2. Dasar Teori

2.1 Data Mining

Data mining adalah proses mengekstrak pengetahuan yang berguna dari jumlah data yang besar. *Data mining* bertujuan untuk menemukan pola dalam basis data yang berkapasitas besar untuk mendukung dalam pengambilan keputusan. *Data mining* menganalisis data menggunakan tool untuk menemukan pola dan aturan dalam himpunan data. Tool *data mining* diharapkan mampu mengenal pola ini dalam data dengan input minimal dari user[6].

2.2 Fungsionalitas Data Mining

Data mining dapat diklasifikasikan sesuai dengan fungsi yang dilakukan yaitu :

1. *Characterization*
   Memberikan ringkasan mengenai karakteristik objek-objek dalam suatu kelas tertentu. Contoh : menentukan karakteristik pelanggan potensial.
2. *Discrimination*
   Membandingkan karakteristik objek-objek pada kelas tertentu dengan kelas lainnya. Contoh : membandingkan pelanggan potensial dengan yang tidak.
3. *Association*
   Mempelajari frekuensi item-item yang terjadi secara bersamaan dalam transaksi database. Contoh : beli (x, roti) $\rightarrow$ beli (x, mentega).
4. *Classification*
   Mengklasifikasikan data ke dalam kelas yang diberikan berdasarkan nilai atribut (*supervised classification*). Kelas-kelas klasifikasi sudah didefinisikan dahulu. Tipe datanya biasanya bersifat kategorik. Contoh : mengklasifikasikan produk berdasar respon terhadap iklan : baik, biasa, tidak ada respon.
5. *Prediction*
   Meramalkan nilai atribut yang hilang atau tidak diketahui berdasarkan informasi lain. Tipe datanya bersifat kontinyu. Contoh : meramalkan apakah suatu produk akan laku atau tidak berdasarkan data yang ada.
6. *Clustering*
   Berbeda dengan *classification, clustering* (klasterisasi) merupakan *unsupervised classification*. Dalam klasterisasi, kelas-kelas tidak didefinisikan terlebih dahulu. Contoh : aplikasi pengelompokan dokumen untuk mesin pencarian.
7. *Outlier Analysis*
   Mengidentifikasi dan menjelaskan noise dan *outlier. Outlier* adalah objek data yang tidak memenuhi model dan persyaratan secara umum, yang berbeda dan inkonsisten dengan data set yang ada. Contoh aplikasi : *fraud detection*.
8. *Evolution analysis*
   Menggambarkan dan memodelkan regulasi atau tren untuk obyek yang berubah setiap saat. Mungkin melibatkan semua fungsionalias data mining yang lain.

2.3 Teks Mining

Teks mining merupakan pencarian pola yang menarik atau pola yang berguna pada sebuah informasi tekstual yang tidak terstruktur (teks natural language), atau bisa juga didefinisikan sebagai proses dari menganalisa teks untuk mengekstraksi informasi dengan tujuan tertentu.

Kebanyakan pendekatan teks mining menggunakan algoritma mining pada atribut yang dihubungkan pada setiap dokumen. Atribut tersebut bisa berupa ekstraksi keyword dari dokumen atau hanya daftar kata dalam dokumen yang bersangkutan.

2.4 *Text Analysis*

Pada *text analysis* sangat penting untuk dapat mengekstraksi feature dengan "benar" supaya didapatkan ketepatan dan hasil yang berguna. Pada tahap ini akan dihilangkan ambiguitas/kerancuan dari teks, dihitung frekuensi kata-kata atau frase yang menggambarkan isi dari masing-masing dokumen. Hanya kata-kata tertentu atau frase yang sesuai dengan aturan grammar yang akan diekstraksi.

Pada tahap ini akan dilakukan *parsing* atau segmentasi dokumen itu sehingga diperoleh daftar kata-kata yang ada di dalamnya. Daftar kata itu kemudian disaring dengan membuang kata-kata yang ada di daftar *stopword*. Kata-kata yang tersisa itu kemudian dihilangkan imbuhan-imbuhannya melalui proses *stemming* sehingga didapatkan daftar kata dasar yang dapat mewakili dokumen tersebut. Selain daftar kata juga akan didapatkan frekuensi dari kemunculan masing-masing kata/frase dalam suatu dokumen.

