KLASIFIKASI BERITA BERBAHASA INDONESIA DENGAN ALGORITMA *ITERATIVE DICHOTOMIZER TREE*

SKRIPSI

Oleh:

Mohammad Rulisk Fahrobi

0610963028-96

PROGRAM STUDI ILMU KOMPUTER

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS BRAWIJAYA

MALANG

2012


UNIVERSITAS BRAWIJAYA

KLASIFIKASI BERITA BERBAHASA INDONESIA DENGAN ALGORITMA *ITERATIVE DICHOTOMIZER TREE*

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Oleh:

Mohammad Rulisk Fahrobi

0610963028-96

PROGRAM STUDI ILMU KOMPUTER

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS BRAWIJAYA

MALANG

2012


UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN SKRIPSI

Klasifikasi Berita Berbahasa Indonesia Dengan Algoritma *Iterative Dichotomizer Tree*

Oleh :
Mohammad Rulisk Fahrobi
0610963028-96

Setelah dipertahankan di depan Majelis Penguji
Pada tanggal 26 April 2012

dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Dewi Yanti L.,S.Kom,M.Kom	Dany Primanita K., S. T
NIP. 198111162005012004	NIP. 197711162005012003

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP.196709071992031001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama	**: Mohammad Rulisk Fahrobi**
NIM	**: 0610963028-96**
Jurusan	**: Matematika**
Program Studi	**: Ilmu Komputer**
Penulis skripsi berjudul	**: Klasifikasi Berita Berbahasa Indonesia Dengan Algoritma *Iterative Dichotomizer Tree***

Dengan ini menyatakan bahwa :

1. **Isi dari Skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.**
2. **Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 26 April 2012
Yang menyatakan,

Mohammad Rulisk Fahrobi
NIM. 0610963028-96

KLASIFIKASI BERITA BERBAHASA INDONESIA DENGAN ALGORITMA ITERATIVE DICHOTOMIZER TREE

ABSTRAK

Perkembangan teknologi mengakibatkan penyampaian berita semakin beragam. Internet merupakan salah satu teknologi untuk penyampaian berita. Banyaknya dokumen berita diperlukan pengelompokan berita kedalam kategori. Dengan dibuatnya pengkategorian berita dapat memudahkan pembaca dalam mencari berita. Banyaknya dokumen berita mengakibatkan tidak bisa dilakukan pengkategorian berita secara manual. *Text mining* merupakan bagian dari *data mining* yang memiliki solusi untuk permasalahan tersebut. *Iterative Dichotomizer Tree* (ID3) adalah salah satu algoritma dalam *data mining* yang dapat menyelesaikan permasalahan tersebut. ID3 adalah algoritma pembangkitan pohon keputusan dan termasuk metode *supervised learning* karena memerlukan sejumlah data untuk proses pembelajaran. Penerapan ID3 dalam pengelompokan berita dapat diterapkan dengan melalui *preprocessing*, data fitur dan data transformasi. Transformasi data mengunakan distribusi frekuensi, karena ID3 tidak bisa membaca data numerik maka data numerik dirubah ke dalam data kategori. Hasil tertinggi dari perhitungan *recall*, *precision* dan *f-measure* pada perbandingan data latih 80 % dengan nilai *recall* 88.658333 %, *precision* 86.458333 % dan *f-measure* 86.608333 %. Dengan hasil pengujian ini maka algoritma ID3 dapat digunakan untuk pengklasifikasian berita berbahasa Indonesia.

CLASSIFICATION OF INDONESIAN LANGUAGE NEWS USING ITERATIVE DICHOTOMIZER TREE ALGORITHM

ABSTRACT

Technological developments resulting in the delivery of news to be increasing diversely. Internet is one of the technologi for news delivery. The number of news documents, required a certain grouping of news into categories to facilitate the reader to look for news. With so many news documents, categorization of news may not be possible manually. Text mining is part of the data mining that has a solution to these problems. Iterative Dichotomizer Tree is one of the data mining algorithms that can solve these problems. ID3 is a decision tree generation algorithms and includes a supervised learning method because it requires a certain amount of data to the learning process. The application of ID3 in the grouping can be applied to the news through preprocessing, feature data, and data transformation. Data transformation is using frequency distributions, because ID3 can not read the numeric data then converted numeric data into categorical data. The highest results of the calculation of recall, precision and f-measure on a comparison of the data train 80% of the value of 88.658333% recall, precision and 86.458333% 86.608333% f-measure. With the results of this test, the ID3 algorithm can be used to classify the news in Bahasa Indonesia.

Kata Pengantar

Alhamdulillahi rabbil 'alamin. Puji syukur penulis panjatkan kehadirat *Allah* SWT, karena atas segala rahmat dan limpahan hidayah-Nya, Skripsi yang berjudul **" Klasifikasi Berita Berbahasa Indonesia Dengan Algoritma *Iterative Dichotomizer Tree* "** ini dapat berjalan dengan baik. Skripsi ini disusun dan diajukan sebagai syarat untuk memperoleh gelar sarjana pada program studi Ilmu Komputer, jurusan Matematika, fakultas MIPA, universitas Brawijaya.

Shalawat serta salam tetap tercurahkan kepada Baginda Rasulullah Muhammad SAW, makhluk paling mulia yang senantiasa memberikan cahaya petunjuk, seorang uswatun hasanah yang telah membawa agama Allah yaitu agama Islam menjadi agama yang *Rahmatan Lil 'Alamin*.

Dalam penyelesaian tugas akhir ini, penulis telah mendapat begitu banyak bantuan baik moral maupun materiil dari banyak pihak. Atas bantuan yang telah diberikan, penulis ingin menyampaikan penghargaan dan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Dewi Yanti Liliana, S.Kom., M.Kom, selaku dosen pembimbing utama yang telah meluangkan waktu untuk memberikan pengarahan dan masukan bagi penulis.
2. Dany Primanita Kartikasari, S.T., selaku dosen pendamping yang telah banyak memberikan bimbingan serta bantuan.
3. Drs. Marji, M.T., selaku ketua program studi Ilmu Komputer.
4. Dr. Abdul Rouf Alghofari, M.Sc., selaku ketua Jurusan Matematika.
5. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
6. Segenap staf dan karyawan di Jurusan Matematika Fakultas MIPA Universitas Brawijaya yang telah banyak membantu penulis dalam pelaksanaan penyusunan skripsi ini.
7. Kedua orang tua dan keluarga tercinta, terima kasih atas semua doa, kasih sayang dan perhatian yang tulus serta dukungan yang telah diberikan.

8. Rekan-rekan di Program Studi Ilmu Komputer Fakultas MIPA Universitas Brawijaya yang telah banyak memberikan bantuannya demi kelancaran pelaksanaan penyusunan skripsi ini.
9. Dan semua pihak yang telah terlibat baik secara langsung maupun tidak langsung yang tidak dapat penulis sebutkan satu persatu terima kasih atas semua bantuan yang telah diberikan.

Semoga skripsi ini bermanfaat bagi pembaca sekalian. Akhirnya, penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan dan memiliki banyak kekurangan, sehingga dengan segala kerendahan hati penulis mengharapkan kritik dan saran dari pembaca.

Malang, 26 April 2012

Penulis

Daftar Isi

Daftar Gambar

Daftar Tabel

Daftar *Sorce Code*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Berita menjadi kebutuhan penting bagi manusia. Setiap hari manusia mencari informasi terbaru tentang berbagai macam peristiwa yang terjadi. Tokoh jurnalis Indonesia Dja'far H. Assegaff mengartikan berita sebagai laporan tentang fakta atau ide yang baru, yang dipilih oleh staf redaksi suatu harian untuk disiarkan, yang dapat menarik perhatian pembaca (Dja'far H. Assegaff, 1991).

Seiring berkembangnya teknologi, penyampaian berita ikut berkembang dan semakin beragam. Awalnya berita disampaikan melalui media cetak dan berkembang melalui media elektronik. Internet sebagai salah satu media elektronik membawa dampak besar terhadap teknologi penyampaian berita. Dengan teknologi internet banyak bermunculan surat kabar berbasis web, di antaranya : kompas, liputan6, okezone, dan vivanews.

Semakin meningkatnya kebutuhan berita berbasis web, diperlukan suatu pengkategorian untuk mempermudah pembaca berita memilih berita yang hendak dibaca. Dokumen berita yang semakin banyak tidak mungkin dilakukan pengkategorian berita secara manual, diperlukan suatu pengkategorian otomatis.

Algoritma klasifikasi teks mulai dikembangkan. Rizal Ramadan (2007) menerapkan empat algoritma klasifikasi teks dalam pengklasifikasian bahasa Inggris dengan algoritma *Iterative Dichotomizer Tree* (ID3), *K-Nearest Neighbor, Naïve Bayes Classifier,* dan *Neural Network*. Hasil yang didapat dari penelitian tersebut, *Iterative Dichotomizer tree* dengan nilai akurasi 79,48 % dan waktu rata-rata 0,5 menit, *K-Nearest Neighbor* dengan nilai akurasi 40,72 % dan waktu rata-rata 2 menit, *Naïve Bayes Classifier* dengan nilai akurasi 74,56 % dan waktu rata-rata 0,5 menit, dan *Neural Network* dengan nilai akurasi 31,18 % dan waktu rata-rata 9 menit. Pembentukan pohon keputusan mengunakan algoritma *Iterative Dichotomizer tree* merupakan algoritma terbaik dalam melakukan klasifikasi dokumen teks. Hal ini terlihat dari nilai rata-rata tingkat akurasi yang didapatkan dari hasil eksperimen yang mendekati 80% (Riza Ramadan, 2007).

Iterative Dichotomizer Tree merupakan salah satu algoritma *text mining* dengan menggunakan metode *supervised learning*. Diperlukan data latih sebagai pembelajaran. *Preprocessing*, *featuring data* dan *transformation data* dilakukan sebelum masuk tahap pembelajaran. *Transfrmation data* menggunakan distribusi frekuensi untuk merubah data numerik menjadi data kategori.

Menerapkan algoritma *Iterative Dichotomizer Tree* pada klasifikasi berita bahasa Indonesia diharapkan memiliki tingkat akurasi yang tinggi. Berdasarkan latar belakang yang telah dipaparkan, maka judul yang diambil dalam skripsi ini adalah **" Klasifikasi Berita Berbahasa Indonesia Dengan Algoritma *Iterative Dichotomizer Tree* "**.

1.2 Rumusan Masalah

Rumusan masalah dalam skripsi ini adalah :

1. Bagaimana merancang dan mengimplementasikan algoritma *Iterative Dichotomizer Tree* dalam pengklasifikasian berita berbahasa Indonesia?
2. Berapa besar tingkat *precision, recall* dan *f-measure* algoritma *Iterative Dichotomizer Tree* pada sejumlah sampel berita berbahasa Indonesia?

1.3 Tujuan

Beberapa tujuan yang ingin dicapai dalam penelitian skripsi ini adalah :

1. Merancang dan membangun sistem yang dapat mengklasifikasikan berita berbahasa Indonesia dengan pengklasifikasian *Iterative Dichotomizer Tree.*
2. Menghitung *precision, recall* dan *f-measure* pada perangkat lunak klasifikasi berita berbahasa Indonesia.

1. 4 Batasan Masalah

Batasan masalah yang diangkat dalam skripsi ini adalah :

1. Dokumen berita berbahasa Indonesia yang akan digunakan dalam penelitian ini berupa dokumen dalam format **.html*.
2. Dokumen berita yang digunakan sebagai dokumen latih dan dokumen uji tidak menggunakan dokumen yang sama, kedua

dokumen tersebut hanya bersumber dari situs berita online vivanews periode Juli 2011 (www.vivanews.com).
3. Kata yang diproses berdasarkan jumlah kemunculan dan tidak mempermasalahkan posisi kata.
4. Suatu kata berdiri sendiri, tidak berhubungan dengan kata disekitarnya untuk menyederhanakan proses pengolahan.
5. Proses *stemming* tidak meperhatikan infiks (sisipan), hanya menghitung prefiks (awalan) dan sufiks (akhiran).
6. Terdiri dari 4 kategori, yaitu politik, bisnis, metropolis dan olahraga.
7. Jumlah data setiap kategori sama.
8. Proses penelitian dilakukan secara offline.

1.5 Manfaat

Manfaat hasil skripsi ini akan menghasilkan perangkat lunak yang digunakan dalam mengklasifikasikan berita berbahasa Indonesia kedalam suatu kategori tertentu.

1.6 Sistematika Penulisan

BAB I : PENDAHULUAN
Berisi latar belakang, permasalahan, tujuan, batasan masalah, dan manfaat serta sistematika penulisan skripsi.

BAB II : DASAR TEORI
Berisi teori tentang berita, teks mining, metode ID3 dan teori-teori yang berhubungan dengan metode pengklasifikasian.

BAB III : METODOLOGI DAN PERANCANGAN
Berisi metode-metode yang digunakan dalam pembuatan sistem pengklasifikasian berita berbahasa Indonesia kedalam suatu kategori dengan mengunakan pengklasifikasian ID3.

BAB IV : HASIL DAN PEMBAHASAN
Berisi tentang penjelasan implementasi sistem dan hasil pengujian.

BAB V : PENUTUP
Berisi kesimpulan yang diperoleh dari hasil pengujian dan saran-saran untuk pengembangan.


UNIVERSITAS BRAWIJAYA

BAB II
DASAR TEORI

Pada bab II berisi pembahasan dasar teori yang digunakan dalam penelitian. Sub bab pertama berisi penjelasan tentang berita, apa yang dimaksud dengan berita dan berbagai macam jenis berita. Sub bab kedua membahas tentang morfologi bahasa Indonesia dan metode *stemming* dalam bahasa Indonesia. Sub bab berikutnya adalah *teks mining*, penjelasan tentang *teks mining* dan tahapan dalam *teks mining*. *Decision tree* dan algoritma *Iterative Dichotomizer Tree* dijelaskan pada bab ini. Sub bab terakhir membahas metode dalam melakukan evaluasi penelitian.

2.1 Berita

2.1.1 Definisi Berita

Seorang tokoh jurnalis Indonesia Dja'far Husin Assegaff mengartikan berita sebagai laporan tentang fakta atau ide yang termasa (baru), dipilih oleh staf redaksi suatu harian untuk disiarkan, yang dapat menarik perhatian pembaca (Dja'far Husin Assegaff, 1991).

Definisi berita dapat juga dikutip dari Nancy Nasution, laporan tentang peristiwa-peristiwa yang terjadi, yang ingin diketahui oleh umum, dengan sifat-sifat aktual, terjadi di lingkungan pembaca, mengenai tokoh terkemuka, akibat peristiwa tersebut berpengaruh terhadap pembaca (Basuki, 1983).

Dalam kamus besar bahasa Indonesia yang ditulis oleh W.J.S. Poerwadarminta, menjelaskan bahwa berita adalah laporan tentang suatu kejadian yang terbaru. Dari beberapa pendapat tersebut dapat di simpulkan berita merupakan ringkasan dari suatu peristiwa penting yang menarik untuk di ketahui masyarakat umum dan bersifat fakta atau benar terjadi.

2.1.2 Bagian Berita

Pada umumnya suatu berita terdiri dari : *headline*, *deadline*, *lead*, dan *body* (Basuki, 1983).

1. *Headline* (judul berita)

Headline disebut juga judul bagian terpenting dari berita, sering dilengkapi dengan anak judul. Judul digunakan sebagai :

a. Penolong untuk pembaca agar segera mengetahui peristiwa yang akan diberitakan,
b. Menonjolkan satu berita dengan dukungan teknik grafika.

2. *Dateline* (batas tanggal berita)

Ada yang terdiri atas nama media massa, tempat kejadian dan tanggal kejadian. Ada pula yang terdiri atas nama media massa, tempat kejadian dan tanggal kejadian. Tujuannya adalah untuk menunjukkan tempat kejadian dan inisial media.

3. *Lead* (teras berita)

Lead biasanya ditulis pada paragraf pertama sebuah berita. Ia merupakan unsur yang paling penting dari sebuah berita, yang menentukan apakah isi berita akan dibaca atau tidak. Ia merupakan sari pati sebuah berita, yang melukiskan seluruh berita secara singkat.

4. *Body* (tubuh berita)

Isinya menceritakan peristiwa yang dilaporkan dengan bahasa yang singkat, padat, dan jelas. Dengan demikian *body* merupakan perkembangan berita.

2.2 *Stemming* Pada Bahasa Indonesia

Dalam melakukan *stemming* bahasa Indonesia terlebih dulu memahami morfologi bentuk kata dari bahasa Indonesia. Berikut akan menjelaskan morfologi kata dalam bahasa Indonesia.

2.2.1 Struktur Morfologi Kata Bahasa Indonesia

Morfologi adalah bagian dari ilmu bahasa yang membicarakan atau yang mempelajari seluk beluk kata serta pengaruh perubahan-perubahan bentuk kata terhadap golongan dan arti kata, atau dengan kata lain dapat dikatakan bahwa morfologi mempelajari seluk-beluk bentuk kata serta fungsi perubahan-perubahan bentuk kata itu (Ramlan, 1995).

Secara umum, setiap struktur terbina dari unsur-unsur yang lebih kecil. Struktur bahasa juga mempunyai unsur yang dicantumkan agar menjadi satu perkataan. Morfologi memiliki dua unsur yang terlibat dalam membentuk perkataan yaitu morfem dan kata.

Unsur-unsur kata seperti kereta, pokok, udara dan duduk tidak dapat diperkecil lagi karena tidak akan mempunyai makna. Dalam *morfologi* kata dapat digunakan untuk membentuk perkataan lain.

Morfologi kata bahasa Indonesia bisa terdiri dari struktur infleksional dan derivasional. Infleksional adalah struktur yang paling sederhana dalam penambahan sufiks dimana tidak mempengaruhi arti sebenarnya dari kata dasar yang dilekati (Tala, 2003). Sufiks infleksional dapat dibagi menjadi 2 jenis :

1. Sufiks -lah, -kah, -pun, -tah. Sufiks ini sebenarnya adalah partikel yang tidak mempunyai arti. Keberadaannya pada suatu kata adalah untuk penekanan. Contoh :
   dia + kah → diakah
   duduk + lah → duduklah
2. Sufiks -ku, -mu, -nya. Sufiks ini berfungsi sebagai kata ganti kepunyaan. Contoh :
   tas + ku → tasku
   buku + mu → bukumu

Sufiks-sufiks diatas dapat melekat pada kata dasar secara bersama-sama. Adapun aturan urutannya adalah sufiks pada jenis kedua selalu diletakkan sebelum sufiks jenis pertama. Sehingga struktur morfologi pada kata infleksional adalah :

Infleksional = (kata dasar +kata ganti) | (kata dasar + partikel) |
(kata dasar + kata ganti + partikel)

Penambahan sufiks infleksional tidak akan merubah bentuk dasar dari kata berimbuhan (Tala, 2003). Dengan kata lain, tidak ada penghilangan atau peleburan kata dasar pada kata berimbuhan. Kata dasar dapat ditentukan dengan mudah pada struktur infleksional. Struktur derivasional dalam bahasa Indonesia terdiri dari prefiks, sufiks dan kombinasi dari keduanya. Prefiks yang sering dipakai adalah : ber-, di-, ke-, meng-, peng-, per-, ter-. Contoh penggunaan prefiks adalah :

ber + lari → berlari
di + makan → dimakan
ke + kasih → kekasih
meng + ambil → mengambil
peng + atur → pengatur
per + lebar → perlebar
ter + baca → terbaca

Beberapa prefiks seperti ber-, meng-, peng-, per-, ter- mungkin akan berubah menjadi beberapa bentuk yang berbeda. Bentuk dari setiap prefiks bergantung pada karakter pertama dari kata dasar yang dilekatinya. Tidak seperti struktur infleksional, pada struktur derivasional pengucapan kata mungkin berubah setelah adanya penambahan prefiks. Seperti contoh menyapu yang terdiri dari prefiks meng- dan kata dasar sapu. Prefiks meng- berubah menjadi meny- dan karakter pertama dari kata dasar mengalami peleburan.

Sufiks derivasional adalah -i, -kan, -an (Tala, 2003). Contoh penggunaan sufiks derivasional adalah :

gula + i → gulai
makan + an → makanan
sampai + kan → sampaikan

Penambahan sufiks tidak mengubah bentuk dasar dari suatu kalimat.

Disebutkan sebelumnya, struktur derivasional juga terdiri dari konfiks, yaitu gabungan dari prefiks dan sufiks yang melekat secara bersama-sama pada suatu kata. Contoh :

per + main + an → permainan
ke + kalah + an → kekalahan
ber + jatuh + an → berjatuhan
meng + ambil + i → mengambili

Tidak semua prefiks dan sufiks dapat dikombinasikan menjadi sebuah konfiks. Ada beberapa kombinasi prefiks dan sufiks yang tidak diperbolehkan. Kombinasi tersebut ditunjukkan pada tabel 2.1.

Tabel 2.1 Pasangan Konfiks yang tidak diperbolehkan (Tala, 2003)

Prefiks	Sufiks
Ber	i
Di	an
Ke	i | kan
meng	an
peng	i | kan
ter	an

Prefiks/konfiks dapat ditambahkan pada suatu kata yang telah terdapat konfiks/prefiks, yang menghasilkan struktur prefiks ganda. Seperti pada pembentukan sebuah konfiks, pada pembentukan prefiks ganda, tidak semua prefiks/konfiks dapat ditambahkan pada kata yang telah mendapatkan prefiks/konfiks. Ada beberapa aturan dalam urutan pembentukan prefiks ganda. Aturan-aturan tersebut ditunjukkan pada tabel 2.2.

Tabel 2.2 Urutan prefiks ganda (Tala, 2003)

Prefiks 1	Prefiks 2
Meng	Per
Di	ber
Ter	
ke	

Derivasional = (prefiks +kata dasar) | (kata dasar |sufiks) |
(prefiks + kata dasar +sufiks) |
(prefiks 1 + prefiks 2 + kata dasar) |
(prefiks 1 + prefiks 2 +kata dasar +sufiks).

Contoh dari struktur kata derivasional tersebut adalah sebagai berikut :

mem + beri → memberi
ambil + kan → ambilkan
meng + ambil + kan → mengambilkan
mem + per + indah → memperindah
mem + per + main + kan → mempermainkan

Struktur lain yang mungkin terjadi adalah sufiks *infleksional* pada struktur *derivasional*, yang dinamakan *multiple* sufiks. Dapat disimpulkan secara umum struktur morfologi kata bahasa Indonesia adalah :

Sturtur morfologi = [prefiks 1] + [prefiks 2] + kata dasar + [sufiks] + [kata ganti] + [partikel].

Penjelasan lebih lanjut untuk penggolongan prefiks dan sufiks dijelaskan pada sub bab selanjutnya.

2.2.2 Proses *Stemming* Bahasa Indonesia

Daplam proses *stemming* bahasa Indonesia terdapat lima aturan (Tala, 2003). Aturan-aturan tersebut adalah :

a. Aturan yang menangani partikel infleksional

Tabel 2.3 Aturan partikel infleksional

Sufiks	Penganti	Kondisi Tambahan	Contoh
Kah	NULL	NULL	Diakah → dia
Lah	NULL	NULL	Adalah → ada
Tah	NULL	NULL	Apatah → apa
Pun	NULL	NULL	Bukupun → buku

b. Aturan yang menangani kata ganti infleksional

Tabel 2.4 Aturan kata ganti infleksional

Sufiks	Penganti	Kondisi Tambahan	Contoh
Ku	NULL	NULL	bukuku → buku
Mu	NULL	NULL	bukumu → buku
nya	NULL	NULL	bukunya → buku

c. Aturan yang menangani urutan prefiks derivasional pertama

Tabel 2.5 Aturan prefiks derivasional pertama

Prefiks	Penganti	Kondisi Tambahan	Contoh
Meng	NULL	NULL	Mengukur → ukur
Meny	S	V…*	Menyapu → sapu
Men	T	V…*	Menuduh→ tuduh
Men	NULL	NULL	Menduga → duga
Mem	P	V…*	Memukul → pukul
Mem	NULL	NULL	Membakar → bakar
Me	NULL	NULL	Merusak → rusak
Peng	NULL	NULL	Pengukur → ukur
Peny	S	V…*	Penyelam → selam
pen	T	V…*	Penari → tari

Pen	NULL	NULL	Penduga → duga
Pem	P	V…*	Pemandu → pandu
Pem	NULL	NULL	Pembaca → baca
Di	NULL	NULL	Diukur → ukur
Ter	NULL	NULL	Tersipu → sipu
ke	NULL	NULL	Kekasih → kasih

*kata dasar dimulai dari huruf vocal

d. Aturan yang menangani urutan prefiks derivisional kedua

Tabel 2.6 Aturan prefiks derivasional kedua

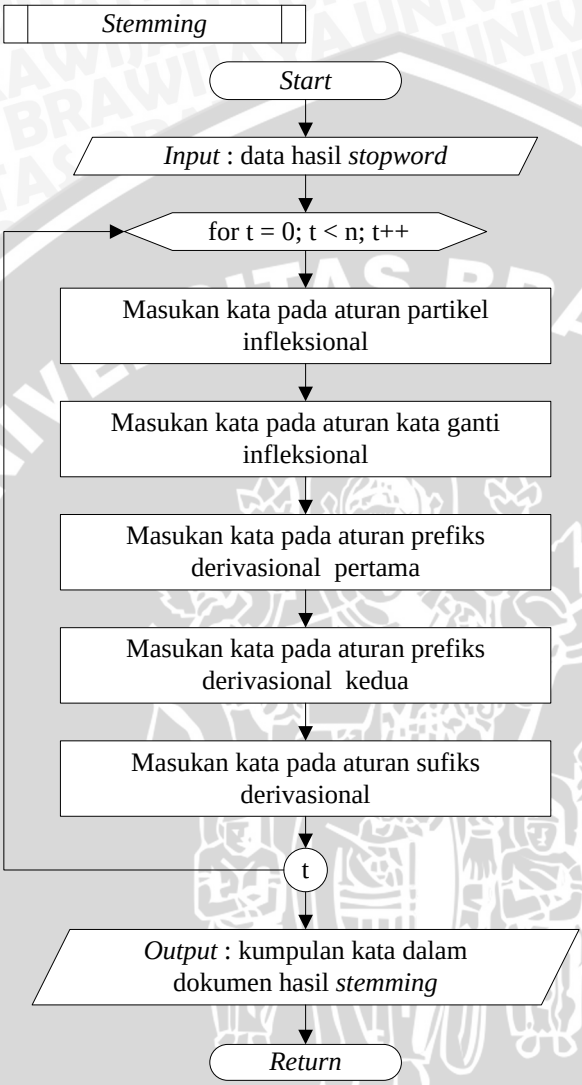
Prefiks	Penganti	Kondisi Tambahan	Contoh
Ber	NULL	NULL	berlari → lari
Bel	NULL	Ajar	belajar → ajar
Be	NULL	kerja	bekerja → kerja
Per	NULL	NULL	perjelas → jelas
Pel	NULL	ajar	Pelajar → ajar
Pe	NULL	NULL	Pekerja → kerja

e. Aturan yang menangani sufiks dervisional

Tabel 2.7 Aturan sufiks derivasional

Sufiks	Penganti	Kondisi Tambahan	Contoh
Kan	NULL	Prefiks ¢ {ke,peng}	Tarikkan → tarik (meng) ambilkan → ambil
An	NULL	Prefik ¢ {di, meng, ter}	makanan → makan (per)janjian → janji
i	NULL	Prefiks ¢ {ber,ke,peng}	Tanda → tanda (men)dapati → dapat

Berikut *flowchart stemming* untuk pengambaran dari setiap proses aturan *stemming* bahasa Indonesia.



Gambar 2.1 *Stemming* bahasa Indonesia

2.3 Teks Mining

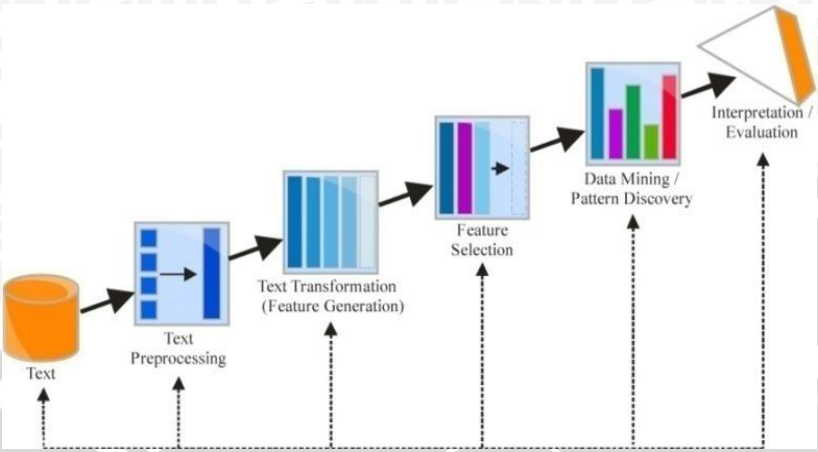
Teks mining merupakan salah satu bentuk eksplorasi dan analisis data teks yang bertujuan mendapatkan pengetahuan baru baik itu melalui cara otomatis maupun semi otomatis (Evan Yair Zohar, 2002). *Teks mining* dapat didefinisi menambang data yang berupa teks dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen (Raymond J. Mooney, 2006).

Dari dua definisi *teks mining* diatas dapat disimpulkan *teks mining* merupakan proses analisa teks untuk menemukan informasi yang mewakili suatu dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen lainya.

Penerapan *teks mining* banyak digunakan dan dikembangkan, salah satunya dalam pengklasifikasian berita. Aplikasi teks mining juga diaplikasikan dalam berbagai kasus seperti *clustering, web mining,* pengelompokan spam, pengklasifikasian emosi dan berbagai macam kasus dengan data teks. Dalam pengklasifikasian berita, dari sekumpulan berita akan dilakukan pencarian informasi yang menjadi ciri suatu berita tersebut kemudian dilakukan perbandingan dengan kumpulan beita lainya. Informasi dari suatu berita dapat berbentuk angka-angka probabilitas, vektor, set aturan atau bentuk lainya.

2.4 Tahap Teks Mining

Dari definisi diatas, dalam melakukan teks mining terdapat tahapan-tahapan proses untuk mendapatkan pengetahuan, tahapan-tahapan itu ada lima yaitu: *preprocessing*, *transformation, feature selection,* data mining dan *interpretation/evaluation*. Namun secara lengkap hanya tiga yang digunakan dalam teks mining yaitu *preprocessing, transformation* , dan *pattern discovery* atau *data mining* (Even Yahir Zohar, 2002). Penggambaran tahap teks mining menurut Evan Yahir Zohar dapat dilihat pada gambar 2.2 .



Gambar 2.2 Tahap dalam teks mining (Even, Yahir, Zohar, 2002)

2.4.1 Tahap *Preprocessing*

Pada tahap *preprocessing* dilakukan proses pengolahan teks menjadi data yang akan digunakan dalam proses selanjutnya. Tujuan dilakukannya *preprocessing* untuk melakukan seleksi informasi yang tidak diperlukan dalam proses selanjutnya seperti tag-tag HTML, kata umum yang biasanya muncul dalam jumlah besar tetapi tidak memiliki makna dan pembuangan imbuhan kata untuk diambil kata dasar dari setiap kata yang berimbuhan. *Preprocessing* penting dilakukan untuk meningkatkan ketepatan dalam proses klasifikasi. Pada *preprocessing* terdapat proses pengolahan teks seperti *tokenizing*, *stemming*, *stopwords* dan frekuensi kata.

