuk  $\widetilde{tf}_i$  dari Dokumen 1 (D1), *term* “pimpin”, dengan menggunakan Persamaan 2.2.

$$\begin{aligned}
 v_1 &= 5 \\
 v_2 &= 3 \\
 \widetilde{tf}_i &= \sum_{s=1}^2 v_s \frac{tf_{si}}{B_s} \\
 \widetilde{tf}_{pimpin} &= \left( 5 \times \frac{1}{1,409091} \right) + \left( 3 \times \frac{0}{1,27273} \right) \\
 &= 3,548387
 \end{aligned}$$

#### 4.5.9 Perhitungan BM25F Dokumen Uji

Setelah melalui berbagai proses sebelumnya, tahap pembobotan terakhir adalah perhitungan BM25F dari setiap *term* pada masing-masing dokumen latih. Untuk menghitung BM25F, dibutuhkan nilai  $k_1$ , yaitu konstanta saturasi yang wajib digunakan. Pada penelitian ini,  $k_1$  ditetapkan sebesar 1.5. Nilai BM25F setiap *term* akan digabung menjadi nilai akhir BM25F untuk setiap dokumen latih ditunjukkan pada Tabel 4.22.

**Tabel 4.22 Hasil Perhitungan BM25F**

<i>term</i>	BM25F				
	D1	D2	D3	D4	D5
ketik	0	0	0	0	0
serius	0	0	0	0	0
tolong	0	0,385198	0	0	0
presiden	0	0	0	0	0
tegak	0	0	0	0	0
adil	0	0	0	0	0
boneka	0	0	0	0	0,372883
mulu	0	0	0	0	0
rakyat	-0,636411	-0,710407	-0,947699	-0,764962	-0,886505
indonesia	0	0	0	0	0
butuh	0	0	0	0	0
pimpin	0,102709	0	0	0	0,114203
terima	0	0	0	0	0
kasih	0	0	0	0	0
hidup	0	0	0	0,325478	0
mahasiswa	0	0	0	0,325478	0
tolak	0	0	0	0	0
rukuhp	0	0	0	0	0

**Tabel 4.22 Hasil Perhitungan BM25F (lanjutan)**

<i>term</i>	BM25F				
	D1	D2	D3	D4	D5
save	0	0	0	0	0
kpk	0,335356	0	0	0	0
<b>Total BM25F</b>	-0,198340	-0,325209	-0,947700	-0,114007	-0,399420

Berikut adalah contoh perhitungan manual untuk BM25F dari Dokumen 1 (D1), dimana *term* data uji yang muncul pada data latih hanya “rakyat”, “pimpin”, dan “kpk”, dengan menggunakan Persamaan 2.4.

$$\begin{aligned}
 k_1 &= 1,5 \\
 w_d^{BM25F} &= \sum_{i=1}^j \frac{\widetilde{tf}_i}{k_1 + \widetilde{tf}_i} w_i^{IDF} \\
 w_1^{BM25F} &= \left( \frac{\widetilde{tf}_{rakyat}}{k_1 + \widetilde{tf}_{rakyat}} w_i^{IDF} \right) + \left( \frac{\widetilde{tf}_{pimpin}}{k_1 + \widetilde{tf}_{pimpin}} w_i^{IDF} \right) \\
 &\quad + \left( \frac{\widetilde{tf}_{kpk}}{k_1 + \widetilde{tf}_{kpk}} w_i^{IDF} \right) \\
 w_1^{BM25F} &= \left( \frac{2,357143}{1,5 \times 2,357143} \times -1,04139 \right) \\
 &\quad + \left( \frac{3,548387}{1,5 \times 3,548387} \times 0,146128 \right) \\
 &\quad + \left( \frac{3,5483874}{1,5 \times 3,548387} \times 0,477121 \right) \\
 &= (-0,636411) + (0,102709) + (0,335356) \\
 &= -0,198340
 \end{aligned}$$

#### 4.5.10 Perhitungan IKNN Dokumen Uji

IKNN diawali dengan menentukan nilai *k* awal, untuk mendapatkan nilai *k* baru, yaitu *n*. Pada penelitian ini, untuk perhitungan manual, nilai *k* awal yang dipilih adalah 3. Hasil perhitungan nilai *n* (*k* baru) dapat dilihat pada Tabel 4.23.

**Tabel 4.23 Hasil Perhitungan *n***

Kelas	<i>n</i> ( <i>k</i> baru)
1 (UK)	3
2 (NUK)	2

Berikut adalah contoh perhitungan manual untuk nilai  $n$  dari Kelas 2 (non ujaran kebencian atau NUK), dengan menggunakan Persamaan 2.5.

$$\begin{aligned}
 k &= 3 \\
 n &= \left\lceil \frac{k \times N(C_m)}{\max\{N(C_j) | j = 1, \dots, Nc\}} \right\rceil \\
 n &= \frac{3 \times 2}{3} \\
 &= 2
 \end{aligned}$$

Setelah nilai  $n$  sudah ditemukan untuk masing-masing kelas, maka klasifikasi dapat dilakukan dengan menggunakan  $n$  tersebut untuk menghitung probabilitas suatu dokumen uji masuk ke tiap kelas yang ada pada dokumen latih. Sedangkan peran dari hasil pembobotan BM25F yang sebelumnya sudah dilakukan adalah untuk sebagai *similarity* pada klasifikasi.

Hasil perhitungan probabilitas dokumen uji (dipersingkat menjadi  $q$ ) masuk ke masing-masing kelas ditunjukkan pada Tabel 4.24.

**Tabel 4.24 Probabilitas Kelas Dokumen Latih**

Kelas	P(DokUji, Kelas)
UK	0,821181
NUK	0,365001

Berikut adalah contoh perhitungan manual untuk probabilitas dari Kelas ke-2 yaitu non ujaran kebencian (NUK). Karena sebelumnya sudah didapatkan nilai  $k$  baru atau  $n$  untuk Kelas NUK adalah 2, maka yang digunakan dalam menghitung probabilitas hanya 2 dokumen dengan nilai BM25F tertinggi, yakni dokumen latih 1 (D1) dan dokumen latih 4 (D4). Perhitungan dilakukan dengan menggunakan Persamaan 2.6.

$$\begin{aligned}
 k &= 3 \\
 P(x, C_m) &= \operatorname{argMax}_m \frac{\sum_{i=1}^n \operatorname{sim}(x, d_j) y(d_j, C_m)}{\sum_{i=1}^n \operatorname{sim}(x, d_j)} \\
 P(q, C_{NUK}) &= \frac{(\operatorname{sim}(q, D1) y(D1, C_{NUK})) + (\operatorname{sim}(q, D4) y(D4, C_{NUK}))}{\operatorname{sim}(q, D1) + \operatorname{sim}(q, D4)} \\
 &= \frac{(-0,198340 \times 0) + (-0,114007 \times 1)}{-0,198340 - 0,114007} \\
 &= 0,365001
 \end{aligned}$$

Maka, sesuai dengan hasil yang sudah ditunjukkan pada Tabel 4.24 dan contoh perhitungan probabilitas kelas NUK, didapatkan bahwa dokumen uji yang diberikan adalah bagian dari kelas ujaran kebencian atau UK, sebab nilai

probabilitasnya, yaitu sebesar 0,821181, lebih tinggi dari nilai probabilitas NUK, yaitu sebesar 0,365001.

#### 4.6 Rancangan Pengujian

Pada penelitian ini, pengujian yang digunakan dalam adalah *5-Fold Cross Validation*. Penggunaan *5-Fold*, dengan rasio 4:1 untuk jumlah data latih dan data uji merupakan salah satu jumlah *fold* yang umum digunakan. Penelitian yang menggunakan *K-Fold Cross Validation* dengan  $K = 5$  di antaranya adalah penelitian Yu, dkk. (2008) mengenai prediksi resiko kredit finansial menggunakan *Least-Squares Support Vector Machine*. Selain itu, dilakukan pengamatan distribusi kelas data tanpa *shuffle* apabila dikenakan jumlah *fold* yang paling sering digunakan dalam *k-Fold Cross Validation*, yaitu *4-Fold Cross Validation* dan *5-Fold Cross Validation*. Hasilnya adalah *5-Fold Cross Validation* memiliki persebaran kelas data yang sedikit lebih baik, sehingga *5-Fold Cross Validation* dipilih. Pengujian dimulai dengan menghitung nilai *F-measure*, *Precision*, *Recall*, dan *Accuracy* untuk sejumlah variasi  $k$  pada IKNN dengan *5-fold*, kemudian dilanjutkan dengan uji nilai  $b_s$ ,  $v_s$ , dan  $k_1$  juga akan divariasikan.

Penelitian oleh Ma, dkk. (2014) menyebutkan bahwa, tidak ada ketentuan tetap dalam memilih rentang nilai  $k$  untuk pengujian metode klasifikasi KNN. Sehingga dipilih  $k = 3, 4, 5, 10$  dan semua kelipatan 10 hingga 100. Penggunaan kelipatan 10 hingga 100 juga digunakan dalam penelitian oleh Miao, dkk. (2014) mengenai *Improved K-Nearest Neighbor* dengan data latih yang *Imbalanced*. Sehingga penelitian ini juga memilih menggunakan kelipatan 10 hingga 100 supaya pengujian  $k$  tidak terlalu banyak. Rancangan pengujian variasi  $k$  dapat dilihat pada Tabel 4.25.

**Tabel 4.25 Tabel Rancangan Pengujian  $k$**

$k$	Kelas	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	0				
	1				
4	0				
	1				
5	0				
	1				
⋮					
100	0				
	1				

Dilanjutkan dengan pengujian  $b_s$ , dengan  $k$  yang hasil  $F-measure$  nya terbaik. Kisaran untuk nilai  $b_s$  adalah  $0,5 < b_s < 0,8$ . Rancangan pengujian Parameter  $b_s$  terdapat pada Tabel 4.26.

**Tabel 4.26 Rancangan Pengujian  $b_s$**

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5				
0,6				
0,7				
0,8				

Dilanjutkan dengan pengujian  $v_s$ , dengan  $b_s$  yang hasil  $F-measure$  nya terbaik. Kisaran untuk nilai  $v_s$  adalah  $2 \leq v_s \leq 5$ , terdapat pada Tabel 4.27.

**Tabel 4.27 Rancangan Pengujian  $v_s$**

$v_1$	$v_2$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
2	3				
2	4				
2	5				
3	2				
3	4				
3	5				
⋮					
5	4				

Terakhir adalah pengujian  $k_1$ , dengan  $v_s$  yang hasil  $F-measure$  nya terbaik. Kisaran untuk nilai  $k_1$  adalah  $1,3 < k_1 \leq 2$ , terdapat pada Tabel 4.28.

**Tabel 4.28 Rancangan Pengujian  $k_1$**

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
1,3				
1,4				
1,5				
⋮				
2,0				

## BAB 5 IMPLEMENTASI

Pada bab 5, akan ada penjabaran mengenai implementasi dari sistem berupa kode program. Pembahasan yang terdapat pada bab implementasi terdiri dari lingkungan penerapan, penerapan algoritme dan hasil implementasi.

### 5.1 Lingkungan Penerapan

Lingkungan penerapan yang terdapat pada penelitian ini menjelaskan mengenai perangkat-perangkat yang digunakan dalam proses klasifikasi *tweet* berbahasa Indonesia berisi ujaran kebencian menggunakan metode *Improved K-Nearest Neighbor* dengan pembobotan BM25F. Perangkat yang digunakan dijabarkan sebagai berikut.

#### 5.1.1 Lingkungan Penerapan Perangkat Lunak

Penelitian mengenai klasifikasi *tweet* berbahasa Indonesia berisi ujaran kebencian menggunakan metode *Improved K-Nearest Neighbor* dengan pembobotan BM25F diterapkan pada lingkungan perangkat lunak sebagai berikut.

1. *Python 3.6*

Bahasa pemrograman yang digunakan untuk implementasi kode program dalam penelitian ini adalah *Python* dengan versi 3.6.

2. *Jupyter Notebook 5.7.8*

Perangkat lunak yang digunakan untuk implementasi penelitian ini dan juga sebagai *text editor* adalah *Jupyter Notebook* versi 5.7.8.