2.4.1 Kata henti (*Stopword)*

Merupakan kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan bag-of-words. Contoh stopwords adalah "at","to","an" dan seterusnya. Stopword juga didefenisikan sebagai sejumlah kata-kata yang dihilangkan sewaktu proses pengindeksan secara otomatis karena jika kata-kata ini tidak dihilangkan akan menghasilkan pengindeksan yang kurang baik. Atau ada juga yang menyebutkan bahwa *stopword* ini sebagai kata negatif atau noise.[8]

Jika kata henti atau stopword ini tidak dihilangkan maka proses kategorisasi bisa saja menghasilakan kelas suatu dokumen yang tidak tepat karena menggunakan term yang tidak relevan. Kata henti atau stopword ini juga akan mengambil bagian yang besar dalam suatu dokumen teks. Tapi dengan menghindarkan atau dengan melakukan pengapusan pada saat tahap pengindeksan akan mempercepat proses, menghemat tempat penyimpanan data dan meningkatkan akurasi dalam pengkategorisasian[8].

Stopwords removal adalah sebuah proses untuk menghilangkan kata yang 'tidak relevan' pada hasil *parsing* sebuah dokumen teks dengan cara membandingkannya dengan *Stoplist* (*stopword list*) yang ada Contoh dari stopword misalnya, kata sambung, artikel dan preposisi.

Pada umumnya untuk membangun daftar *stopword* dilakukan dengan melihat frekuensi kemunculan dari setiap kata yang ada pada dokumen. Dan untuk kata yang memiliki frekuensi yang tinggi dengan nilai thereshold tertentu maka

diperoleh daftar *stopword.* Tapi, dengan meilihat sebatas dari frekuensi kata tidak lah cukup untuk bisa mendapatkan daftar *stopword* yang baik. Hal ini bisa di ilustrasikan oleh Bucley dengan kalimat "The head and president of an American computer system company based in Washington said she expected to make a million systems by the end of the year". Semua kata dalam kalimat tersebut, dalam lebih dari 10% dari dokumen yang ada di koleksi TREC dan menghilangkan kata-kata ini maka tidak akan meninggalkan satu katapun nantinya untuk diproses. Jadi untuk menghasilkan daftar stopword yang baik, maka disampingkan berdasarkan frekuensi dari term tapi juga melihat tingkat kepentingan dari kata tersebut dalam dokumen[7].

Pada tugas akhir ini yang menjadi fokus pengerjaan adalah preproses untuk menghilangkan kata-kata yang tidak penting dalam suatu dokumen. Jadi dilakukan pembangunan daftar *stopword* yang cocok untuk dokumen yang akan diproses.

2.4.2 Pembangunan Daftar Stopword Secara Umum

Untuk memperoleh daftar *stopword* maka akan dilihat frekuensi kemunculan dari term-term yang ada dalam sebuah dokumen teks. Berikut langkah-langkah pembangunan daftar stopwordnya :

1. Semua kata-kata yang ada dalam koleksi dokumen di-extrak atau diambil semua.
2. Kemudian masing-masing kata-kata ini akan diurutkan berdasarkan frekuensi kemunculannya
3. Kata-kata yang paling sering muncul biasanya akan dianggap sebagai kata henti (*stopword*)

- Dari proses diatas akan dibentuk sejumlah kata-kata yang termasuk kata henti yang akan dibuang dalam proses berikutnya

2.4.3 Pendekatan *Term Based Random Sampling*

Salah satu algoritma yang digunakan dalam membangun daftar stopword adalah : ***"****Term Based Random Sampling Approach****".***

1. Pendekatan ini didasarkan pada tingkat seberapa pentingnya suatu term dalam suatu dokumen.
2. Dengan pendekatan ini kita dapat menentukan apakah suatu *term* tertentu merupakan suatu stopword berdasarkan tingkat kepentingannya dalam dokumen.
3. Semakin kecil tingkat kepentingan suatu term atau kata maka term tersebut akan cenderung dikategorikan sebagai *stopword*.
4. Tingkat kepentingan suatu term dapat diakses dengan menggunakan Kullback-Leibler *divergence measure*. Berikut adalah rumus yang digunakan :

$$\omega(t) = Px * \log_2 \frac{Px}{Pc}$$

5. Idenya adalah menemukan term-term yang memiliki arti yang sama dengan istilah yang diberikan, kemudian menemukan semua dokumen-

dokumen yang mengandung term-term yang diberikan dan dokumen ini digunakan sebagai dokumen sample untuk proses berikutnya dalam pembangunan daftar stopword.