2.4.1.1 *Preprocessing* Tahap *Tokenizing*

Tahap *tokenizing* adalah tahap pemotongan *string input* berdasarkan tiap kata yang menyusunnya (Raymond J. Mooney, 2006). Pada tahap *tokenizing* dilakukan dengan menghilangkahn tanda baca dan memisahkan per spasi.

2.4.1.2 *Preprocessing* Tahap *Stemming*

Tahap *stemming* adalah tahap mencari *root* kata dari tiap kata hasil *filtering* (Raymond J. Mooney, 2006). Pengertian *stemming* bisa juga diartikan sebagai proses mengembalikan kata menjadi kata dasarnya. Proses pencarian kata dasar dapat dilakukan dengan

membuang awalan kata dan akhiran kata yang terdapat dalam kata berimbuhan. Implementasi proses *stemming* beragam tergantung dengan bahasa dari dokumen. Dengan dilakukannya proses *stemming* setiap kata berimbuhan akan berubah menjadi kata dasar, dengan demikian dapat lebih mengoptimalkan proses teks mining.

2.4.1.3 *Preprocessing* Tahap *Stopword*

Stopword adalah kata umum (*common words*) yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna (Yudi Wibisono, 2005). Tahap *stopword* dilakukan dengan pembuangan kata-kata yang sering muncul seperti kata sambung, kata tanya dan kata-kata lain yang tidak memiliki makna. *Stopword* yang digunakan harus sesuai dengan dokumen yang akan diproses.

Dalam *teks mining* proses *stemming* dan *stopword* dapat dilakukan dalam proses berbeda atau dalam satu proses tergabung. Dilakukan proses penghilangan *stopword* terlebih dahulu kemudian dilanjutkan proses *stemming*. Dalam skripsi ini data berasal dari surat kabar berbahasa Indonesia maka *stopword* menyimpan kata-kata dalam bahasa Indonesia. Dalam proses *stemming* atau pencarian kata dasar dalam bahasa Indonesia dibahas pada sub bab 2.2.

2.4.1.4 *Preprocessing* Tahap Frekuensi Kata

Proses terakhir dari *preprocessing* adalah frekuensi kata. Dalam frekuensi kata dilakukan proses perhitungan kemunculan kata pada setiap dokumen. Kata-kata yang memiliki frekuensi tertentu menjadi ciri dari setiap dokumen. Oleh karena itu, hanya kata-kata yang muncul minimal 3 kali didalam keseluruhan dokumen akan digunakan sebagai ciri dari masing-masing dokumen (Joachims, 1998).

2.4.2 Tahap Perubahan Teks Dan Pemilihan Fitur

Fitur adalah suatu pola menarik dari dokumen yang dianggap bisa merepresentasikan suatu dokumen. Tidak semua fitur bermanfaat dalam proses selanjutnya. Tujuan dilakukannya proses ini untuk mengambil fitur - fitur yang bermanfaat. Tahap pemilihan fitur dapat dilakukan dengan bermacam-macam metode, bergantung pada kebutuhan. Pada skripsi ini tahap pemilihan fitur dengan mengunakan pembobotan TF-IDF *(term frequency-inverse document*

frequency). Frekuensi dari sebuah *term* dalam satu dokumen dapat merepresentasikan makna sebuah term pada suatu dokumen.

2.4.3 Tahap Penemuan Pola

Tahap penemuan pola atau *pattern discovery* adalah tahap terpenting dari seluruh proses *text mining*, tahap ini berusaha menemukan pola atau pengetahuan dari keseluruhan teks (Aziz Musthafa, 2009). *Feature* yang lolos pada tahap sebelumnya akan memasuki tahap *data mining* untuk menghasilkan pola menarik dari data.

Pada tahap *pattern discovery* terdapat dua teknik pembelajaran, *unsupervised* dan *supervised learning*. Perbedaan dari kedua pembelajaran tersebut, *supervised learning* terdapat label atau nama kelas pada data latih (*supervisi*) dan data dikelasifikasikn berdasarkan data latih. *Unsupervised learning* tidak terdapat label atau nama kelas pada data latih, data dikelompokkan berdasarkan ukuran kemiripan pada suatu kelas.

Berdasarkan keluaran dari fungsi, *supervised learning* dibagi menjadi 2, regresi dan klasifikasi. Regresi terjadi jika output dari fungsi merupakan nilai yang kontinyu sedangkan klasifikasi terjadi jika keluaran dari fungsi adalah nilai tertentu dari suatu atribut tujuan (tidak kontinyu). Tujuan dari *supervised learning* adalah untuk memprediksi nilai dari fungsi sebuah data masukan yang sah setelah melihat sejumlah data latih. (Luz, 2006).

Berikut adalah tahapan umum yang biasanya dilakukan pada *supervised learning* :

1. Menentukan tipe dari data latih.
2. Mengumpulkan data latih. Data latih yang digunakan seharusnya memiliki karakterisktik dunia nyata. Karena itu data latih dapat berasal baik dari hasil pengukuran atau dari pakar.
3. Menentukan representasi fitur masukan dari fungsi yang ingin dibentuk karena tingkat akurasi dari fungsi dapat dipengaruhi oleh representasi dari masukan.
4. Menentukan struktur dari pengetahuan (fungsi) dan algoritma yang akan digunakan.
5. Jalankan algoritma terhadap data latih.

Terdapat banyak teknik *supervised learning* yang telah dikembangkan oleh para ahli namun sesuai dengan cakupan dari skripsi ini, maka pada bahasan selanjutnya hanya akan dibahas *Iterative Dichotomizer Tree*.

2.5 *Term Frequency - Inverse Document Frequency*

Setelah melalui *preprocessing* hasil data berbentuk *token* yang terpisah dari kata yang lain dan sudah dalam bentuk dasar. Pada langkah selanjutnya *term* akan dirubah kedalam bentuk numerik untuk diketahui bobot setiap kata dari satu dokumen ke dokumen lainya. *Term Frequency* sering di sebut *TF*, merupakan banyak kata yang keluar dari satu dokumen. *Invers Dokumen Frequency* yang sering dikenal dengan *IDF*, merupakan pembobotan kemunculan jumlah kata pada suatu dokumen. Metode *TF-IDF* merupakan metode pembobotan dalam bentuk sebuah metode yang merupakan *integrasi* antar *term frequency (tf)*, dan *inverse document frequency (idf)* (Yiming Y dan Liu, 1999). Berikut rumus untuk mencari bobot kata dengan metode *TF-IDF*.

$$w_{i\,j} = tf_{ij}.idf \qquad (2.1)$$

$$idf = log\frac{N}{df_j} \qquad (2.2)$$

Keterangan :

- W_{ij} = bobot kata *i* pada dokumen *j*
- N = jumlah koleksi dokumen
- tf_{ij} = jumlah kehadiran kata *i* yang akan dihitung bobotnya dalam dokumen *j*
- df_j = dokumen *j* mengandung kata yang akan dihitung bobotnya
- *idf = inverse document frecuency* didapat dari hasil *log.N/df*

2.6 Transformasi Data

Proses data mining membutuhkan integrasi data dari berbagai data penyimpanan. Transformasi data dilakukan sebelum proses pengenalan pola. Dilakukan data transformasi ke dalam bentuk yang sesuai untuk pengenalan pola (Jiawei H dan Michelin Kamber, 2006). Menurut Jiawei dan Kamber data transformasi dapat dilakukan dengan cara berikut :

a) ***Smoothing***, yang bekerja untuk menghilangkahn *noise* dari data. Teknik-teknik tersebut meliputi *binning*, *regresi*, dan *clustering*.
b) ***Aggregation***, digunakannya ringkasan atau agregasi operasi diterapkan pada data. Misalkan untuk penjumlahn data harian dapat dikumpulkan sehingga untuk menghitung total bulanan dan tahunan. Langkah ini digunakan dalam membangun sebuah data kubus untuk analisa data pada beberapa *multiple granularities.*
c) ***Generalization,*** dari data tingkat rendah atau primitif, data diganti dengan *higher-level concepts* dengan menggunakan *concept hirerarchies*. Sebagai contoh atribut kategori, seperti jalan dapat digeneralisasi untuk tingkat yang lebih tinggi dengan menggunakan nama kota atau negara. Demikian juga nilai untuk atribut numerik, seperti usia dapat dipatenkan ketingkat lebih tinggi, seperti pemuda, setengah baya dan senior.
d) ***Normalization,*** dimana data atribut ditingkatkan sehingga jatuh dalam jangkau terkecil, seperti -1.0 menjadi 1.0 atau untuk 0.0 menjadi 1.0.
e) ***Attribute construction,*** (konstruksi fitur), dimana atribut dibangun dan menambahkan dari himpunan atribut untuk membantu proses data mining.

2.6.1 Distribusi Frekuensi

Dalam kasus skripsi ini, data numerik yang dihasilkan tidak dapat begitu saja dikelompokan dalam kategori tertentu. Metode distribusi frekuensi dapat digunakan untuk membantu dalam pembuatan kelas dan interval suatu kategori. Data yang didapat dari dataset bertipe numerik. Algoritma *Iterative Dichotomizer Tree* memerlukan data tipe kategori. Teknik yang digunakan untuk mengubah data numerik menjadi data kategori adalah teknik distribusi frekuensi (Sofi Defiyanti, 2010).

Berikut beberapa langkah dalam menentukan interval kelas pada metode distribusi frekuensi (J. Supranto, 2000).

1. Urutkan data, untuk mencari nilai terbesar dan terkecil dari data. Gunakan persamaan berikut untuk mencari range.

$$R = X_{max} - X_{min} \qquad (2.3)$$

R = Rentang antara data terbesar dengan data terkecil.
X_{max} = Nilai data terbesar dari kumpulan data.
X_{min} = Nilai data terkecil dari kumpulan data.

2. Menentukan banyak kelas yang akan digunakan dengan menggunakan rumus Sturges.

$$K = 1 + 3{,}3 \log N \qquad (2.4)$$

K = Banyak kelas.
N = Jumlah data observasi.

3. Menetukan panjang interval setiap kelas, dengan menggunakan persamaan berikut.

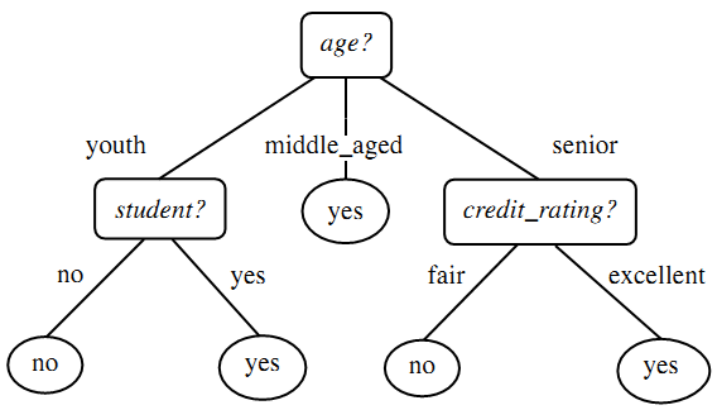
$$Interval = \frac{R}{K} \qquad (2.5)$$

$Interval$ = Panjang range data dalam tiap kelas.
R = Rentang antara data terbesar dan terkecil.
K = Banyak kelas.

4. Pilih ujung bawah kelas interval pertama, dapat menggunakan data terkecil atau nilai data yang lebih kecil dari data.
5. Membuat daftar distribusi untuk mengetahui frekuensi masing-masing kelas.

2.7 *Decision Tree*

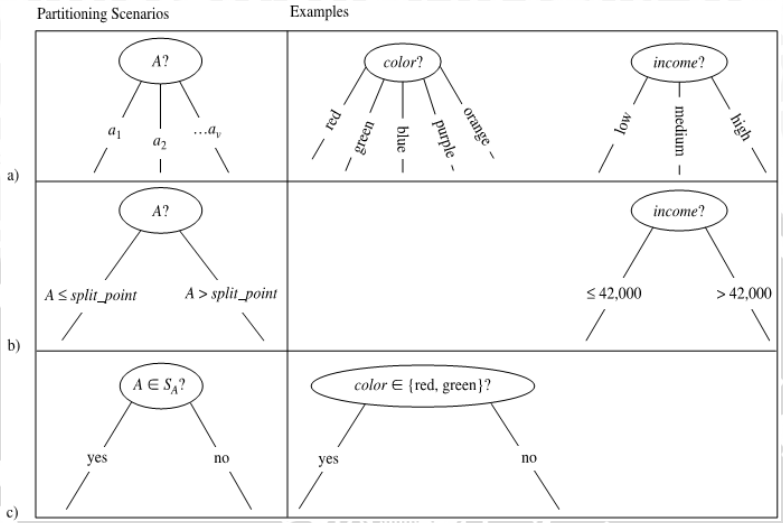
Penggunaan pohon keputusan dimulai pada tahun 1970-an dan awal 1980-an, J. Ross Quinlan, melakukan penelitian dan pengembangan algoritma *Iterative Dichotomizer Tree* (ID3). Algoritma ID3 diperluas untuk konsep sistem pembelajaran yang dijelaskan oleh EB Hunt, J. Marin, dan P.T. Stone. Quinlan kemudian mepresentasikan C4.5 yang menjadi penerus ID3. Pada tahun 1984, sekelompok ahli statistik L. Breiman, J. Friendman, R. Olshen dan C. Stone mempublikasikan buku pengklasifikasian dan regresi tree (CART), yang menggambarkan pohon keputusan biner. ID3 dan CART dikembangkan secara bebas pada waktu yang sama, namun mengikuti pendekatan yang sama untuk pembelajaran *decision tree* dari pelatihan *tuple*.



Gambar 2.3 *Decision tree* untuk pembelian komputer (Jiawei H dan Michelin Kamber, 2006)

Decision tree adalah pembelajaran pohon keputusan dari *class* yang berlebel. *Decision tree* adalah struktur pohon seperti *flowchart*, dimana setiap *node internal (node non leaf*) menunjukkan tes pada atribut, setiap cabang berasal dari hasil pengujian, dan masing-masing daun (*node terminal*) memiliki label kelas. *Node* paling atas adalah *root.* Pada contoh *decision tree* digambar 2.3, pohon keputusan untuk pembelian komputer pada AllElectronics. Pada gambar tersebut menjelaskan apakah pelanggan AllElectronics memungkinkan untuk membeli komputer di tokoh tersebut. Setiap *node internal* (*non leaf*) merupakan tes dari atribut. Setiap simpul daun mewakili sebuah kelas, baik kelas berlabel Yes atau No.

Secara singkat bahwa *Decision Tree* merupakan salah satu metode klasifikasi pada *text mining*. Klasifikasi adalah proses menemukan kumpulan pola atau fungsi-fungsi yang mendeskripsikan dan memisahkan kelas data satu dengan lainnya, untuk dapat digunakan memprediksi data yang belum memiliki kelas data tertentu (Jianwei Han, 2001).



Gambar 2.4 *Decision tree* yang memungkinkan dalam proses partisi data. (Jiawei H dan Michelin Kamber, 2006).

Tiga kemungkinan untuk partisi data dalam decision tree berdasarkan kriteria pemisahnya. Ditunjukkan pada gambar 2.4, dengan contoh A sebagai atribut pemisah.

a) Jika A bernilai diskrit, salah satu cabang tumbuh untuk setiap nilai A. Penggunaan partisi seperti ini digunakan dalam pemodelan ID3, dijelaskan lebih lanjut pada sub bab berikutnya.
b) Jika A adalah *continuous_valued,* lalu dua cabang berkembang, sesuai denagn A $\leq$ *splite_point* dan A > *split_poin*. Penggunaan partisi dengan data split digunakan dalam pemodelan C4.5 dengan menggunakan gain ratio.
c) Jika A adalah nilai diskrit dan pohon biner harus diproduksi, maka tes bentuk $A \in S_A$, dimana S_A adalah pemisah subset untuk A. Penggunaan partisi seperti ini digunakan dalam pemodelan CART.

2.9 *Iterative Decotomister Tree*

Iterative Dichotomizer Tree adalah algoritma *decision tree learning* (algoritma pembelajaran pohon keputusan) yang paling dasar. Algoritma ini melakukan pencarian secara *greedy* pada semua kemungkinan pohon keputusan (Wahyudi, 2009).

ID3 menggunakan *informasi gain* sebagai ukuran seleksi atribut. Ukuran ini didasarkan pada penyelesaian pionering oleh *Claude Shannon* pada teori informasi, yang mempelajari nilai atau "informasi isi" dari pesan. Membiarkan node N mewakili atau terus mempartisi tupel D. Atribut dengan informasi gain tertinggi dipilih sebagai informasi pemisah node N. Atribut ini meminimalkan informasi yang dibutuhkan untuk mengklasifikasikan tupel dalam menghasilkan partisi dan menggambarkan keacakan atau ketidakmurnian dalam partisi ini. Pendekatan seperti yang diharapkan meminimalkan jumlah tes yang dibutuhkan untuk mengklasifikasikan sebuah tuple yang diberikan dan menjamin kesederhanaan (tetapi belum tentu sederhana) tree yang ditemukan. Informasi yang diharapkan untuk mengklasifikasikan sebuah tuple dalam D diberikan oleh (Jiawei H dan Michelin Kamber, 2006) :

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i), \qquad (2.6)$$

Keterangan :

- *Info (D)* = *Entropy* atribut untuk membandingkan data sample.
- *pi* = Data sample ke *i* pada suatu atribut.

Dari hasil representasi *entropy* atribut yang di tunjuk sebagai *class*, dilakukan perhitungan seluruh *entropy* atribut untuk setiap data sampel. Data sample data atribut akan dibandingkan dengan data sampel dari atribut *class.* Dari hasil peritungan *entropy*, dilakukan perbandingan dengan *entropy class* dengan rumus *information gain* sbagai berikut :

$$Gain(A) = Info(D) - Info_A(D). \qquad (2.7)$$

Keterangan :

- *Gain (A)* = *Information gain* dari perbandingn *entropy class* dengan *entropy* setiap sampel atribut.
- *Info(D)* = perhitugan *entropy class.*
- $Info_A(D)$ = perhitungan *entropy* dari atribut.

Hasil dari *information gain* pada satu atribut dibandingkan dengan atribut lainya, nilai informasi gain terbesar akan dijadikan *root*. Data sampel dari root utama menjadi cabang dari tree. Dilakukan rekursi setiap cabang, memilih atribut terbaik dan dijadikan sebagai node cabang tersebut. Dilakukan rekursi sampai pada attribute terakhir untuk setiap cabang.

Berikut contoh penerapan ID3 dalam penentuan keputusan untuk permainan tenis. Beberapa faktor yang mempengaruhi keputusan dalam permainan tenis dijadikan atribut seperti outlook, temperature, humidity dan wind. Berikut tabel keputusan dalam penentuan permainan tenis.

Tabel 2.8 Tabel penentuan dalam bermain tenis

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Pada tabel 2.8 *Play Tennis* merupakan atribut tujuan dalam menentukan jadi tidaknya bermain tenis. Pada atribut target memiliki dua *value Yes* dan *No*. Atribut tujuan atau *entropy(S)* adalah koleksi dari 14 contoh data dengan 9 contoh positif "Yes" dan 5 contoh negatig "No", dituliskan dengan notasi [9+,5-]. Dilakukan perhitungan *entropy* pada atribut target dengan persamaan 2.6 berikut perhitungannya.

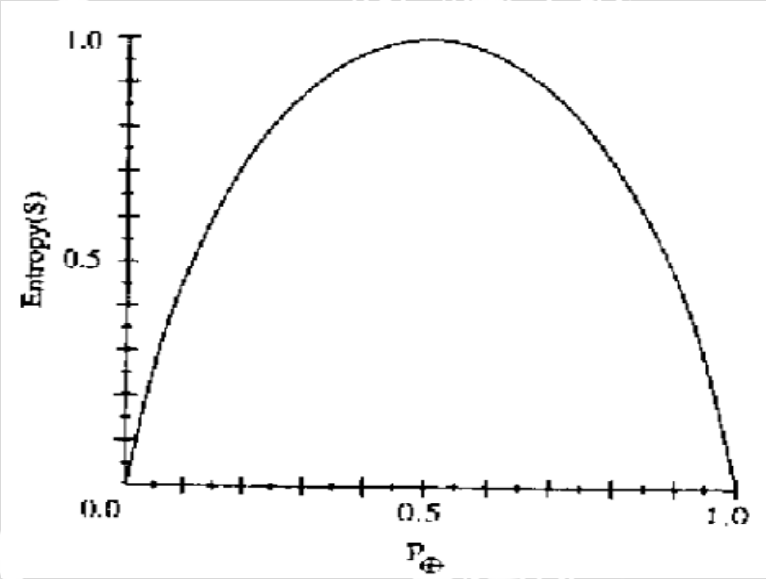
Entropy dari S adalah:

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

$$\text{Entropy}([9+,5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94029$$

Catatan :

- Entropy(S) = 0, jika semua contoh pada S berada dalam kelas yang sama.
- Entropy(S) = 1, jika jumlah contoh positif dan jumlah contoh negatif dalam S adalah sama.
- 0 < Entropy(S) < 1, jika jumlah contoh positif dan negative dalam S tidak sama.
-



Gambar 2.5 Grafik Perbandingan nilai *entropy*

Setelah didapatkan nilai *entropy* atribut tujuan, dilakukan perhitungan informasi gain dengan persamaan 2.7. Berikut perhitungan *information gain.*

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \quad (2.8)$$

Values(Wind) = Weak, Strong
S_{Weak} = [6+,2-]
S_{Strong} = [3+,3-]
Gain(S,Wind) = **Entropy(S) - (8/14)Entropy(S_{Weak}) - 6/14)Entropy(S_{Strong})**
= 0.94029 - (8/14)0.81128 - (6/14)1.0000
= 0.04813

Values(Humidity) = High, Normal
S_{High} = [3+,4-]
S_{Normal} = [6+,1-]
Gain(S,Humidity) = Entropy(S) - (7/14)Entropy(S_{High}) - (7/14)Entropy(S_{Normal})
= 0.94029 - (7/14)0.98523 - (7/14)0.59167
= 0.15184

Values(Temperature) = Hot, Mild, Cool
S_{Hot} = [2+,2-]
S_{Mild} = [4+,2-]
S_{Cool} = [3+,1-]
Gain(S,Temperature) = Entropy(S) - (4/14)Entropy(S_{Hot}) - (6/14)Entropy(S_{Mild}) - (4/14)Entropy(S_{Cool})
= **0.94029 - (4/14)1.00000 - (6/14)0.91830 - (4/14)0.81128**
= 0.02922

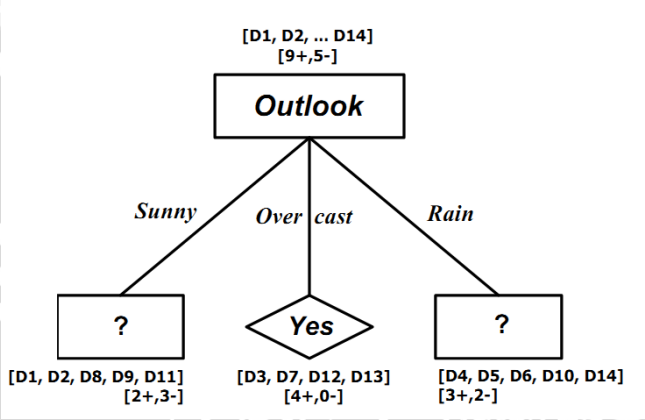
Values(Outlook) = Sunny, Overcast, Rain
S_{Sunny} = [2+,3-]
$S_{Overcast}$ = [4+,0-]
S_{Rain} = [3+,2-]
Gain(S,Outlook) = Entropy(S) - (5/14)Entropy(S_{Sunny}) - (4/14)Entropy($S_{Overcast}$) - (5/14)Entropy(S_{Rain})
= **0.94029 - (5/14)0.97075 - (4/14)1.000000 - (5/14)0.97075**
= **0.24675**

Information gain yang dihasilkan dari perhitungan :

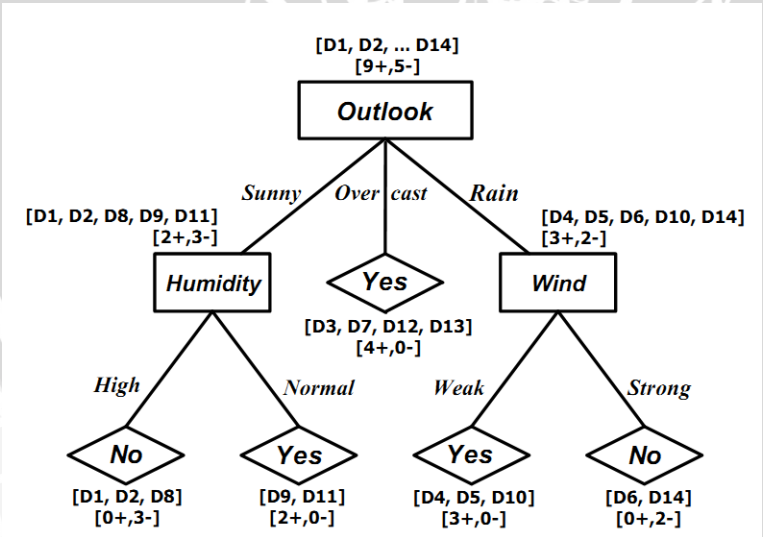
Gain(S,Wind) = 0.04813
Gain(S,Humidity) = 0.15184
Gain(S,Temperature) = 0.02922
Gain(S,Outlook) = **0.24675**

Dari hasil perhitungan *information gain* diperoleh data masing-masing *information gain* dari atribut. Pada data yang dihasilkan, atribut *Outlook* memiliki nilai *information gain* tertinggi, *outlook* digunakan sebagai *root*. *Decision tree* yang dihasilkan terdapat pada

gambar 2.6. Dari hasil *tree* akan dikelompokan berdasarkan *record* data yang dimiliki atribut *outlook* yaitu *sunny*, *over cast* dan *rain*. Dilakukan pengecekan setiap cabang, jika atribut target terdapat dua jenis data *record* dilakukan perhitungan sama seperti pada perhitungan pencarian *root*. Pada contoh ini cabang *sunny* dan *rain* masi memiliki atribut target lebih dari satu. Hasil dari *tree* akhir pada gambar 2.7.



Gambar 2.6 *Tree* hasil perhitungan *root*.



Gambar 2.7 *Tree* akhir dari penentuan permainan tenis.

2.10 Evaluasi

Pengukuran validitas pada penelitian pengklasifikasian berita ini menggunakan *F-Measure* karena sudah ada data pembanding eksternal yang dapat digunakan dalam evaluasi. Untuk pengujian dengan *F-Measure*, digunakan suatu standar yang disebut matriks *confusion*. Matriks *confusion* berisi informasi mengenai pengklasifikasian yang sebenarnya dan prediksi pengklasifikasian yang dilakukan oleh sistem (Hamilton, 2003). Tabel 2.9 menunjukkan matriks *confusion* (Lewis, 1995).

Tabel 2.9 Matriks *Confusion*

		Klasifikasi Sebenarnya	
		Kelas Sama	**Kelas Beda**
Klasifikasi Sistem	**Kelas Sama**	*True Positive*	*False Negative*
	Kelas Beda	*False Positive*	*True Negative*

1. *True Positive (TP)* menunjukkan bahwa *record* yang termasuk dalam hasil pengklasifikasian oleh sistem memang merupakan anggota klaster. Pengklasifikasian sebenarnya ya dan pengklasifikasian oleh sistem ya.
2. *False Negative (FN)* menunjukkan bahwa *record* yang termasuk dalam hasil pengklasifikasian oleh sistem ternyata seharusnya bukan merupakan anggota klaster. Pengklasifikasian sebenarnya bukan dan pengklasifikasian oleh sistem ya.
3. *False Positive (FP)* menunjukkan bahwa *record* yang tidak termasuk dalam hasil pengklasifikasian oleh sistem ternyata seharusnya merupakan anggota klaster. Pengklasifikasian sebenarnya ya dan pengklasifikasian oleh sistem bukan.
4. *True Negative (TN)* menunjukkan bahwa *record* yang tidak termasuk dalam hasil pengklasifikasian oleh sistem ternyata seharusnya bukan merupakan anggota klaster.

2.10.1 *Recall*

Ada beberapa standar pengukuran yang digunakan dalam pengklasifikasian, diantaranya adalah *recall*, *precision* dan *F-measure*. *Recall* adalah tingkat keberhasilan mengenali suatu *event* dari seluruh *event* yang seharusnya dikenali. Dari pengertian tersebut, *recall* berarti ukuran dari jumlah *record* benar yang berhasil dikelompokan oleh sistem dari keseluruhan *record* yang seharusnya benar (Tala, 2003). Cara mencari nilai *recall* ini adalah dengan membagi jumlah *record* hasil pengklasifikasian sistem yang benar dengan jumlah *record* yang sebenarnya dalam klaster tersebut.

$$recall = \frac{TP}{TP + FP} \quad (2.9)$$

Keterangan :

Recall = tingkat keberhasilan mengenali suatu *event* dari seluruh *event* yang seharusnya dikenali.

TP = True positif

FP = False Positif

2.10.2 Precision

Sedangkan *precision* adalah tingkat ketepatan hasil pengelompokan terhadap suatu *event*. Artinya, dari seluruh *record* hasil pengklasifikasian, berapa persenkah yang dinyatakan benar (Tala, 2003).

$$precision = \frac{TP}{TP + FN} \quad (2.10)$$

Keterangan :

precision = tingkat ketepatan hasil pengelompokan terhadap suatu *event*.

TP = True positif

FN = False Negatif

2.10.3 *F- Measure*

F-measure merupakan gabungan antara *recall* dan *precision* yang didefinisikan dengan persamaan 2.5 (Yang dan Liu, 1999):

$$F - Measure = \frac{2 \text{ x } recall \text{ x } precision}{recall + precision} \quad (2.11)$$

Keterangan :

precision = tingkat ketepatan hasil pengelompokan terhadap suatu *event*.

Recall = tingkat keberhasilan mengenali suatu *event* dari seluruh *event* yang seharusnya dikenali.


UNIVERSITAS BRAWIJAYA

BAB III
METODOLOGI DAN PERANCANGAN

Dalam BAB III ini berisi methodologi penelitian yang dilakukan serta perancangan dari sistem yang akan dibuat. Pada penelitian ini, dibagi menjadi beberapa langkah.

1. Melakukan pencarian dan mempelajari berbagai literatur yang dibutuhkan dalam permasalahan klasifikasi berita berbahasa Indonesia menggunakan metode *Iterative Dichotomizer Tree* (ID3) berasal dari buku dan beberapa dari internet.
2. Persiapan data berita yang dibagi dalam dua bagian yaitu data latih dan data *testing*.
3. Mengimplementasikan metode klasifikasi berita dengan *Iterative Dichotomizer Tree* menggunakan data latih yang telah disiapkan sebagai proses pembelajaran, hasil pembelajaran akan digunakan sebagai pengenalan ke sebuah sistem perangkat lunak.
4. Melakukan uji coba data *testing* terhadap hasil klasifikasi yang dilakukan oleh sistem. Hasil klasifikasi merupakan informasi pengklasifikasian dari berita yang diujikan (data *testing*).
5. Melakukan evaluasi dan menganalisis hasil klasifikasi yang dihasilkan oleh sistem berupa tingkat akurasi.

Alur penelitian skripsi ini dapat dilihat pada gamabar diargam 3.1.