#### 5.1.2 Lingkungan Penerapan Perangkat Keras

Penelitian mengenai klasifikasi *tweet* berbahasa Indonesia berisi ujaran kebencian menggunakan metode *Improved K-Nearest Neighbor* dengan pembobotan BM25F diterapkan pada lingkungan perangkat keras sebagai berikut.

1. Laptop Lenovo G40 AMD A4-6210 APU
2. 64-bit OS
3. RAM 4.00 GB
4. Radeon R3 Graphics

### 5.2 Penerapan Algoritme

Penerapan algoritme pada penelitian ini membahas tentang proses implementasi program menggunakan algoritme yang sesuai dengan alur yang telah dibuat di bab perancangan. Proses yang terdapat pada penelitian ini berupa proses *preprocessing*, pembobotan BM25F dan klasifikasi IKNN.

### 5.2.1 Preprocessing

Tahap *preprocessing* yang terdapat pada penelitian ini terdiri dari tahap *cleaning*, tokenisasi, *case folding*, *Split By Uppercase*, *filtering*, dan *stemming*. Berikut adalah penerapan *preprocessing* secara general untuk masing-masing dokumen latih dan dokumen uji, yang ditampilkan di Kode Program 5.1 dan Kode Program 5.2.

Algoritme: <i>Preprocessing</i> Dokumen Latih	
1	def preprocessing_latih(listkalimat):
2	wordsperstc=[]
3	for i,x in enumerate(listkalimat):
4	wordsperstc.append([])
5	kalimat = cleaning(x)
6	tokens = tokenisasi(kalimat)
7	clean,ht = casefolding(tokens)
8	tagar = splitbyuppercase(ht)
9	nontagar = [y for y in clean if y not in ht]
10	filter_kt = filtering(nontagar)
11	filter_ht = filtering(tagar)
12	prep_kt = stemming(filter_kt)
13	prep_ht = stemming(filter_ht)
14	wordsperstc[i].append(prepare_kt)
15	wordsperstc[i].append(prepare_ht)
16	return wordsperstc

**Kode Program 5.1 *Preprocessing* Dokumen Latih**

Method *preprocessing* dokumen latih bernama *preprocessing\_latih*. Baris 2 dan 3 adalah inisialisasi *list* untuk hasil akhir *preprocessing*, dan perulangan for untuk kode di baris 4-15. Baris 4-8 memanggil method *cleaning*, *tokenisasi*, *casefolding* dan *splitbyuppercase*. Baris 9 berisi perulangan for yang memasukkan *term* ke dalam *list* bernama *nontagar* jika *term* tersebut bukan *hashtag*. Baris 10-15 memanggil method *filtering* dan *stemming* untuk *list* tagar dan *nontagar*, lalu menambahkan kedua *list* tersebut menjadi *sublist* dalam *list* hasil *preprocessing*, yang merupakan *return value* method ini.

Algoritme: <i>Preprocessing</i> Dokumen Uji	
1	def preprocessing_uji(listkalimat):
2	token=[]
3	for i,x in enumerate(listkalimat):
4	kal=cleaning(x)
5	tokens = tokenisasi(kal)
6	bersih,ht = casefolding(tokens)
7	tagar = splitbyuppercase(ht)
8	for y in bersih:
9	if y not in ht:
10	token.append(y)
11	for t in tagar:
12	token.append(t)
13	token = list(dict.fromkeys(stemming(filtering(token))))
14	return token

**Kode Program 5.2 *Preprocessing* Dokumen Uji**

Method *preprocessing* dokumen uji bernama *preprocessing\_uji*. Baris 2 dan 3 adalah inialisasi *list* untuk hasil akhir *preprocessing*, dan perulangan for untuk kode di baris 4-13. Baris 4-7 memanggil method *cleaning*, *tokenisasi*, *casefolding* dan *splitbyuppercase*. Baris 8-10 berisi perulangan for yang memasukkan *term* ke dalam *list* hasil akhir *preprocessing* jika *term* tersebut bukan *hashtag*. Baris 11-12 berisi perulangan for yang memasukkan semua *term* hasil pemisahan *hashtag* ke *list* hasil akhir *preprocessing*. Baris 13 memanggil method *filtering* dan *stemming* untuk *list* hasil akhir *preprocessing*, yang akan menjadi *return value* method ini.

### 5.2.2 Cleaning

Tahap *preprocessing* pertama adalah *cleaning*. Kode programnya ditampilkan di Kode Program 5.3.

Algoritme: <i>Cleaning</i>	
1	def cleaning(kalimat):
2	ganti =
3	re.sub(r'^\w\s# \d \w+(?:\.me) (?:.*)   (?:@) (?:.*)   (?:pic.tw
4	itter\S+)  \w+(?:\.com)   (?: (https? s?ftp):\/\/\/) (?:.*) ?   (?:www
5	\.) (?:.*) ? ',',',', kalimat)
6	kalimat = ganti
7	return kalimat

**Kode Program 5.3 Cleaning**

Method *cleaning* akan menerima parameter input berupa string dokumen latih/uji. Baris 2-6 adalah untuk mengganti bagian dalam string dengan ' ' atau spasi kosong jika bagian dalam tersebut cocok dengan *regular expression (regex)* yang telah ditentukan, misalkan bagian dalam string yang mengandung url laman web. Setelah diganti, string baru akan menjadi *return value* method ini.

### 5.2.3 Tokenisasi

Tahap *preprocessing* selanjutnya adalah tokenisasi. Kode programnya ditampilkan di Kode Program 5.4.

Algoritme: Tokenisasi	
1	def tokenisasi(kalimat):
2	tkn=kalimat.split()
3	return tkn

**Kode Program 5.4 Tokenisasi**

Method *tokenisasi* akan menerima parameter input berupa string dokumen latih/uji. Baris 2-3 adalah untuk memisahkan kalimat tersebut berdasarkan spasi dan memasukkannya ke *list* tkn, yang merupakan *return value* method ini.

### 5.2.4 Case Folding

Tahap *preprocessing* selanjutnya adalah *case folding*. Kode programnya ditampilkan di Kode Program 5.5.



Algoritme: <i>Case Folding</i>	
1	def casefolding(tkn):
2	wordtagar = [y for y in tkn if re.match(r"#\S+",
3	str(y))]
4	tkn = [y.lower() for y in tkn if y not in wordtagar]
5	tkn = list(filter(None, tkn))
6	return tkn, wordtagar

#### Kode Program 5.5 *Case Folding*

Method *casefolding* akan menerima parameter input berupa *list* term dokumen latih/uji. Baris 2-3 adalah untuk perulangan for untuk mengisi *list* wordtagar, jika *term* dalam *list* input cocok dengan *regex* yang diawali tanda pagar (*hashtag*). Baris 4 adalah untuk perulangan for untuk mengubah huruf dalam *list* tkn menjadi huruf kecil, jika *term* tidak ada dalam *list* wordtagar, kemudian baris 5 akan menghapus item kosong dari *list*, dan baris 6 menjadikan *list* tkn dan wordtagar sebagai *return value* method ini.

#### 5.2.5 *Split By Uppercase*

Tahap *preprocessing* selanjutnya adalah *split by uppercase*. Kode programnya ditampilkan di Kode Program 5.6.

Algoritme: <i>Split By Uppercase</i>	
1	def splitbyuppercase(wordtagar):
2	tagar=[]
3	for x,y in enumerate(wordtagar):
4	caps = re.findall('[A-Z][a-z]+ [A-Z]{2,}','
5	wordtagar[x])
6	[tagar.append(i.lower()) for i in caps]
7	return tagar

#### Kode Program 5.6 *Split By Uppercase*

Method *splitbyuppercase* akan menerima parameter input berupa *list* wordtagar pada dokumen latih/uji. Inisialisasi *list* bernama tagar ada di baris 2 diikuti perulangan for di baris 3-7 yang memisahkan tiap *hashtag* yang ditemukan dalam dokumen *tweet* berdasarkan huruf besar dalam string, kemudian memasukkannya ke *list* tagar sebagai *return value* method ini.

#### 5.2.6 *Filtering*

Tahap *preprocessing* selanjutnya adalah *filtering*. Kode programnya ditampilkan di Kode Program 5.7.

Algoritme: <i>Filtering</i>	
1	def filtering(unfiltered):
2	filtered=[kata for kata in unfiltered if kata not in
3	stoplist]
4	return filtered

#### Kode Program 5.7 *Filtering*

Method *filtering* akan menerima parameter input berupa *list* bernama unfiltered dari dokumen latih/uji. Inisialisasi *list* bernama filtered ada di baris 2-3

sekaligus menampung sebuah *term* dari *list* tersebut dengan perulangan *for*, jika *term* tersebut tidak ada dalam daftar *stopword*. *List filtered* adalah *return value* method ini.

### 5.2.7 Stemming

Tahap *preprocessing* selanjutnya adalah *stemming*. Kode programnya ditampilkan di Kode Program 5.8.

Algoritme: <i>Stemming</i>	
1	def stemming(filtered):
2	inputstem = ' '.join(filtered)
3	outputstem = stemmer.stem(inputstem)
4	hasil_stem = outputstem.split()
5	return hasil_stem

**Kode Program 5.8 Stemming**

Method *stemming* menerima parameter input berupa *list* term setelah *filtering* dokumen latih/uji, bernama *filtered*. Baris 2 akan menggabungkan semua *term* dalam *list* menjadi sebuah string dengan menambah spasi di antaranya. Baris 3 adalah *stemming* pada semua kata di string tersebut. Baris 4 memisah string setelah *stemming* dan memasukkannya ke *list* *hasil\_stem*. *List* tersebut adalah *return value* method ini.

### 5.2.8 Perhitungan *tf*

Tahap yang ditempuh setelah *preprocessing* adalah perhitungan *tf*, dimana *tf* dilakukan pada tahap pelatihan maupun pengujian. Kode programnya ditampilkan di Kode Program 5.9 dan 5.10.

Algoritme: Perhitungan <i>tf</i> Dokumen Latih	
1	def tf_latih(datalatih,term):
2	tf=[]
3	for x in range(len(datalatih)):
4	tf.append([])
5	for y in range(len(datalatih[x])):
6	tf[x].append([])
7	for l in term:
8	if l not in datalatih[x][y]:
9	tf[x][y].append(0)
10	elif l in datalatih[x][y]:
11	he = datalatih[x][y].count(l)
12	tf[x][y].append(he)
13	return tf

**Kode Program 5.9 Perhitungan *tf* Dokumen Latih**

Method *tf\_latih* menerima parameter input berupa *list* bernama *datalatih* (hasil method *preprocessing\_latih*) berupa semua *term* pada masing-masing *stream* di tiap dokumen dan *list* bernama *term*, berisi *term* apa saja yang ada di sebuah dokumen latih (tidak membedakan *stream* nya). Baris 2 adalah inisialisasi *list* *tf*. Baris 3 adalah perulangan *for* sebanyak jumlah dokumen latih, baris 4 menambahkan *sublist* kosong pada *list* *tf*. Baris 5 dan 7 secara berurutan adalah perulangan *for* sebanyak jumlah *stream* pada sebuah dokumen dan jumlah item

dalam *list* term. Baris 6 menambahkan *sublist* ke dalam *sublist* di baris 4 sebelumnya. Baris 8-12 berisi seleksi kondisi jika sebuah item dari *list* term tidak ada dalam *sublist* datalatih (jika item tidak ditemukan dalam *sublist* stream suatu dokumen latih), maka pada *tf* ditambahkan item *integer* 0. Jika sebaliknya, maka jumlah item tersebut dalam *sublist* stream di *list* datalatih akan dihitung dan ditambahkan ke dalam *list* *tf*, yaitu *return value* method ini.

Setelah itu, *list* *tf* akan diubah menjadi dataframe pandas dan disimpan dalam sebuah file berekstensi .csv untuk meningkatkan efisiensi waktu pengambilan nilai *tf* untuk dicocokkan dengan *tf* dokumen uji.

Perhitungan *tf* dokumen uji akan menggunakan kode program berikut.