6. Kemudian meng-ekstrak kata-kata yang memiliki informasi yang paling sedikit dari sampel dengan mengukur divergence dari distribusi kata-kata yang diberikan dengan himpunan dokumen sample dari distribusinya.
7. Dengan Kullback-Leibler *divergence measure*, maka bisa ditentukan tingkat kepentingan masing-masing term.

$$\omega(t) = Px * \log_2 \frac{Px}{Pc}$$

Untuk formula di atas, $Px = \frac{tfx}{lx}$ dan $Pc = \frac{F}{tokenc}$, dimana *tfx* adalah frekuensi dari query term dalam himpunan dokumen sampel, sedangakan *lx* merupakan panjang dari himpunan dokumen sampel, *F* adalah frekuensi dari wuery term di dalam keseluruhan koleksi dokumen, *tokenc* adalah jumlah total token dalam keseluruhan koleksi dokumen[8].

2.4.4 Algoritma dari Term Based Random Sampling Approach

Ulang Y kali, dimana Y adalah sebuah parameter:

1. Secara acak pilih sebuah terms dalam file kamus, kita sebut sebagai ω_{random}
2. Retrieve/ambil semua dokumen dalam corpus/kitab yang terdiri dari ω_{random}
3. Dengan menggunakan Kullback_Leibler divergence measure untuk meng-*assign* berat ke setiap term pada dokumen sampel yang diperoleh. Berat yang di-*assign* akan memberikan indikasi tentang seberapa pentingnya suatu istilah dalam suatu dokumen.
4. Bagi masing-masing berat *term* dengan maksimum berat dari semua *term*. Sebagai hasilnya nanti, semua berat dikontrol/berada dalam range [1,0]. Dengan kata lain, menormalisasikan masing-masing term dengan berat maksimum.
5. Urutkan term-term tersebut secara ascending sesuai dengan berat masing-masing term. Semakin kurang informatifnya suatu istilah, maka kurang berguna term tersebut dalam suatu dokumen dan akan semakin cocok masuk kedalam daftar *stopword*.
6. Extrak ranking X paling atas, dimana X adalah sebuah parameter.

Dengan memiliki sebuah panjang array X * Y. masing-masing elemen dalam array berasosiasi dengan sebuah berat.

7. Menyusutkan array dengan merging elemen-elemen mengandung istilah yang sama dan mengambil rata-rata berat dari term. Misalnya : jika term "retrieval" terjadi 3 kali dalam array dan beratnya adalah 0.5, 0.4, 0.3 secara terurut, kita gabungkan 3 elemen ini secara bersamaan ke dalam "one single one" dan berat dari istilah "retrieval" menjadi :
   0.5 + 0.4 + 0.3/3 = 0.4

8. Rank array yang disusutkan secara ascending bergantung berat term. Dengan kata lain urutkan array secara ascending.
9. Extrak L top-ranked terms sebagai daftar stopword. L adalah suatu parameter.

Misalkan dokumen yang digunakan (koleksi dokumen dalam berekstensi .txt)

Doc ID	Doc Text
1	I bought a new mat.
2	A cat sat on the mat.
3	The cat is white and the mat is blue

Koleksi dokumen di-parsing, untuk masing masing kata di konversi menjadi huruf kecil, dilakukan proses *stemming*, dan penghilangan tanda-tanda baca. Hasilnya adalah

idberita	term	freq
1	a	1
1	cat	1
1	sat	1
1	on	1
1	the	1
1	mat	1
2	i	1
2	bought	1
2	a	1
2	new	1
2	mat	1
3	the	2
3	cat	1
3	is	2
3	white	1
3	and	1
3	mat	1
3	blue	1

Ulangi sebanyak Y kali, dimana Y adalah parameter :

1. Secara acak pilih 1 term dari kamus, disebut sebagai ωrandom
2. Ambil semua dokumen yang mengandung ωrandom dalam korpus

Gunakan Kullbak Leibler divergence measure untuk memberikan nilai/bobot untuk masing-masing term yang ada di dokumen yang terambil (Misalkan Y=2) Misalkan ωrandom adalah **cat** (untuk Y=1), maka dokumen-dokumen yang mengandung ωrandom = cat adalah

Doc ID	Doc Text
2	A cat sat on the mat.
3	The cat is white and the mat is blue

Maka bobot untuk masing-masing term yang ada dalam dokumen dengan id = 2 dan 3 yaitu

Doc id	term	bobot
2	a	0.2748318729
2	cat	1.0084952223
2	sat	0.5042476111
2	on	0.5042476111
2	the	1.5127428334
2	mat	0.7400960144
3	is	1.0084952223
3	white	0.5042476111
3	and	0.5042476111
3	blue	0.5042476111

Misalkan ωrandom adalah **and** (untuk Y=2), maka dokumen-dokumen yang mengandung ωrandom = and adalah