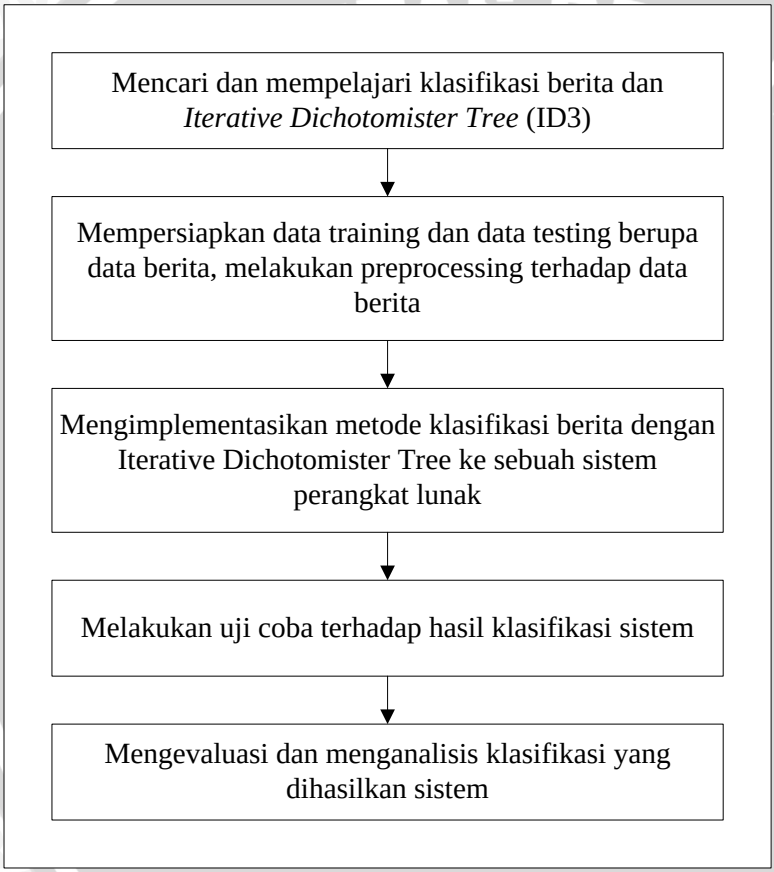
3.1 Analisa Data

Analisa data berisi pembahasan sumber data berupa dokumen berita yang digunakan untuk penelitian. Pembagian data untuk dokumen latih dan *testing* berdasarkan perbandingan dalam pengujian data.

Dokumen berita dalam peneliti ini bersumber dari situs vivanews, vivanews merupakan salah satu media surat kabar berbasis web. Dokumen berita yang digunakan adalah dokumen berita bahasa Indonesia dan periode Juli 2011.

Pada situs vivanews telah memberika kategori pada berita yang disajikan, dokumen berita pada vivanews ini nanti akan menjadi acuan untuk dokumen pembelajaran sistem. Contoh kategori yang

telah disediakan oleh vivanews adalah politik, bisnis, nasional, metro, dunia, saintek, sport, bola, otomotif, showbiz, kosmo, sorot, analisis, makanan, fokus dan forum. Untuk penelitian ini digunakan empat katgori berita, kategori berita yang digunakan adalah politik, bisnis, metropolis dan olahraga. Diambilnya keempat kategori ini untuk memperoleh data latih yang tepat, memaksimalkan proses pembelajaran, dan meningkatkan nilai akurasi pada hasil pengujian dokumen.



Gambar 3.1 Alur penelitian

Dokumen berita dari situs vivanews dibagi menjadi dua bagian, bagian pertama digunakan sebagai dokumen berita pembelajaran yang selanjutnya disebut data latih dan bagian kedua digunakan dokumen berita pengujian yang selajutnya disebut data *testing*. Dalam pengujian data dilakukan perbandingan data latih dan data *testing* untuk mendapatkan nilai keakurasian yang maksimal. Pembahasan untuk perbandingan data akan dijelaskan pada sub bab pengujian data.

3.2 Deskripsi Umum Sistem

Secara keseluruhan penelitian ini dibagi dalam 3 tahap, yaitu *preprocessing*, pembelajaran dan klasifikasi. Pada gambar 3.2 menujukkan tahap dalam penelitian.

Tahap pertama adalah *preprocessing*, semua data input berupa dokumen berita data latih dan *testing* akan melalui proses *preprocessing*. Pada proses *preprocessing* data akan melalu beberapa proses diantaranya pengambilan isi berita dari file **.html*, pembersihan *tag* HTML, *tokenizing, stopword, stemming* dan *frekuensi* kata. Detail setiap proses pada *preprocessing* akan dijelaskan di sub bab 3.3 perancangan proses.

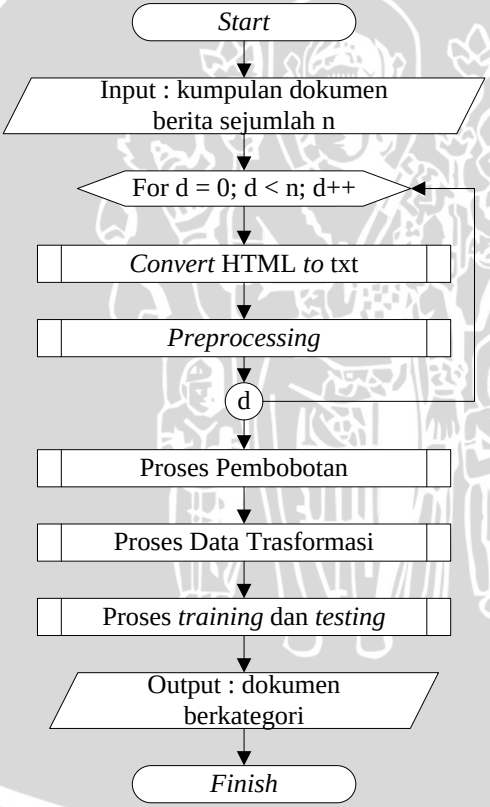
Tahap kedua adalah tahap pembelajaran atau latih. Setelah melalui tahap *preprocessing*, data yang telah disiapkan akan melalui proses pembelajaran untuk membentuk *decision tree* dengan algoritma *Iterative Dichotomizer Tree*.

Tahap ketiga adalah tahap klasifikasi. Setelah sistem melalui proses pembelajaran dari data latih, akan terbentuk suatu *rule* berbentuk *decision tree*. Dari *rule* tersebut, data *testing* yang disiapkan akan melaui proses pengklasifikasian.

Tahapan dalam deskripsi umum sistem, berdasarkan *flowchart* pada gambar 3.2 adalah :

1. *User* memasukan data berita dengan tipe **.html*, data latih dan data *testing* dimasukkan secara bersamaan pada tahap ini.
2. Dilakukan pemrosesan satu persatu terhadap dokumen berita yang sudah dimasukkan. Kedua data akan melalui proses yang sama, proses pengambilan teks berita dan *preprocessing*.
3. Dokumen berita yang berupa file **.html* akan melalui proses *html cleaner*, pada tahap ini dilakukan pengambilan *body* berita dan pemebrsian *tag html*.

4. Setelah isi berita berupa teks diambil, akan melalui tahap *preprocessing*. Pada tahap ini akan dilakukan *tokenizing*, *stemming* dan *stopword*.
5. Dari hasil *preprocessing*, setiap kata akan dihitung bobotnya dengan metode perhitungan bobot TF-IDF.
6. Data hasil perhitungan TF-IDF dilakukan proses trasnformasi data untuk mendapatkan data berupa kategori.
7. Dalam proses ini terbagi menjadi dua, bila data latih akan dilakukan proses pembelajaran. Bila data tesing akan dilakukan proses pengklasifikasian berdasarkan hasil pengetahuan dari proses pembelajaran.
8. Hasil berupa data *testing* yang sudah memiliki kategori yang sudah ditentukan.



Gambar 3.2 *Flowchart* deskripsi umum sistem

3.3 Prancangan Proses

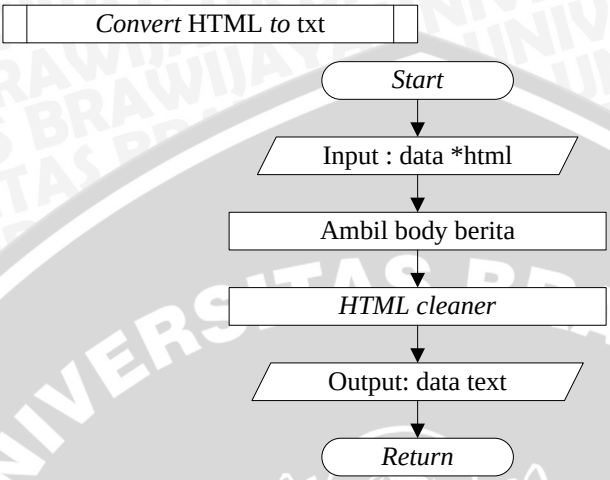
Pada sub bab perancangan proses dijelaskan mengenai proses-proses dalam membangun sistem. Seluruh dokumen berita akan melalui proses ini baik data latih atau data uji. Pada sub bab 3.3 perancangan proses akan menjelaskan proses pengambilan teks berita dari file *html*, *preprocessing*, pembobotan, *generalization*, pembelajaran dan klasifikasi.

3.3.1 Proses Pengambilan Teks Berita Dari File HTML

Didalam sub bab 3.3.1 Proses pengambilan teks berita, dijelaskan proses pengambilan teks berita dari file **.html*, tidak semua isi dari file **.html* melalui proses *html cleaner*. Pada web berita vivanews letak isi berita berada antara <*div class*=”isiberita”> dan <*div id*=”bodykanan”>. Perancangan proses pengambilan teks berita dari file *html* dapat dilihat pada gambar 3.3 *flowchart* proses pengambilan *body* berita dan *html cleaner*.

Tahapan proses pengambilan teks berita sesuai pada *flow chart* 3.3.

1. *User* memasukan file berita bertipe *html.*
2. Mengambil teks berita dari file *html*, teks berada pada <*div class*="isiberita"> sampai <*div id*="bodykanan">.
3. Pembersihan *tag html* dilakukan setelah proses kedua selesai.
4. Hasil dari proses *html cleaner* berupa teks tanpa *tag html.*



Gambar 3.3 *Flowchart* pembersihan *tag html*.

3.3.2 Perancangan *Preprocessing*

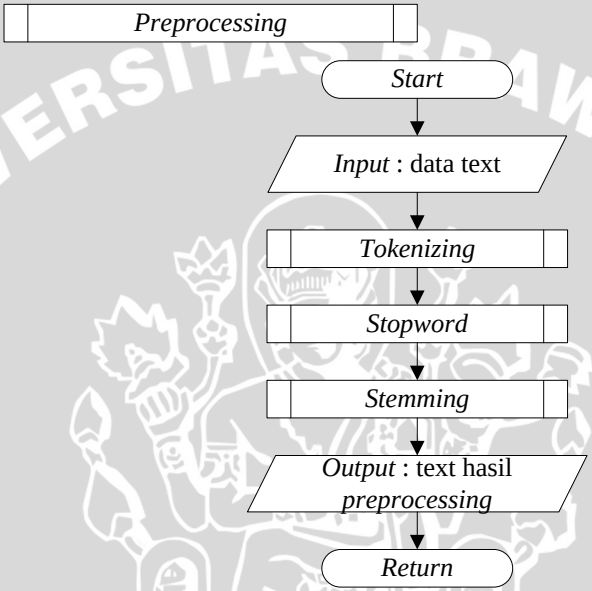
Pada sub bab 3.32 menjelaskan perancangan tahap *preprocessing*, pada tahapan ini dilakukan proses pemecahan dokumen string panjang menjadi string token berdasar beberapa variabel pemisahnya (*tokenizing*), penghapusan kata sering muncul namun kurang bermakna (*stopword*) dan pencarian kata dasar (*stemming*). Proses *preprocessing* digambarkan pada *flowchart* 3.4.

Tahap *prerocessing* terbagi dalam tiga sub proses, berikut alur pemprosesan teks pada gambar 3.4 *flowchart* perancangan *preprocessing*.

1. Data teks hasil pengambilan *body* berita dan *html cleaner* akan menjadi *input* dalam proses *preprocessing*.
2. Data teks pertama kali akan dilakukan *tokenizing*, pemecahan partikel, penghapusan tanda baca dan merubah alphabet menjadi huruf kecil. Hasil dari proses *tokenizing* berupa string dalam bentuk token.
3. Hasil *tokenizing*, selanjutnya melalui proses *stopword* penghapusan kata-kata yang kurang memiliki makna. Hasil dari proses ini berupa kumpulan string yang tidak ada pada daftar *stopword*.

4. Hasil dari *stopword* akan dilakukan pencarian kata dasar dari sejumlah kata berimbuhan. Hasil dari proses *stemming* berupa kumpulan string dalam bentuk kata dasar.
5. Hasil dari *preprocessing* disimpan pada string *list* .

Detail proses dari setiap proses pada tahap *preprocessing* akan di jelaskan pada sub bab berikutnya.



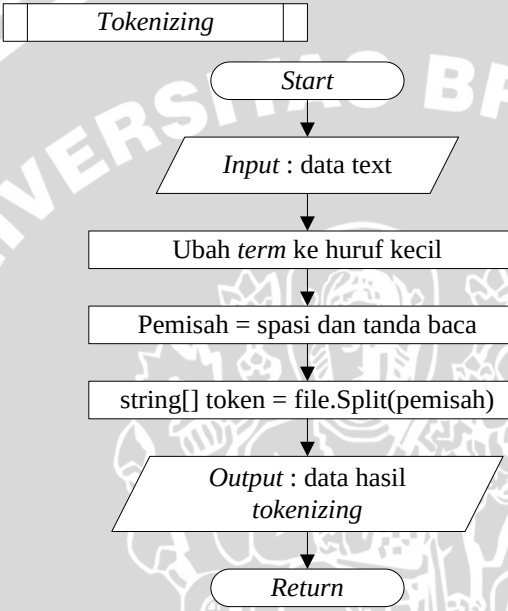
Gambar 3.4 *Flowchart* perancangan *preprocessing*

3.3.2.1 Perancangan Proses *Tokenizing*

Pada tahap *tokenizing* terdapat dua sub proses, *case folding* dan *parsing*. Perancangan proses *tokenizing* digambarkan pada gambar 3.5 *flowchart* proses *tokenizing*. Tahap *tokenizing* terbagi dalam dua tahap, *case folding* dan *parsing*. Berikut penjelasan untuk proses *tokenizing* pada gambar 3.5.

1. Data hasil pembersihan *html* akan di proses dalam proses *tokenizing*.
2. Data teks akan dirubah kebentuk huruf kecil.

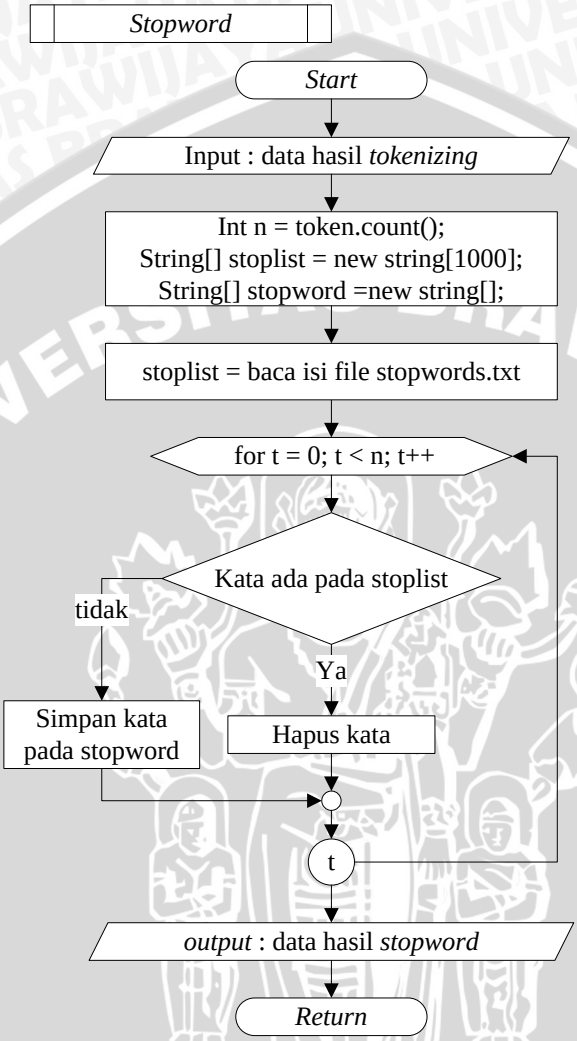
3. Variabel pemisah mengunakan semua tanda baca, sepasi dan karakter bukan alfabet.
4. Data teks bertipe string akan dipisah sesuai dengan *variable* pemisah.
5. Hasil data pemisahan berupa data teks yang terpisah.



Gambar 3.5 *Flowchart* proses *tokenizing*

3.3.2.2 Perancangan Proses *Stopword*

Proses *stopword* adalah proses penghapusan kata-kata yang sering muncul tapi tidak memberikan informasi. Hasil proses *stopword* adalah kata-kata yang merepresntasikan informasi dari suatu dokumen berita. Data *stopword* diambil dari http://fpmipa.upi.edu/staff/yudi/stop_words_list.txt. Sistem akan membaca dokumen teks berisi data *stopword*, kemudian disimpan pada *array string* stoplist. Dilakukan pengecekan setiap kata pada dokumen berita, jika kata terdapat pada *stopword* dilakukan penghapusan. Gammbar perancangan proses *stopword* terdapat pada gambar 3.6 *flowchart* proses *stopword*.



Gambar 3.6 *Flowchart* proses *stopword*

Tahap *stopword*, dilakukan pengecekkan pada *list stopword* yang sudah disiapkan. Berikut deskripsi dari proses *stopword* pada gambar 3.6 *flowchart* proses *stopword*.

1. Data berupa kumpulan kata hasil *tokenizing* dimasuk dalam sistem.

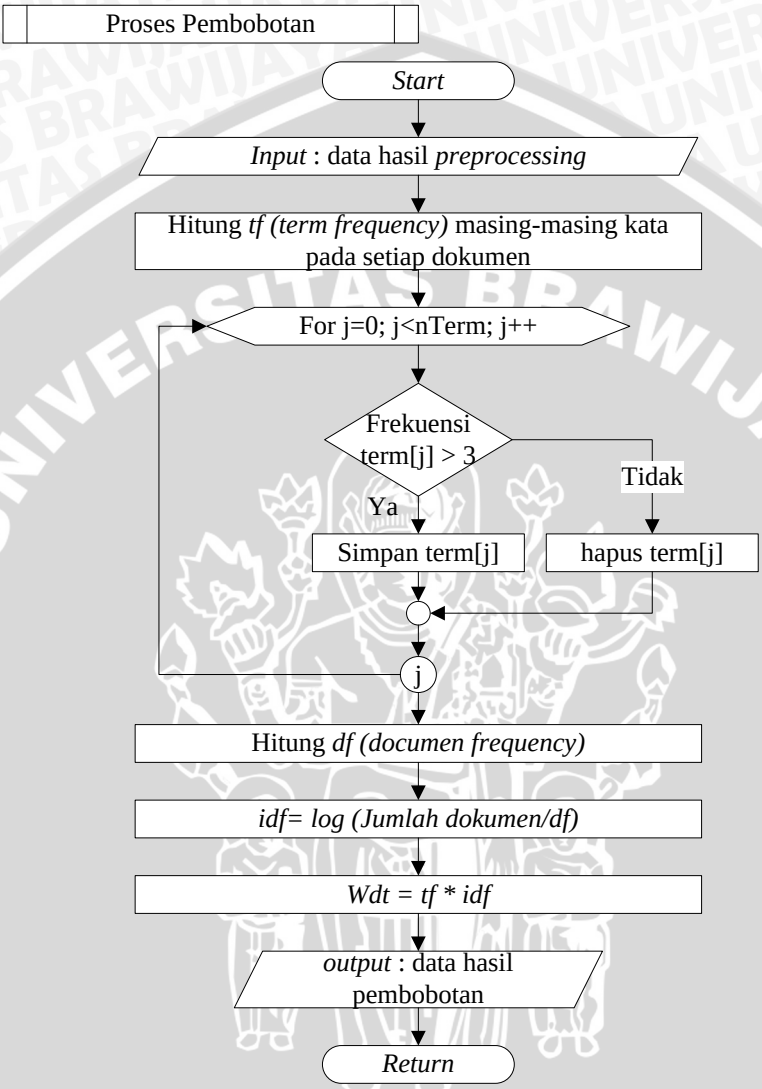
2. Baca file stopwords.text simpan pada string[] stoplist.
3. Satu persatu kata melalui proses pengecekan, jika kata ada dalam *list stopword* dilakukan penghapusan kata.
4. Kata tidak ada dalam *stopword*, dilakukan penyimpanan kata pada stopword.
5. Hasil dari sistem adalah kumpulan kata yang tidak ada pada stoplist.

3.3.3 Perancangan Proses *tf-idf*

Proses selanjutnya setelah melalui pencarian kata dasar, melakukan pembobotan kata. Pembobotan kata adalah proses pencarian bobot suatu kata dari satu dokumen ke dokumen lainnya. Dalam proses pembobotan ini menggunakan metode *term frequency - inverse document frequency* yang sering dikenal dengan metode pembobotan *tf-idf*. Proses pembobotan kata dapat dilihat dari gambar 3.7 *flowchart* proses pembobotan.

Pada gambar 3.7 dijelaskan proses pembobotan menggunakan metode tf-idf, berikut penjelasan proses pembobotan.

1. User memasukan kumpulan kata dalam suatu dokumen berita.
2. Dilakukan perhitungan banyaknya kemunculan suatu kata dalam satu dokumen.
3. Dilakukan proses *feature selection*, kata yang akan diproses selanjutnya adalah kata yang memiliki frekuensi kemunculan lebih dari samadengan 3 dalam satu dokumen.
4. Selanjutnya dilakukan perhitungan jumlah kata yang muncul dalam suatu dokumen, proses tersebut dinamkan perhitungan df atau *document frequency*.
5. Setelah diketahui hasil df, proses berlanjut pada perhitungan *invers documen frequency* dengan menggunakan persamaan 2.2.
6. Diketahui hasil idf dari setiap dokumen, proses selanjutnya menggunakan persamaan 2.1, dengan mengalikan setiap kemunculan kata dengan hasil idf dari tiap dokumen.
7. Output hasil dari pembobotan berupa kata dan bobot kata.

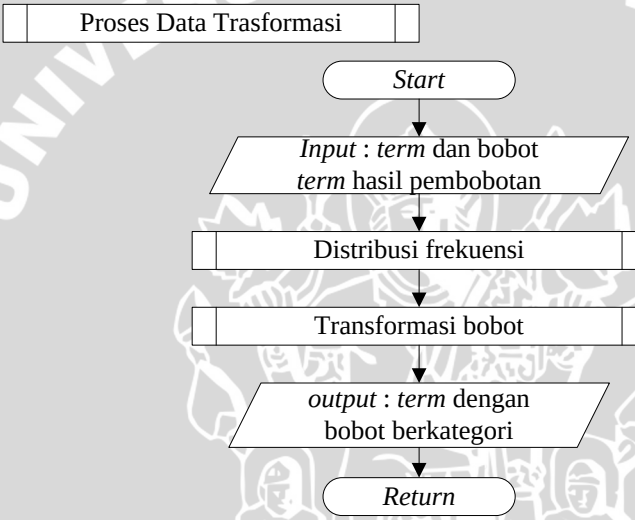


Gambar 3.7 *Flowchart* proses pembobotan

3.3.4 Perancangan Data Transformasi

Pada sub-bab perancangan transformasi data, dilakukan perubahan dari data numerik menjadi data kategori. Dilakukan transformasi data dengan cara seperti ini karena ID3 tidak dapat

memproses data dengan tipe numerik, maka data pembobotan yang bertipe numerik akan diletakan dalam kelompok-kelompok dengan range data tertentu. Terdapat dua proses dalam perancangan transformasi data, distribusi frekuensi yang merupakan metode pembuatan range dari data hasil pembobotan dan pengkategorian bobot merupakan proses pemberian kategori pada bobot sesuai hasil range kategori yang dihasilkan dari distribusi frekuensi. Untuk perancangan proses data transformasi digambarkan pada gambar 3.8 *flowchart* proses data transformasi.



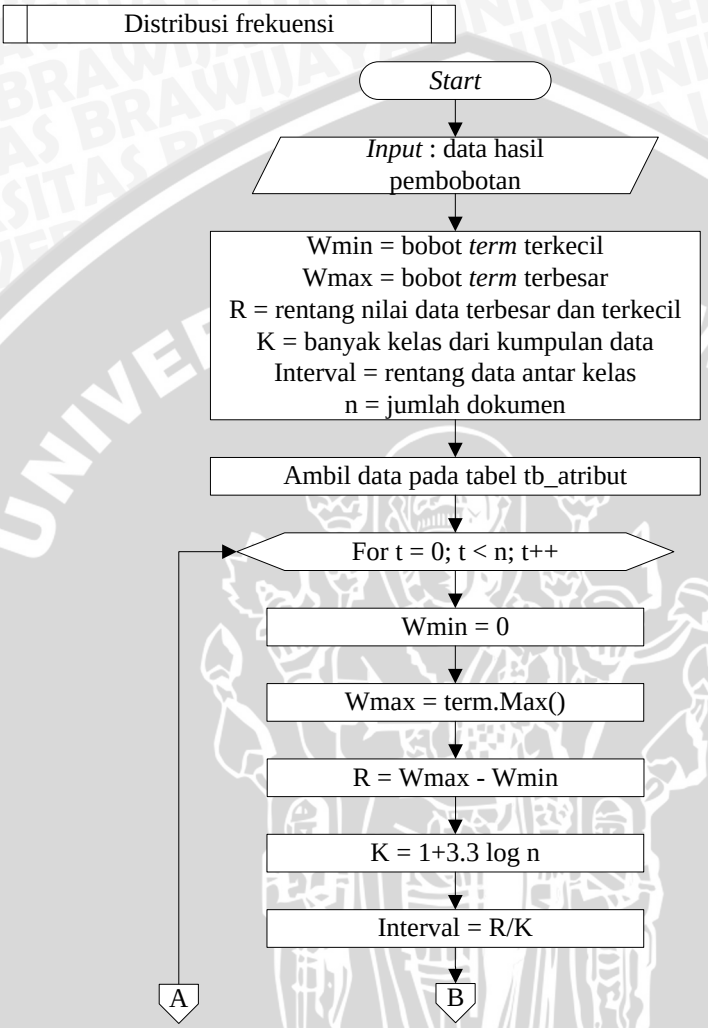
Gambar 3.8 *Flowchart* proses data transformasi

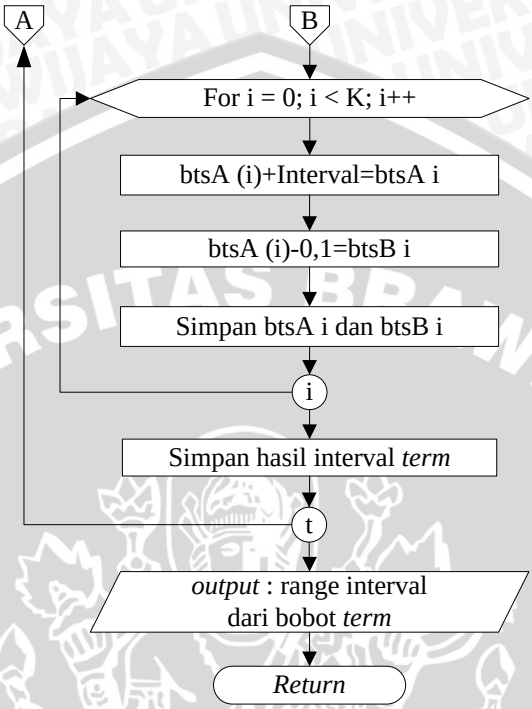
Untuk sub proses pertama pada proses data transformasi adalah distribusi frekuensi. Metode distribusi frekuensi digunakan untuk menentukan pembuatan kelompok data berdasarkan persebaran kumpulan data numerik hasil proses pembobotan.

Dari gambar 3.9 *flowchart* proses distribusi frekuensi, berikut tahap pembuatan interval kelas dengan metode distribusi frekuensi.

1. Sejumlah kata (*term*) dalam suatu dokumen di inputkan kedalam sistem.
2. Dilakukan perulangan sebanyak jumlah kata.

3. Pada nilai Wmin ditetapkan 0, karena kata yang tidak ada dalam dokumen memiliki nilai 0 sehingga tidak perlu dilakukan lagi pencarian nilai minimum.
4. Mencari nilai maksimum dari bobot kata.
5. Menghitung jumlah kelas yang dapat terbentuk dari kumpulan bobot kata.
6. menghitung interval setiap kelas.
7. Untuk menentukan batas atas kelas (btsA) dan batas bawah kelas (btsB), btsA dimulai dari 0 dijumlahkan dengan interval kelas dan btsB didapat dari btsA di kurangi nilai terkecil dari dokumen pada penelitian ini menggunakan 0,1 sebagai nilai terkecil karna data numerik dari bobot berbentuk desimal.
8. Simpan hasil btsA dan btsB untuk tiap perulangan sampai dengan sejumlah kelas yang terbentuk.
9. Simpan hasil range interval *term*.
10. Output yang dihasilkan berupa kelompok range interval dari bobot term.