Algoritme: Perhitungan tf Dokumen Uji	
1	def tf_uji(file,datauji,dokumen,streams):
2	tfread = pd.read_csv(file, index_col=0)
3	tf=[]
4	for x in range(dokumen):
5	tf.append([])
6	for y in range(streams):
7	tf[x].append([])
8	for l in datauji:
9	if l not in tfread.columns:
10	tf[x][y].append(0)
11	elif l in tfread.columns:
12	n = (stream*x)+y
13	he = tfread.at[n,l]
14	tf[x][y].append(he)
15	return tf

**Kode Program 5.10 Perhitungan *tf* Dokumen Uji**

Method *tf\_uji* menerima parameter input berupa *list* bernama datauji (hasil *preprocessing\_uji*) berupa *term* apa saja yang ada di sebuah dokumen latih (tidak membedakan *stream* nya), nama file .csv penyimpanan *tf* dokumen latih, serta jumlah dokumen dan jumlah *stream* pada semua dokumen. Baris 2 membaca file .csv dengan pandas. Baris 3 adalah inisialisasi *list* *tf*. Baris 4 adalah perulangan for sebanyak jumlah dokumen latih, baris 5 menambahkan *sublist* kosong pada *list* *tf*. Baris 6 dan 8 secara berurutan adalah perulangan for sebanyak jumlah *stream* pada sebuah dokumen dan jumlah item dalam *list* term. Baris 7 menambahkan *sublist* ke dalam *sublist* di baris 5 sebelumnya. Baris 9-12 berisi seleksi kondisi jika sebuah item dari *list* term tidak ada dalam *sublist* datalatih (jika item tidak ditemukan dalam *sublist* stream suatu dokumen latih), maka pada *tf* ditambahkan item *integer* 0. Jika sebaliknya, maka nilai *tf* untuk dokumen dan *stream* yang bersangkutan akan diambil dari file .csv dan ditambahkan ke dalam *list* *tf*, yaitu *return value* method ini.

### 5.2.9 Perhitungan *df*

Tahap yang ditempuh setelah *tf* adalah perhitungan *df*, dimana *df* dilakukan pada tahap pelatihan maupun pengujian. Kode program *df* untuk dokumen latih ditampilkan di Kode Program 5.11.

Algoritme: Perhitungan DF Dokumen Latih	
1	def df_latih(tf,term):
2	df=[0]*len(term)
3	for x in range(len(tf)):
4	tftotal = [sum(i) for i in zip(*tf[x])]
5	for y in range(len(term)):
6	if tftotal[y] != 0:
7	df[y] += 1
8	return df

**Kode Program 5.11 Perhitungan *df* Dokumen Latih**

Method *df\_latih* menerima parameter input berupa *list* bernama *tf* (hasil *tf\_latih*) berupa *tf term* pada masing-masing *stream* di tiap dokumen dan *list* bernama *term*, berisi *term* apa saja yang ada di sebuah dokumen latih (tidak membedakan *stream* nya). Baris 2 adalah inisialisasi *list* *df*, berukuran sejumlah semua *term* pada kumpulan dokumen latih. Baris 3 adalah perulangan *for* sebanyak jumlah dokumen latih, baris 4 menjumlahkan *tf* antar *stream* dalam satu dokumen dan menambahkannya ke dalam sebuah *list* bernama *tftotal*, lalu pada baris 5-7 ada perulangan *for* dengan kondisi jika item ke-*y* pada *tftotal* lebih dari 0 maka item ke-*y* pada *list* *df* akan ditambah 1. *List* *df* menjadi *return value* method ini.

Setelah itu, *list* *df* juga akan diubah menjadi dataframe pandas dan disimpan dalam sebuah file berekstensi .csv untuk dicocokkan dengan *df* dokumen uji. Perhitungan *df* dokumen uji akan menggunakan kode program 5.12.

Algoritme: Perhitungan df Dokumen Uji	
1	def df_uji(file,datauji):
2	dfread = pd.read_csv(file, index_col=0)
3	df=[dfread.at[0,1] if 1 in dfread.columns else 0 for 1
4	in datauji]
5	return df

**Kode Program 5.12 Perhitungan *df* Dokumen Uji**

Method *df\_uji* menerima parameter nama file dan *list* bernama *datauji*, yaitu *list* *term* dari dokumen uji tanpa membedakan *stream*. Baris 2 membaca file .csv yang berisi hasil *df\_latih*. Baris 3 menambahkan item di file tersebut ke *list* *df* jika nama kolomnya ada dalam *list* *datauji*. *List* *df* adalah *return value* method ini.

### 5.2.10 Perhitungan wIDF

Tahap yang ditempuh setelah *df* adalah perhitungan wIDF, dan tahap ini sudah memasuki tahap pengujian. Kode programnya ditampilkan di Kode Program 5.13.

Algoritme: Perhitungan wIDF	
1	def modified_idf(df):
2	mod_idf=[math.log10(((dok-i)+0.5)/(i+0.5)) for i in df]
3	return mod_idf

**Kode Program 5.13 Perhitungan wIDF**

Method *modified\_idf* menerima input *list* bernama *df*, yaitu hasil dari method *df\_uji*. Baris 2 melakukan perulangan *for* pada nilai dalam *list* *df* dan

menambahkan hasil modifikasi nilai  $df$  sesuai perhitungan  $wIDF$  di Persamaan 2.3 ke dalam  $list$   $mod\_idf$ .  $List$  tersebut akan menjadi *return value* method ini.

### 5.2.11 Perhitungan $sl_s$

Tahap yang ditempuh setelah  $wIDF$  adalah perhitungan  $sl_s$ . Kode programnya ditampilkan di Kode Program 5.14.

Algoritme: Perhitungan $sl_s$	
1	def $sl(dataframe\_tf, data\_uji, jml\_dok, streams)$ :
2	$sl = [None] * jml\_dok$
3	$count\_row = list(dataframe\_tf.sum(axis=1))$
5	for $x$ in $range(jml\_dok)$ :
6	$sl[x] = [count\_row[(stream * x) + y]$ for $y$ in
7	$range(streams)]$
8	$avsl = np.mean(sl, axis=0)$ #axis=0 artinya mengarah baris
9	return $sl, avsl$

**Kode Program 5.14 Perhitungan  $sl_s$**

Method  $sl$  menerima input *list* bernama  $dataframe\_tf$ , yaitu *dataframe* yang diambil dari file  $tf$  dokumen latih, *list* hasil method  $preprocessing\_uji$  untuk dokumen uji, serta jumlah dokumen latih dan *stream*. Baris 2 inisialisasi *list*  $sl$  berukuran sebanyak jumlah dokumen latih, baris 3 menjumlahkan nilai per baris di  $dataframe\_tf$  (semua kata pada tiap *stream* di tiap dokumen latih), mengubahnya sebagai *list* bernama  $count\_row$ . Baris 5-8 melakukan perulangan for di seluruh dokumen latih untuk menghitung rata-rata pada tiap *stream*, yang dimasukkan ke *list*  $sl$  dan  $avsl$ . Kedua *list* tersebut akan menjadi *return value* nya.

### 5.2.12 Perhitungan $B_s$

Tahap yang ditempuh setelah perhitungan  $sl_s$  adalah  $B_s$ . Kode programnya ditampilkan di Kode Program 5.15.

Algoritme: Perhitungan $B_s$	
1	def $Bs(sl, avsl, b)$ :
2	$Bs = []$
3	for $x$ in $range(len(sl))$ :
5	$Bs.append([])$
6	for $y$ in $range(len(sl[x]))$ :
7	$j = (1-b) + (b * (sl[x][y] / avsl[y]))$
8	$Bs[x].append(j)$
9	return $Bs$

**Kode Program 5.15 Perhitungan  $B_s$**

Method  $Bs$  menerima input nilai hasil metod  $sl$  (*list*  $sl$  dan  $avsl$ ), dan nilai konstanta  $b_s$ . Baris 2 adalah inisialisasi *list*  $Bs$ . Baris 3 melakukan perulangan for sejumlah item di *list*  $sl$  (sebanyak jumlah dokumen latih), baris 4 menambahkan *sublist* kosong di *list*  $Bs$ , baris 5 melakukan perulangan sebanyak jumlah *stream*, baris 6-7 menambahkan hasil perhitungan  $B_s$  untuk tiap *stream* di tiap dokumen sesuai Persamaan 2.1. *Return value* adalah *list*  $Bs$ .

### 5.2.13 Perhitungan $\widetilde{tf}_i$

Tahap yang ditempuh setelah perhitungan  $B_s$  adalah perhitungan  $\widetilde{tf}_i$ . Kode programnya ditampilkan di Kode Program 5.16.

Algoritme: Perhitungan tf	
1	def tfi(bobot,tf,hasilBs,jt):
2	tfi=[]
3	tfperword =[(list(zip(*i))) for i in tf]
4	for x,k in enumerate(hasilBs):
5	tfi.append([])
6	for y in range(jt):
7	tpa=0
8	for z,m in enumerate(k):
9	tpa +=
10	(bobot[z]*(tfperword[x][y][z]/hasilBs[x][z]))
11	tfi[x].append(tpa)
12	return tfi

**Kode Program 5.16 Perhitungan  $\widetilde{tf}_i$**

Method *tfi* menerima input nilai hasil metod Bs berupa *list* bernama hasilBs, bobot unik ( $v_s$ ) tiap *stream*, hasil method *tf\_uji*, dan jumlah term dokumen uji. Baris 2 ialah inisialisasi *list* tfi. Baris 3 yaitu *zip* jumlah tf untuk tiap term di semua dokumen uji. Baris 4 melakukan perulangan for sejumlah item di *list* hasilBs (sebanyak jumlah dokumen latih), baris 5 menambahkan *sublist* kosong di *list* tfi, baris 6 ialah perulangan sebanyak jumlah term, baris 7-11 menambahkan hasil  $\widetilde{tf}_i$  tiap term di tiap dokumen sesuai Persamaan 2.2. *Return value* nya *list* tfi.

### 5.2.14 Perhitungan BM25F

Tahap yang ditempuh setelah perhitungan  $\widetilde{tf}_i$  adalah perhitungan total BM25F. Kode programnya ditampilkan di Kode Program 5.17.

Algoritme: Perhitungan BM25F	
1	def bm25f(tfi,widf,k1):
2	bm25f=wbm25f=[]
3	for x,k in enumerate(tfi):
4	bm25f.append([])
5	for y,n in enumerate(k):
6	bm25f_i = (n/(k1+n))*widf[y]
7	bm25f[x].append(bm25f_i)
8	wbm25f.append(sum(bm25f[x]))
9	label_dl = [int(i) for i in label_latih]
10	bm_label = pd.DataFrame(zip(label_dl,wbm25f))
11	bm_label.columns = ['label','Nilai wBM25F']
12	return bm25f, bm_label

**Kode Program 5.17 Perhitungan BM25F**

Method *bm25f* menerima input nilai hasil metod tfi berupa *list* bernama tfi, hasil method modified\_idf berupa *list* widf, dan nilai konstanta  $k_1$ . Baris 2-3 adalah inisialisasi *list* bm25f dan wbm25f. Baris 3 melakukan perulangan sebanyak jumlah item di *list* tfi. Baris 4 menambahkan *sublist* kosong di *list* bm25f, baris 5 melakukan perulangan for sejumlah *sublist* di *list* tfi (sebanyak

jumlah term), baris 7-9 menambahkan hasil perhitungan BM25F untuk tiap term di tiap dokumen sesuai Persamaan 2.4 ke dalam *list* *bm25f* dan memasukkan *list* *bm25f* ke dalam *list* *wbm25f* sebagai *sublist*. Baris 10-12 menggabung *list* *wbm25f* dan daftar label kelas tiap data latih, ke dalam sebuah dataframe bernama *bm\_label*. *Return value* adalah *list* *bm25f* dan dataframe *bm\_label*.

### 5.2.15 Perhitungan IKNN

Tahap yang ditempuh setelah perhitungan BM25F adalah penentuan peluang tiap kelas dengan IKNN. Terdapat 2 tahap, yaitu penentuan nilai *k* baru tiap kelas, penentuan *k* dokumen dengan BM25F tertinggi. Secara berurutan, kode programnya ditampilkan di Kode Program 5.18, Kode Program 5.19, dan Kode Program 5.20.