Distribusi frekuensi
Start
Input : data hasil pembobotan
Wmin = bobot term terkecil
Wmax = bobot term terbesar
R = rentang nilai data terbesar dan terkecil
K = banyak kelas dari kumpulan data
Interval = rentang data antar kelas
n = jumlah dokumen
Ambil data pada tabel tb_atribut
For t = 0; t < n; t++
Wmin = 0
Wmax = term.Max()
R = Wmax - Wmin
K = 1+3.3 log n
Interval = R/K
A
B

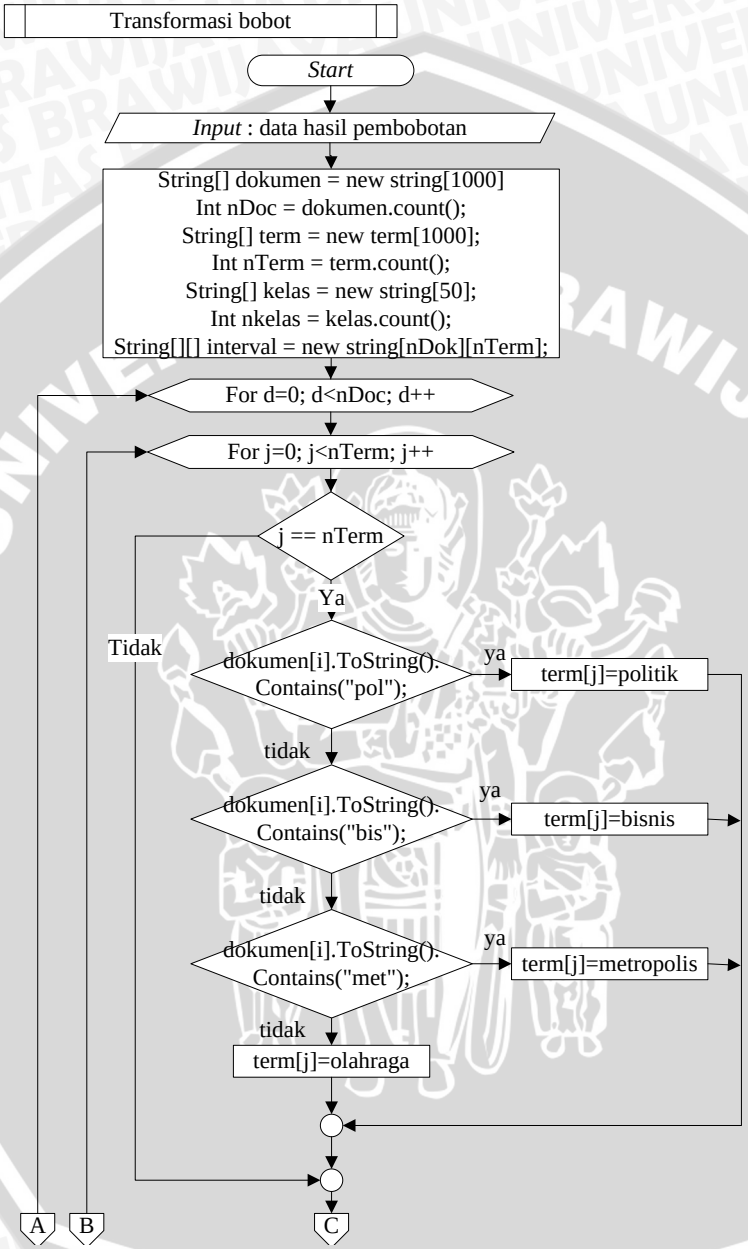


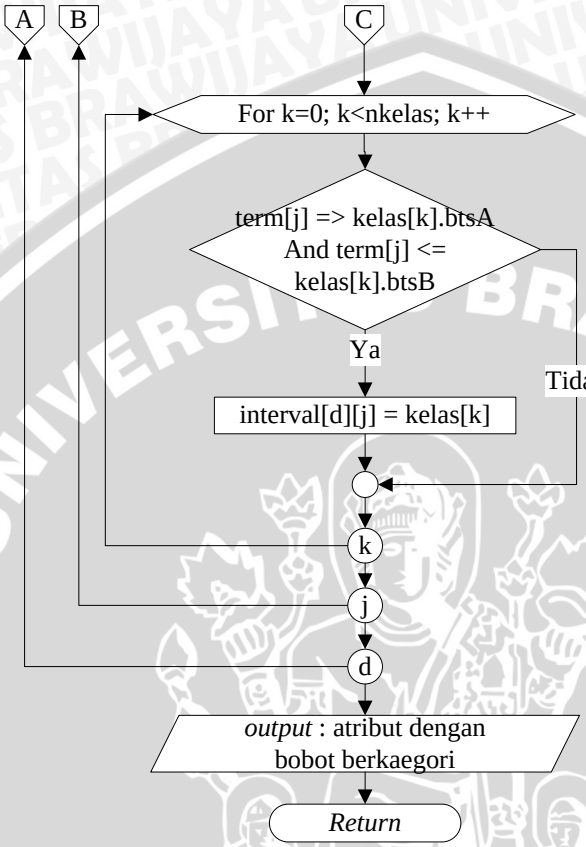
Gambar 3.9 *Flowchart* proses distribusi frekuensi

Setelah dilakukan pembuatan interval setiap kelas dengan metode distribusi frekuensi, selanjutnya dilakukan proses transformasi data. Pada proses ini data akan dimasukan ke-dalam interval sesuai dengan hasil distribusi frekuensi. Berikut proses transformasi data yang ditunjukan pada gambar 3.10 *flowchart* proses transformasi bobot.

1. Data inputan adalah hasil proses pembobotan.
2. Bentuk *array string* untuk menyimpan dokumen. Bentuk *integer* nDoc untuk menyimpan banyak dokumen. Bentuk *array* term tipe *string* untuk menyimpan kata. Bentuk *integer* nTerm untuk menyimpan banyak kata. Bentuk *array* kelas bertipe *string* untuk menyimpan data kelas. Bentuk *integer* nkelas untuk menyimpan banyak kelas.
3. Lakukan perulangan sejumlah dokumen.
4. Lakukan perulangan sejumlah kata pada dokumen indek ke-d.

5. Kondisi j sama dengan nTerm, dilakukan pengkategorian kelas tujuan.
6. Dokumen ke-d string pertama diawali "pol". Jika kondisi terpenuhi, atribut tujuan term ke-j adalah politik. Jika kondisi tidak terpenuhi lanjut ke proses selanjutnya.
7. Dokumen ke-d string pertama diawali "bis". Jika kondisi terpenuhi, atribut tujuan term ke-j adalah bisnins. Jika kondisi tidak terpenuhi lanjut ke proses selanjutnya.
8. Dokumen ke-d string pertama diawali "met". Jika kondisi terpenuhi, atribut tujuan term ke-j adalah metropolis.Jika kondisi tidak terpenuhi, atribut tujuan term ke-j adalah olahraga.
9. Indek ke-j sama dengan nTerm. tidak terpenuhi, dilakukan pengkategorian bobot term sesuai hasil distribusi frekuensi.
10. Lakukan perulangan sebanyak nKelas yang terbentuk.
11. term[j] => kelas[k].btsA dan term[j] <= kelas[k].btsB. Kondisi terpenuhi, interval dokumen ke-d term ke-j memiliki kelas interval ke-k. Kondisi tidak terpenuhi, kembali ke perulangan poin 10.
12. Hasil dari proses transformasi bobot adalah atribut dengan bobot berkategori.
13. Proses berlajut ke sub proses selanjutnya.

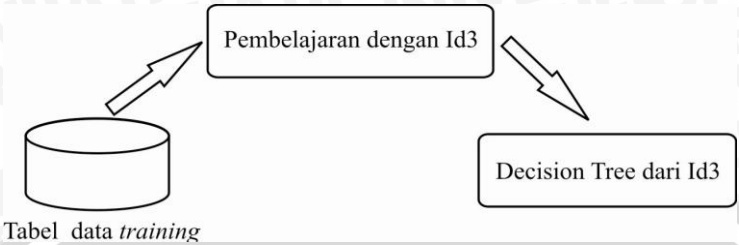

Transformasi bobot
Start
Input : data hasil pembobotan
String[] dokumen = new string[1000]
Int nDoc = dokumen.count();
String[] term = new term[1000];
Int nTerm = term.count();
String[] kelas = new string[50];
Int nkelas = kelas.count();
String[][] interval = new string[nDok][nTerm];
For d=0; d<nDoc; d++
For j=0; j<nTerm; j++
j == nTerm
Ya
Tidak
dokumen[i].ToString().
Contains("pol");
ya
term[j]=politik
tidak
dokumen[i].ToString().
Contains("bis");
ya
term[j]=bisnis
tidak
dokumen[i].ToString().
Contains("met");
ya
term[j]=metropolis
tidak
term[j]=olahraga
A
B
C



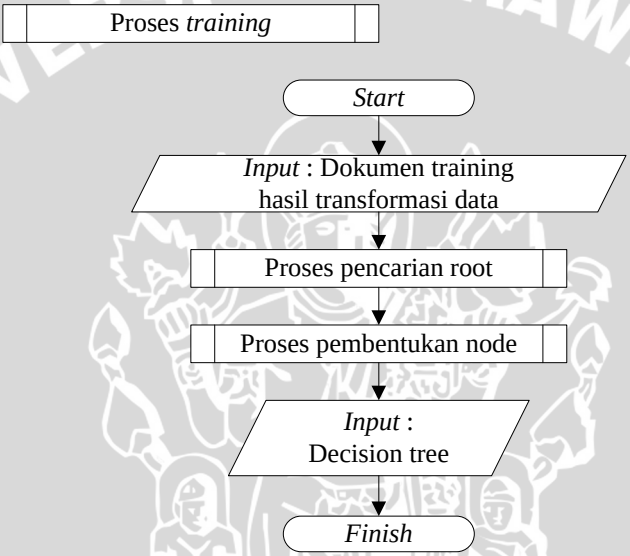
Gambar 3.10 *Flowchart* proses transformasi bobot

3.3.5 Perancangan Proses Latih

Pada tahap pembelajaran dokumen berita yang disiapkan untuk pemebelajaran akan melalui proses pembelajarang dengan ID3. Skema proses pembelajaran dijelaskan pada gambar 3.11, dari data dokumen pembelajaran akan melalu proses pengklasifikasian secara manual untuk dikenali oleh sistem. Hasil dari pembelajaran sistem dengan ID3 berupa *decision tree*. Berikut gambar penjelasan proses pembelajaran dijelaskan pada Gambar 3.12 *Flowchart* proses latih ID3.

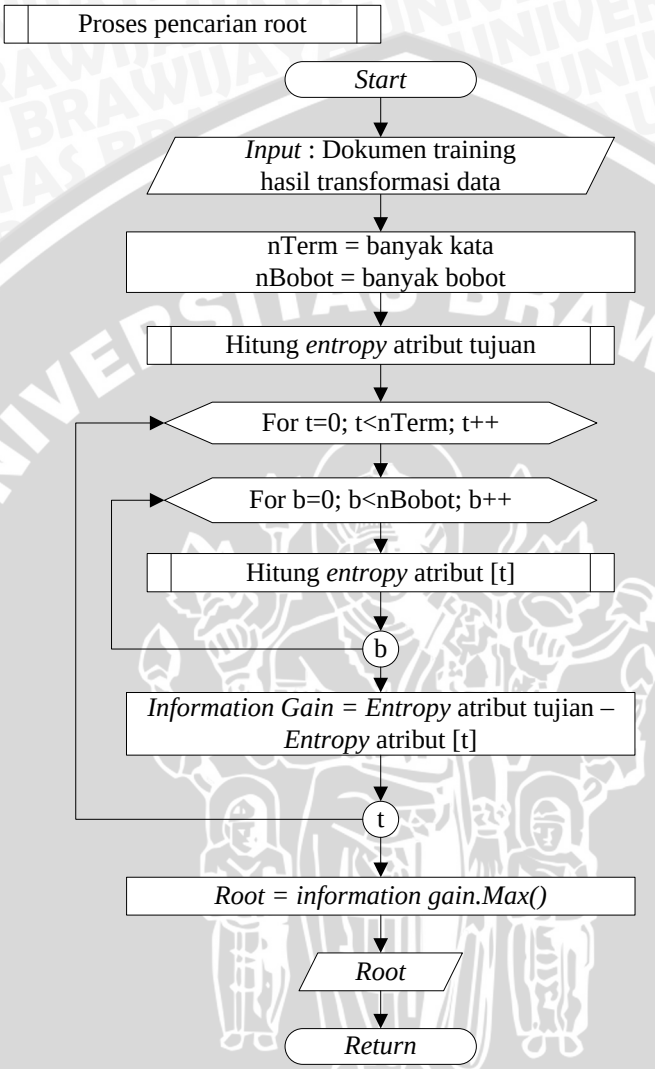


Gambar 3.11 Proses pembelajaran



Gambar 3.12 *Flowchart* proses latih ID3

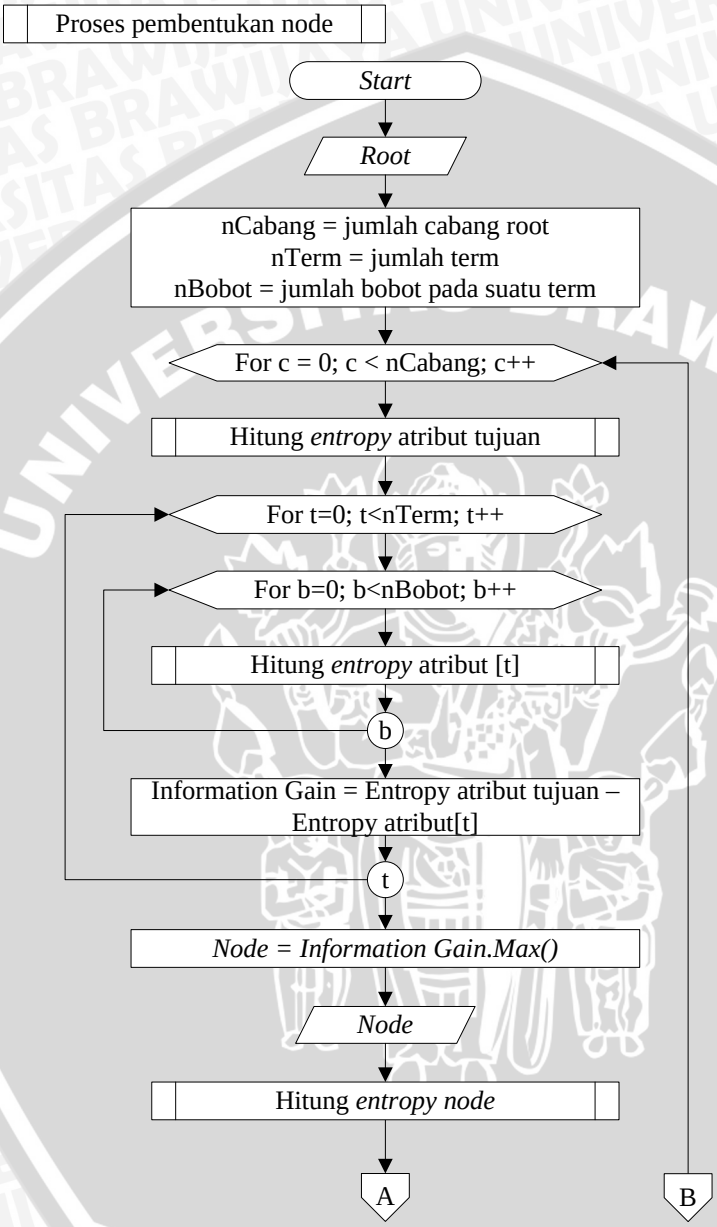
Pada penelitian ini pembelajaran ID3 dibagi dalam dua sub proses, sub proses pertama adalah proses pencarian *root* utama dari *decision tree* dan sub proses kedua adalah proses pebentukan *node decision tree*. Perancangan proses latih digambarkan pada gambar 3.12 *flowchart* .

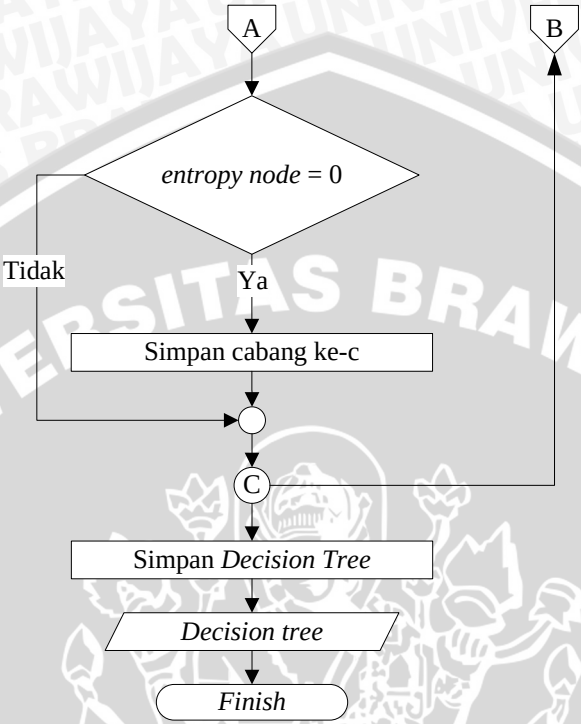


Gambar 3.13 *Flowchart* proses pencarian *root*

Pada tahap pertama dijelaskan proses penentuan *root* dari perbandingan *entropy class* dengan *entropy* atribut. Penjelasan proses pencarian *root* dijelaskan pada gambar 3.13 *flowchart* proses pencarian *root*. Berikut penjelasan proses penentuan *root* dari *decision tree*.

1. Data masukan pada proses ini berasal dari hasil transformasi data.
2. Bentuk *integer* nTerm untuk menyimpan banyak kata. Bentuk *integer* nBobot untuk menyimpan banyak bobot.
3. Hitung *entropy* kelas tujuan.
4. Lakukan perulangan sebanyak nTerm.
5. Lakukan perulangan sebanyak nBobot.
6. Hitung *entropy* atribut pada term ke-t bobot ke-b.
7. Hitung informasi *gain* pada setiap term ke-t.
8. *Root* adalah hasil perhitungan setiap term informasi *gain* yang memiliki nilai tertinggi.
9. Hasil dari proses pencarian *root* adalah satu kata sebagai *root*.
10. Proses berlanjut ke pencarian node.


Proses pembentukan node
Start
Root
nCabang = jumlah cabang root
nTerm = jumlah term
nBobot = jumlah bobot pada suatu term
For c = 0; c < nCabang; c++
Hitung entropy atribut tujuan
For t=0; t<nTerm; t++
For b=0; b<nBobot; b++
Hitung entropy atribut [t]
b
Information Gain = Entropy atribut tujuan – Entropy atribut[t]
t
Node = Information Gain.Max()
Node
Hitung entropy node
A
B



Gambar 3.14 *Flowchart* proses pembentukan *node*

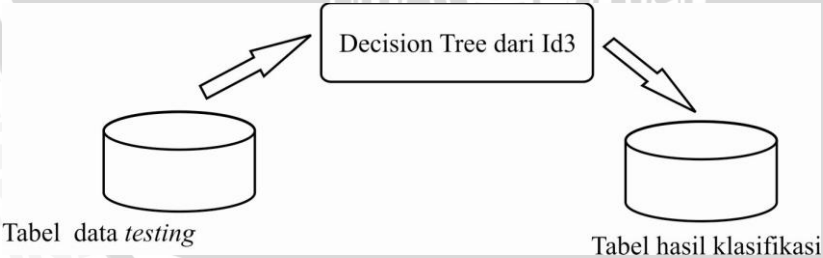
Pada tahap selanjutnya adalah tahap pembentukan *node*, dari hasil penentuan *root* pada proses sebelumnya selanjutnya cabang setiap root akan dilakukan perhitungan *entropy* dan perbandingan *information gain* sampai pada cabang terdapat satu *class* yang sama atau hasil *entropy* dari node bernilai nol. Berikut penjelasn proses pembentukan node yang digambarkan pada gambar 3.14 *flowchart* proses pembentukan *node* .

1. Dari hasil *root* yang dihasilkan pada proses sebelumnya, kemudian dilakukan perhitungan *entropy class* setiap cabang root.
2. Dilakukan pengecekan pada cabang *root*, dan dilakukan perulangan pada setiap cabang *root*.
3. Pada tiap cabang akan dilakukn perhitungan masing – masing *entropy class* pada tiap cabang dari *root.*

4. Lakukan perulangan sejumlah term atribut yang ada pada cabang.
5. Perulangan kedua dimana setiap *term* memiliki bobot kategori, dilakukan perulangan sampai sejumlah bobot kategori yang ada pada *term* atribut.
6. Dilakukan perhitungan *entropy* pada atribut.
7. Kembali ke perulangan kedua pada poin 5, sampai bobot pada *term* atribut tersebut habis.
8. Dilakukan perhitungan *information gain* dengn membandingkan *entropy class* dengan *entropy* atribut.
9. Setelah didapat *information gain*, diambil nilai terbesar dari *information gain* tersebut untuk digunakan sebagai *node*.
10. Dari hasil *node* dilakukan pengecekan *entropy class* yang berada pada cabang tersebut.
11. Jika *entropy class* bernilai nol, maka cabang tersebut merupakan akhir dari tree.
12. Menyimpan hasil perhitungan cabang ke-c tersebut.
13. *Entropy* tidak bernilai nol maka kembali ke cabang selanjutnya pada point ke dua.
14. Hasil dari proses ini berupa pohon keputusan dari proses pembelajaran.

3.3.5 Perancangan Proses *Testing*

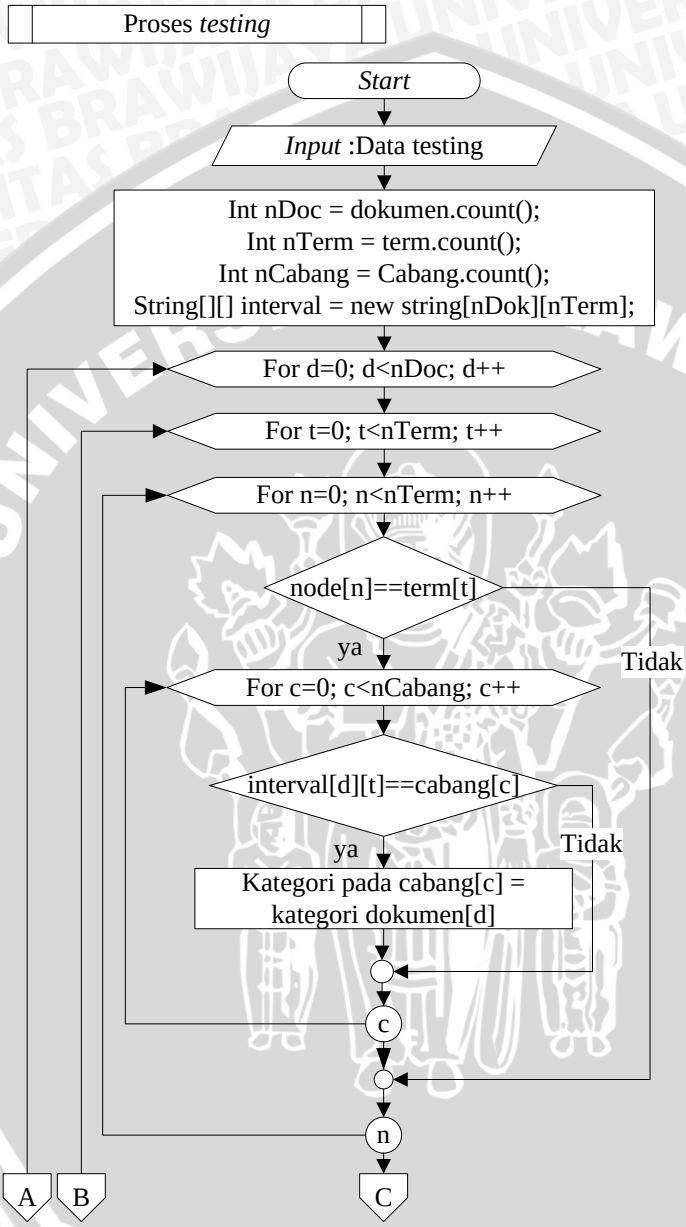
Berikut merupakan skema dari proses klasifikasi ditunjukan pada gambar 3.15. Dokumen testing yang telah disiapkan akan dilakukn pengklasifikasian sesuai hasil dari pembelajaran sistem pada proses pembelajaran. Hasil dari proses klasifikasi berupa dokumen yang sudah memiliki kategori.



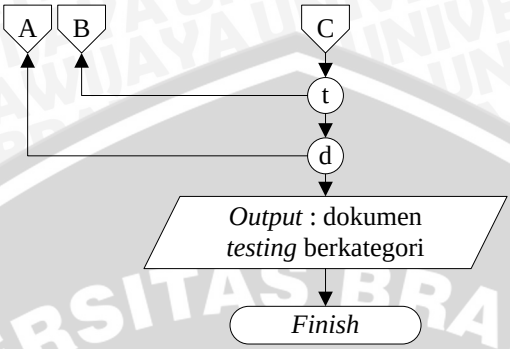
Gambar 3.15 Proses klasifikasi

Pada gamar *flowchart* 3.16 yang merupakan proses klasikasi penemuan kategori untuk dokumen, berikut penjelasan proses pengklasifikasian.

1. Data masukan pada proses testing adalah data tes hasil transformasi data.
2. Bnetuk integer nDoc untuk menyimpan banyak dokumen test. Bentuk integer nTerm unutk menyimpan banyak kata. Bentuk integer nCabang untuk menyimpan banyak cabang dicesion tree. Bnetuk array interval bertipe string untuk menyimpan interval bobot kata.
3. Lakukan perulangan sejumlah dokumen test.
4. Lakukan perulangan sejmlah kata pada dokumen ke-d.
5. lakukan perulangan sejumlah node.
6. NOde ke-n sama dengan term ke-t. Kondisi terpenuhi berlanjut ke pengecekan cabang.
7. Lakukan pengecekan cabang node ke-n.
8. interval dokumen ke-d term ke-t sama dengan cabang ke-c. Kondisi memenuhi, kategori cabang ke-c adalah kategori dokumen ke-d.
9. Kondisi poin 8 tidak terpenuhi kembali pada perulangan point 7.
10. Hasil dari proses ini adalah dokumen tes berkategori.


Proses *testing*
Start
Input :Data testing
Int nDoc = dokumen.count();
Int nTerm = term.count();
Int nCabang = Cabang.count();
String[][] interval = new string[nDok][nTerm];
For d=0; d<nDoc; d++
For t=0; t<nTerm; t++
For n=0; n<nTerm; n++
node[n]==term[t]
ya
Tidak
For c=0; c<nCabang; c++
interval[d][t]==cabang[c]
ya
Tidak
Kategori pada cabang[c] =
kategori dokumen[d]
c
n
A
B
C

11.



Gambar 3.16 *Flowchart* proses klasifikasi ID3

3.4 Perhitungan Manual

Pada sub bab perhitungan manual akan dibahas contoh perhitungan manual untuk data latih, testing, pembobotan tf-idf, proses generalisasi dengan distribusi frekuensi dan pengklasifikasian dengn algoritma ID3. Perhitungan manual dimulai setelah kata melalui tahap *preprocessing.*

3.4.1 Sumber Data

Data dalam perhitungan manual berjumlah 9 dokumen berita. Sembilan dokumen berita tersebut dibagi menjadi dua bagian, 6 dokumen berita sudah memiliki kategori yang nantinya akan di gunakan sebagai contoh perhitungan dokumen latih. 3 dokumen berita sisanya tidak berkategori dan digunakan sebagai data *testing.* 6 dokumen latih terbagi menjadi dua kategori, 3 dokumen dalam kategori politik dan 3 dokumen dalam kategori olahraga. Dokumen dalam perhitungan manual dan perhitungan jumlah frekuensi kata ada pada lampiran 2.

3.4.2 Proses Pembobotan

Setelah melakukan perhitungan frekuensi setiap kata pada 9 dokumen, kemudian dilakukan *stopword* dan *stimming* secara manual. Data yang diambil dalam proses selanjutnya adalah kata yang memiliki frekuensi lebih dari 3. Daftar kata yang dihasilkan dari tahap processing terdapat dalam dua tabel berikut, tabel 3.1 hasil *preprocessing* dari data *latih* dan tabel 3.2 hasil *preprocessing* untuk data *testing*. Kedua tabel yang berisi kata tersebut akan dilakukan

proses pembobotan. Berikut hasil perhitungan *term frequency*, *document frequency* dan *invers document frequency*, perhitungan idf dengan menggunakan persamaan 2.1 dan dengan jumlah dokumen sebanyak 6 untuk data latih dan 3 untuk data *testing.*

Tabel 3.1 Data latih untuk *term frequency, document frequency* dan *invers document frequnecy*

tf	fraksi	choi	kader	partai	me	fifa
D1	7					
D2		6				
D3			6	5		
D4					7	
D5						5
D6						8
df	1	1	1	1	1	2
idf	0.7782	0.7782	0.7782	0.7782	0.7782	0.4771

Tabel 3.2 Data *testing* untuk *term frequency, document frequency* dan *invers document frequnecy*

tf	anggota	dpr	me	calon	komite	fifa
D7	5	10	13			
D8				7		
D9	5				5	5
df	2	1	1	1	1	1
idf	0.1761	0.4771	0.4771	0.4771	0.4771	0.4771

Setelah diketahui hasil *idf* selanjutnya dilakukan pembobotan kata pada setiap dokumen dengan menggunakan persamaan 2.2. Berikut contoh perhitungan pembobotan untuk kata fraksi. Tabel hasil pembobotan berada pada tabel 3.8 untuk data *latih* dan 3.6 untuk tabel data *testing.*

$$wdt = 7 \times 0.7782 = 5.4471$$

Tabel 3.3 Data *latih* hasil pembobotan kata

wdt	fraksi	choi	kader	partai	me	fifa
D1	5.4471	0	0	0	0	0
D2	0	4.6689	0	0	0	0
D3	0	0	4.6689	3.8908	0	0
D4	0	0	0	0	5.4471	0
D5	0	0	0	0	0	2.3856
D6	0	0	0	0	0	3.8170

Tabel 3.4 Data *testing* hasil pembobotan kata

wdt	anggota	dpr	me	calon	komite	fifa
D7	0.8805	4.7712	5.2483	0	0	0
D8	0	0	0	3.3398	0	0
D9	0.8805	0	0	0	2.3856	2.3856

3.4.3 Proses Data Transformasi

Pada proses data transformasi, data berupa numerik tidak dapat diterima oleh algoritma *Iterative Dichotomizer Tree*. Karna algoritma *Iterative Dichotomizer Tree* memproses data berbntuk data berkategori. Dalam masalah ini digunakan data transformasi *generalization*, dimana pengkategorian data dalam kelas – kelas tertentu dengan metode distribusi frekuensi. Dalam penerapan kedalam sisitem distribusi frekuensi dilakukan pada setiap atribut karna setiap atribut memiliki data bobot berbeda. Pada perhitungan manual ini distribusi frekuensi tidak dilakukan ke seluru atribut karna julah dokumen terbatas, maka dilakukan distribusi frekuensi keseluruh data hasil pembobotan dari data latih. Berikut langkah - langkah untuk mebuat distribusi frekuensi.

1. Hasil data pembobotan data latih dilakukan pengurutan data dan mencari data terkecil dan data terbesar (X_{min} dan X_{max}). Dilakukan pencarian *range* anatara kedua data tersebut dengan persamaan 2.3.
   Hasil pengurutan pembobotan :

Tabel 3.5 Tabel hasil pengurutan data pembobotan

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	2.3856
3.817	3.8908	4.6689	4.6689	5.4471	5.4471

Dari tabel 3.10 diketehui nilai maksimal dari data adalah 5.4471 dan nilai minimum dari data adalah 0 dengan jumlah data sampel sejumlah 36. Dilakukan perhitungan rentang nilai minimum dan maksimum dengan persamaan 2.3.

$R = 5.4471 - 0 = \mathbf{5.4471}$

2. Menghitung banyak kelas yang akan digunakan dengan menggunakan persamaan 2.4.

   $K = 1 + 3.3 \log 36 = 6.1357$

   Dari hasil perhitungan kelas diatas maka kelas yang dapat dibuat dari data tersebut sebanyak **6 kelas**.
3. Menghitung panjang interval setiap kelas dengan menggunakan persamaan 2.5.

   $interval = \frac{5.4471}{6} = 1.07$

   Dari hasil perhitungan interval diatas, maka interval kelas dengan panjang **1**.
4. Memilih ujung bawah kelas interval pertama. Untuk ini, bisa diambil sama dengan data terkecil atau mulai data yang lebih kecil, tapi selisihnya harus kurang dari panjang kelasnya. Maka dimabil 0 sebagai sampel data terkecil, menentukan batas kelas atas dengan menambah kan 0 dengan interval kemudian mengurangi batas atas kelas dengan skala terkecil dari data. Skala terkecil data menggunakan 0.1. Maka didapat interval kelas pertama 0 – 0.9, dan interval kelas kedua 1 – 1,9 dan demikian seterusnya.
5. Membuat tabel distribusi frekuensi sesuai aturan sebelumnya. Setiap *range* interval memiliki nama untuk memudahkan pemanggilan *record* data. Untuk mengatasi data dengan nilai bobot lebih dari data X_{max} dalam distribusi frekuensi, maka pada

data maksimal dilakukan perlakuan berbeda dengan menggunakan kondisi data x > max. Hasil distribusi frekuensi ditunjukan pada tabel 3.6.