Algoritme: Perhitungan <i>k</i> baru	
1	def <i>k_baru</i> ( <i>k</i> , dataframe_bm):
2	jumlah_label =
3	dataframe_bm['label'].value_counts().rename_axis('label').r
4	eset_index(name='counts')
5	kls_max = jumlah_label['counts'].max()
6	jumlah_label['k baru'] = [( <i>k</i> *i)/int(kls_max) for i in
7	jumlah_label['counts']]
8	return jumlah_label

**Kode Program 5.18 Penentuan Nilai *k* baru**

Method *k\_baru* menerima input nilai *k* awal dan dataframe *wbm25f* hasil method *bm25f*. Baris 2-4 menghitung jumlah kelas yang ada pada pada dokumen latih dan memasukkannya ke dataframe bernama *jumlah\_label*. Baris 5 mendapatkan kelas dengan jumlah dokumen latih terbanyak. Baris 6-7 menambah kolom baru bernama '*k\_baru*' pada dataframe *jumlah\_label* yang diisi dengan hasil perhitungan nilai *n* sebagai *k* baru sesuai Persamaan 2.5. *Return value* adalah dataframe *jumlah\_label*. Setelah itu akan dilakukan penentuan *k* dokumen dengan BM25F tertinggi pada method selanjutnya.

Algoritme: Penentuan <i>k</i> dokumen dengan BM25F tertinggi	
1	def <i>index_highest</i> (dataframe_bm, jlabel):
2	jln_kls = len(dataframe_bm['label'].unique().tolist())
3	index = []
4	for i in range(jln_kls):
5	df_max =
6	dataframe_bm.nlargest(int(jlabel.loc[jlabel['label'] == i,
7	'k baru'].iloc[0]), 'Nilai wBM25F')
8	max_index = df_max.index.values.tolist()
9	index.append(max_index)
10	return index, jln_kls

**Kode Program 5.19 Penentuan *k* dokumen dengan BM25F tertinggi**

Method *index\_highest* menerima input dataframe *wbm25f* hasil method *bm25f* dan dataframe hasil method *k\_baru*. Baris 2 menghitung jumlah kelas yang ada pada kumpulan dokumen latih di dataframe. Baris 3 menginisialisasi *list* bernama *index*. Baris 4 adalah perulangan for sejumlah kelas yang ada di dokumen latih. Baris 5-7 memilih *k* dokumen dengan BM25F tertinggi bagi tiap

kelas sesuai dengan jumlah  $k$  baru kelas masing-masing. Baris 8 mendapatkan nomor index dari  $k$  dokumen dengan BM25F tertinggi. Baris 9 menambahkan hasil nomor index pada *list* index. *Return value* method ini adalah *list* index dan jumlah kelas.

Setelah itu akan dilakukan penentuan peluang IKNN tiap kelas dan peluang yang tertinggi pada method selanjutnya.

Algoritme: Penentuan Peluang IKNN tiap Kelas	
1	def iknn_probability(index, jlh_kls, dataframe_bm, jlabel):
2	probabilitas = []
3	for i in range(jlh_kls):
4	bm_i =
5	dataframe_bm.nlargest(int(jlabel.loc[jlabel['label'] == i,
6	'k baru'].iloc[0]), 'Nilai wBM25F')
7	sum_nom = 0
8	sum_denom = 0
9	for j in index[i]:
10	if bm_i.at[j, 'label'] == i:
11	sum_nom += bm_i.at[j, 'Nilai wBM25F']
12	sum_denom += bm_i.at[j, 'Nilai wBM25F']
13	pro = sum_nom/sum_denom
14	probabilitas.append(pro)
15	high = max(probabilitas)
16	return probabilitas, probabilitas.index(high)

#### Kode Program 5.20 Penentuan peluang IKNN tiap kelas

Method *iknn\_probability* menerima input indeks dari  $k$  dokumen tertinggi, jumlah kelas, dataframe wbm25f dan jumlah dokumen tiap kelas. Baris 2 menginisialisasi *list* bernama probabilitas. Baris 3 melakukan perulangan sebanyak jumlah kelas, baris 4-6 menemukan nilai BM25F dari  $k$  dokumen tertinggi. Baris 7-8 menginisialisasi variable untuk menampung nilai pembilang dan penyebut dari Persamaan 2.6. Baris 9 melakukan perulangan for pada item dalam *list* index, baris 10-12 adalah seleksi kondisi untuk menjumlahkan nilai BM25F jika indeks pada dataframe wbm25f cocok dengan indeks salah satu dari  $k$  dokumen dengan BM25F tertinggi, untuk mendapatkan nilai pembilang dan penyebut yang telah dibahas sebelumnya. Baris 13-15 adalah menghitung peluang kelas dengan membagi pembilang dengan penyebut tersebut sesuai Persamaan 2.6, memasukkan hasilnya ke *list* peluang untuk masing-masing kelas dan mendapatkan kelas dengan peluang tertinggi. *Return value* adalah nilai peluang masing-masing kelas dan kelas yang berpeluang tertinggi sebagai hasil akhir dari klasifikasi IKNN.



## BAB 6 PENGUJIAN DAN ANALISIS

Pada Bab ini akan dibahas pengujian yang telah dilakukan berdasarkan pada rancangan, manualisasi serta implementasi dari bab sebelumnya. Dan pada bab ini pula dibahas hasil dari pengujian yang telah dilakukan.

### 6.1 Pengujian Sistem

Pada pengujian sistem terdapat uraian dari hasil pengujian yang telah dilakukan. Pengujian sistem dilakukan berdasarkan perbandingan *F-Measure* dari proses klasifikasi dengan parameter yang telah ditentukan. Pengujian sistem terdiri dari pengujian nilai  $k$  pada metode IKNN dengan menggunakan *5-Fold Cross Valiation*, dan total data yang digunakan sebanyak 500 dokumen, sehingga Pengujian yang mencakup *Accuracy*, *Precision*, *Recall* dan *F-measure* dilakukan sebanyak 5 kali pada 5 *Fold* data uji dan data latih yang berbeda-beda. Kemudian akan diambil salah satu  $k$  pada rata-rata semua *Fold*, yang hasil pengujiannya yang memiliki *F-measure* tertinggi. *F-measure* memperhitungkan *false positive* dan *false negative*, biasanya lebih relevan daripada *Accuracy*, apabila dataset yang digunakan memiliki distribusi kelas yang tidak merata. Nilai  $k$  tersebut akan digunakan sebagai variabel kontrol untuk menguji variasi penggunaan nilai  $b_s$ ,  $v_s$ , dan  $k_1$  sebagai variabel bebasnya (variabel yang nilainya diubah-ubah).

#### 6.1.1 Pengujian 5-Fold Cross Validation pada Variabel $k$

Untuk pengujian *5-Fold Cross Valiation*, *Fold* ke-1 akan menggunakan dokumen ke 1-100 sebagai dokumen latih dan dokumen ke 101-500 sebagai dokumen uji. Hasil ditampilkan di Tabel 6.1.

**Tabel 6.1 *Fold* ke-1 Variabel  $k$**

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	67,16%	75,00%	70,86%	63,00%
4	65,75%	80,00%	72,18%	63,00%
5	63,88%	76,67%	69,69%	60,00%
10	66,66%	80,00%	72,72%	64,00%
20	66,23%	85,00%	74,45%	65,00%
30	65,00%	86,67%	74,29%	64,00%
40	62,65%	86,67%	72,72%	61,00%
50	62,35%	88,33%	73,10%	61,00%
60	63,33%	95,00%	75,99%	64,00%
70	62,37%	96,67%	75,82%	63,00%
80	61,7%	96,67%	75,32%	62,00%
90	61,7%	96,67%	75,32%	62,00%
100	61,05%	96,67%	74,84%	61,00%

*Fold* ke-2 akan menggunakan dokumen ke 101-200 sebagai dokumen latih dan dokumen ke 1-100 dan 201-500 sebagai dokumen uji. Hasil ditampilkan di Tabel 6.2.

**Tabel 6.2 *Fold* ke-2 Variabel  $k$**

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	73,68%	87,5%	79,99%	72%
4	72,15%	89,06%	79,72%	71,00%
5	73,08%	89,06%	80,28%	72,00%
10	72,5%	90,63%	80,55%	72,00%
20	70,24%	92,19%	79,73%	70,00%
30	71,43%	93,75%	81,08%	72,00%
40	67,42%	93,75%	78,43%	67,00%
50	66,66%	96,88%	79,98%	67,00%
60	67,02%	98,44%	79,75%	68,00%
70	67,02%	98,44%	79,75%	68,00%
80	65,63%	98,44%	78,75%	66,00%
90	64,95%	98,44%	78,26%	65,00%
100	64,95%	98,44%	78,26%	65,00%

*Fold* ke-3 akan menggunakan dokumen ke 201-300 sebagai dokumen latih dan dokumen ke 1-200 dan 301-500 sebagai dokumen uji. Berikut adalah hasil pengujiannya.

**Tabel 6.3 *Fold* ke-3 Variabel  $k$**

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	73,02%	64,79%	68,66%	58,00%
4	72,22%	73,24%	72,73%	61,00%
5	76,06%	76,06%	76,06%	66,00%
10	75,64%	83,1%	79,2%	69,00%
20	74,07%	84,51%	78,95%	68,00%
30	73,49%	85,91%	79,22%	68,00%
40	71,77%	85,92%	78,21%	66,00%
50	71,59%	88,73%	79,25%	67,00%
60	71,59%	88,73%	79,25%	67,00%
70	72,22%	91,55%	80,75%	69,00%
80	72,22%	91,55%	80,75%	69,00%
90	72,22%	91,55%	80,75%	69,00%
100	72,53%	92,96%	81,48%	70,00%

*Fold* ke-4 akan menggunakan dokumen ke 301-400 sebagai dokumen latih dan dokumen ke 1-300 dan 401-500 sebagai dokumen uji. Berikut adalah hasil pengujiannya.

**Tabel 6.4 *Fold* ke-4 Variabel  $k$**

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	62,32%	64,18%	63,24%	50,00%
4	67,12%	73,13%	70%	58,00%
5	67,53%	77,61%	72,22%	60,00%
10	70,89%	83,58%	76,71%	66,00%
20	69,31%	91,04%	78,71%	67,00%
30	68,54%	91,04%	78,21%	66,00%
40	67,02%	94,03%	78,26%	65,00%
50	67,02%	94,03%	78,26%	65,00%
60	67,71%	97,02%	79,76%	67,00%
70	67,71%	97,02%	79,76%	67,00%
80	68,42%	97,02%	80,25%	68,00%
90	67,71%	97,02%	79,76%	67,00%
100	67,71%	97,02%	79,76%	67,00%

*Fold* ke-5 akan menggunakan dokumen ke 401-500 sebagai dokumen latih dan dokumen ke 1-400 sebagai dokumen uji. Berikut adalah hasil pengujiannya.

**Tabel 6.5 *Fold* ke-5 Variabel  $k$**

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	66,66%	74,19%	70,23%	61,00%
4	63,08%	66,13%	64,57%	55,00%
5	68,75%	70,97%	69,84%	62,00%
10	70,00%	79,03%	74,24%	66,00%
20	69,74%	85,48%	76,81%	68,00%
30	71,79%	90,32%	80,00%	72,00%
40	70,37%	91,94%	79,72%	71,00%
50	68,97%	96,77%	80,54%	71,00%
60	68,61%	95,16%	79,73%	70,00%
70	68,97%	96,77%	80,54%	71,00%
80	68,18%	96,77%	80,00%	70,00%
90	67,39%	100%	80,52%	70,00%
100	67,39%	100%	80,52%	70,00%

Dari semua *Fold* di atas, mengikuti prinsip *K-Fold*, maka akan dihitung rata-ratanya untuk masing-masing  $k$ . Berikut adalah hasil perhitungan rata-rata *Accuracy*, *Precision*, *Recall* dan *F-measure* dari masing-masing  $k$ .