Tabel 3.6 Tabel hasil pengurutan data pembobotan

Data interval	Nilai interval	frekuensi
Int1	0 – 0.9	29
Int2	1 – 1.9	0
Int3	2 – 2.9	1
Int4	3 – 3.9	2
Int5	4 – 4.9	2
Int6	5 – 5.9	2
Int7	X > 6	0
Jumlah data		36

Dari hasil tabel 3.6 dihasilkan pengkategorian untuk pembobotan nilai data, selanjutnya melakukan transformasi bobot pada tabel 3.5. Nilai setiap bobot akan digantikan dengan data interval sesuai dengan pengkategorian data pada tabel 3.6. Berikut hasil transformasi data dan label kategori dari setiap dokumen ditunjukan pada tabel 3.7 Data latih hasil data transformasi.

Tabel 3.7 Data latih hasil data transformasi

	fraksi	choi	kader	partai	me	fifa	**Kategori**
D1	Int6	Int1	Int1	Int1	Int1	Int1	**Pol**
D2	Int1	Int5	Int1	Int1	Int1	Int1	**Pol**
D3	Int1	Int1	Int5	Int4	Int1	Int1	**Pol**
D4	Int1	Int1	Int1	Int1	Int6	Int1	**Ola**
D5	Int1	Int1	Int1	Int1	Int1	Int3	**Ola**
D6	Int1	Int1	Int1	Int1	Int1	Int4	**Ola**

3.4.4 Perancangan Pohon Keputusan

Pada sub bab perancangan pohon keputusan, membahas perhitungan dalam pembentukan pohon keputusan dan contoh pembuatan pohon keputusan. Dari tabel 3.7 yang dihasilkan pada proses transformasi data, data sudah dalam bentuk kategori sehingga sudah bisa dilakukan perhitungan menggunakan persamaan ID3. Berikut langkah perhiungan pembentukan tree.

1. Pembentukan *root* utama.

Melakukan perbandingan *entropy* atribut kategori dengan atribut kata dengan menggunakan persamaan 2.6.

Menghitung *entropy* kategori (S)

S adalah 6 data dari tabel kategori pada tabel 3.12 dengan prediksi 3 dokumen berkategori politik dan 3 dokumen lainya dalam kategori olahraga.

$$Entropy\ kategori\ (S) = -\left(\frac{3}{6}\right) log_2 \left(\frac{3}{6}\right) - \left(\frac{3}{6}\right) log_2 \left(\frac{3}{6}\right) = \mathbf{1}$$

Menghitung *entropy* atribut dan menetukan *information gain*

Berikut contoh berhitungan entropy dan information gain untuk atribut fraksi, *information gain* dengan persamaan 2.7. Dengan jumlah data 6, record data int6 dengan jumlah 1 (dalam kategori politik 1 dan dalam kategori olahraga 0), record data int1 dengan jumlah data 5 (dalam kategori politik 2 dan dalam kategori olahraga berjumlah 3).

***Entropy* fraksi** int6 1 (1,0) dan int1 5 (2,3)

Entropy int6 1 (1,0)

$$Entropy\ int6\ (1{,}0) = -\left(\frac{1}{1}\right) log_2 \left(\frac{1}{1}\right) - \left(\frac{0}{1}\right) log_2 \left(\frac{0}{1}\right) = \mathbf{0}$$

Entropy int1 5 (2,3)

$$Entropy\ int1\ (2{,}3) = -\left(\frac{2}{5}\right) log_2 \left(\frac{2}{5}\right) - \left(\frac{3}{5}\right) log_2 \left(\frac{3}{5}\right) = \mathbf{0.9709}$$

***Information Gain* fraksi**

Perbandingan *entropy* yang dimiliki fraksi dengan *entropy* pada kelas kategori.

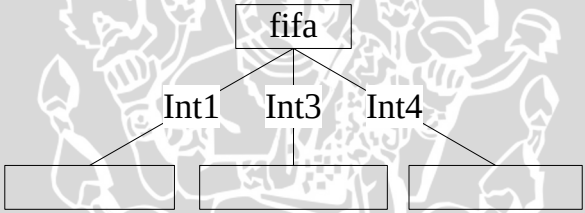
$$IG\ fraksi = 1 - \left(\frac{1}{6}\right) 0 - \left(\frac{5}{6}\right) 0.9709 = \mathbf{0.1909}$$

Berikut hasil perhitungan *entropy* dan *information gain* pada seluru atribut ditampilkan pada tabel 3.13.

Tabel 3.8 Hasil perhitungan *entropy* dan *information gain*

	fraksi	**choi**	**kader**	**partai**	**me**	**fifa**
Int1	0.9709	0.9709	0.9709	0.9709	0.9709	0.81127
Int2	-	-	-	-	-	-
Int3	-	-	-	-	-	0
Int4	-	-	-	0	-	0
Int5	-	0	0	-	-	-
Int6	0	-	-	-	0	-
IG	0.1909	0.1909	0.1909	0.1909	0.1909	**0.4591**

Dari hasil perhitungan *information gain* pada Tabel 3.8 Hasil perhitungan *entropy* dan *information gain*, atribut fifa merupakan atribut terbaik karna memiliki *information gain* terbesar dibandingkan *information gain* atribut yang lainya. Berikut gambar tree yang terbentuk dari perhitungan root pada gambar 3.18.



Gambar 3.17 Tree hasil perhitungan root

2. Node cabang Int1 dari atribut fifa.

Dilakukan pengecekan atribut mana saja yang berada pada cabang Int1, data atribut yang berada pada cabang Int1 terdapat pada tabel 3.9.

Tabel 3.9 Data atribut pada cabang Int1 untuk atribut fifa

	fraksi	choi	kader	partai	me	**Kategori**
D1	Int6	Int1	Int1	Int1	Int1	**Pol**
D2	Int1	Int5	Int1	Int1	Int1	**Pol**
D3	Int1	Int1	Int5	Int4	Int1	**Pol**
D4	Int1	Int1	Int1	Int1	Int6	**Ola**

Dari data tabel diatas dilakukan perhitungan *entropy* untuk kelas kategori, jumlah data 4 dengan 3 data berada pada kategori politik dan 1 data berada pada kategori olahraga.

Entropy kategori 4 (3,1)

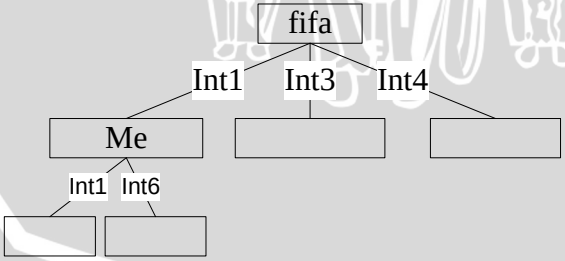
$$Entropy \text{ kategori } (S_{Int1}) = -\left(\frac{3}{4}\right) log_2 \left(\frac{3}{4}\right) - \left(\frac{1}{4}\right) log_2 \left(\frac{1}{4}\right)$$
$$= \mathbf{0.8112}$$

Sama dengan perhitungn sebelumnya dilakukan perhitungan *entropy* dan *information gain* pada seluru atribut. Hasil perhitungan terdapat pada tabel 3.10.

Tabel 3.10 Hasil perhitungan *entropy* dan *information gain* untuk cabang Int1

	fraksi	**choi**	**kader**	**partai**	**me**
Int1	0.9182	0.9182	0.9709	0.9709	0
Int2	-	-	-	-	-
Int3	-	-	-	-	-
Int4	-	-	-	0	-
Int5	-	0	0	-	-
Int6	0	-	-	-	0
IG	0.1225	0.1225	0.1225	0.1225	**0.8112**

Dari hasil perhitungan *information gain* pada tabel 3.10 didapat atribut me yang memiliki nilai *information gain* terbesar, maka me digunakan sebagai node dalam cabang Int1 atribut fifa. Gambar tree dari hasil perhitungan cabang Int atribut fifa ditunjukan pada gambar 3.19.



Gambar 3.18 Tree hasil cabang Int1 untuk atribut fifa

2.1 Node cabang Int1 dari atribut me.

Dilakukan rekursif pemangilan fungsi ID3 untuk aribut me dengan *record* data Int1, berikut data atribut yang berada pada cabang Int1 untuk atribut me pada tabel 3.16.

Tabel 3.11 Data atribut pada cabang Int1 untuk atribut me

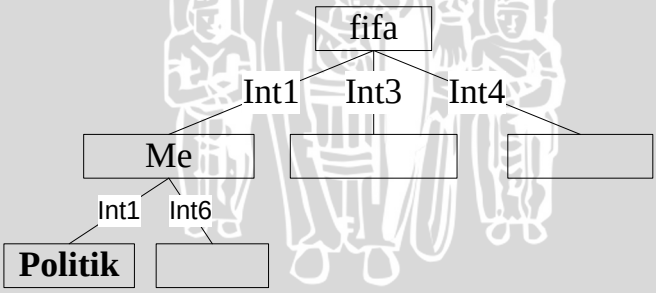
	fraksi	choi	kader	partai	**Kategori**
D1	Int6	Int1	Int1	Int1	**Pol**
D2	Int1	Int5	Int1	Int1	**Pol**
D3	Int1	Int1	Int5	Int4	**Pol**

Dari data tabel 3.11 dilakukan perhitungan *entropy* untuk kelas kategori, jumlah data 3 dengan 3 data berada pada kategori politik dan 0 data berada pada kategori olahraga.

Entropy kategori 3 (3,0)

$$Entropy \text{ kategori } (S_{Int1}) = -\left(\frac{3}{3}\right) log_2 \left(\frac{3}{3}\right) - \left(\frac{0}{3}\right) log_2 \left(\frac{0}{3}\right) = 0$$

Dari hasil *entropy* kelas diperoleh nilai 0 yang berarti sampel data yang berada pada cabang Int1 untuk atribut me sudah dalam satu kelas tujuan yang sama, maka tidak dilakukan rekursif dan proses kembali ke node sebelumnya. Pada cabang Int1 untuk atribut um memiliki label politik. Hasil tree dari cabang ini ditunjukan pada gambar 3.20.



Gambar 3.19 Tree cabang Int1 untuk atribut me

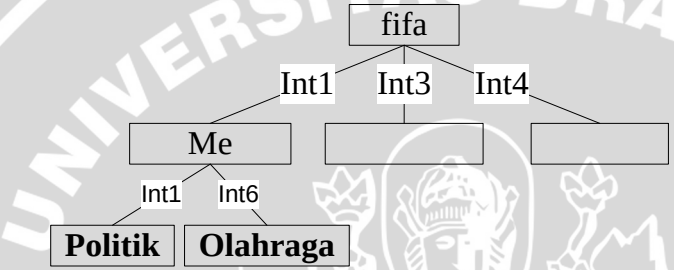
2.2 Node cabang Int6 dari atribut me.

Dilakukan pemanggilan fungsi ID3 untuk cabang Int6 pada atribut me, data berada pada tabel 3.13. Pada sampel data tersebut

data sudah berada pada atribut target atau kelas utama, maka tidak dilakukan proses selanjutnya. Cabang Int6 untuk atribut me memiliki label olahraga, gambar tree yang dihasilkan ditunjukkan pada gambar 3.21.

Tabel 3.13 Data atribut pada cabang Int1

	fraksi	choi	kader	partai	me	**Kategori**
D4	Int1	Int1	Int1	Int1	Int6	**Ola**



Gambar 3.20 Tree cabang Int6 untuk atribut me

3. Node cabang Int3 dan Int4 dari atribut fifa.

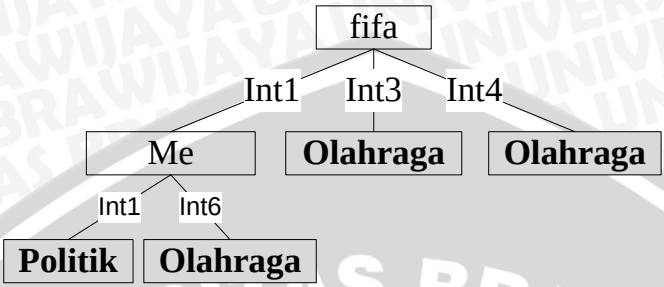
Pada cabang Int3 dan Int4 pada atribut fifa sudah dalam atribut target, maka tidak dilakukan rekursi dan label masing – masing cabang diberikan sesuai atribut yang tersedia pada atribut target. Gambar *tree* ditunjukan pada gambar 3.22.

Tabel 3.14 Data atribut pada cabang Int3

	fraksi	choi	kader	partai	me	**Kategori**
D5	Int1	Int1	Int1	Int1	Int1	**Ola**

Tabel 3.15 Data atribut pada cabang Int4

	fraksi	choi	kader	partai	me	**Kategori**
D6	Int1	Int1	Int1	Int1	Int1	**Ola**



Gambar 3.21 Tree cabang Int3 dan Int4 untuk atribut fifa.

Ketika semua data sampel sudah dalam satu atribut target maka rekusi sudah tidak berlanjut, ada gambar 3.22 merupakan hasil tree dari proses pembelajaran dokumen berita.

3.4.5 Perancangan *Rule Tree*

Setelah dihasilkan *decision tree* dari proses pembelajaran, dilakukan pengekstrakan hasil *decision tree* untuk mendapatkan aturan dalam pengklasifikasian dari *decision tree* tersebut. Berikut aturan klasifikasi yang didapat :

1. **IF** atribut fifa = Int1 **AND** atribut me = Int1 **THEN** dokumen berita termasuka dalam kategori Politik.
2. **IF** atribut fifa = Int1 **AND** atribut me = Int6 **THEN** dokumen berita termasuk dalam kategori Olahraga.
3. **IF** atribut fifa = Int3 **THEN** dokumen berita termasuk dalam kategori Olahraga.
4. **IF** atribut fifa = Int4 **THEN** dokumen berita termasuk dalam kategori Olahraga.

3.4.6 Perancangan Proses Klasifikasi

Pada sub bab sebelumnya hasil *decision tree* telah terbentuk dan telah dilakukan pengekstrakan untuk menghasilkan aturan dalam pengklasifikasian. Pada tahap ini dilakukan proses klasifikasi sesuai dengan hasil aturan klasfikasi yang terbentuk pada dokumen pembelajaran.

Pada data testing dilakukan proses yang sama dengan data latih, sampai pada proses pembobotan. Setelah setiap atribut mendapatkan nilai bobot, dilakukan data transformasi. Pada data transformasi tidak dilakukan perhitungan distribusi frekuensi seperti pada data latih,

data bobot yang ada pada data testing tinggal mengikuti hasil distribusi frekuensi dari hasil data latih. Berikut hasil data transformasi untuk data testing pada tabel 3.20.

Tabel 3.16 Data *testing* hasil *genreralisasi*

	anggota	dpr	me	calon	komite	fifa	
D7	Int1	Int5	Int6	Int1	Int1	Int1	Pol
D8	Int1	Int1	Int1	Int4	Int1	Int1	Ola
D9	Int1	Int1	Int1	Int1	Int3	Int3	Ola

Pada tabel 3.6 dilakukan pengklasifikasin terhadap data testing sesuai aturan klasfikasi yang sudah terbentuk. Langkah pengklasifikasian dilakukan perdokumen, dimulai dengan atribut yang memiliki nilai bobot tertinggi (pada distribusi frekuensi ini kelas data tertinggi adalah Int7) proses berhenti ketika atribut tertinggi telah mendapatkan label kategori.

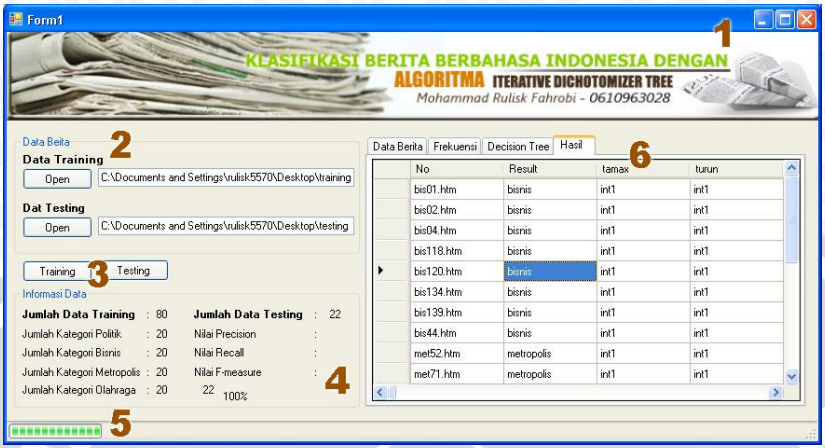
Pada data testing D7 dilakukan pengurutan nilai bobot, dimulai dari Int7 atribut me, Int5 dari atribut dpr, Int1 dari atribut fifa, komite, calon dan anggota. Dari dokumen D7 atribut yang bisa masuk dalam aturan klasifikasi adalah atribut fifa dengan int1 dan atribut me dengan atribut Int6 yang memenuhi aturan dua (2) pada aturan klasfikasi, label untuk D7 adalah Olahraga.

Pada data testing D8 atribut yang bisa masuk dalam aturan klasifikasi adalah atribut fifa dengan Int1 dan atribut me dengan Int1, aturan klasifikasi yang sesuai untuk kondisi D8 adalah aturan ke satu maka dokumen berita D8 berlabel politik.

Pada data testing D9 atribut yang memenuhi aturan klasifikasi adalah atribut fifa dengan Int3 terdapat pada aturan ke tiga (3) maka label dokumen berita D9 berlabel olahraga.

3.5 Perancangan Antarmuka

Sistem dibuat dengan basis visual menggunakan bahasa pemrograman visual C#. Sistem memiliki satu form utama. Dari form utama tersebut terdapat beberapa bagian, berikut penjelasan bagian-bagian form dari gambar 3.22.



Gambar 3.22 Antarmuka sistem.

Pada gambar 3.23 antarmuka sistem terdapat 7 bagian dari *form* tersebut yang memiliki kegunaan berbeda, berikut keterangan dari bagian gambar 3.23.

1. *Form* utama dari sistem klasifikasi berita.
2. Data Berita, terdapat dua *button*, digunakan untuk mengambil dokumen berita. Keterangan teks di samping *button* digunakan sebagai informasi lokasi data berita.
3. Proses data, terdapat dua *button* yaitu :
   - *Button latih*,digunakan untuk memulai proses pembelajaran.
   - *Button testing*, digunakan untuk memulai proses pengklasifikasian.
4. Informasi data, terdapat beberapa label sebagai informasi jumlah data *latih*, jumlah data setiap kategori dan hasil proses *testing*. Terdapat informasi untuk pengujian data.
5. *Progress bar*, penanda berjalannya dari setiap proses yang berjalan.
6. Terdapat empat tab sebagai hasil dari pemprosesan data.
   a. Tab data berita untuk menampilkan hasil pembersihan tag HTML.
   b. Tab frekuensi data, untuk menampilkan hasil dari proses *preprocessing* dan jumlah kemunculan kata dari setiap dokumen.

c. Tab *decision tree,* menampilkan hasil pembangkita pohon keputusan setelah proses pembelajaran.
d. Tab hasil, menampilkan hasil klasifikasi setelah dilakukan proses *testing*.

3.6 Metode Pengujian

Setelah selesai melakukan pembuatan sistem, dilakukan pengujian terhadap sistem. Dalam penelitian ini metode pengujian dengan mengunakan perbandingan jumlah data terhadap perubahan prosentase *precision, recall* dan *accuracy*. Pada tabel perbandingan data pengujian terdapat 8 macam pengujian dengan prosentase data *latih* terhadap data *testing*, dengan jumlah data *latih* 50%, 60%, 66.7%, 70%, 80%, 90%, 95% dan 97.5% data. Berikut tabel hasil untuk penelitian ditunjukan pada tabel 3.17.

Tabel 3.17 Tabel hasil pengkategorian pengujian

	Data latih 50%		…….		Data latih 97.5%	
	Manual	Sistem	Manual	Sistem	Manual	Sistem
D1						
D2						
D3						
D4						
…….						
D ke n						
Total data						

Pada tabel 3.17 tabel data pengujian hasil klasifikasi manual dibandingkan dengan hasil klasifiksai sistem. Pada bagian kolom data pengujian berisi perbandingan data antara data *latih* dengan data *testing*. Perbandingan data *latih* dengan data *testing* dimulai dari 50% : 50%, 60% : 40%, 66.7% : 33.3% sampai dengan pengujian dengan perbandingan 97.5% : 2.5%. Pada baris tabel data, merupakan indeks dari dokumen berita sampai kedokumen berita ke-n. Pada kolom kedua atas terdapat manual dan sistem, berisi hasil pengklasifikasian manual dengan hasil pengklasifikasian sistem.

Dari tabel pengklasifikasian pada tabel 3.10, kemudian dilakukan penilaian terhadap hasil pengklasifikasian. Nilai hasil pengklasifikasian mengacu pada tabel 2.9 Matriks *confusion*, dengan nilai TP, FP, dan FN. Setelah didapat nilai TP, FP dan FN dari hasil pengklasifikasian sistem, selanjutnya dilakukan perhitungan *recall, precision dan F-measure* pada setiap kategori. Dilakukan perhitungan *recall, precision* dan *F-measure* rata-rata dari hasil perhitungan setiap kategori. Berikut perancangan tabel untuk perhitungan *recall, precision* dan *F-Measure* setiap kategori dan hasil rata-rata *recall, precision dan F-measure* ditampilkan pada tabel 3.18.

Tabel 3.18 Hasil Perhitungan *recall, precision* dan *F-measure.*

n	Politik			Bisnis			Metropolis			Olahraga			Rata-rata		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
1															
2															
3															
4															
5															
6															
7															
8															
9															

Keterangan :
n : Banyak pengujian.
R : Nilai *Recall* yang didapat dari pengujian sistem.
P : Nilai *Precision* yang didapat dari pengujian sistem.
F : Nilai *F-measure* yang didapat dari pengujian sistem.
Politik : Hasil pengujian kategori politik.
Bisnis : Hasil pengujian kategori bisnis.
Metropolis : Hasil pengujian kategori metrolpolis
Olahraga : Hasil pengujian kategori olahraga.
Rata – rata : Hasil rata, untuk semua pengujian kategori.

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

Pada BAB IV akan dibahas implementasi dari algoritma ID3 dalam penanganan pengkategorian berita berbahasa Indonesia. Akan dijelaskan proses berjalanya program pengklasifikasian dan evaluasi data.

4.1 Lingkungan Implementasi

Pada lingkungan implementasi, terdapat dua faktor yang mempengaruhi yaitu lingkungan perangkat keras dan lingkungan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan untuk implementasi algoritma ID3 memiliki spesifikasi sebagai berikut:

1. Genuine Intel(R) CPU T2300 @ 1.66GHz (2 CPUs).
2. Memory 2038MB RAM.
3. Hardisk 60GB.
4. System Model Aspire 5570.
5. Monitor 14,1".
6. Keyboard.
7. Mouse.

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan untuk mengembangkan sistem klasifikasi *e-mail spam* dengan metode ID3 adalah sebagai berikut:

1. Operating System Windows XP Professional (5.1, Build 2600) Service Pack 3.
2. Microsoft Visual Studio C# 2010 Express Edition.

4.2 Implementasi Program

Pada permasalahan sub bab implementasi program akan di jelaskan penerapan sistem pengklasifikasian berita dengan ID3. Data terbagi dua, data *latih* dan data *testing*. Data *latih* dimasukan terlebih dulu untuk melakukan tahap pelatihan, kemudian data *testing* diproses untuk mengetahui hasil sistem. Pada sub bab ini dibahas implementasi mulai dari tahap pembersihan *tag* HTML,

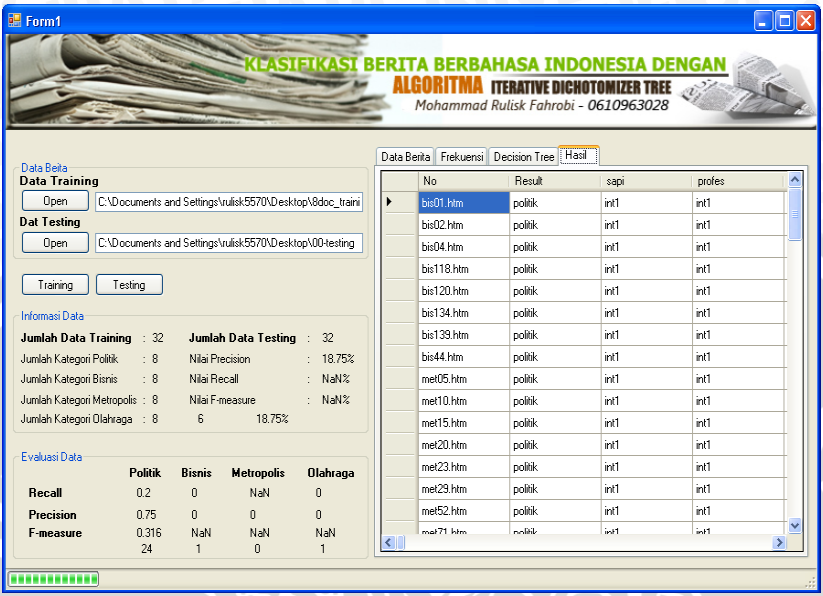
preprocessing, pembobotan, distribusi frekuensi dan klasifikasi. Berikut daftar *class* yang digunakan pada implementasi pengklasifikasin berita, dapat dilihat pada tabel 4.1 *Class* pada program.

Tabel 4.1 *Class* pada program

Nama *Class*	Keterangan
Form1.cs	Kelas utama dari program, terdapat *method preprocessing*, distribusi frekuensi dan pemanggilan *method* lain di luar kelas Form1.cs.
DecisionTree.cs	Kelas ini berisi pembentukan pohon keputusan dan pengklasifikasian.
htmlcleaner.cs	Kelas htmlcleaner.cs berisi *method* untuk mengambil isi berita dan pembersihan *tag* HTML.
Stemming.cs	Kelas stemming.cs berfungsi mencari bentuk kata dasar dari kata berimbuhan.

4.2.1 Implementasi Antarmuka Proses Data

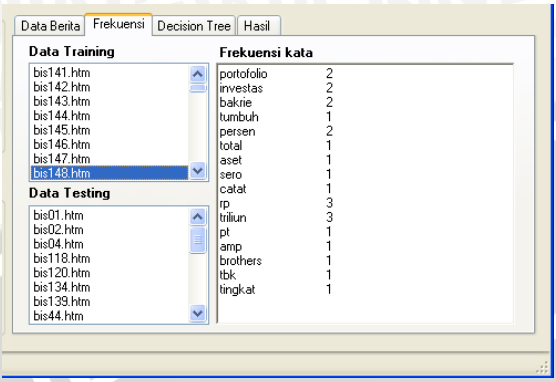
Implementasi tampilan ditunjukan pada gambar mainform 4.1. Pada tampilan awal data dimasukan kedalam sistem, brows folder berisi data berita dengan format .html. Pada tampilan tab Data Berita, terlihat list data yang sudah dimasukkan. Rich TextBox Task sebelah kanan menampilkan hasil dari proses pengambilan *body* berita dan pembersihan *tag* HTML. Pada GroupBox informasi data ditampilkan jumlah data *latih*, *testing* dan jumlah dokumen *latih* setiap kategori berita (Politik, Bisnis, Metropolis dan Olahraga).



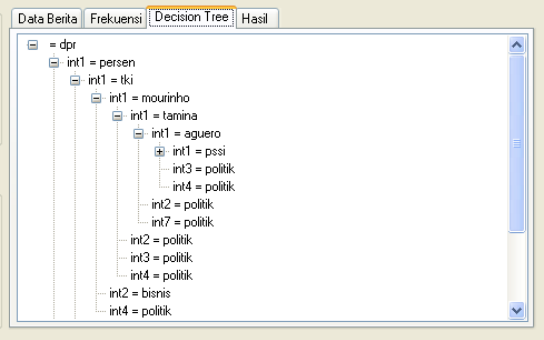
Gambar 4.1 Antarmuka mainform

Tampilan tab kedua untuk informasi frekuensi kemunculan kata pada setiap dokumen. Hasil dari *preprocessing* dan perhitungan frekuensi kata pada setiap dokumen. Ditampilkan seperti pada gambar 4.2 Antarmuka tab frekuensi.

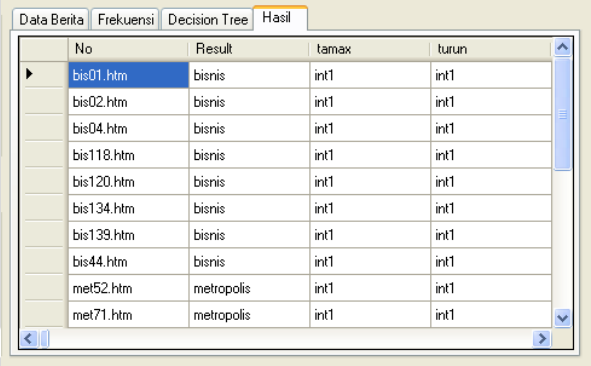
Tab ketiga menampilkan hasil dari proses *latih* berupa pohon keputusan terbentuk dari dokumen *latih*, terlihat hasil tree yang terbentuk pada gambar 4.3 Antarmuka tab *Decision tree*. Prose selanjutnya melakukan *testing* terhadap dokumen uji yang sudah dimasukkan. Testing dimulai dengan menekan tombol *testing*. Hasil dari proses *testing* ditunjukkan pada gambar 4.4 Antarmuka tab hasil klasifikasi. Pada GroubBox akan ditampilkan hasil evaluasi data dari sistem. Informasi evaluasi berupa nilai *recall*, *precision*, dan *F-measure* dari setiap kategori, serta hasil nilai rata-rata yang di dapat.



Gambar 4.2 Antarmuka tab frekuensi



Gambar 4.3 Antarmuka tab Decision Tree



Gambar 4.4 Antarmuka tab hasil klasifikasi

4.2.2 Implementasi HTML *cleaner*

Pada implementasi sub bab 4.2.2 HTML *cleaner* dilakukan proses pengambilan teks berita dari file HTML. *Source code* 4.1 berisi pemanggilan *method* yang dgunakan dalam pengambilan teks berita dan pembersihan HTML. Pada data berita diambil tiga bagian berita paragraph utama berita, judul berita dan teras berita atau deskripsi berita. Ketiga string tersebut digabungkan menjadi satu string dan kemudian melalui proses pembersihan *tag* HTML. Berikut daftar *method* yang digunakan dalam kelas htmlcleaner.cs dijelaskan pada tabel 4.2 *method* pada kelas htmlcleaner.

Tabel 4.2 *method* pada kelas htmlcleaner

Method	**Keterangan**
`public static string GetPargrap(string file)`	Digunakan untuk mengambil paragraph pada dokumen berita HTML. Ambil teks antara *tag* sampai tag .
`public static string GetJudul(string file)`	Digunakan untuk mengambil judul berita pada dokumen berita HTML. Ambil judul dari *tag* "judul\s* sampai dengan *tag* s*</h1>".
`public static string GetDes(string file)`	Digunakan untuk mengambil teras berita atau deskripsi berita pada dokumen berita HTML. Ambil *tag* deskripsi\s* sampai tag *</h2>.
`public static string BasmiHTML(string source)`	Digunakan untuk menghapus *tag* HTML dari dokumen berita HTML. Membersihkan semua karakter didalam *tag* < dan > .

Form1.cs

```
1.  private string html_cleaner(string isifile)
2.  {
3.  String paragrap =
    (htmlcleaner.GetPargrap(isifile));
4.  string judul = (htmlcleaner.GetJudul(isifile));
5.  string deskrip = (htmlcleaner.GetDes(isifile));
```

```
6.  //Menggabungkan judul berita dengn body berita.
7.  string abody = (judul + deskrip + paragrap);
8.  isifile = htmlcleaner.BasmiHTML(abody);
9.        return isifile;
10. }
```

***Source Code* 4.1** Mengambil isi berita

htmlcleaner.cs

```
1.  class htmlcleaner
2.    {
3.  // ambil Paragraph Dokumen HTML
4.  public static string GetPargrap(string file)
5.      {
6.  Match m = Regex.Match(file,
    @"<p>\s*(.+?)\s*</p>");
7.      if (m.Success)
8.        {
9.         return m.Groups[1].Value;
10.       }
11.     else
12.       {
13.         return "";
14.       }
15.     }
16. //Ambil Judul berita
17. public static string GetJudul(string file)
18.     {
19. Match m = Regex.Match(file,
    @"judul\s*(.+?)\s*</h1>");
20.      if (m.Success)
21.        {
22.         return m.Groups[1].Value;
23.        }
24.       else
25.        {
26.         return "";
27.        }
28.     }
29. //Ambil Deskripsi Brita
30. public static string GetDes(string file)
31.   {
32. Match m = Regex.Match(file,
    @"deskripsi\s*(.+?)\s*</h2>");
33.       if (m.Success)
```

```
34.             {
35.              return m.Groups[1].Value;
36.             }
37.           else
38.             {
39.              return "";
40.             }
41.     }
42.  //Pembasmi HTML
43.  public static string BasmiHTML(string source)
44.       {
45.  char[] array = new char[source.Length];
46.  int arrayIndex = 0;
47.  bool inside = false;
48.
49.       for (int i = 0; i < source.Length; i++)
50.          {
51.            char let = source[i];
52.            if (let == '<')
53.             {
54.              inside = true;
55.              continue;
56.             }
57.            if (let == '>')
58.             {
59.              inside = false;
60.              continue;
61.             }
62.            if (!inside)
63.             {
64.              array[arrayIndex] = let;
65.              arrayIndex++;
66.             }
67.          }
68.       return new string(array, 0, arrayIndex);
69.  }
```

Source Code 4.2 Pembersihan *tag* HTML

4.2.3 Implementasi *Preprocessing*

Sub bab implementasi *preprocessing* dibahas implementasi dari proses *preprocessing* yang meliputi *tokenizing*, *stopwords* dan *stemming*. Berikut daftar *method* yang digunakan dalam proses *preprocessing* terdapat pada tabel 4.3 *Method preprocessing*.

Tabel 4.3 *Method preprocessing*

Method	**Keterangan**
`private List<string> tokenizing(string isifile)`	Digunakan sebagai proses *tokenizing*, memecah kumpulan term menjadi token.
`private List<string> stopword(List<string> token)`	Digunakan sebagai proses *stopword*, dilakukan pengecekan terhadap kata yang kurang memiliki makna, untuk dilakukan penghapusan.
`private List<string> stemming(List<string> term)`	Digunakan untuk proses setemming, pengecekan kata berimbuhan.

4.2.3.1 Implementasi *Tokenizing*

Sub bab implementasi *tokenizing*, perubahan kumpulan *term* dirubah menjadi token berdasarkan *variable* pemisah. Sebelumnya semua isi file teks diubah kedalam huruf kecil. Kemudian dilakukan pendeklarasian *variable* pemisah yang terdiri dari spasi, tanda baca dan character bukan alphabet. Setiap teks hasil HTML *cleaner* akan dipecah menjadi token berdasarkan *variable* pemisah.

Form1.cs

```
1.  // Tokenizing
2.  private List<string> tokenizing(string isifile)
3.  {
4.  isifile = isifile.ToLower();
5.  char[] pemisah = new char[] { '~', '}', '{', '|',
    '*', ':', '\\', '_', '[', ']', '/', ';', '<',
    '>', ' ', '-', '=', '+', '.', ',', '"', '!',
    '?', '@', '#', '$', '%', '^', '&', '(', ')',
    '\r', '\n', '\t', '\v', '1', '2', '3', '4', '5',
    '6', '7', '8', '9', '0' };
6.  List<string> token = isifile.Split(pemisah,
    StringSplitOptions.RemoveEmptyEntries).ToList();
7.          return token;
8.  }
```

Source Code 4.3 Proses *tokenizing*

4.2.3.2 Implementasi *Stopwords*

Proses *stopwords* mengimplementasikan pembuangan kata yang dinilai kurang berarti atau tidak memiliki informasi. Kamus kata terlampir pada lampiran 2. Semua kata tersimpan pada dokumen berformat *.txt dengan nama stopwords.txt. *Filestream* membaca semua daftar kata yang berada pada file stopwords.txt. Kemudian dilakukan pengecekan setiap dokumen, dari satu dokumen dilakukan pengecekan perkata, bila kata terdapat pada stopword maka dilakukan penghapusan. Kata yang tidak terdapat pada *stopwords* akan disimpan pada *string list word*. *Sorce code* implementasi *stopwords* dapat dilihat pada *Source Code* 4.3 Proses *stopwords*.

Form1.cs

```
// Stopword
private List<string> stopword(List<string> token)
    {
     string[] stopword = new string[1000];
     int m = stopword.GetLength(0);
     int n = token.Count();
     string[] _ceksplit = new string[n];
     bool sama = true;
     int jmlh = 0;
FileStream sr = new FileStream("stopwords.txt", FileMode.Open);
StreamReader str = new StreamReader(sr);
     int a = 0;
     while (!str.EndOfStream)
     {
      stopword[a] = str.ReadLine();
      a++;
     }
      sr.Close();
      str.Close();
      for (int j = 0; j < n; j++)
      {
       sama = true;
       for (int k = 0; k < m; k++)
         {
          if (token[j] == stopword[k])
             sama = false;
         }
```

```
28.             if (sama != false)
29.             {
30.                 _ceksplit[j] = token[j];
31.                 jmlh++;
32.             }
33.         }
34. List<string> words = new List<string>();
35.         for (int l = 0; l < n; l++)
36.             {
37.                 if (_ceksplit[l] != null)
38.                   {
39.                     words.Add(_ceksplit[l]);
40.                   }
41.             }
42.                 return words;
43. }
```

***Source Code* 4.3** Proses *stopwords*

4.2.3.3 Implementasi *Stemming*

Pada tahapan *stemming*, , proses perubahan dari kata berimbuhan kedalam bentuk kata dasar menggunakan *stemming* Porter untuk bahasa Indonesia yang telah dikembangkan oleh Fadillah Talla. *Source Code* 4.4 Mengakses kelas *Stemming*, berfungsi sebagai *method* untuk mengakses kelas *stemming*.

Form1.cs

```
    // Stemming
2.  private List<string> stemming(List<string> term)
3.      {
4.      List<string> stem_term = new List<string>();
5.      Stemming stm = new Stemming();
6.          for (int q = 0; q < term.Count; q++)
7.              {
8.  stem_term.Add(Stemming.stemPrefiks1(term[q]));
9.              }
10.         return stem_term;
11. }
```

***Source Code* 4.4** Mengakses kelas *Stemming*

4.2.4 Implementasi Pembobotan

Sub bab pembobotan ini terdiri dari dua sub proses besar, sub proses pertama adalah proses frekuensi kata dan yang kedua adalah proses pembobotan dengn TF-IDF.

4.2.4.1 Implementasi Frekuensi Kata

Pada proses frekuensi kata terdiri dari 3 proses, pertama proses pengambilan satu kata dan dimasukan kedalam list `wordsNew`. Proses kedua perhitungan frekuensi kemunculan kata, dan selanjutnya proses *filtering* kata. Pada sub proses frekuensi kata memiliki dua *method*, daftar *method* dapat dilihat pada Tabel 4.4 Method frekuensi.

Tabel 4.4 *Method* frekuensi

Method	**Keterangan**
`private List<string> wordSingle(List<string> words)`	Wordsingle berfungsi untuk mengambil satu kata dari beberapa kata yang muncul.
`private List<int> frekuensi(List<string> wordsNew, List<string> words)`	Frekuensi berfungsi untuk menghitung jumlah kemunculan satu kata.

Proses pertama dalam frekuensi adalah *word single*, dilakukan pengambilan satu kata dan dimasukan kedalam string list word. Dari hasil list kata akan dihitung jumlah kemunculan setiap kata. Dua proses ini terlihat pada Source Code 4.5 Proses frekuensi kata.

Form1.cs

```
1.  // Pengambilan satu kata
2.  private List<string> wordSingle(List<string>
    words)
3.  {
4.  var sList = new ArrayList();
5.  for (int i = 0; i < words.Count; i++)
6.      {
7.       if (sList.Contains(words[i]) == false)
8.           {
9.           sList.Add(words[i]);
10.          }
```

```
11.      }
12. var wordsNew_ = sList.ToArray();
13. string[] wordsNew = new
    string[wordsNew_.Length];
14. for (int i = 0; i < wordsNew_.Length; i++)
15.     {
16.      wordsNew[i] = (string)wordsNew_[i];
17.     }
18. return wordsNew.ToList();
19.     }
20. // frekuensi kata
21. private List<int> frekuensi(List<string>
    wordsNew, List<string> words)
22.     {
23. int[] wordsFrek = new int[wordsNew.Count];
24.     for (int i = 0; i < wordsNew.Count; i++)
25.         {
26.          for (int j = 0; j < words.Count; j++)
27.              {
28.          if
    (words[j].Equals((string)wordsNew[i]))
29.              {
30.               wordsFrek[i]++;
31.              }
32.           }
33.      }
34. return wordsFrek.ToList();
35. }
```

***Source Code* 4.5** Proses frekuensi kata.

Selanjutnya hasil frekuensi kata dilakukan *filtering*. Kata yang dihapus adalah kata dengan kemunculan kurang dari tiga. Berikut proses *filtering* kata terdapat pada Source Code 4.6 Proses filter kata.

```
1  for (int j = 0; j < pro_wsgl.Count; j++)
2     {
3        if (pro_frek[j] > 2)
4          {
5        for (int k = 0; k < nTerm; k++)
6            {
7         if (!term.Contains(pro_wsgl[j]))
8           {
9         term.Add(pro_wsgl[j]);
10        dataLatih[i, nTerm] = pro_frek[j];
```

```
11          nTerm++;
12        break;
13          }
14        if (term[k].Equals(pro_wsgl[j]))
15        {
16      dataLatih[i, k] = pro_frek[j];
17    }
18  }
19      if (nTerm == 0)
20    {
21    term.Add(pro_wsgl[j]);
22    dataLatih[i, nTerm] = pro_frek[j];
23    nTerm++;
24      }
25    }
26  }
```

***Source Code* 4.6** Proses filter kata

4.2.4.2 Implementasi Pembobotan

Proses selanjutnya setelah dilakukan *filtering* dan data dismpan pada list `pro_frek` dilakukan pengambilan *feature*. *Feature* yang diambil berupa kemunculan kata dengan metode TF-IDF. Proses pembobotan pertama dilakukan pencarian nilai dokumen frekuensi (DF). Nilai DF didapat dari jumah adanya suatu kata dalam satu dokumen. Selanjutnya dilakukan perhitungan idf dan disimpan pada `dataLatih`. Dilakukan perhitungan bobot kata, dengan mengkalikan hasil idf setiap term ke semua *term* frekuensi pada tiap dokumen. Perhitungan pembobotan TF-IDF dpat dilihat pada Source Code 4.7 Proses pembobotan.

Form1.cs

```
1.  //TF-IDF
2.  int df = 0;
3.  double[,] dataPembobotan = new
    double[lb_teks.Items.Count, nTerm];
4.  for (int i = 0; i < nTerm; i++)
5.      {
6.    for (int j = 0; j < lb_teks.Items.Count; j++)
7.        {
8.        if (dataLatih[j, i] != 0)
9.        {
10.         df++;
```

```
11.         }
12.     }
13. dataLatih[lb_teks.Items.Count, i] = df;
14. df = 0;
15. dataLatih[(lb_teks.Items.Count + 1), i] =
    Math.Round(Math.Log10(lb_teks.Items.Count /
    dataLatih[lb_teks.Items.Count, i]), 4,
    MidpointRounding.AwayFromZero);
16. idf.Add(dataLatih[(lb_teks.Items.Count + 1),
    i]);
17.   for (int k = 0; k < lb_teks.Items.Count; k++)
18.         {
19.   dataPembobotan[k, i] = dataLatih[k, i] *
    dataLatih[(lb_teks.Items.Count + 1), i];
20.             }
21. }
22.
```

***Source Code* 4.7** Proses pembobotan.

4.2.5 Implementasi Transformasi Data

Transformasi data digunakan untuk melakukan generalisasi data numerik menjadi data kategori. Terdapat dua proses, proses pertama adalah proses distribusi frekuensi. Variabel `double max` digunakan mencari data terbesar dan `double min` digunakan untuk menyimpan data terkecil dari hasil pembobotan. Setelah mengetahui nilai max dan min dari bobot, dihitung *range* dari data dan disimpan pada variabel `double range`. Selanjutnya mencari kelas yang terbentuk dengan rumus Sturges (2.4) dapat dilihat pada *Source Code* 4.7 Proses pembobotan pada bagian mencari jumlah kelas. Selanjutnya dilakukan pembentukan label kelas dan menghitung batas atas dan bawah kelas.

Form1.cs

```
1 // Distribusi Frekuensi
2 toolStripProgressBar1.Value += 20;
3 double max =
  dataPembobotan.Cast<double>().Max();
4 double min =
  dataPembobotan.Cast<double>().Min();
5 double range = max - min;
6 // mencari jumlah kelas
7 kelas = 1 + (3.3 *
```

```
    Math.Log10(dataPembobotan.Cast<double>().Count()
    ));
8   kelas = Math.Ceiling(kelas);
9   double pInterval = Math.Round(range / kelas, 4,
    MidpointRounding.AwayFromZero);
10  inv _interval = new inv();
11
12  for (int i = 0; i <= kelas; i++)
13    {
14      if (i == 0)
15        {
16          _interval.a = min;
17          _interval.b = _interval.a + pInterval;
18          interval.Add(_interval);
19        }
20       else
21      {
22          _interval.a = interval[i - 1].b + 0.0001;
23          _interval.b = _interval.a + pInterval;
24         interval.Add(_interval);
25      }
26  }
27  //transformasi bobot kata
28  string[,] dataKategorisasi = new
    string[lb_teks.Items.Count, nTerm];
29  //Memerikan label pada kelas tujuan
30  var hasil = new List<bool>();
31  for (int i = 0; i < lb_teks.Items.Count; i++)
32      {
33       for (int j = 0; j <= nTerm; j++)
34          {
35
36            if (j == nTerm)
37              {
38  if (lb_teks.Items[i].ToString().Contains("pol"))
39       {
40        sb.Append("politik");
41       }
42  if (lb_teks.Items[i].ToString().Contains("bis"))
43       {
44        sb.Append("bisnis");
45       }
46  if (lb_teks.Items[i].ToString().Contains("met"))
47       {
48        sb.Append("metropolis");
```

```
49          }
50      if (lb_teks.Items[i].ToString().Contains("ola"))
51          {
52           sb.Append("olahraga");
53          }
54      }
55
56           else
57          {
58      for (int k = 0; k <= kelas; k++)
59          {
60      if ((dataPembobotan[i, j] >= interval[k].a) &&
        (dataPembobotan[i, j] <= interval[k].b) && (k !=
        kelas))
61      { dataKategorisasi[i, j] = (string)("int" + (k +
        1)); }
62      if ((dataPembobotan[i, j] >= interval[k].a) &&
        (k == kelas))
63      { dataKategorisasi[i, j] = (string)("int" + (k +
        1)); }
64          }
65      sb.Append(dataKategorisasi[i, j] + ",");
66              }
67      }
68      if (i < (lb_teks.Items.Count - 1))
        sb.Append("\n");
69      }
```

***Source Code* 4.8** Proses transformasi data.

Setelah terbentuk label dari setiap *range* bobot, kemudian data ditransformasikan berdasarkan distribusi yang sudah dihasilkan. Dilakukan transformasi boot keseluruh dokumen latih dan uji. Pada proses ini juga dilakukan penamaan kelas tujuan. Label atribut tujuan terdiri dari empat kategori yaitu politik, bisnis, metropolis, dan olahraga. Penamaan kategori ini dilihat dari nama file dokumen *latih* yang dimasukkan dengan menghitung tiga karakter pertama pada nama dokumen. Misalnya untuk label kategori politik setiap nama file dengan nama depan `"pol"` akan dikategorikan sebagai kelas politik.

4.2.6 Implementasi Pembentukan Pohon Keputusan

Pada sub bab 4.2.6 dilakukan pembahasan implementasi pembentukan pohon keputusan berdasarkan data *latih*, berikut beberapa *method* yang digunakan dalam kelas decisionTree terdpat pada tabel 4.5 Method pada kelas DecisionTree.

Tabel 4.5 *Method* pada kelas DecisionTree

Method	**Keterangan**
`private int countTotalPolitik (DataTable samples)`	Menghitung jumlah kategori politik pada atribut tujuan.
`private int countTotalBisnis (DataTable samples)`	Menghitung jumlah kategori bisnis pada atribut tujuan.
`private int countTotalMetropolis (DataTable samples)`	Menghitung jumlah kategori metropolis pada atribut tujuan.
`private int countTotalOlahraga (DataTable samples)`	Menghitung jumlah kategori olahraga pada atribut tujuan.
`private double getCalculatedEntropy (int politik, int bisnis, int metropolis, int olahraga)`	Menghitung nilai entropy pada kelas target atribut.
`private void getValuesToAttribute (DataTable samples, TreeAttribute attribute, string value, out int politik, out int bisnis, out int metropolis, out int olahraga)`	Melakukan perhitungan *Information gain* untuk setiap atribut tujuan.
`private double gain (DataTable samples, TreeAttribute attribute)`	Menghitung *information gain* untuk atribut selain kelas tujuan.
`private TreeAttribute getBestAttribute (DataTable samples, TreeAttributeCollection attributes)`	Menentukan *information gain* terbaik.
`private TreeNode internalMountTree`	Pembentukan pohon

(DataTable samples, string targetAttribute, TreeAttributeCollection attributes)	keputusan.

Ada beberapa proses dalam melakukan pembentukan pohon keputusan, pertama melakukan perhitungan kelas atribut tujuan untuk setiap kaegoti. `countTotalPolitik` digunakan sebagai menghitung jumlah atribut tujuan politik. Dilakukan pengecekan, jika didapat kata Politik didalam tabel atribut tujuan akan ditambahkan satu kedalam `countTotalPolitik`. Begtu juga dengan kategori lainya, dilakukan pengecekan smpai ketegori terakhir yaitu olahraga. Implementasi proses pengecekan dapat dilihat pada *source code* 4.8 Proses pengecekan atribut tujuan.

DecisionTree.cs

```
1  // Menghitung jumlah kategori tujuan
2  private int countTotalPolitik(DataTable samples)
3          {
4           int result = 0;
5
6      foreach (DataRow aRow in samples.Rows)
7              {
8  if
   (aRow[mTargetAttribute].ToString().ToUpper().Trim
   () == "POLITIK")
9           result++;
10             }
11
12             return result;
13          }
14
15 private int countTotalBisnis(DataTable samples)
16          {
17             int result = 0;
18
19         foreach (DataRow aRow in samples.Rows)
20             {
21 if
   (aRow[mTargetAttribute].ToString().ToUpper().Trim
   () == "BISNIS")
22             result++;
23             }
```

```
24
25              return result;
26          }
27
28  private int countTotalMetropolis(DataTable
    samples)
29          {
30              int result = 0;
31
32              foreach (DataRow aRow in
    samples.Rows)
33              {
34                  if
    (aRow[mTargetAttribute].ToString().ToUpper().Trim
    () == "METROPOLIS")
35                      result++;
36              }
37
38              return result;
39          }
40
41  private int countTotalOlahraga(DataTable samples)
42          {
43          int result = 0;
44
45      foreach (DataRow aRow in samples.Rows)
46              {
47  if
    (aRow[mTargetAttribute].ToString().ToUpper().Trim
    () == "OLAHRAGA")
48              result++;
49              }
50
51              return result;
52          }
```

***Source Code* 4.9** Proses pengecekan atribut tujuan.

Setelah didapat jumlah atribut tiap kategori dilakukan perhitungan *entropy* kelas tujuan. *Method* `getCalculatedEntropy` berfungsi melakukan perhitungan *entropy* dari setiap atribut tujuan, hasil perhitungan entropy akan disimpan `ratioPolitik` untuk perhitungan entropy kelas tujuan politik. Implementasi program

dapat dilihat pada source code 4.10 Proses perhitungan entropy atribut tujuan.

DecisionTree.cs

```
1   private double getCalculatedEntropy(int politik,
    int bisnis, int metropolis, int olahraga)
2           {
3   int total = politik + bisnis + metropolis +
    olahraga;
4     double ratioPolitik = (double)politik / total;
5       double ratioBisnis = (double)bisnis / total;
6   double ratioMetropolis = (double)metropolis /
    total;
7   double ratioOlahraga = (double)olahraga / total;
8
9           if (ratioPolitik != 0)
10  ratioPolitik = -(ratioPolitik) *
    System.Math.Log(ratioPolitik, 2);
11          if (ratioBisnis != 0)
12  ratioBisnis = -(ratioBisnis) *
    System.Math.Log(ratioBisnis, 2);
13  if (ratioMetropolis != 0)
14  ratioMetropolis = -(ratioMetropolis) *
    System.Math.Log(ratioMetropolis, 2);
15  if (ratioOlahraga != 0)
16  ratioOlahraga = -(ratioOlahraga) *
    System.Math.Log(ratioOlahraga, 2);
17
18  double result = ratioPolitik + ratioBisnis +
    ratioMetropolis + ratioOlahraga;
19
20              return result;
21      }
```

***Source Code* 4.10** Proses perhitungan *entropy* atribut tujuan.

Setelah diketahui hasil *entropy* atribut tujuan, dilakuan perhitungan *Information Gain* `getValuesToAttribute`, Selain dilakukan perhitungan *informasi gain* pada setiap atribut tujuan dilakukan pula perhitungan IG setiap atribut dengan *method* `gain`, semua atribut dan record pada atribut akan dilakukan perhitungan. Proses perhitungan *information gain* untuk atribut tujuan, atribut selain

tujuan dan data record dapat dilihat pada source code 4.9 Menghitung IG tujuan dan IG setiap atribut.

DecisionTree.cs

```
1   // IG tiap tujuan
2   private void getValuesToAttribute(DataTable
    samples, TreeAttribute attribute, string value,
    out int politik, out int bisnis, out int
    metropolis, out int olahraga)
3   {
4      politik = 0;
5      bisnis = 0;
6      metropolis = 0;
7      olahraga = 0;
8
9   foreach (DataRow aRow in samples.Rows)
10    {
11  if (((string)aRow[attribute.AttributeName] ==
    value))
12     {
13       if
    (aRow[mTargetAttribute].ToString().Trim().ToUpper
    () == "POLITIK") politik++;
14  if(aRow[mTargetAttribute].ToString().Trim().ToUpp
    er() == "BISNIS")bisnis++;
15  if
    (aRow[mTargetAttribute].ToString().Trim().ToUpper
    () == "METROPOLIS") metropolis++;
16  if
    (aRow[mTargetAttribute].ToString().Trim().ToUpper
    () == "OLAHRAGA") olahraga++;
17                  }
18              }
19          }
20  // IG tiap atribut
21  private double gain(DataTable samples,
    TreeAttribute attribute)
22  {
23  PossibleValueCollection values =
    attribute.PossibleValues;
24   double sum = 0.0;
25
26   for (int i = 0; i < values.Count; i++)
27      {
```

```
28      int politik, bisnis, metropolis, olahraga;
29
30      politik = bisnis = metropolis = olahraga = 0;
31
32  getValuesToAttribute(samples, attribute,
    values[i], out politik, out bisnis, out
    metropolis, out olahraga);
33
34  double entropy = getCalculatedEntropy(politik,
    bisnis, metropolis, olahraga);
35  sum += -(double)(politik + bisnis + metropolis +
    olahraga) / mTotal * entropy;
36          }
37    return mEntropySet + sum;
38  }
```

***Source Code* 4.11** Menghitung IG tujuan dan IG setiap atribut.

Dari hasil perhitungan *information gain* antara atribut tujuan dan atribut setiap kelas, dilakukan perbandingan antara satu kelas atribut dengan atribut lainya. Kelas atribut dengan IG terbaik digunakan sebagai *root* utama dalam decision tree. Hasil IG terbaik disimpan pada variabel `double maxGain = 0.0;`. Keseluruhan proses dapat dilihat pada source code 4.10 Menentukan atribut terbaik.

DecisionTree.cs

```
1   private TreeAttribute getBestAttribute(DataTable
    samples, TreeAttributeCollection attributes)
2       {
3        double maxGain = 0.0;
4        TreeAttribute result = null;
5
6   foreach (TreeAttribute attribute in attributes)
7           {
8   double aux = gain(samples, attribute);
9            if (aux > maxGain)
10              {
11                 maxGain = aux;
12                 result = attribute;
13              }
14          } return result; }
```

***Source Code* 4.12** Menentukan atribut terbaik.

Proses terakhir dari klasifikasi adalah pembentukan pohon keputusan, `DecisionTreeImplementation sam = new DecisionTreeImplementation()` memanggil kelas `DecisionTreeImplementation` dan diganti dengn variabel sam. Pada kelas DicisionTreeImplementation terdapat tiga pemanggilan proses penting, melakukan pembacaan data tabel yang berada di RawDataSorce, pengambilan atribut dan pemanggilan *method* perhitungan id3 pada kelas DecisionTree. Proses ditunujukan pada *source code 4.11* berikut ini.

DecisionTree.cs

```
1   public System.Windows.Forms.TreeNode
    GetTree(string sourceFile)
2           {
3            _sourceFile = sourceFile;
4   //mebaca baris dan kolom
5   RawDataSource samples = new
    RawDataSource(_sourceFile);
6   // mengambil atribut
7   TreeAttributeCollection attributes =
    samples.GetValidAttributeCollection(); //term
8   //memanggil perhitungan ID3
9   DecisionTree id3 = new DecisionTree();
10  TreeNode root = id3.mountTree(samples, "result",
    attributes);
11
12        return PrintNode(root, "");
13
14  }
```

***Source Code* 4.13** Kelas DecisionTreeImplementation.

Selanjutnya dilakukan pembentukan pohon keputusan sesuai data yang telah didapat pada proses pada kelas DecisinTreeImplementation. *Method* internalMountTree berguna dalam pembentukan pohon keputusan. Proses pertama dilakukan pembacaan atribut target dari setiap kategori politik, bisnis, metropolis dan olahraga. Pada variabel `mTotal = samples.Rows.Count` dilakukan perhitungan jumlah data record pada suatu atribut. Variabel mTargetAttribute digunakan menyimpan jumlah target atribut. Variabel mTotalPolitik digunakan sebagai

menghitung banyak data record yang berada pada target kategori politik, begtu juga dengan variabel mTotalBisnis, mTotalMetropolis dan mTotalOlahraga. Variabel mEntropySet berfungsi menghitung total seluruh *entropy*.

DecisionTree.cs

```
1   private TreeNode internalMountTree(DataTable
    samples, string targetAttribute,
    TreeAttributeCollection attributes)
2           {
3   if (samplesArePolitik(samples, targetAttribute)
    == true)
4   return new TreeNode(new
    OutcomeTreeAttribute("politik"));
5
6   if (samplesAreBisnis(samples, targetAttribute) ==
    true)
7   return new TreeNode(new
    OutcomeTreeAttribute("bisnis"));
8
9   if (samplesAreMetropolis(samples,
    targetAttribute) == true)
10  return new TreeNode(new
    OutcomeTreeAttribute("metropolis"));
11
12  if (samplesAreOlahraga(samples, targetAttribute)
    == true)
13  return new TreeNode(new
    OutcomeTreeAttribute("olahraga"));
14
15  if (attributes.Count == 0)
16  return new TreeNode(new
    OutcomeTreeAttribute(getMostCommonValue(samples,
    targetAttribute)));
17
18  mTotal = samples.Rows.Count;
19  mTargetAttribute = targetAttribute;
20  mTotalPolitik = countTotalPolitik(samples);
21  mTotalBisnis = countTotalBisnis(samples);
22  mTotalMetropolis = countTotalMetropolis(samples);
23  mTotalOlahraga = countTotalOlahraga(samples);
24
25  mEntropySet = getCalculatedEntropy(mTotalPolitik,
    mTotalBisnis, mTotalMetropolis, mTotalOlahraga);
26  //menghitung atribut terbaik
```

```
27  TreeAttribute bestAttribute =
    getBestAttribute(samples, attributes);
28  // atribut terbaik ditempatkn sebagai root
29  TreeNode root = new TreeNode(bestAttribute);
30
31     if (bestAttribute == null)
32         return root;
33
34         DataTable aSample = samples.Clone();
35
36         foreach (string value in
    bestAttribute.PossibleValues)
37                 {
38
39          aSample.Rows.Clear();
40
41          DataRow[] rows =
    samples.Select(bestAttribute.AttributeName + " =
    " + "'" + value + "'");
42
43            foreach (DataRow row in rows)
44          {
45  aSample.Rows.Add(row.ItemArray);
46          }
47
48  TreeAttributeCollection aAttributes = new
    TreeAttributeCollection();
49
50  for (int i = 0; i < attributes.Count; i++)
51      {
52  if (attributes[i].AttributeName !=
    bestAttribute.AttributeName)
53  aAttributes.Add(attributes[i]);
54       }
55
56  if (aSample.Rows.Count == 0)
57         {
58  return new TreeNode(new
    OutcomeTreeAttribute(getMostCommonValue(aSample,
    targetAttribute)));
59          }
60           else
61       {
62         DecisionTree dc3 = new DecisionTree();
63  TreeNode ChildNode = dc3.mountTree(aSample,
    targetAttribute, aAttributes);
```

```
64  root.AddTreeNode(ChildNode, value);
65          }
66    }
67
68  return root;
69  }
```

***Source Code* 4.14** Pembentukan pohon keputusan.

Melakukan pengambilan atribut terbaik dan ditempatkan sebagai root. Melakukan pengecekan terhadap data sampel atribut yang digunakan sebagai root. Pada setiap cabang root dicek terhadap target atribut, jika masi terdapat dua kategori dalam satu cabang dilakukan kembali perhitungan IG untuk mencari atribut terbaik dan digunakan sebagain root dari cabang tersebut. Proses ini terus berlangsung sampai cabang memiliki satu target atribut.

4.2.7 Implementasi Keputusan ID3

Pada proses keputusan ID3 atau pengklasifikasian oleh sistem data *testing* akan melalu proses yang sama dengan data *latih* sampai pada proses transformasi data. Data tabel pada dataGrid *testing* akan melalu proses pengecekan perdokumen, dari satu dokumen dilakukan pengecekan tiap kata. Pada proses ini, data *latih* melakukan pengecekan langsung terhadap hasil tree yang sudah terbentuk. Dilakukan penelusuran perkata sampai mendapatkan kategori sesuai dengan *decision tree* yang telah dihasilkan pada proses latih .