**Tabel 6.6 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $k$** 

$k$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
3	68,57%	73,13%	70,60%	60,80%
4	68,06%	76,31%	71,84%	61,60%
5	69,86%	78,07%	73,62%	64,00%
10	71,14%	83,27%	76,68%	67,40%
20	69,92%	87,64%	77,73%	67,60%
30	70,05%	89,54%	78,56%	68,40%
40	67,85%	90,46%	77,47%	66,00%
50	67,32%	92,95%	78,23%	66,20%
60	67,65%	94,87%	78,90%	67,20%
70	67,66%	96,09%	79,32%	67,60%
80	67,23%	96,09%	79,01%	67,00%
90	66,79%	96,74%	78,92%	66,60%
100	66,73%	97,02%	78,97%	66,60%

Dari hasil perhitungan rerata di atas, dapat dilihat bahwa pada  $k = 70$ , rerata nilai *F-measure* mencapai titik tertinggi yaitu 79,32%, sementara *F-measure* mencapai . Berdasarkan nilai *Accuracy* tertinggi inilah,  $k = 70$  dipilih untuk menjadi nilai  $k$  yang akan diujikan pada variabel  $b_s$ , dimana variabel  $b_s$  tetap akan menjalani pengujian *5-Fold Cross Validation* dan akan dihitung pula reratanya untuk digunakan pada pengujian variabel selanjutnya, seperti pada variabel  $k$ .

### 6.1.2 Pengujian 5-Fold Cross Validation pada Variabel $b_s$

Untuk pengujian *5-Fold Cross Validation* pada variabel  $b_s$ , sama seperti pada variabel  $k$ , *Fold* ke-1 akan menggunakan dokumen ke 1-100 sebagai dokumen latih dan dokumen ke 101-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.7.

**Tabel 6.7 Fold ke-1 Variabel  $b_s$** 

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	62,37%	96,67%	75,82%	63,00%
0,6	63,04%	96,67%	76,32%	64,00%
0,7	62,37%	96,67%	75,82%	63,00%
0,8	62,37%	96,67%	75,82%	63,00%

*Fold* ke-2 akan menggunakan dokumen ke 101-200 sebagai dokumen latih dan dokumen ke 1-100 dan 201-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.8.

**Tabel 6.8 Fold ke-2 Variabel  $b_s$** 

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	67,02%	98,43%	79,75%	68,00%
0,6	67,02%	98,43%	79,75%	68,00%
0,7	67,02%	98,43%	79,75%	68,00%
0,8	67,02%	98,43%	79,75%	68,00%

*Fold* ke-3 akan menggunakan dokumen ke 201-300 sebagai dokumen latih dan dokumen ke 1-200 dan 301-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.9.

**Tabel 6.9 Fold ke-3 Variabel  $b_s$** 

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	71,91%	90,14%	80,00%	68,00%
0,6	72,22%	91,55%	80,75%	69,00%
0,7	72,22%	91,55%	80,75%	69,00%
0,8	72,22%	91,55%	80,75%	69,00%

*Fold* ke-4 akan menggunakan dokumen ke 301-400 sebagai dokumen latih dan dokumen ke 1-300 dan 401-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.10.

**Tabel 6.10 Fold ke-4 Variabel  $b_s$** 

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	67,71%	97,01%	79,75%	67,00%
0,6	67,71%	97,01%	79,75%	67,00%
0,7	67,71%	97,01%	79,75%	67,00%
0,8	67,71%	97,01%	79,75%	67,00%

*Fold* ke-5 akan menggunakan dokumen ke 401-500 sebagai dokumen latih dan dokumen ke 1-400 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.11.

**Tabel 6.11 Fold ke-5 Variabel  $b_s$** 

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	69,32%	98,39%	81,33%	72,00%
0,6	68,97%	96,77%	80,54%	71,00%
0,7	68,97%	96,77%	80,54%	71,00%
0,8	68,97%	96,77%	80,54%	71,00%

Dari semua *Fold* di atas, mengikuti prinsip *K-Fold*, maka akan dihitung rata-ratanya untuk masing-masing  $b_s$ . Berikut adalah hasil perhitungan rata-rata *Accuracy*, *Precision*, *Recall* dan *F-measure* dari masing-masing  $b_s$  pada Tabel 6.12.

**Tabel 6.12 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $b_s$**

$b_s$	<i>Precision</i>	<i>Recall</i>	<i>F-measure</i>	<i>Accuracy</i>
0,5	67,66%	96,13%	79,33%	67,60%
0,6	67,79%	96,09%	79,42%	67,80%
0,7	67,66%	96,09%	79,32%	67,60%
0,8	67,66%	96,09%	79,32%	67,60%

Dari hasil perhitungan rerata di atas, dapat dilihat bahwa pada  $b_s = 0.6$ , rerata nilai *F-measure* mencapai titik tertinggi yaitu 79,42%. Berdasarkan nilai *F-measure* tertinggi inilah,  $b_s = 0.6$  dipilih untuk menjadi nilai  $b_s$  yang akan diujikan pada variabel  $v_s$ , bersama dengan nilai variabel  $k = 70$ . dimana variabel  $v_s$  tetap akan menjalani pengujian *5-Fold Cross Validation* dan akan dihitung kembali reratanya untuk digunakan pada pengujian variabel selanjutnya.

### 6.1.3 Pengujian 5-Fold Cross Validation pada Variabel $v_s$

Untuk pengujian *5-Fold Cross Validation* pada variabel  $v_s$ , *Fold* ke-1 akan menggunakan dokumen ke 1-100 sebagai dokumen latih dan dokumen ke 101-500 sebagai dokumen uji seperti pengujian-pengujian selanjutnya. Variabel  $v_1$  adalah bobot *tweet* dan  $v_2$  adalah bobot tagar atau *hashtag*. Hasilnya ditampilkan pada Tabel 6.13.

**Tabel 6.13 Fold ke-1 Variabel  $v_s$**

$v_1$	$v_2$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
2	3	63,33%	95,00%	76,00%	64,00%
2	4	63,74%	96,67%	76,82%	65,00%
2	5	64,44%	96,67%	77,33%	66,00%
3	2	62,37%	96,67%	75,82%	63,00%
3	4	62,64%	95,00%	75,50%	63,00%
3	5	63,33%	95,00%	76,00%	64,00%
4	2	61,70%	96,67%	75,32%	62,00%
4	3	61,70%	96,67%	75,32%	62,00%
4	5	63,04%	96,67%	76,32%	64,00%
5	2	62,37%	96,67%	75,82%	63,00%
5	3	63,04%	96,67%	76,32%	64,00%
5	4	62,37%	96,67%	75,82%	63,00%

*Fold* ke-2 akan menggunakan dokumen ke 101-200 sebagai dokumen latih dan dokumen ke 1-100 dan 201-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.14.

**Tabel 6.14 *Fold* ke-2 Variabel  $v_s$**

$v_1$	$v_2$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
2	3	67,02%	98,44%	79,75%	68,00%
2	4	67,39%	96,88%	79,49%	68,00%
2	5	67,39%	96,88%	79,49%	68,00%
3	2	67,02%	98,44%	79,75%	68,00%
3	4	67,02%	98,44%	79,75%	68,00%
3	5	67,02%	98,44%	79,75%	68,00%
4	2	66,32%	98,44%	79,75%	67,00%
4	3	66,32%	98,44%	79,75%	67,00%
4	5	66,32%	98,44%	79,75%	67,00%
5	2	65,26%	96,88%	77,99%	65,00%
5	3	67,02%	98,44%	79,75%	68,00%
5	4	66,32%	98,44%	79,25%	67,00%

*Fold* ke-3 akan menggunakan dokumen ke 201-300 sebagai dokumen latih dan dokumen ke 1-200 dan 301-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.15.

**Tabel 6.15 *Fold* ke-3 Variabel  $v_s$**

$v_1$	$v_2$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
2	3	70,97%	92,96%	80,49%	68,00%
2	4	70,97%	92,96%	80,49%	68,00%
2	5	70,97%	92,96%	80,49%	68,00%
3	2	71,91%	90,14%	80,00%	68,00%
3	4	71,43%	91,55%	80,25%	68,00%
3	5	71,43%	91,55%	80,25%	68,00%
4	2	75,59%	88,73%	79,25%	67,00%
4	3	72,22%	91,55%	80,75%	69,00%
4	5	71,43%	91,55%	80,25%	68,00%
5	2	71,91%	90,14%	80,00%	68,00%
5	3	72,22%	91,55%	80,75%	69,00%
5	4	72,22%	91,55%	80,75%	69,00%

*Fold* ke-4 akan menggunakan dokumen ke 301-400 sebagai dokumen latih dan dokumen ke 1-300 dan 401-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.16.

**Tabel 6.16 Fold ke-4 Variabel  $v_s$** 

$v_1$	$v_2$	Precision	Recall	F-Measure	Accuracy
2	3	68,42%	97,01%	80,25%	68,00%
2	4	68,09%	95,52%	79,50%	67,00%
2	5	68,82%	95,52%	80,00%	68,00%
3	2	67,71%	97,01%	79,75%	67,00%
3	4	68,42%	97,01%	80,25%	68,00%
3	5	68,09%	95,52%	79,50%	67,00%
4	2	67,71%	97,01%	79,50%	67,00%
4	3	67,71%	97,01%	79,50%	67,00%
4	5	68,42%	97,01%	80,25%	68,00%
5	2	67,01%	97,01%	79,27%	66,00%
5	3	67,71%	97,01%	79,75%	67,00%
5	4	67,71%	97,01%	79,75%	67,00%

Fold ke-5 akan menggunakan dokumen ke 401-500 sebagai dokumen latih dan dokumen ke 1-400 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.17.

**Tabel 6.17 Fold ke-5 Variabel  $v_s$** 

$v_1$	$v_2$	Precision	Recall	F-Measure	Accuracy
2	3	69,88%	93,55%	80,00%	71,00%
2	4	69,88%	93,55%	80,00%	71,00%
2	5	69,88%	93,55%	80,00%	71,00%
3	2	69,88%	96,77%	80,54%	71,00%
3	4	69,05%	93,55%	79,45%	70,00%
3	5	70,24%	95,16%	80,82%	72,00%
4	2	68,54%	98,39%	80,79%	71,00%
4	3	69,32%	98,39%	81,33%	72,00%
4	5	69,41%	95,16%	80,27%	71,00%
5	2	67,03%	98,39%	79,74%	69,00%
5	3	68,97%	96,77%	80,54%	71,00%
5	4	69,32%	98,39%	81,33%	72,00%

Dari semua Fold di atas, mengikuti prinsip *K-Fold*, maka akan dihitung rata-ratanya untuk masing-masing  $v_s$ . Hasil perhitungan rata-ratanya di Tabel 6.18.

**Tabel 6.18 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $v_s$** 

$v_1$	$v_2$	Precision	Recall	F-Measure	Accuracy
2	3	67,92%	95,39%	79,30%	67,80%
2	4	68,01%	95,11%	79,26%	67,80%



**Tabel 6.18 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $v_s$  (lanjutan)**

$v_1$	$v_2$	Precision	Recall	F-Measure	Accuracy
2	5	68,30%	95,11%	79,46%	68,20%
3	2	67,78%	95,81%	79,17%	67,40%
3	4	67,71%	95,11%	79,04%	67,40%
3	5	68,02%	95,13%	79,26%	67,80%
4	2	67,97%	95,85%	78,92%	66,80%
4	3	67,45%	96,41%	79,33%	67,40%
4	5	67,72%	95,77%	79,37%	67,60%
5	2	66,72%	95,82%	78,56%	66,20%
5	3	67,79%	96,09%	79,42%	67,80%
5	4	67,59%	96,41%	79,38%	67,60%

**Tabel 6.18 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $v_s$  (lanjutan)**

Dari hasil perhitungan rerata di atas, dapat dilihat bahwa pada  $v_1 = 2$  dan  $v_2 = 5$  rerata nilai *F-Measure* mencapai titik tertinggi yaitu 79,46%. Begitu pula dengan *Precision* dan *Accuracy*-nya yang juga mencapai nilai tertinggi, yaitu 68,30% dan 68,20%.

Berdasarkan nilai *F-Measure* tertinggi inilah,  $v_1 = 2$  dan  $v_2 = 5$  dipilih untuk menjadi nilai  $v_1$  dan  $v_2$  yang akan diujikan pada variabel  $k_1$ , bersama dengan nilai variabel  $k = 70$  dan  $b_s = 0,6$ . dimana variabel  $k_1$  tetap akan menjalani pengujian 5-Fold Cross Validation dan dihitung reratanya untuk hasil akhir.

#### 6.1.4 Pengujian 5-Fold Cross Validation pada Variabel $k_1$

Untuk pengujian 5-Fold Cross Validation pada variabel  $k_1$ , Fold ke-1 akan menggunakan dokumen ke 1-100 sebagai dokumen latih dan dokumen ke 101-500 sebagai dokumen uji seperti pengujian-pengujian selanjutnya. Hasilnya ditampilkan pada Tabel 6.19.