DecisionTree.cs

```
1   public string PrintResult(TreeNode root,
    List<string> dataInv, List<string> term)
2   {
3     if (root.Attribute.PossibleValues == null)
4       {
5         result += root.Attribute;
6       }
7         else
8       {
9         for (int i = 0; i < term.Count; i++)
10        {
11  if
    (term[i].Equals(root.Attribute.AttributeName))
12          {
13
```

```
14    if (root.Attribute.PossibleValues != null)
15    {
16    for (int j = 0; j < root.Attribute.PossibleValues.Count; j++)
17    {
18    if (dataInv[i].Equals(root.Attribute.PossibleValues[j]))
19    {
20    TreeNode childNode = root.GetChildByBranchName(root.Attribute.PossibleValues[j]);
21    if (childNode.Attribute != null)
22    {
23    result = PrintResult(childNode, dataInv, term);
24    }
25    }
26    }
27    if (result == String.Empty)
28    return "result";
29    }
30    }
31    }
32    }
33
34    return result;
35    }
```

***Source Code* 4.15** Pengklasifikasian.

Data yang dihasilkan pada proses ini akan ditampilkan pada datagrit tabel di *tab* Hasil, dapat dilihat pada Gambar 4.4 Antarmuka *tab* hasil klasifikasi.

4.2.8 Implementasi Evaluasi Data

Untuk menguji kinerja sistem dalam melakukan pengklasifikasian data dilakukan pengujian dengan metode *precison, recall* dan *f-measure* sesuai pada perancangan evaluasi data sebelumnya. Nilai *f-measure* didapat dari hasil perhitungan *precision*

dan *recall. Precision* dan *recall* didapat dari perhitungan jumlah *true positive, false negative* dan *false positive* dari setiap kategori.

Perhitungan pertama dengan menghitung jumlah *true positive* pada setiap kategori. Variabel `TPpol` digunakan untuk menghitung jumlah *true positive* pada kategori politik. Jika pada data dokumen nyata termasuk politik dan pada kelasifikasi sistem dikategorikan politik, maka ditambahkan satu pada varaiabel `TPpol`. Proses yang sama digunakan dalam menghitung *true positive* untuk kategori lainnya. *Souce code* 4.13 implmentasi dari perhitungan *true posirive*.

```
1  //Variabel true positive, false positife dan Fale Negatife
2  int TPpol = 0, TPbis = 0, TPmet = 0, TPola = 0,
3      FPpol = 0, FPbis = 0, FPmet = 0, FPola = 0,
4      FNpol = 0, FNbis = 0, FNmet = 0, FNola = 0;
5  foreach (var kata in term)
6        {
7         dataGridView1.Columns.Add(kata, kata);
8        }
9  for (int i = 0; i < lb_frekTest.Items.Count; i++)
10   {
11    toolStripProgressBar1.Value += 100 / lb_frekTest.Items.Count;
12
13 dataGridView1.Rows.Add();
14 dataGridView1[0, i].Value = lb_teksTest.Items[i].ToString();
15
16 var disFrekTesting = DisFrekTesting(i);
17
18 for (int j = 0; j < disFrekTesting.Count; j++)
19    {
20 dataGridView1[2 + j, i].Value = disFrekTesting[j];
21    }
22 if (lb_frekTest.Items[i].ToString().Contains(dataGridView1["Result",
   i].Value.ToString().Substring(0,3)))
23    jumCocok++;
24 if (lb_frekTest.Items[i].ToString().Contains("pol") &&
   lb_frekTest.Items[i].ToString().Contains(dataGridView1["Result",
```

```
    i].Value.ToString().Substring(0,3)))
25      TPpol++;
26  if
    (lb_frekTest.Items[i].ToString().Contains("bis")
    &&
    lb_frekTest.Items[i].ToString().Contains(dataGridV
    iew1["Result",     i].Value.ToString().Substring(0,
    3)))
27      TPbis++;
28  if
    (lb_frekTest.Items[i].ToString().Contains("met")
    &&
    lb_frekTest.Items[i].ToString().Contains(dataGridV
    iew1["Result",     i].Value.ToString().Substring(0,
    3)))
29      TPmet++;
30  if
    (lb_frekTest.Items[i].ToString().Contains("ola")
    &&
    lb_frekTest.Items[i].ToString().Contains(dataGridV
    iew1["Result",     i].Value.ToString().Substring(0,
    3)))
31      TPola++;
```

***Source Code* 4.16** Menghitung *true positive*.

Langkah kedua melakukan perhitungan terhadap jumlah *false positive*. Misalkan dalam melakukan perhitungan *false positive* untuk politik hasil akan disimpan pada variabel FPpol. Pengecekan dilakukan dari sudut pandang sistem, bila pada sistem dikategorikan politik tapi pada manual pengkategorian bukan politik, maka ditambahkan satu poin pada variabel FPpol. Begitu juga pada kategori lainnya.

```
1   for (int k = 0; k < lb_frekTest.Items.Count; k++)
2               {
3                   //toolStripProgressBar1.Value    +=
    100 / lb_frekTest.Items.Count;
4
5                   dataGridView1.Rows.Add();
6                   dataGridView1[0,     k].Value     =
    lb_teksTest.Items[k].ToString();
7
8                   var         disFrekTesting         =
```

```
        DisFrekTesting(k);
9
10                  for (int l = 0; l < disFrekTesting.Count; l++)
11                  {
12                      dataGridView1[2 + l, k].Value = disFrekTesting[l];
13                  }
14                  if (lb_frekTest.Items[k].ToString().Contains(dataGridView1["Result", k].Value.ToString().Substring(0, 3)))
15                      jumFP++;
16                  if (!lb_frekTest.Items[k].ToString().Contains("pol") && dataGridView1["Result", k].Value.ToString().Substring(0, 3).Contains("pol"))
17                      FPpol++;
18                  if (!lb_frekTest.Items[k].ToString().Contains("bis") && dataGridView1["Result", k].Value.ToString().Substring(0, 3).Contains("bis"))
19                      FPbis++;
20                  if (!lb_frekTest.Items[k].ToString().Contains("met") && dataGridView1["Result", k].Value.ToString().Substring(0, 3).Contains("met"))
21                      FPmet++;
22                  if (!lb_frekTest.Items[k].ToString().Contains("ola") && dataGridView1["Result", k].Value.ToString().Substring(0, 3).Contains("ola"))
23                      FPola++;
24              }
```

***Source Code* 4.17** Menghitung *false positive*.

Langkah ketiga melakukan perhitungan terhadap pengkategorian manual benar tapi dikategorikan sistem salah, atau pencarian nilai *false neative*. *False negative* politik disimpan pada variabel FNpol, jika pada pengkategorian manual dikategorikan

sebagai politik tetapi oleh sistem dimasukkan dalam kategori lain maka ditambahkan satu pada variabel FNpol. Kondisi seperti kategori politik juga berlaku pada kategori bisnis, metropolis dan olahraga.

```
1   for (int m = 0; m < lb_frekTest.Items.Count; m++)
2       {
3
4
5    dataGridView1.Rows.Add();
6   dataGridView1[0,              m].Value              =
    lb_teksTest.Items[m].ToString();
7
8   var disFrekTesting = DisFrekTesting(m);
9
10  for (int n = 0; n < disFrekTesting.Count; n++)
11      {
12  dataGridView1[2 + n, m].Value = disFrekTesting[n];
13      }
14  if
    (lb_frekTest.Items[m].ToString().Contains(dataGrid
    View1["Result",    m].Value.ToString().Substring(0,
    3)))
15      jumFN++;
16  if
    (lb_frekTest.Items[m].ToString().Contains("pol")
    &&                          !dataGridView1["Result",
    m].Value.ToString().Substring(0,
    3).Contains("pol"))
17      FNpol++;
18  if
    (lb_frekTest.Items[m].ToString().Contains("bis")
    &&                          !dataGridView1["Result",
    m].Value.ToString().Substring(0,
    3).Contains("bis"))
19      FNbis++;
20  if
    (lb_frekTest.Items[m].ToString().Contains("met")
    &&                          !dataGridView1["Result",
    m].Value.ToString().Substring(0,
    3).Contains("met"))
21      FNmet++;
22  if
    (lb_frekTest.Items[m].ToString().Contains("ola")
    &&                          !dataGridView1["Result",
```

```
    m].Value.ToString().Substring(0,
    3).Contains("ola"))
23      FNola++;
24  }
```

***Source Code* 4.18** Menghitung *false negative*.

Pada tahap keempat dilakukan perhitungan *recall*, *precision* dan *f-measure* untuk setiap kategori sesuai perhitungan *true positive*, *false positive* dan *false negative*. Rumus perhitungan telah dibahas pada bab 2 pada bagian evaluasi data. Setelah diketahui hasil *recall*, *precision* dan *f-measure* untuk setiap kategori kemudian dilakukan perhitungan rata-rata ketiganya. Berikut perhitungan ada pada *sorce code* 4.16.

```
1   //recall
2   double recalpol = Math.Round((double)TPpol /
    (double)(TPpol + FPpol), 3,
    MidpointRounding.AwayFromZero);
3   double recalbis = Math.Round((double)TPbis /
    (double)(TPbis + FPbis), 3,
    MidpointRounding.AwayFromZero);
4   double recalmet = Math.Round((double)TPmet /
    (double)(TPmet + FPmet), 3,
    MidpointRounding.AwayFromZero);
5   double recalola = Math.Round((double)TPola /
    (double)(TPola + FPola), 3,
    MidpointRounding.AwayFromZero);
6   recpol.Text = recalpol.ToString();
7   recbis.Text = recalbis.ToString();
8   recmet.Text = recalmet.ToString();
9   recola.Text = recalola.ToString();
10  //precision
11  double precipol = Math.Round((double)TPpol /
    (double)(TPpol + FNpol), 3,
    MidpointRounding.AwayFromZero);
12  double precibis = Math.Round((double)TPbis /
    (double)(TPbis + FNbis), 3,
    MidpointRounding.AwayFromZero);
13  double precimet = Math.Round((double)TPmet /
    (double)(TPmet + FNmet), 3,
    MidpointRounding.AwayFromZero);
14  double preciola = Math.Round((double)TPola /
    (double)(TPola + FNola), 3,
    MidpointRounding.AwayFromZero);
```

```
15              prepol.Text = precipol.ToString();
16              prebis.Text = precibis.ToString();
17              premet.Text = precimet.ToString();
18              preola.Text = preciola.ToString();
19  //F-measure
20  double fmpol_ = Math.Round((double)2 * (recalpol * precipol) / (double)(recalpol + precipol), 3, MidpointRounding.AwayFromZero);
21  double fmbis_ = Math.Round((double)2 * (recalbis * precibis) / (double)(recalbis + precibis), 3, MidpointRounding.AwayFromZero);
22  double fmmet_ = Math.Round((double)2 * (recalmet * precimet) / (double)(recalmet + precimet), 3, MidpointRounding.AwayFromZero);
23  double fmola_ = Math.Round((double)2 * (recalola * preciola) / (double)(recalola + preciola), 3, MidpointRounding.AwayFromZero);
24              fmpol.Text = fmpol_.ToString();
25              fmbis.Text = fmbis_.ToString();
26              fmmet.Text = fmmet_.ToString();
27              fmola.Text = fmola_.ToString();
28  // Rata - rata recall, precision dan F-Measure
29  double rtpreci_ = ((double)(precipol + precibis + precimet + preciola) / 4) * 100;
30  double rtrecall_ = ((double)(recalpol + recalbis + recalmet + recalola) / 4) * 100;
31  double rtfm_ = ((double)(fmpol_ + fmbis_ + fmmet_ + fmola_) / 4) * 100;
32  rtpreci.Text = rtpreci_.ToString() + "%";
33  rtrecall.Text = rtrecall_.ToString() + "%";
34  rtfms.Text = rtfm_.ToString() + "%";
```

***Source Code* 4.19** Menghitung *recall*, *precision* dan *fmeasure*.

4.4 Skenario Pengujian

Pengujian dilakukan sesuai pada skenario pengujian pada sub bab 3.6 Metode Pengujian. Data dibagi menjadi dua jenis data latih dan data uji. Untuk data latih dilakukan beberapa perbandingan jumlah data dengan data uji. Perbandingan jumlah data latih dan uji sesuai dengan skenario pengujian pada metode pengujian disub bab 3.6.

4.4.1 Data pengujian

Total keseluruhan data berita bulan Juli 2011 adalah 640 dokumen berita, dengan 160 dokumen berita pada kategori politik, bisinis, metropolis dan olahraga. Dokumen berita Data *testing* berjumlah 32 dokumen berita, pengujian tetap menggunakan data yang sama untuk mendapatkan keterkaitan antar pengujian.

4.4.2 Lingkungan Pengujian

Jumlah data uji sebanyak 32 dokumen berita, dengan masing-masing terdiri dari 8 dokumen berita untuk setiap kategorinya. Pengujian data dilakukan sampai perbandingan 80 % data *latih* 20 % data *testing*, dengan komposisi data berita 128 dokumen berita sebagai *latih* dan 32 dokumen berita sebagai *testing.* Perbandingn komposisi data tiap pengujian ada pada tabel 4.6.

Tabel 4.6 Perbandingan jumlah data pengujian.

Prosentase data (%)		Data Berita	
Latih	*Testing*	*Latih*	*Testing*
50	50	32	32
60	40	48	32
66.7	33.3	64	32
70	30	72	32
80	20	128	32

Pengujian hanya dilakukan sampai 80% *latih*, pada saat dilakukan pengujian dengan perbandingan data 90% *latih* pengujian data tidak dapat dilakukan karna keterbatasan perangkat keras.

4.4.3 Hasil Pengujian

Setiap perbandingan data *latih* dilakukan pengujian sebanyak tiga kali, kemudian dicari nilai rata-rata untuk setiap *precision*, *recall* dan *f-measure*. Hasil perhitungan ditampilkan pada tabel 4.7. Dari tabel hasil pengujian 4.7 dapat dilihat baik prosentase hasil *precision, recall* dan *F-measure* semakin meningkat dengan bertambahnya jumlah data latih.

Tabel 4.7 Hasil perhitungan rata-rata *precision, recall* dan *f-measure.*

Data *latih*	*Precision*	*Recall*	*F-measure*
50 %	33.3333	38.6917	27.1333
60 %	43.75	52.4125	43.483333
66.7 %	67.708333	71.414899	67.557277
70 %	71.875	77.075	72.166667
80 %	86.458333	88.658333	86.608333

Untuk keseluruhan hasil perhitungan *recall, precision dan f-measure* terlampir pada bagian lampiran ketiga. Berikut hasil perhitungan dari tiap kategori pada perbandingan 50 % data *latih* ditampilkan pada tabel 4.8.

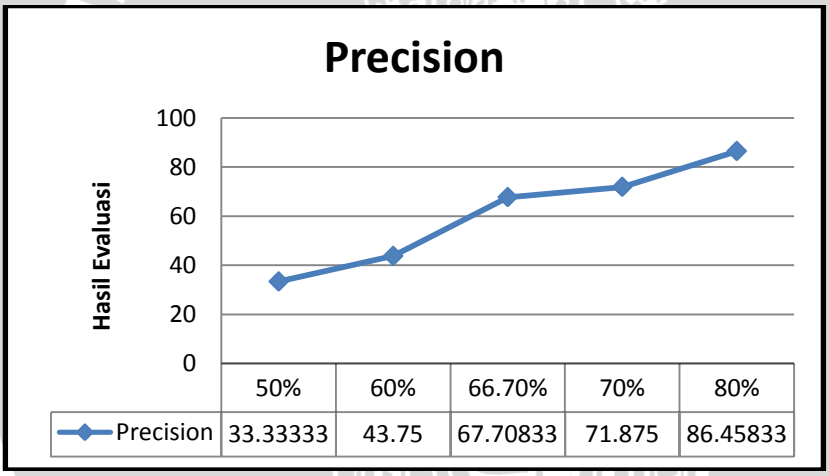
Tabel 4.8 Hasil perhitungan *precision, recall* dan *f-measure* untuk data *latih* 50%.

n uji		*Precision*	*Recall*	*F-measure*
1	Politik	0.75	0.206	0.324
	Bisnis	0.125	0.5	0.2
	Metropolis	0.125	1	0.222
	Olahraga	0	0	0
	Rata-rata	33.3333333	56.8666666	24.8666666
2	Politik	0.25	0.285	0.266
	Bisnis	0.25	0.666	0.363
	Metropolis	0.375	1	0.545
	Olahraga	0.875	0.368	0.518
	Rata-rata	43.75	57.975	42.3
3	Politik	0.375	0.25	0.3
	Bisnis	0	0	0
	Metropolis	0	0	0
	Olahraga	0.875	0.368	0.518
	Rata-rata	62.5	30.9	40.9
	Rata-rata Total	33,33333	48.5805555	36.0222222

Pada tabel 4.8 untuk hasil pengujian 50% data *latih*, terdapat beberapa kolom kategori yang tidak memiliki nilai *precision, recall* dan *f-measure* hal ini dikarenakan kegagalan sistem dalam melakukan klasifikasi.

4.5 Analisa Hasil

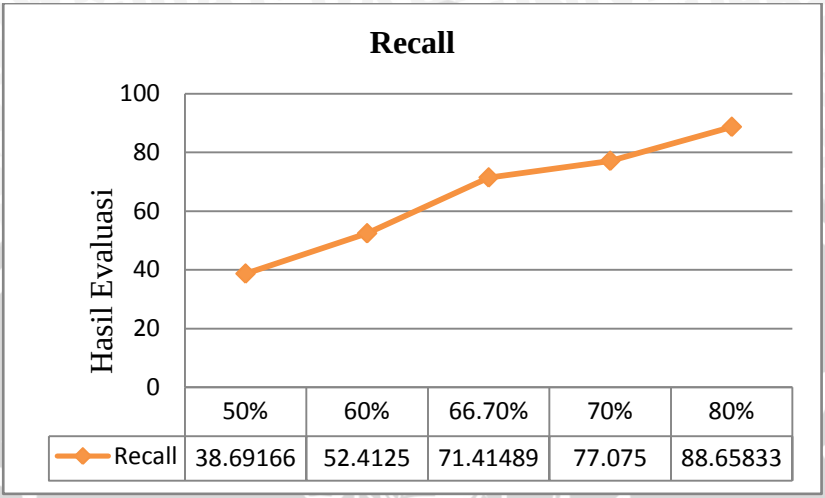
Dari data hasil perngujian pada tabel 4.7 dapat dibuat grafik sebagai analisa data. Grafik *precision* menunjukkan hasil prosentase ketepatan data manual terhadap pengkategorian sistem. Pada gambar 4.5 gambar hasil *precision* , dapat dilihat hasil terendah dari rata-rata *precision* pada prosentase data latih 60 %, sedangkan pada prosentase data 50% lebih tinggi karna ada beberapa kategori yang tidak memiliki nilai *precision* sehingga tidak diikutkan dalam perhitungan rata-rata *precision.*



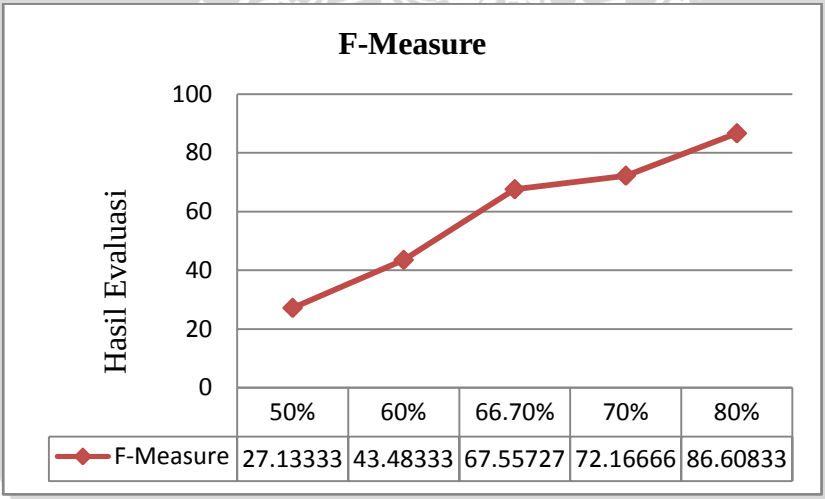
Gambar 4.5 Grafik hasil *precision*

Nilai *precision* semakin bertambah seiring bertambahnya jumlah data latih yang dimasukkan. Pada perbandingan data latih 80 % hasil precision mencapai nilai tertinggi pada 86,4 %.

Pada grafik gambar 4.6 hasil dari perhitungan rata-rata nilai *recall*. Data terendah yang didapat adalah pada perbandingan 50 % data *latih* yang hanya memiliki nilai 48,5 %. Data *recall* semakin bertambah dengan bertambahnya jumlah data *latih*. Pada data *latih* 80 % merupakan nilai tertinggi yang didapat dari *recall.*



Gambar 4.6 Grafik hasil *recall*



Gambar 4.7 Grafik hasil *F-measure*

Analisa selanjutnya adalah penggabungan dari nilai *precision* dan *recall*, perhitungan nilai *f-measure*. Pada gambar grafik 4.7 didapat nilai terendah dari hasil perhitungan *f-measure* rata-rata pada

saat komposisi 50 % data *latih*, nilai *f-measure* mencapai 36 %. Sedangkan pencapain nilai *f-measure* tertinggi pada data latih 80 % dengan perolehan nilai *f-measure* 86,6. Nilai *f-measure 100 %* berarti seluruh dokumen berhasil diklasifikasikan oleh sistem. Semakin banyak dokumen dalam pembelajaran akan semakin meningkatkan jumlah ketepatan dokumen pengujian.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan dan implementasi sistem maka dapat disimpulkan :

1. *Iterative Dichotomizer Tree* dapat diterapkan pada proses pengklasifikasian berita berbahasa Indonesia. ID3 memerlukan proses *preprocessing*, pembobotan dan *generalisasi* data sebelum melakukan proses pengenalan pola.
2. Hasil *tertinggi* dari perhitungan *recall, precision* dan *f-measure* pada perbandingan data *latih* 80 % dengan nilai *recall* 88.658333 %, *precision* 86.458333 % dan *f-measure* 86.608333 %. Dengan hasil pengujian ini maka algoritma ID3 dapat digunakan untuk pengklasifikasian berita berbahasa Indonesia.

5.2 Saran

Sistem yang dibangun pada penelitian ini belum mencapai kesempurnaan, adapun saran untuk penelitian lebih lanjut:

1. Data berita masih bersumber dari satu situs berita yang sama, sehingga *layout* halaman web sama. Untuk penelitian selanjutnya diharapkan bisa bersumber dari situs beragam dan memiliki metode pembersihan *tag* HTML yang bisa dipergunakan untuk semua halaman situs berita. Dengan semakin beragamnya sumber pengetahuan data akan semakin bervariasi hasil yang didapat.
2. Pada penelitian ini data berita diproses secara offline, untuk penelitian lebih lanjut dapat dilakukan proses pengambilan dan klasifikasi secara online.

Daftar Pustaka

Assegaf, Dja'far H. 1991. *Jurnalistik Masa Kini*. Jakarta: Ghalia Indonesia.

Basuki, Sulistyo. 2008. Teknik dan Jasa dokumentasi. Jakarta : Gramedia Pustaka Utama.

Defiyanti, Sofi dan Pardede, Crispina D. L. 2010. *Perbandingan Kinerja Algoritma ID3 dan C4.5 Dalam Klasifikasi Spam-Mail*. Depok: Universitas Gunadarma.

Even,Yahir dan Zohar. 2002. *Introduction to Teks Mining. Automated Learning Group National Center For Supercomputing Applications*. Chicago: University of Illionis.

Han, Jiawei dan Micheline Kamber. 2006. *Data Mining:Concepts and Techniques*. San Francisco: Morgan Kaufmann Publishers is an imprint of Elsevier.

Hearst, M.A. 1999. *Untangling text data mining. In: Proceedings of ACL'99: the 37 th Annual Meeting of the Association for Computational Linguistics*. Maryland: University of Maryland.

Hamilton, H dan Olive, W. 2003. Confusion Matrix. http://www2.cs.uregina.ca/dbd/cs831/ notes/confusion_matrix/ confusion_matrix.pdf tanggal akses: 21 Februari 2012

Musthafa, Aziz. 2009. *Klasifikasi Otomatis Dokumen Berita Kejadian Berbahasa Indonesia*. Malang: Universitas Islam Negri (UIN) Maulana Malik Ibrahim Malang.

Ramos, Juan. 2006. *Using TF-IDF to Determine Word Relevance in Document Queries*. Department of Computer Science, Rutgers University. (http://www.cs.rutgers.edu/~mlittman/courses/ml03/iCML03/papers/ramos.pdf tanggal akses 21 Februari 2011

Ramlan, M. 1995. *Ilmu Bahasa Indonesia : Morfologi Suatu Tinjauan Deskriptif*. CV. Karyono. Yogyakarta.

Sugono, Dendy. 2008. *Kamus Bahsa Indonesia*. Pusat Bahasa Departemen Pendidikan Nasional Jakarta.

Supranto, J. 2000. *Statistika : Teori Dan Aplikasi Edisi Keenam*. Erlangga Jakarta.

Tala, Fadillah Z. 2003. *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*. Netherlands: Universiteit van Amsterdam The Netherlands. http://www.illc.uva.nl/Research/Reports/MoL-2003-02.teks.pdf tanggal akses : 18 Juli 2011.

Wahyudi, JB. 2002. *Dasar-dasar jurnalistik radio dan televise*. Perpustakaan Utan Kayu Jakarta.

Mooney, Raymond J. 2006. Machine Learning Text Categorization. University of Texas at Austun.

Wibisono, yudi. Masayu Leylia Khodra. 2005. *Clustering Berita Berbahasa Indonesia*. FPMIPA Universitas Pendidikan Indonesia, Jln Setiabudhi 229 Bandung.

Yiming Yang, Jaime G.Carbonell, Rulf D. Brown, Thomas Pierce, Brian T. Achiba ld, Xin Liu. 1999. LearningApproaches for

Dete cting and Tracking News Events. IEEE Intelligent Systems, Language Techlonolies Institute, Carbegie Mellon University.

Lampiran 1
Data stopword

1	ada	35	awal	69	benar
2	adalah	36	awalnya	70	benarkah
3	adanya	37	bagai	71	benarlah
4	adapun	38	bagaikan	72	berada
5	agak	39	bagaimana	73	berakhir
6	agaknya	40	bagaimanakah	74	berakhirlah
7	agar	41	bagaimanapun	75	berakhirnya
8	akan	42	bagi	76	berapa
9	akankah	43	bagian	77	berapakah
10	akhir	44	bahkan	78	berapalah
11	akhiri	45	bahwa	79	berapapun
12	akhirnya	46	bahwasanya	80	berarti
13	aku	47	baik	81	berawal
14	akulah	48	bakal	82	berbagai
15	amat	49	bakalan	83	berdatangan
16	amatlah	50	balik	84	beri
17	anda	51	banyak	85	berikan
18	andalah	52	bapak	86	berikut
19	antar	53	baru	87	berikutnya
20	antara	54	bawah	88	berjumlah
21	antaranya	55	beberapa	89	berkali
22	apa	56	begini	90	berkata
23	apaan	57	beginian	91	berkehendak
24	apabila	58	beginikah	92	berkeinginan
25	apakah	59	beginilah	93	berkenaan
26	apalagi	60	begitu	94	berlainan
27	apatah	61	begitukah	95	berlalu
28	artinya	62	begitulah	96	berlangsung
29	asal	63	begitupun	97	berlebihan
30	asalkan	64	bekerja	98	bermaksud
31	atas	65	belakang	99	bermula
32	atau	66	belakangan	100	bersama
33	ataukah	67	belum	101	bersiap
34	ataupun	68	belumlah	102	bertanya

103	berturut	140	demi	177	dilihat
104	bertutur	141	demikian	178	dimaksud
105	berujar	142	demikianlah	179	dimaksudkan
106	berupa	143	dengan	180	dimaksudkannya
107	besar	144	depan	181	dimaksudnya
108	betul	145	di	182	diminta
109	betulkah	146	dia	183	dimintai
110	biasa	147	diakhiri	184	dimisalkan
111	biasanya	148	diakhirinya	185	dimulai
112	bila	149	dialah	186	dimulailah
113	bilakah	150	diantara	187	dimulainya
114	bisa	151	diantaranya	188	dimungkinkan
115	bisakah	152	diberi	189	dini
116	boleh	153	diberikan	190	dipastikan
117	bolehkah	154	diberikannya	191	diperbuat
118	bolehlah	155	dibuat	192	diperbuatnya
119	buat	156	dibuatnya	193	dipergunakan
120	bukan	157	didapat	194	diperkirakan
121	bukankah	158	didatangkan	195	diperlihatkan
122	bukanlah	159	digunakan	196	diperlukan
123	bukannya	160	diibaratkan	197	diperlukannya
124	bulan	161	diibaratkannya	198	dipersoalkan
125	bung	162	diingat	199	dipertanyakan
126	cara	163	diingatkan	200	dipunyai
127	caranya	164	diinginkan	201	diri
128	cukup	165	dijawab	202	dirinya
129	cukupkah	166	dijelaskan	203	disampaikan
130	cukuplah	167	dijelaskannya	204	disebut
131	cuma	168	dikarenakan	205	disebutkan
132	dahulu	169	dikatakan	206	disebutkannya
133	dalam	170	dikatakannya	207	disini
134	dan	171	dikerjakan	208	disinilah
135	dapat	172	diketahui	209	ditambahkan
136	dari	173	diketahuinya	210	ditandaskan
137	daripada	174	dikira	211	ditanya
138	datang	175	dilakukan	212	ditanyai
139	dekat	176	dilalui	213	ditanyakan

214	ditegaskan	251	ibaratkan	288	kalaupun
215	ditujukan	252	ibaratnya	289	kalian
216	ditunjuk	253	ibu	290	kami
217	ditunjuki	254	ikut	291	kamilah
218	ditunjukkan	255	ingat	292	kamu
219	ditunjukkannya	256	ingin	293	kamulah
220	ditunjuknya	257	inginkah	294	kan
221	dituturkan	258	inginkan	295	kapan
222	dituturkannya	259	ini	296	kapankah
223	diucapkan	260	inikah	297	kapanpun
224	diucapkannya	261	inilah	298	karena
225	diungkapkan	262	itu	299	karenanya
226	dong	263	itukah	300	kasus
227	dua	264	itulah	301	kata
228	dulu	265	jadi	302	katakan
229	empat	266	jadilah	303	katakanlah
230	enggak	267	jadinya	304	katanya
231	enggaknya	268	jangan	305	ke
232	entah	269	jangankan	306	keadaan
233	entahlah	270	janganlah	307	kebetulan
234	guna	271	jauh	308	kecil
235	gunakan	272	jawab	309	kedua
236	hal	273	jawaban	310	keduanya
237	hampir	274	jawabnya	311	keinginan
238	hanya	275	jelas	312	kelamaan
239	hanyalah	276	jelaskan	313	kelihatan
240	hari	277	jelaslah	314	kelihatannya
241	harus	278	jelasnya	315	kelima
242	haruslah	279	jika	316	keluar
243	harusnya	280	jikalau	317	kembali
244	hendak	281	juga	318	kemudian
245	hendaklah	282	jumlah	319	kemungkinan
246	hendaknya	283	jumlahnya	320	kemungkinannya
247	hingga	284	justru	321	kenapa
248	ia	285	kala	322	kepada
249	ialah	286	kalau	323	kepadanya
250	ibarat	287	kalaulah	324	kesampaian