**Tabel 6.19 Fold ke-1 Variabel  $k_1$** 

$k_1$	Precision	Recall	F-measure	Accuracy
1,3	63,74%	96,67%	76,82%	65,00%
1,4	64,44%	96,67%	77,33%	66,00%
1,5	64,44%	96,67%	77,33%	66,00%
1,6	64,44%	96,67%	77,33%	66,00%
1,7	64,44%	96,67%	77,33%	66,00%
1,8	64,44%	96,67%	77,33%	66,00%
1,9	64,44%	96,67%	77,33%	66,00%
2	64,44%	96,67%	77,33%	66,00%

*Fold* ke-2 akan menggunakan dokumen ke 101-200 sebagai dokumen latih dan dokumen ke 1-100 dan 201-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.20.

**Tabel 6.20 *Fold* ke-2 Variabel  $k_1$**

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,3	67,74%	98,44%	80,25%	69,00%
1,4	67,74%	96,88%	79,49%	68,00%
1,5	67,74%	96,88%	79,49%	68,00%
1,6	67,74%	96,88%	79,49%	68,00%
1,7	67,74%	96,88%	79,49%	68,00%
1,8	67,74%	96,88%	79,49%	69,00%
1,9	68,89%	96,88%	80,52%	70,00%
2	68,89%	96,88%	80,52%	70,00%

*Fold* ke-3 akan menggunakan dokumen ke 201-300 sebagai dokumen latih dan dokumen ke 1-200 dan 301-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.21.

**Tabel 6.21 *Fold* ke-3 Variabel  $k_1$**

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,3	71,74%	92,96%	80,98%	69,00%
1,4	70,97%	92,96%	80,49%	68,00%
1,5	70,97%	92,96%	80,49%	68,00%
1,6	70,97%	92,96%	80,49%	68,00%
1,7	70,97%	92,96%	80,49%	68,00%
1,8	70,97%	92,96%	80,49%	68,00%
1,9	70,97%	92,96%	80,49%	68,00%
2	70,97%	92,96%	80,49%	68,00%

*Fold* ke-4 akan menggunakan dokumen ke 301-400 sebagai dokumen latih dan dokumen ke 1-300 dan 401-500 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.22.

**Tabel 6.22 *Fold* ke-4 Variabel  $k_1$**

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,3	68.82%	95.52%	80.00%	68.00%
1,4	68.82%	95.52%	80.00%	68.00%
1,5	68.82%	95.52%	80.00%	68.00%
1,6	68.82%	95.52%	80.00%	68.00%
1,7	68.82%	95.52%	80.00%	68.00%
1,8	68.82%	95.52%	80.00%	68.00%

**Tabel 6.22 Fold ke-4 Variabel  $k_1$  (lanjutan)**

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,9	68.82%	95.52%	80.00%	68.00%
2	68.82%	69.57%	80.50%	69.00%

*Fold* ke-5 akan menggunakan dokumen ke 401-500 sebagai dokumen latih dan dokumen ke 1-400 sebagai dokumen uji. Hasilnya ditampilkan pada Tabel 6.23.

**Tabel 6.23 *Fold* ke-5 Variabel  $k_1$** 

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,3	69,88%	93,55%	80,00%	71,00%
1,4	69,88%	93,55%	80,00%	71,00%
1,5	69,88%	93,55%	80,00%	71,00%
1,6	69,88%	93,55%	80,00%	71,00%
1,7	69,88%	93,55%	80,00%	71,00%
1,8	69,88%	93,55%	80,00%	71,00%
1,9	69,88%	93,55%	80,00%	71,00%
2	69,88%	93,55%	80,00%	71,00%

Dari semua *Fold* di atas, mengikuti prinsip *K-Fold*, maka akan dihitung rata-ratanya untuk masing-masing  $k_1$ . Berikut adalah hasil perhitungan rata-rata *Accuracy*, *Precision*, *Recall* dan *F-measure* dari masing-masing  $k_1$ . Hasilnya ditampilkan pada Tabel 6.24.

**Tabel 6.24 Rata-rata Hasil 5-Fold Cross Validation untuk nilai  $k_1$** 

$k_1$	<i>Precision</i>	<i>Recall</i>	<i>F-Measure</i>	<i>Accuracy</i>
1,3	68,38%	95,43%	79,61%	68,40%
1,4	68,37%	95,11%	79,46%	68,20%
1,5	68,37%	95,11%	79,46%	68,20%
1,6	68,37%	95,11%	79,46%	68,20%
1,7	68,37%	95,11%	79,46%	68,20%
1,8	68,37%	95,11%	79,46%	68,40%
1,9	68,60%	95,11%	79,67%	68,60%
2	68,60%	89,92%	79,77%	68,80%

Berdasarkan nilai *F-measure* tertinggi inilah  $k_1 = 2$  dipilih untuk menjadi nilai  $k_1$  akhir, dimana bersama dengan nilai variabel  $k = 70$  dan  $b_s = 0,6$  dan  $v_1 = 2$  dan  $v_2 = 5$ , menjadi parameter yang menghasilkan *F-Measure* akhir tertinggi.

## 6.2 Analisis Pengujian

Setelah tahap pengujian dilakukan proses analisis pengujian. Hal ini bertujuan untuk memberikan uraian dan penjelasan dari hasil pengujian yang telah dilakukan. Analisis pengujian terdiri dari analisis pengujian nilai  $k$ ,  $b_s$ ,  $v_s$ , dan  $k_1$ . Berdasarkan hasil evaluasi untuk masing-masing pengujian, dapat disimpulkan bahwa semua perubahan parameter memengaruhi klasifikasi dengan menggunakan metode *Improved K-Nearest Neighbor*. Ditemukan bahwa parameter terbaik yang bisa digunakan untuk klasifikasi dengan IKNN dan BM25F adalah  $k = 70$ ,  $b_s = 0,6$ ,  $v_1 = 2$ ,  $v_2 = 5$  dan  $k_1 = 2$ .

Namun hasil akhir klasifikasi menunjukkan bahwa nilai *F-measure* akhir rata-rata *5-Fold* adalah 79,77%, dimana nilai ini kemungkinan didapatkan karena beberapa alasan, di antaranya adalah tidak imbangnya jumlah dokumen latih dan uji hasil dari pelabelan oleh Pakar, yang berlabel Ujaran Kebencian (324 Dokumen) dengan yang berlabel Non Ujaran Kebencian (176 Dokumen). Kemungkinan lainnya adalah kurangnya keberadaan *term* yang spesifik hanya mewakili salah satu dari kedua kelas, serta beberapa tagar yang digunakan belum tentu mencerminkan sentimen/perasaan dari sang penulis tweet. Bahkan terdapat sejumlah *tweet* yg bersifat ambigu untuk masing-masing kelas, karena sejumlah *tweet* tersebut memiliki sentimen berkebalikan dengan tagarnya, sehingga bisa saja diklasifikasikan ke Ujaran Kebencian maupun Non Ujaran Kebencian. Contoh *tweet* seperti yang dijabarkan sebelumnya ditampilkan pada Tabel 6.25.

**Tabel 6.25 Dokumen *tweet* bersifat ambigu**

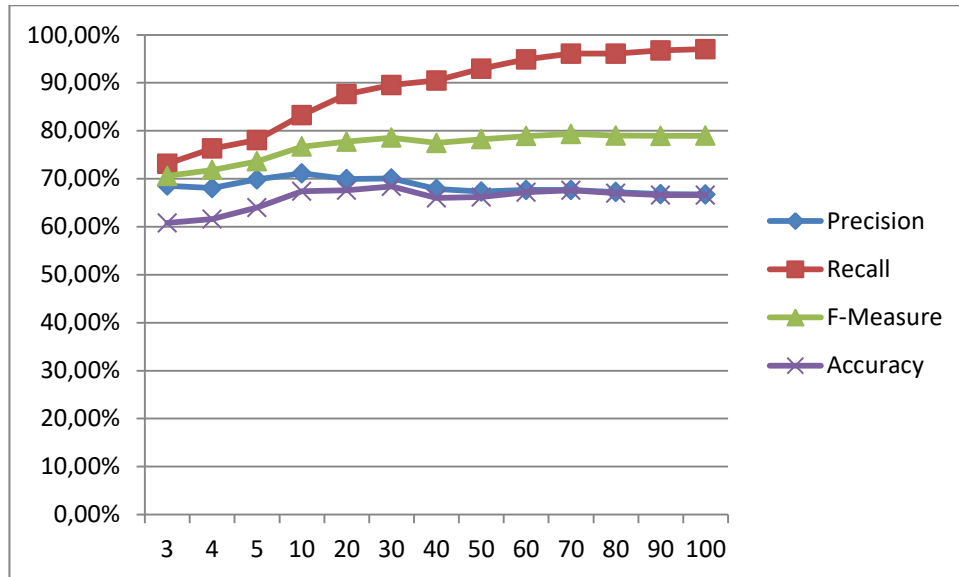
<i>Tweet</i>	<i>Hashtag</i>
Ribuan Buruh Terjun Demo ke DPR Besok. Ramaikan biar dia tahu salahnya. Rezim togog.	#RakyatBersatuBergerak
Ketika para Pemimpin memikirkan rakyatnya.. jalan panjang kan terbuka.	#BpjsRentenir #BanggaJadiRakyatOposisi

Sedangkan untuk analisis pengujian masing-masing variabel, yaitu  $b_s$ ,  $v_s$ , dan  $k_1$ , akan dijabarkan sebagai berikut.

### 6.2.1 Analisis Pengujian *5-Fold Cross Validation* Nilai $k$

Besar kecilnya nilai  $k$  mempengaruhi hasil klasifikasi yang dilakukan. Penggunaan nilai  $k = 3, 4, 5$  didorong oleh percobaan penggunaan nilai 6-9 yang malah menghasilkan penurunan pada hasil *F-Measure* pada rerata *5-fold*. Sedangkan untuk nilai  $k$  kelipatan 10 hasilnya menunjukkan peningkatan bertahap bagi hasil *F-measure* sehingga nilai kelipatan 10 dipertahankan penggunaannya. Pada penelitian ini, didapatkan hasil bahwa  $k = 70$  menghasilkan rerata nilai *F-measure* tertinggi yaitu sebesar 79,32%, dan ketika  $k = 3$  menghasilkan rerata nilai *F-measure* terendah yaitu sebesar 70,60%.

Nilai  $k$  yang rendah bisa jadi mengakibatkan proses klasifikasi hanya mengambil nilai ketetanggaan yang sedikit. Rerata hasil ditampilkan pada Gambar 6.1.



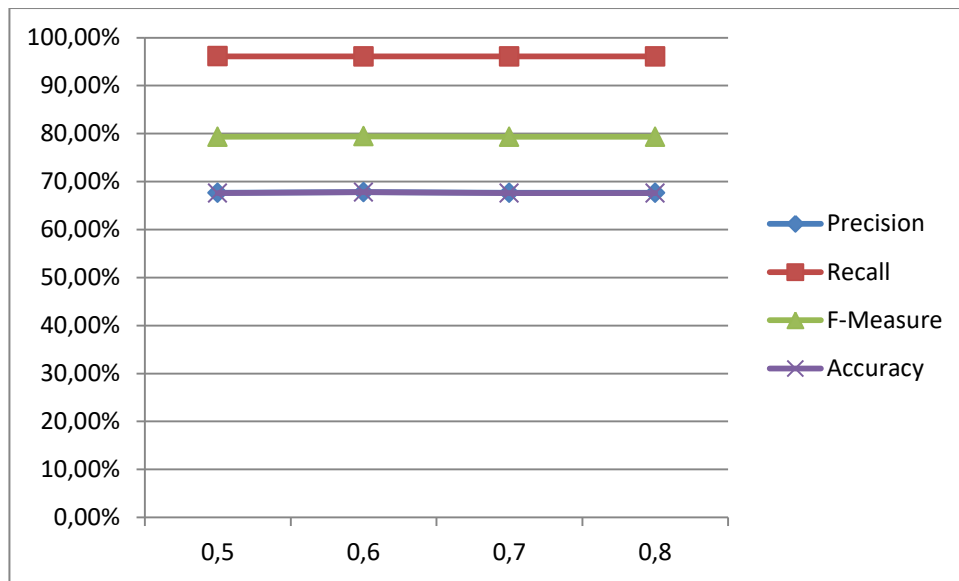
**Gambar 6.1 Grafik Rata-rata 5-Fold Cross Validation Nilai  $k$**

Disimpulkan bahwa nilai ketetanggaan 70 akan menghasilkan klasifikasi terbaik. Dapat dilihat dalam grafik, bahwa *Precision* mengalami penurunan (semakin banyak Non Ujaran Kebencian yang salah diklasifikasikan sebagai Ujaran Kebencian) dan *Recall* terus meningkat (semakin banyak Ujaran Kebencian yang berhasil diklasifikasikan dengan benar sebagai Ujaran Kebencian) seiring dengan semakin banyaknya tetangga yang diambil dalam klasifikasi.

Hal ini disinyalir disebabkan oleh jumlah anggota kelas Non Ujaran Kebencian jauh lebih sedikit (176 dokumen) dari pada Ujaran Kebencian (324 dokumen), sehingga semakin banyak tetangga yang diambil, kemungkinan untuk Non Ujaran Kebencian diklasifikasikan dengan benar akan semakin menurun, akan tetapi Ujaran Kebencian mengalami peningkatan klasifikasi yang benar.

### 6.2.2 Analisis Pengujian 5-Fold Cross Validation Nilai $b_s$

Pada penelitian ini, didapatkan hasil bahwa  $b_s = 0,6$  menghasilkan rerata nilai *F-measure* tertinggi yaitu sebesar 79,42%. Disamping itu, nilai *Accuracy*, *Precision*, *Recall* dn *F-measure* sama sekali tidak menunjukkan penurunan maupun peningkatan yang signifikan pada  $b_s$ . Rerata hasil ditampilkan pada Gambar 6.2.

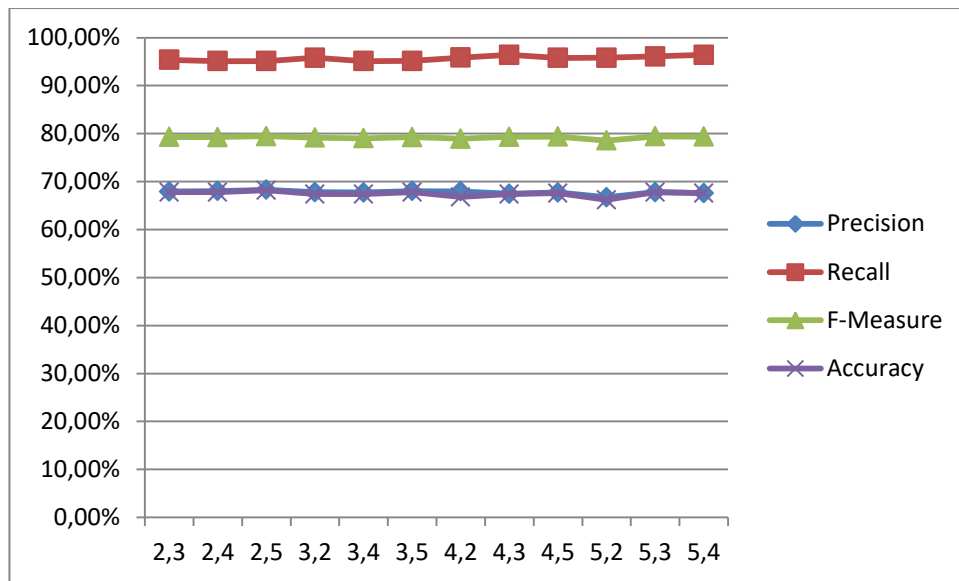


**Gambar 6.2 Grafik Rata-rata 5-Fold Cross Validation Nilai  $b_s$**

Nilai yang didapatkan tidak menunjukkan pola tertentu atau perubahan nilai *F-measure* yang signifikan, masing-masing metrik pengukuran memiliki rerata 5-Fold yang hanya berselisih sangat kecil (semua berselisih kurang dari 0,5%) antara satu  $b_s$  ke  $b_s$  yang lain. Meski begitu, pengukuran *F-measure*, *Precision* dan *Accuracy* menunjukkan nilai terbaik pada saat  $b_s = 0.6$ . Hal ini mengindikasikan bahwa parameter normalisasi yang bernilai tidak terlalu besar dan tidak terlalu kecil pada panjang *stream* tiap dokumen akan membuat hasil klasifikasinya semakin baik, walaupun pengaruhnya sangat kecil.

### 6.2.3 Analisis Pengujian 5-Fold Cross Validation Nilai $v_s$

Pada penelitian ini, didapatkan hasil bahwa  $v_1 = 2$  dan  $v_2 = 5$  menghasilkan rerata nilai *F-measure* tertinggi yaitu sebesar 79,46%. Disamping itu, *Accuracy*, *Precision*, *Recall* dan *F-measure* tidak menurun maupun meningkat secara signifikan. Rerata hasil ditampilkan pada Gambar 6.3.



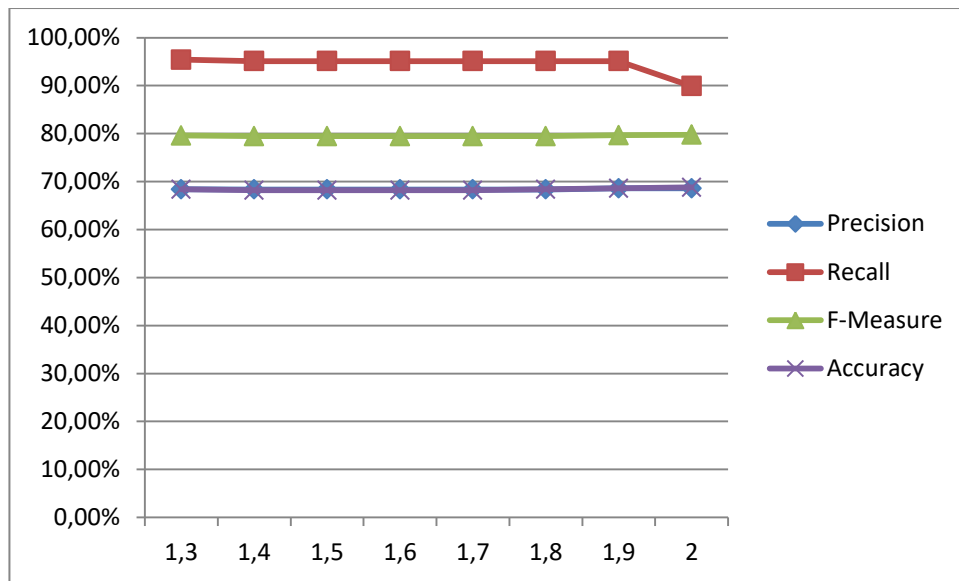
**Gambar 6.3 Grafik Rata-rata 5-Fold Cross Validation Nilai  $v_s$**

Nilai pengukuran yang didapatkan memiliki pola yang mirip dengan pengujian-pengujian sebelumnya, yaitu *Recall* dan *F-measure* selalu lebih tinggi dari pada *Accuracy* dan *Precision*. Bahkan pada beberapa titik  $v_1$  dan  $v_2$ , *Accuracy* dan *Precision* hampir memiliki nilai yang sama. Dan sama seperti hasil pengujian  $b_s$ , *F-measure*, *Precision* dan *Accuracy* menunjukkan nilai terbaik pada saat bersamaan, yaitu  $v_1 = 2$  dan  $v_2 = 5$ , yang menguatkan dugaan bahwa prioritas antara *stream* isi *tweet* dengan tagar memiliki bobot yang cukup berbeda.

Hasil yang didapat setelah pelatihan dan pengujian menunjukkan bahwa Pakar yang terlibat lebih menitikberatkan unsur penentu ujaran kebencian pada *stream* ke-2, yaitu tagar yang digunakan. Sehingga hasil pengujian  $v_s$  menunjukkan bahwa bobot *stream* ke-2 lebih besar.

#### 6.2.4 Analisis Pengujian 5-Fold Cross Validation Nilai $k_1$

Pada penelitian ini, didapatkan hasil bahwa  $k_1 = 2$  menghasilkan rerata nilai *Accuracy* dan *F-Measure* tertinggi yaitu sebesar 68,80% dan 79,77%. Disamping itu, *Accuracy*, *Precision*, *Recall* dan *F-measure* tidak menurun maupun meningkat secara signifikan. Rerata hasil ditampilkan pada Gambar 6.4.



**Gambar 6.4 Grafik Rata-rata 5-Fold Cross Validation Nilai  $k_1$**

Nilai pengukuran yang didapatkan memiliki pola yang mirip dengan pengujian-pengujian sebelumnya, yaitu *Recall* dan *F-measure* selalu lebih tinggi dari pada *Accuracy* dan *Precision*. Dan sama seperti hasil pengujian  $v_1$  dan  $v_2$ , nilai *F-Measure*, *Precision* dan *Accuracy* menunjukkan nilai terbaik pada saat bersamaan, yaitu  $k_1 = 2$ , namun tidak terlihat pola tertentu pada grafik yang signifikan, masing-masing *F-Measure*, *Precision* dan *Accuracy* pada setiap  $k_1$  berada pada rentang yang sama. Hanya *Recall* yang justru menunjukkan penurunan yang hampir tak terlihat pada grafik, namun menjadi sangat jelas antara  $k_1 = 1,9$  ke  $k_1 = 2$ , yang menandakan bahwa semakin besar nilai  $k_1$  sebagai parameter saturasi *term*, maka jumlah dokumen ujaran kebencian yang benar-benar diklasifikasikan ke kelas Ujaran Kebencian semakin menurun.



## BAB 7 PENUTUP

Bab Penutup pada penelitian ini berisi kesimpulan dari penelitian yang telah dilakukan dan saran yang diberikan oleh penulis untuk dipertimbangkan dalam pengembangan penelitian selanjutnya.

### 7.1 Kesimpulan

Berdasarkan hasil dari penelitian dan pengujian yang telah dilakukan, maka kesimpulan yang didapatkan adalah sebagai berikut.

1. Pemilihan bobot *stream* pada BM25F cukup mempengaruhi hasil probabilitas tiap kelas dalam klasifikasi IKNN yang dilakukan terhadap data *tweet* berbahasa Indonesia berisi Ujaran Kebencian. Hal tersebut terbukti ketika hasil akhir pengujian klasifikasi dengan program menunjukkan bahwa klasifikasi dengan *F-Measure* terbaik didapatkan ketika perbedaan bobot masing-masing *stream* cukup jauh, dengan bobot untuk tagar sebesar 5, dan bobot untuk *tweet* sebesar 2. Hal ini disebabkan oleh pelabelan Pakar pada dokumen latih yang lebih menitikberatkan ujaran kebencian melalui tagar yang digunakan, sehingga muncul kecenderungan ini pada hasil pengujiannya.
2. Hasil akhir terbaik untuk *F-Measure*, *Accuracy*, *Precision*, dan *Recall* dari rerata *5-Fold Cross Validation* yang didapatkan adalah 79,77% , 68,80%, 68,80%, dan 89,92% dengan  $k = 70$ ,  $b_s = 0,6$ ,  $v_1 = 2$ ,  $v_2 = 5$  dan  $k_1 = 2$ . Nilai *Accuracy* yang tidak tinggi disebabkan oleh jumlah dokumen latih yang berlabel Ujaran Kebencian tidak seimbang dengan yang berlabel Non Ujaran Kebencian (*imbalanced dataset*).

### 7.2 Saran

Berdasarkan kesimpulan yang telah ditarik dari penelitian ini, penulis memberikan beberapa saran yang dapat digunakan untuk penelitian selanjutnya.

1. Melakukan *oversampling* atau *undersampling* apabila menggunakan dataset yang *imbalance* untuk mengatasi kecenderungan atau bias klasifikasi terhadap kelas mayoritas.
2. Tidak dilakukan *shuffle* sebelum set dokumen dibagi ke dalam *5-Fold*, sehingga pada penelitian selanjutnya dapat di-*shuffle* terlebih dahulu.
3. Dokumen *tweet* yang digunakan mengandung cukup banyak kata tidak baku bahkan terdapat singkatan kata. Pada penelitian selanjutnya dapat dilakukan proses perbaikan kata tidak baku maupun singkatan kata untuk mengurangi adanya *term* yang bermakna sama namun penulisannya berbeda.