325	keseluruhan	362	manalagi	399	menanti
326	keseluruhannya	363	masa	400	menantikan
327	keterlaluan	364	masalah	401	menanya
328	ketika	365	masalahnya	402	menanyai
329	khususnya	366	masih	403	menanyakan
330	kini	367	masihkah	404	mendapat
331	kinilah	368	masing	405	mendapatkan
332	kira	369	mau	406	mendatang
333	kiranya	370	maupun	407	mendatangi
334	kita	371	melainkan	408	mendatangkan
335	kitalah	372	melakukan	409	menegaskan
336	kok	373	melalui	410	mengakhiri
337	kurang	374	melihat	411	mengapa
338	lagi	375	melihatnya	412	mengatakan
339	lagian	376	memang	413	mengatakannya
340	lah	377	memastikan	414	mengenai
341	lain	378	memberi	415	mengerjakan
342	lainnya	379	memberikan	416	mengetahui
343	lalu	380	membuat	417	menggunakan
344	lama	381	memerlukan	418	menghendaki
345	lamanya	382	memihak	419	mengibaratkan
346	lanjut	383	meminta	420	mengibaratkannya
347	lanjutnya	384	memintakan	421	mengingat
348	lebih	385	memisalkan	422	mengingatkan
349	lewat	386	memperbuat	423	menginginkan
350	lima	387	mempergunakan	424	mengira
351	luar	388	memperkirakan	425	mengucapkan
352	macam	389	memperlihatkan	426	mengucapkannya
353	maka	390	mempersiapkan	427	mengungkapkan
354	makanya	391	mempersoalkan	428	menjadi
355	makin	392	mempertanyakan	429	menjawab
356	malah	393	mempunyai	430	menjelaskan
357	malahan	394	memulai	431	menuju
358	mampu	395	memungkinkan	432	menunjuk
359	mampukah	396	menaiki	433	menunjuki
360	mana	397	menambahkan	434	menunjukkan
361	manakala	398	menandaskan	435	menunjuknya

436	menurut	473	padahal	510	sambil
437	menuturkan	474	padanya	511	sampai
438	menyampaikan	475	pak	512	sampaikan
439	menyangkut	476	paling	513	sana
440	menyatakan	477	panjang	514	sangat
441	menyebutkan	478	pantas	515	sangatlah
442	menyeluruh	479	para	516	satu
443	menyiapkan	480	pasti	517	saya
444	merasa	481	pastilah	518	sayalah
445	mereka	482	penting	519	se
446	merekalah	483	pentingnya	520	sebab
447	merupakan	484	per	521	sebabnya
448	meski	485	percuma	522	sebagai
449	meskipun	486	perlu	523	sebagaimana
450	meyakini	487	perlukah	524	sebagainya
451	meyakinkan	488	perlunya	525	sebagian
452	minta	489	pernah	526	sebaik
453	mirip	490	persoalan	527	sebaiknya
454	misal	491	pertama	528	sebaliknya
455	misalkan	492	pertanyaan	529	sebanyak
456	misalnya	493	pertanyakan	530	sebegini
457	mula	494	pihak	531	sebegitu
458	mulai	495	pihaknya	532	sebelum
459	mulailah	496	pukul	533	sebelumnya
460	mulanya	497	pula	534	sebenarnya
461	mungkin	498	pun	535	seberapa
462	mungkinkah	499	punya	536	sebesar
463	nah	500	rasa	537	sebetulnya
464	naik	501	rasanya	538	sebisanya
465	namun	502	rata	539	sebuah
466	nanti	503	rupanya	540	sebut
467	nantinya	504	saat	541	sebutlah
468	nyaris	505	saatnya	542	sebutnya
469	nyatanya	506	saja	543	secara
470	oleh	507	sajalah	544	secukupnya
471	olehnya	508	saling	545	sedang
472	pada	509	sama	546	sedangkan

547	sedemikian	584	semakin	621	seseorang
548	sedikit	585	semampu	622	sesuatu
549	sedikitnya	586	semampunya	623	sesuatunya
550	seenaknya	587	semasa	624	sesudah
551	segala	588	semasih	625	sesudahnya
552	segalanya	589	semata	626	setelah
553	segera	590	semata-mata	627	setempat
554	seharusnya	591	semaunya	628	setengah
555	sehingga	592	sementara	629	seterusnya
556	seingat	593	semisal	630	setiap
557	sejak	594	semisalnya	631	setiba
558	sejauh	595	sempat	632	setibanya
559	sejenak	596	semua	633	setidaknya
560	sejumlah	597	semuanya	634	setinggi
561	sekadar	598	semula	635	seusai
562	sekadarnya	599	sendiri	636	sewaktu
563	sekali	600	sendirian	637	siap
564	sekalian	601	sendirinya	638	siapa
565	sekaligus	602	seolah	639	siapakah
566	sekalipun	603	seolah-olah	640	siapapun
567	sekarang	604	seorang	641	sini
568	sekecil	605	sepanjang	642	sinilah
569	seketika	606	sepantasnya	643	soal
570	sekiranya	607	sepantasnyalah	644	soalnya
571	sekitar	608	seperlunya	645	suatu
572	sekitarnya	609	seperti	646	sudah
573	sekurangnya	610	sepertinya	647	sudahkah
574	sela	611	sepihak	648	sudahlah
575	selain	612	sering	649	supaya
576	selaku	613	seringnya	650	tadi
577	selalu	614	serta	651	tadinya
578	selama	615	serupa	652	tahu
579	selamanya	616	sesaat	653	tahun
580	selanjutnya	617	sesama	654	tak
581	seluruh	618	sesampai	655	tambah
582	seluruhnya	619	sesegera	656	tambahnya
583	semacam	620	sesekali	657	tampak

658	tampaknya	695	tersebut	732	walaupun
659	tandas	696	tersebutlah	733	wong
660	tandasnya	697	tertentu	734	yaitu
661	tanpa	698	tertuju	735	yakin
662	tanya	699	terus	736	yakni
663	tanyakan	700	terutama	737	yang
664	tanyanya	701	tetap		
665	tapi	702	tetapi		
666	tegas	703	tiap		
667	tegasnya	704	tiba		
668	telah	705	tidak		
669	tempat	706	tidakkah		
670	tengah	707	tidaklah		
671	tentang	708	tiga		
672	tentu	709	tinggi		
673	tentulah	710	toh		
674	tentunya	711	tunjuk		
675	tepat	712	turut		
676	terakhir	713	tutur		
677	terasa	714	tuturnya		
678	terbanyak	715	ucap		
679	terdahulu	716	ucapnya		
680	terdapat	717	ujar		
681	terdiri	718	ujarnya		
682	terhadap	719	umum		
683	terhadapnya	720	umumnya		
684	teringat	721	ungkap		
685	terjadi	722	ungkapnya		
686	terjadilah	723	untuk		
687	terjadinya	724	usah		
688	terkira	725	usai		
689	terlalu	726	waduh		
690	terlebih	727	wah		
691	terlihat	728	wahai		
692	termasuk	729	waktu		
693	ternyata	730	waktunya		
694	tersampaikan	731	walau		


UNIVERSITAS BRAWIJAYA

Lampiran 2

Data latih dan data testing untuk contoh perhitungan manual

Dokumen 1

Dokumen *latih*

Kategori : Politik

Isi berita :

5 Anggota DPR Teken Petisi Tolak Gedung Baru

"Karena DPR dipilih rakyat, jadi kembalikan saja soal ini ke rakyat."

VIVAnews - Lima anggota DPR membuat petisi penolakan pembangunan gedung baru. Lima orang yang berasal dari fraksi-fraksi yang berbeda itu mendesak agar pimpinan DPR menghentikan rencana pembangunan gedung baru.

Lima orang itu yakni, Teguh Juwarno dari Fraksi Partai Amanat Nasional (PAN), Roy Suryo dari Fraksi Demokrat, Budiman Sudjatmiko dari Fraksi PDI Perjuangan, Edy Prabowo dari Fraksi Gerindra, dan Abdul Malik Haramain dari Fraksi Kebangkitan Bangsa.

Hasil perhitungan frekuensi kata pada dokumen 1

Kata	Frekuensi	Kata	Frekuensi
Anggota	2	Lima	3
DPR	4	membuat	1
Teken	1	penolakan	1
Petisi	2	pembangunan	2
Tolak	1	orang	2
Gedung	3	yang	2
Baru	3	berasal	1
Karena	1	dari	6
Dipilih	1	fraksi	7
rakyat	2	berbeda	1
jadi	1	itu	2
kembalikan	1	mendesak	1
saja	1	agar	1
soal	1	pimpinan	1
ini	1	menghentikan	1
ke	2	rencana	1

VIVAnews	1	PAN	1
yakni	1	Roy	1
Teguh	1	Suryo	1
Juwarno	1	Demokrat	1
Partai	1	Budiman	1
Amanat	1	Sudjatmiko	1
Nasional	1	PDI	1
Perjuangan	1	Prabowo	1
Edy	1	Gerindra	1
dan	1	Malik	1
Abdul	1	Haramain	1
bangkitan	1	Bangsa	1

Dokumen 2
Dokumen *latih*
Kategori : Politik
Isi berita :
Jazilul & Andi Gantikan Lily & Gus Choi
Pergantian ini diperkirakan berbuntut panjang. Pasalnya, Gus Choi akan melawan.
VIVAnews – Jazilul Fawaid dan Andi Muawiyah Ramly dipastikan menduduki kursi anggota Fraksi Partai Kebangkitan Bangsa Dewan Perwakilan Rakyat yang dalam waktu dekat akan ditinggalkan Lily Wahid dan Effendy Choirie (Gus Choi).
"Penggantinya otomatis dari kader PKB yang meraih suara terbanyak di bawah mereka (Lily dan Choi)," kata Ketua DPP PKB Syaifullah Maksum, Selasa, 15 Maret 2011.
Lily Wahid yang merupakan adik pendiri PKB, Abdurrahman Wahid, dan Gus Choi diusulkan Dewan Pimpinan Pusat PKB kepada pimpinan DPR untuk diganti. Pasalnya, kedua kader ini dinilai tidak mau tunduk pada kebijakan partai untuk menolak Angket Mafia Perpajakan. Proses PAW itu sendiri berjalan rumit dan melalui perdebatan panjang karena di internal partai ada yang pro dan kontra, tetapi akhirnya disepakati untuk PAW.

Hasil perhitungan frekuensi kata pada dokumen 2

Kata	Frekuensi	Kata	Frekuensi
jazilul	2	dan	6

andi	2	muawiyah	1
gantikan	1	ramly	1
lily	4	dipastikan	1
gus	4	menduduki	1
choi	6	kursi	1
pergantian	1	anggota	1
ini	3	fraksi	1
diperkirakan	1	partai	3
berbuntut	1	kebangkitan	1
panjang	2	bangsa	1
pasalnya	2	dewan	2
akan	5	perwakilan	1
melawan	1	rakyat	1
vivanews	1	yang	4
fawaid	1	dalam	1
waktu	1	effendy	1
dekat	1	kata	1
ditinggalkan	1	ketua	1
wahid	3	dpp	1
rie	1	syaifullah	1
penggantinya	1	maksum	1
otomatis	1	selasa	1
dari	1	maret	1
kader	2	abdurrahman	1
pkb	4	usulkan	1
meraih	1	pimpinan	2
suara	1	pusat	1
di	8	kepada	1
bawah	1	dpr	1
mereka	1	untuk	3
ganti	1	kedua	1
tunduk	1	kebij	1
menolak	1	angket	1
mafia	1	perpaj	1
proses	1	paw	2
itu	1	rumit	1
senri	1	melalui	1

berjalan	1	perdebatan	1
karena	1	pro	1
internal	1	kontra	1
ada	1	tetapi	1
akhirnya	1	sepakati	1

Dokumen 3
Dokumen *latih*
Kategori : Politik
Isi berita :
10 Kader Golkar Belajar dari Parpol China
Pendidikan kader muda Golkar ini rencananya berlangsung pada saat liburan sekolah.
VIVAnews – Partai Golkar berencana mengirimkan sebanyak 10 kader untuk belajar manajemen partai dan manajemen kaderisasi dari Partai Komunis Tiongkok (PKT). Langkah serupa akan dilakukan oleh partai penguasa China tersebut terhadap kadernya. Rencana di bidang organisasi dan pengkaderan tersebut merupakan salah satu hasil kesepakatan pertemuan Partai Golkar dan PKT beberapa waktu lalu.

Hasil perhitungan frekuensi kata pada dokumen 3

Kata	Frekuensi	Kata	Frekuensi
Kader	6	mengirimkan	1
Golkar	4	sebanyak	1
Belajar	2	untuk	1
dari	2	manajemen	2
Parpol	1	dan	4
China	2	isasi	2
Pendidikan	1	komunis	1
Muda	1	tiongkok	1
muda	1	pkt	2
sekolah	1	Langkah	1
vivanews	1	Serupa	1
partai	5	akan	2
berencana	1	dilakukan	1
oleh	1	penguasa	1

tersebut	1	merup	1
terhadap	1	salah	1
rencana	1	satu	1
di	1	kesepakatan	1
big	1	pertemuan	1
organ	1	beberapa	1
pengan	1	waktu	1

Dokumen 4
Dokumen *latih*
Kategori : Olahraga
Isi berita :
PSM Minta Kelompok 78 Patuhi FIFA
Terakhir Kelompok 78 mengancam akan melengserkan Ketua Normalisasi, Agum Gumelar.
VIVAnews - Ketua Umum PSM Makassar, Ilham Arief Siradjuddin, meminta Kelompok 78 untuk tetap mematuhi perintah FIFA dan tidak memaksakan pencalonan George Toisutta dan Arifin Panigoro.
Sembilan hari jelang pelaksanaan Kongres PSSI, Kelompok 78 terus berusaha mencalonkan George dan Arifin meski keduanya telah dilarang oleh FIFA untuk ikut dalam pencalonan.
Terakhir Kelompok 78 mengancam akan melengserkan Ketua Normalisasi, Agum Gumelar, dalam Kongres yang rencananya akan dilakukan di Hotel Sultan, Jakarta, 20 Mei mendatang.

Hasil perhitungan frekuensi kata pada dokumen 4

Kata	Frekuensi	Kata	Frekuensi
PSM	2	normalisasi	2
Minta	2	agum	2
kelompok	5	gumelar	2
patuhi	1	vivanews	1
fifa	3	umum	1
terakhir	2	Makassar	1
mengancam	2	Ilham	1
akan	4	Arief	1
melengserkan	2	siradjuddin	1
ketua	3	me	7

untuk	2	jelang	1
tetap	1	pelaksanaan	1
matuhi	1	kongres	2
perintah	1	pssi	1
dan	3	terus	1
tidak	1	berusaha	1
maks	1	ncalonkan	1
pencalonan	2	ski	1
george	2	keduanya	1
toisutta	1	telah	1
arifin	2	dilarang	1
panigoro	1	oleh	1
sembilan	1	ikut	1
hari	1	sultan	1
hotel	1	jakarta	1
rencananya	1	ndatang	1

Dokumen 5
Dokumen *latih*
Kategori : Olahraga
Isi berita :
Pandangan Dirigen Aremania Soal 'Pemilu' PSSI
'Konyol kalau sampai melanggar keputusan FIFA," ujar Yuli Sumpil.
VIVAnews - Dirigen Aremania (pendukung Arema FC), Yuli Sumpil berharap kisruh yang mewarnai pemilihan pengurus PSSI (2011-2015) tidak mengorbankan sepakbola nasional. Yuli meminta semua pihak tetap tunduk pada keputusan FIFA.
"Konyol kalau sampai melanggar keputusan yang sudah dibuat oleh FIFA. Kalau kena sanksi, justru hanya merugikan sepakbola Indonesia secara keseluruhan,"kata Yuli saat dihubungi VIVAnews, Jumat, 30 April 2011.
Silang pendapat memang sedang mewarnai proses pemilihan pengurus PSSI periode 2011-2015. Pemilik suara yang menamakan diri sebagai kelompok 78 tidak sejalan dengan Komite Normalisasi yang dibentuk FIFA.
Pemicunya adalah keputusan FIFA yang melarang George Toisutta dan Arifin Panigoro untuk ikut bursa pencalonan. Tak hanya sebagai

ketua umum, keduanya juga dilarang maju sebagai calon anggota exco.

Hasil perhitungan frekuensi kata pada dokumen 5

Kata	Frekuensi	Kata	Frekuensi
pandangan	1	tetap	1
dirigen	2	tunduk	1
aremania	2	pada	1
soal	1	sudah	1
pemilu	1	dibuat	1
pssi	3	oleh	1
konyol	2	kena	1
kalau	3	sanksi	1
sampai	2	justru	1
melanggar	2	hanya	2
keputusan	4	merugikan	1
fifa	5	indonesia	1
ujar	1	secara	1
yuli	4	keseluruhan	1
sumpil	2	kata	1
vivanews	2	saat	1
pendukung	1	dihubungi	1
arema	1	jumat	1
fc	2	april	1
berharap	1	silang	1
kisruh	1	pendapat	1
yang	5	memang	1
mewarnai	2	sedang	1
pemilihan	2	proses	1
pengurus	2	periode	1
tidak	2	Pemilik	1
mengorbankan	1	suara	1
sepakbola	2	diri	1
nasional	1	sebagai	3
meminta	1	kelompok	1
semua	1	sejalan	1
pihak	1	melarang	1

toisutta	1	maju	1
dan	1	calon	2
panigoro	1	anggota	1
umum	1	exco	1
keduanya	1		

Dokumen 6
Dokumen *latih*
Kategori : Olahraga
Isi berita :
Menpora: Hormati Apapun Putusan FIFA
"Kalau itu semua diikuti, Insya Allah terbentuk kepengurusan PSSI yang kredibel,"
VIVAnews - Ketua Komite Normalisasi (KN) PSSI Agum Gumelar usai bertemu dengan Presiden FIFA Joseph Blatter di markas FIFA di Zurich, Swiss menegaskan bahwa Komite Normalisasi harus tetap berpatokan pada surat FIFA tanggal 4 April lalu terkait pelaksanaan Kongres PSSI. Atas hasil putusan tersebut, Agum Gumelar meminta semua pihak untuk menghormati putusan FIFA tersebut. Hal senada disampaikan Menteri Pemuda dan Olahraga (Menpora), Andi Mallarangeng yang meminta kepada semua pihak agar mematuhi apa yang diputuskan oleh FIFA nantinya. "Aturan FIFA perlu dihormati. Kalau itu semua diikuti dan berjalan dengan baik, Insya Allah terbentuk kepengurusan PSSI yang kredibel," ujar Andi Mallarangeng di kantor Menpora, Senayan, Rabu, 20 April 2011.

Mengenai rencana penolakan yang dilakukan oleh 78 pemilik suara terhadap putusan FIFA, Andi menuturkan, "Enggak usah berandai-andai dulu. Pokoknya kami harapkan semua berjalan dengan baik sesuai aturan sampai kongres terwujud dengan baik dan kepengurusan baru terbentuk," tambah Andi.

Hasil perhitungan frekuensi kata pada dokumen 6

Kata	Frekuensi	Kata	Frekuensi
Menpora	3	fifa	8
hormati	3	kalau	2
apapun	1	itu	2
putusan	4	semua	5

diikuti	2	menegaskan	1
insya	2	bahwa	1
allah	2	harus	1
terbentuk	3	tetap	1
kepengurusan	3	berpatokan	1
pssi	4	pada	2
yang	5	surat	2
kredibel	2	tanggal	1
vivanews	1	april	2
ketua	1	lalu	1
komite	2	terkait	1
normalisasi	2	pelaksanaan	1
kn	2	kongres	2
agum	2	atas	1
gumelar	2	hasil	1
usai	1	tersebut	2
bertemu	1	meminta	2
dengan	4	pihak	2
presiden	1	untuk	1
joseph	1	meng	2
blatter	1	hal	1
di	11	senada	1
markas	1	menteri	1
zurich	1	pemuda	1
swiss	1	dan	3
olahraga	1	berjal	2
an	2	baik	3
mallargeng	2	ujar	1
ke	1	ktor	1
agar	1	senay	1
mematuhi	1	rabu	1
apa	1	enai	1
putusk	1	renca	1
oleh	2	penolak	1
ntinya	1	pemilik	1
atur	2	suara	1
perlu	1	terhadap	1

menuturk	1	kami	1
usah	1	harapk	1
berdai	1	sesuai	1
dai	1	sampai	1
dulu	1	terwujud	1
baru	1	tambah	1

Dokumen 7
Dokumen *testing*
Kategori : tidak berkategori
Isi berita :
Anggota DPR ke Lokalisasi, Laporkan ke BK
"Apakah bukti-bukti tersebut absah, ini untuk menghindari pemerasan."
VIVAnews - Wakil Ketua Badan Kehormatan DPR, Nudirman Munir, meminta kepada masyarakat melaporkan jika menemukan ada anggota dewan yang pergi ke lokasi-lokasi pelacuran. DPR tidak akan segan memberikan sanksi jika anggotanya terbukti pergi ke lokalisasi.
"Kalau ada bukti bahwa dia memang pergi ke lokasi pelacuran, foto. Masyarakat boleh melaporkannya ke BK," kata Nudirman Munir di Gedung DPR, Jakarta, Rabu 16 Februari 2011.
Menurutnya, BK DPR akan meneliti laporan masyarakat itu. "Apakah bukti-bukti tersebut absah, ini untuk menghindari pemerasan," ujarnya.
Jika terbukti ada anggota DPR yang terlihat ke lokasi pelacuran, maka DPR akan memberikan sanksi tegas kepada mereka. "Dari peringatan tertulis, pemberhentian sementara, sampai pemecatan," ujarnya.
Dalam Rapat Paripurna DPR, Rabu 16 Februari 2011, mengagendakan pengesahan Rancangan Peraturan DPR RI tentang Kode Etik yang disusun oleh Badan Kehormatan DPR. Salah satu poin penting adalah adanya klausul yang menyebut, anggota Dewan dilarang masuk ke lokalisasi pelacuran dan perjudian.

Hasil perhitungan frekuensi kata pada dokumen 7

Kata	Frekuensi	Kata	Frekuensi
Anggota	5	DPR	10
ke	13	ter	4
Lokalisasi	3	kalau	1
Laporkan	3	bahwa	1
bk	3	dia	2
apakah	2	mang	1
bukti	7	foto	1
tersebut	2	boleh	1
absah	2	kata	1
ini	2	di	3
untuk	2	gedung	1
menghindari	2	jakarta	1
pemerasan	2	rabu	2
vivanews	1	februari	2
wakil	1	nurut	1
tua	1	neliti	1
badan	2	laporan	1
hormatan	2	itu	1
nudirman	2	ujar	2
munir	2	lihat	1
meminta	1	maka	1
pada	2	tegas	1
masyarakat	3	reka	1
me	13	dari	1
jika	3	peringatan	1
nemukan	1	tulis	1
ada	5	pemberhentian	1
dewan	2	sentara	1
yang	4	sampai	1
pergi	3	pecatan	1
lokasi	4	dalam	1
pelacuran	4	rapat	1
tidak	1	paripurna	1
akan	4	ngagend	1
segan	1	pengesahan	1

mberikan	2	rancangan	1
sanksi	2	peraturan	1
nya	6	tentang	1
kode	1	penting	1
etik	1	lah	1
susun	1	klausul	1
oleh	1	nyebut	1
salah	1	larang	1
satu	1	masuk	1
poin	1	perjun	1
dan	1		

Dokumen 8

Dokumen *testing*

Kategori : tidak berkategori

Isi berita :

Calon Ketum PSSI, Iman Tunggu Komite Banding

Iman dicalonkan menjadi ketua umum PSSI oleh klub Divisi II PS Serdang Bedagai.

VIVAnews - Direktur Bidang Teknis Badan Tim Nasional (BTN), Iman Arif, santai mengenai pencalonan dirinya sebagai ketua umum PSSI.

Nama Iman masuk dalam daftar 19 calon ketua umum PSSI periode 2011-2015 yang lolos verifikasi Komite Pemilihan, pekan lalu. Tidak hanya menjadi calon ketua umum, Iman juga dicalonkan sebagai wakil ketua.

Ditanya mengenai persiapannya dalam pencalonan tersebut, Iman mengaku santai dan belum melakukan persiapan apa pun. Direktur Indonesia Football Academy (IFA) dan Soccer School Indonesia (SSI) Arsenal tersebut memilih untuk menunggu keputusan Komite Banding Pemilihan.

"Saya sih santai-santai saja. Saya ingin melihat dulu keputusan Komite Banding nanti, karena kan bisa mengarah ke sanksi FIFA," ujar Iman saat dihubungi VIVAnews.com di Jakarta, Jumat, 6 Mei 2011.

Hasil perhitungan frekuensi kata pada dokumen 8

Kata	Frekuensi	Kata	Frekuensi
calon	7	iman	1
ketum	1	tunggu	1
pssi	4	komite	4
banding	3	hanya	1
dikan	2	wakil	1
menjadi	2	ditanya	1
ketua	2	tersebut	2
umum	4	dan	2
oleh	1	indonesia	2
klub	1	football	1
divisi	1	academy	1
ps	1	ifa	2
serdang	1	soccer	1
bedagai	1	school	1
vivanews	2	ssi	1
direktur	2	arsenal	1
bidang	1	untuk	1
teknis	1	menunggu	1
badan	1	keputusan	2
tim	1	saya	2
nasional	1	sih	1
btn	1	saja	1
arif	1	ingin	1
santai	4	melihat	1
mengenai	2	dulu	1
penan	2	nanti	1
dirinya	1	karena	1
sebagai	2	kan	1
nama	1	bisa	1
masuk	1	mengarah	1
dalam	2	ke	1
daftar	1	sanksi	1
periode	1	f	1
yang	1	ujar	1
lolos	1	saat	1

verifikasi	1	dihubungi	1
pemilihan	2	com	1
pekan	1	di	1
lalu	1	jakarta	1
tidak	1	jumat	1

Dokumen 9
Dokumen *testing*
Kategori : tidak berkategori
Isi berita :
Hasil Verifikasi PSSI Diputuskan Lewat Voting
Anggota Komite Normalisasi, Dityo Pramono menilai semua kandidat layak diverifikasi.
VIVAnews - Komite Normalisasi tidak satu suara saat memutuskan hasil verifikasi bakal calon yang akan maju pada pemilihan pengurus PSSI periode 2011-2015. Keputusan terpaksa diambil melalui mekanisme voting.
Ketua Komite Normalisasi, Agum Gumelar sebenarnya tidak terbuka mengenai mekanisme pengambilan keputusan. Agum dengan tegas menyatakan bahwa hasil verifikasi diputuskan lewat kesepakatan bersama.
"Keputusan ini hasil kesepakatan bersama seluruh anggota Komite Normalisasi. Ini keputusan utuh satu komite. Dan perbedaan sudah tidak ada lagi," ujar Agum kepada wartawan Jumat, 29 April 2011.
Anggota KN lainnya, Dityo Pramono justru berkata lain. Menurut Dityo, pengambilan keputusan hasil verifikasi berjalan alot dan terpaksa dilakukan melalui mekanisme pemungutan suara atau voting.
Sebanyak 7 dari 8 anggota KN ikut dalam pemungutan suara tersebut. Hasilnya, lima orang menyatakan setuju, satu menolak, dan satu lainnya abstain. Sedangkan anggota yang tak hadir adalah Syamsul Ashar.
"Hingga akhir, saya menegaskan tidak setuju terhadap keputusan tersebut," ujar Dityo kepada wartawan.

Hasil perhitungan frekuensi kata pada dokumen 9

Kata	Frekuensi	Kata	Frekuensi
Hasil	6	komite	5
Verifikasi	5	normalisasi	5
PSSI	3	dityo	4
diputuskan	2	pramono	2
lewat	2	menilai	1
voting	3	semua	1
anggota	5	kandidat	1
layak	1	utuh	1
di	4	dan	4
vivanews	1	perbedaan	1
tidak	4	sudah	1
satu	4	ada	3
suara	3	lagi	1
saat	1	ujar	2
memutuskan	1	ke	2
bakal	1	wartawan	2
calon	1	jumat	1
yang	2	april	1
akan	3	kn	2
maju	1	lainnya	2
pada	3	justru	1
pemilihan	1	berkata	1
pengurus	1	lain	1
periode	1	menurut	1
keputusan	6	berjalan	1
terpaksa	2	lakukan	1
ambil	3	pemungutan	2
melalui	2	atau	1
mekanisme	3	sebanyak	1
ketua	1	dari	1
agum	3	ikut	1
gumelar	1	dalam	1
sebenarnya	1	tersebut	2
terbuka	1	nya	1
mengenai	1	lima	1

pengan	2	orang	1
dengan	1	setuju	2
tegas	1	menolak	1
menyat	2	abstain	1
bahwa	1	segkan	1
kesepakatan	2	tak	1
bersama	2	har	1
ini	2	lah	1
seluruh	1	syamsul	1
as	2	saya	1
hingga	1	menegkan	1
akhir	1	terhp	1

Lampiran 3
Hasil perhitungan setiap kategori

		50 - 50			60 – 40			66,7 - 33,3			7 - 30			80 - 20		
		P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
1	Politik	0.75	0.206	0.324	0.375	0.375	0.375	0.625	0.833	0.714	0.75	0.666	0.705	0.875	1	0.933
	Bisnis	0.125	0.5	0.2	0.5	0.21	0.296	0.5	0.571	0.533	0.875	1	0.933	0.75	1	0.857
	Metropolis	0.125	1	0.222	0.25	1	0.4	0.5	0.8	0.615	0.625	1	0.769	0.625	0.714	0.666
	Olahraga				0.25	0.666	0.363	1	0.571	0.727	0.75	0.545	0.631	1	0.666	0.8
	Rata-rata	33.3333	56.8667	24.8667	34.375	56.275	35.85	65.625	69.375	64.725	75	80.275	75.95	81.25	84.5	81.4
2	Politik	0.25	0.285	0.266	0.375	0.5	0.428	0.625	0.5	0.555	0.5	0.8	0.615	0.875	1	0.933
	Bisnis	0.25	0.666	0.363	0.25	0.4	0.307	0.75	0.75	0.75	0.5	1	0.666	0.875	1	0.933
	Metropolis	0.375	1	0.545	0.375	0.6	0.461	0.5	0.8	0.615	0.75	0.6	0.666	0.75	0.857	0.8
	Olahraga	0.875	0.368	0.518	0.875	0.4375	0.583	0.75	0.666	0.705	0.875	0.538	0.666	1	0.727	0.842
	Rata-rata	43.75	57.97	42.3	46.87	48.43	44.47	65.62	67.9	65.62	65.62	73.45	65.32	87.5	89.6	87.7
3	Politik	0.375	0.25	0.3	0.375	0.333	0.352	0.75	0.5455	0.6316	0.75	0.75	0.75	1	0.8	0.888
	Bisnis				0.625	0.714	0.666	0.625	0.8333	0.7143	0.75	1	0.857	0.875	1	0.933
	Metropolis				0.375	0.6	0.461	0.625	1	0.7692	0.625	0.714	0.666	0.875	1	0.933
	Olahraga	0.875	0.368	0.518	0.625	0.454	0.526	0.875	0.7	0.7778	0.875	0.636	0.736	0.875	0.875	0.875
	Rata-rata	62.5	30.9	40.9	50	52.52	50.12	71.87	76.97	72.32	75	77.5	75.22	90.62	91.87	90.72
	Rata - rata Total	46.5278	48.5806	36.0222	43.75	52.413	43.48333	67.708	71.415	67.557	71.875	77.075	72.167	86.458	88.658	86.608