## DAFTAR REFERENSI

- Akbari, M.I.H.A.D., Novianty, A. & Setianingsih, C., 2017. Analisis Sentimen Menggunakan Metode Learning Vector Quantization. *e-Proceeding of Engineering*, pp.2283-92.
- Alfina, I., Mulia, R., Fanany, M.I. & Ekanata, Y., 2017. Hate Speech Detection in the Indonesian Language: A Dataset and Preliminary Study. In *2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. Bali, 2017. IEEE.
- Angiani, G. et al., 2016. A Comparison between Preprocessing Techniques for Sentiment Analysis in Twitter. In *Proceedings of the 2nd International Workshop on Knowledge Discovery on the WEB*. Cagliari, 2016.
- Bozkir, A.S. et al., 2017. Spam E-Mail Classification by Utilizing N-Gram Features of Hyperlink Texts. In *The 11th IEEE International Conference AICT2017*. Moscow, 2017.
- Broder, A.Z., Glassman, S.C., Manasse, M.S. & Zweig, G., 1997. Syntactic Clustering of the Web. *Computer Networks and ISDN Systems*, 29(8-13), pp.1157-66.
- Cortes, C. & Vapnik, V., 1995. Support-vector networks. *Machine Learning*, 20(3), p.September.
- Ezeibe, C.C., 2015. HATE SPEECH AND ELECTORAL VIOLENCE IN NIGERIA. In *INEC Nigeria*. Nsukka, 2015. Department of Political Science, University of Nigera.
- Fawcett, T., 2006. An Introduction to ROC analysis. *Pattern Recognition Letters*, pp.861-74.
- Hakim, M., Fauzi, M.A. & Indriati, 2019. Klasifikasi Ujaran Kebencian pada Twitter Menggunakan Metode Naïve. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (JPTIIK)*, 3(3), pp.2443-51.
- Hossin, M. & Sulaiman, M.N., 2015. A Review on Evaluation Metrics For Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process (IJDMP)*, 5(2).
- KOMNASHAM RI, 2015. In *Penanganan Ujaran Kebencian (Hate Speech)*. Jakarta: KOMNASHAM. p.3.
- Li, B., Yu, S. & Lu, Q., 2003. *An Improved k-Nearest Neighbor Algorithm for Text Categorization*. [Online] Shenyang: Cornell University Tersedia di: <https://arxiv.org/abs/cs/0306099> [Diakses 15 Juli 2019].
- Luqyana, W.A., Cholissodin, I. & Perdana, R.S., 2018. Analisis Sentimen Cyberbullying pada Komentar Instagram dengan Metode Klasifikasi Support Vector Machine. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, pp.4704-13.

- Ma, C.M., Yang, W.S., Cheng, B.W, 2014. How the Parameters of K-nearest Neighbor Algorithm Impact on the Best Classification Accuracy: In Case of Parkinson Dataset. *Journal of Applied Sciences*, 14(2), pp. 171-76.
- Manning, C.D., Raghavan, & Schütze, H., 2009. *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press.
- Miao, Z., Tang, Y., Sun, L., He, Y., Xie, S., 2014. An Improved KNN Algorithm for Imbalanced Data Based on Local Mean. *Journal of Computational Information Systems*, 10(12), pp. 5139-46.
- Movanita, A.N.K., 2017. *Ini Hasil Kerja Polri Perangi Kejahatan Siber Sepanjang 2017*. [Online] PT. Kompas Cyber Media Tersedia di: <https://nasional.kompas.com/read/2017/12/29/17233911/ini-hasil-kerja-polri-perangi-kejahatan-siber-sepanjang-2017> [Diakses 2 Juli 2019].
- Mujilawati, S., 2016. PRE-PROCESSING TEXT MINING PADA DATA TWITTER. In *Seminar Nasional Teknologi Informasi dan Komunikasi (SENTIKA) 2016*. Yogyakarta, 2016. Universitas Islam Lamongan.
- Munir, M.M., Fauzi, M.A. & Pradana, R.S., 2018. Implementasi Metode Backpropagation Neural Network berbasis Lexicon Based Features dan Bag of Words Untuk Identifikasi Ujaran Kebencian Pada Twitter. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 10(2), pp.3182-91.
- Nugroho, A.S., Witarto, A.B. & Handoko, D., 2003. Support Vector Machine. *Proceeding Indones. Sci. Meeting Cent. Japan*, p.2.
- Olson, D.L. & Delen, D., 2008. *Advanced Data Mining Techniques*. Springer.
- Perez-Aguera, J.R. et al., 2010. Using BM25F for Semantic Search. In *SEMSEARCH '10 Proceedings of the 3rd International Semantic Search Workshop Proceedings of the 3rd International Semantic Search Workshop*. New York, 2010. ACM.
- Prasanti, A.A., Fauzi, M.A. & Furqon, M.T., 2018. Klasifikasi Teks Pengaduan Pada Sambat Online Menggunakan Metode N-Gram dan Neighbor Weighted K-Nearest Neighbor (NW-KNN). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (JPTIIK)*, pp.594-601.
- Puspitasari, A.A., Santoso, E. & Indriati, 2018. Klasifikasi Dokumen Tumbuhan Obat Menggunakan Metode Improved k-Nearest Neighbor. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (JPTIIK)*, 11(2), pp.486-92.
- Robertson, S. & Zaragoza, H., 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4), pp.333-89.

- Shung, K.P., 2018. *Accuracy, Precision, Recall or F1?* [Online] Tersedia di: <https://towardsdatascience.com/Accuracy-Precision-Recall-or-f1-331fb37c5cb9> [Diakses 21 Agustus 2019].
- Singh, S. & Shukla, S., 2016. Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. In *IEEE 6th International Conference on Advanced Computing*. Bhimavaram, 2016. IEEE.
- Sutton, G.M. & King, R.D., 2013. HIGH TIMES FOR HATE CRIMES: EXPLAINING THE TEMPORAL CLUSTERING OF HATE-MOTIVATED OFFENDING. *Criminology*, 51(4), pp.871-94.
- Tala, F.Z., 2003. A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia.
- Vijayakumar, W.S., 1999. Sequential Support Vector Classifier and Regression. *International Conference on Soft Computing*, pp.610-19.
- Williams, L.M. & Burnap, P., 2015. Cyber Hate Speech on Twitter: An Application of Machine Classification and Statistical Modeling for Policy and Decision Making. *Policy & Internet*, 22 April. pp.1944-2866.
- Wah, Y.B., Rahman, H. A. A., He, H. & Bulgiba, A., 2016. Handling imbalanced dataset using SVM and k-NN approach. *AIP Conference Proceedings*, 1750(1).
- Yu, L., Wang, S., Lai, K.K., Zhou, L., 2008. *Bio-Inspired Credit Risk Analysis: Computational Intelligence with Support Vector Machines*. Springer Science & Business Media.

## LAMPIRAN A DATASET

No	<i>Tweet</i>	Label Kelas
1	Lagu sindiran, tapi untung-untung juga yang disindir ngerasa. #RezimTeror #RezimTeror	1
2	Hoax Apapun yang Mereka lakukan, Rezim Tetap Melindungi mereka..? Keadilan Hukum Bagi Semua Cuma Slogan Saja. Sudah Hilang Di negeri 62 Bung..??? * JANJI" MU PALSU *. Ndro:) #BebaskanPelajarDanMahasiswa	1
3	Perjuangan kalian akan ditulis dalam sejarah sebagai pahlawan pemberani, pencari keadilan! #STMBergerak #STMbersatu	0
4	Ternyata salah kita semuanya Daripada kerja bikin rakyat sengsara, demo dimana-mana #DPRlebihBaikTidur	0
5	Ribuan Buruh Terjun Demo ke DPR Besok. Ramaikan biar dia tahu salahnya. Rezim togog #RakyatBersatuBergerak	1
6	Yang memohon pertanyaan dibocorkan KPU adalah pihak Prabowo. Sudah dibocorkan saja Prabowo masih sembarangan, bagaimana kalau tidak dibocorkan? bisa ngamuk-ngamuk doang di acara debat.. #Prabohong #Prabocor #DebatPilpres2019	1
7	Pakar politik: Tindakan rezim Jokowi saat ini semakin represif membungkam suara mahasiswa, hal ini terbukti dari dua mahasiswa yang meregang nyawa. Sikap keras ini berbeda dengan menghadapi aksi yang berbau terorisme dan separatisme di #Papua	1
8	POTRET REZIM DZALIM. APA MAU TETAP KITA BIARKAN SAUDARAKU ? GIMANA ANAK CUCU KITA KELAK, KALAU REZIM INI DIBIARKAN? BUKANKAH LEBIH CEPAT LEBIH BAIK? SEMUA ITU HARUS BERAKHIR SECEPATNYA!!! <a href="https://pbs.twimg.com/media/EFklZA1VUAAYURX?format=jpg&amp;name=small">https://pbs.twimg.com/media/EFklZA1VUAAYURX?format=jpg&amp;name=small</a> #KitaDukungJokowiMundur	1
9	Sedih kalau liat orang2 demo yang ujung2nya ngerusak fasilitas umum. Mereka bayar pajak tidak sih? Serious nanya nih. (pake hastag ini biar dapet opini dari mahasiswa) #HidupMahasiswa	0
10	Entah apa yang merasukimu? @jokowi @DPR_RI #HariTaniNasional #HidupMahasiswa #HidupRakyatIndonesia #TolakRKUHP #MosiTidakPercayaDPR <a href="https://t.co/a5p8v57Y3d">https://t.co/a5p8v57Y3d</a>	0
11	Tuntutan adik2 Mahasiswa di dalam Demonstrasi hari ini sangat wajar kok. Semua mewakili keresahan rakyat. Bukan hanya mewakili kelompok tertentu. yang tidak setuju kepada 7 Desakan ini, mungkin memang perlu dilawan. #HidupMahasiswa #HIDUPRAKYATINDONESIA #RakyatDukungAksiMahasiswa <a href="https://t.co/brPFBLpIIK">https://t.co/brPFBLpIIK</a>	0

No	<i>Tweet</i>	Label Kelas
12	Inti nya mereka sudah muak dengan rezim ini !! #KekuasaanDitanganRakyat	1
13	Indonesia berduka!! Papua mandi darah, minang berduka, Demonstrasi dimana2... eh pemimpin negara belajar ngerap.. ini negara apa?? #JokowiGakMampuUrusNegara #jokowiMUNDUR	1
14	Rezim zalim jokowi pasti akan hancur. #JokowiHarusLengser #jokowiMUNDUR #RezimPenghancurBangsa	1
15	Hak rakyat di revisi, tapi itu pihak yang asal mukul & tendang in orang ga direvisi? Gw ga liat sisi mengayomi & melindungi dari orang yang sudah tumbang tp tetep di injek trus <a href="https://twitter.com/i/status/1176706635676495873">https://twitter.com/i/status/1176706635676495873</a> #HidupMahasiwa #IndonesiaKuDamai #PolisiMengayomiBukanKeroyokan	0
16	Biadab!!! Ini Pemukulan namanya @DivHumas_Polri Terjadi Di Belakang Kantor DPRD Sumut. @DatuakPanduko @Fahrihamzah @GusUmarChelsea #PolisiMengayomiBukanKeroyokan	1
17	Kalau ada daftar profesi yang paling dibenci, pasti polisi jadi salah satunya~ #PolisiMengayomiBukanKeroyokan	1
18	Semoga Allah melindungi kalian & semoga Allah jaga negeri ini #PolisiMengayomiBukanKeroyokan	0
19	Mungkin kalian memakai seragam karna uang, bukan murni. Jadi kalian tidak akan tau bagaimana seharusnya mengayomi masyarakat. #PolisiMengayomiBukanKeroyokan #PolisiMusuhRakyat #polisi	0
	.....	
500	Keparaaaatttgt itu bukan polisi atau polisi keparat! Bukan aparat! #PolisiMengayomiBukanKeroyokan #PolisiMusuhRakyat	1

Keterangan:

Dataset selengkapnya dapat diakses di [bit.ly/dataset\\_skripsi\\_difa](http://bit.ly/dataset_skripsi_difa)