Doc ID	Doc Text
3	The cat is white and the mat is blue

Maka bobot untuk masing-masing term yang ada dalam dokumen dengan id = 3 yaitu

Doc Id	term	bobot
3	the	1.0465782492
3	cat	0.4081784312
3	is	1.3710570692
3	white	0.6855285346
3	and	0.6855285346
3	mat	0.2459390212
3	blue	0.6855285346

Hasil akhir dari pengulangan ini adalah array yang berasosasi dengan bobot

Y/X										
1	0.27	1.00	0.50	0.50	1.51	0.74	1.00	0.50	0.50	0.50
2	1.04	0.40	1.37	0.68	0.68	0.245	0.68			

Kemudian dilakukan proses Normalisasi, sehingga nilainya dikontrol antar [0,1]

Y/X										
1	0.18	0.66	0.33	0.33	1	0.48	0.66	0.33	0.33	0.33
2	0.76	0.29	1	0.5	0.5	0.17	0.5			

Lakukan proses sorting

Y/X										
1	0.18	0.33	0.33	0.33	0.33	0.33	0.48	0.66	0.66	1
2	0.17	0.29	0.5	0.5	0.5	0.76	1			

Ambil X teratas : sehingga hasilnya adalah array berukuran X*Y. misalkan X=6, maka hasilnya adalah

Y/X						
1	0.18	0.33	0.33	0.33	0.33	0.33
2	0.17	0.29	0.5	0.5	0.5	0.76

Lakukan proses merging, dimana setiap term yang sama akan dimerge

0.17	0.18	0.29	0.33	0.33	0.34	0.4	0.4	0.76

Ambil L teratas, misalkan L = 5
Maka term-term yang berasosasi terhadap bobot diatas sebagai daftar *stopword* adalah Mat, a, cat, sat, on

2.5 Kategorisasi

Kategorisasi adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk memperkirakan kelas dari suatu objek yang labelnya tidak diketahui.

Pada klasifikasi terdapat dua proses, yaitu:

1. Proses *Learning* atau *Training*
   Proses ini bertujuan untuk membangun model atau fungsi yang akan digunakan untuk memprediksi label atau kelas dari objek yang belum diketahui label atau kelasnya. Pada proses ini digunakan objek yang telah diketahui label atau kelas datanya yang disebut *training set*. Performansi model dapat diukur dengan menguji model dengan data *training* itu sendiri disebut performansi *training.*
2. Proses *Testing*.
   Pada proses ini akan dilakukan tes dengan menggunakan objek yang telah diketahui label atau kelas yang disebut *test set*, dengan tujuan mengetahui keakurasian dan *error-rate* dari model yang telah dibentuk. Objek yang digunakan pada proses ini berbeda dengan objek yang digunakan pada proses learning. Keakurasian objek yang digunakan pada proses tes ini dilakukan dengan cara membandingkan label atau kelas yang telah diketahui dengan label atau kelas hasil prediksi dari model atau fungsi.

2.5.1 Kategorisasi Teks

Klasifikasi teks merupakan salah satu contoh penggunaan metode klasifikasi. Pada klasifikasi teks akan dilakukan pengklasifikasian suatu dokumen, kedalam satu kelas kategori. Proses pengklasifikasian dokumen juga harus melalui proses *training* terlebih dahulu. Sebelum dilakukan *training*, dilakukan *preprocessing* terlebih dahulu. *Preprocessing* adalah proses yang dilakukan pada data *training* agar data tersebut sesuai dengan masukan yang diminta oleh algoritma *learning.* Terdiri dari pembuangan *stopword,* pembobotan kata*,* dan pemilihan atribut, yaitu kata-kata yang paling sering muncul dalam seluruh data *training*, dilakukan dengan memakai metode *feature selection*. Baru kemudian

atribut-atribut ini menjadi masukan untuk algoritma *learning* sehingga dapat dihasilkan model *learning* yang dapat mengklasifikasi data.

2.5.2 Evaluasi

Keefektifan untuk klasifikasi teks, didefinisikan sebagai pengukuran kemampuan sistem untuk mengkategorikan dokumen ke dalam kelompok yang sesuai. Model tabel kontingensi adalah pengukuran yang paling banyak digunakan untuk mengukur keefektifan.

2.5.3 Tabel Kontingensi

Didefinisikan sebagai tabel statistik yang memperlihatkan frekuensi yang diobservasi dari elemen data yang diklasifikasi berdasar dua variabel, baris mengindikasikan satu variabel dan kolom mengindikasikan variabel yang satunya. Digunakan dua pengukuran penting untuk keefektifan sistem dari jenis tabel ini, yaitu recall dan precision.

Contoh tabel kontingensi yang sederhana dari sistem pengambilan keputusan biner dapat dilihat pada Tabel 2.1 Sistem membuat n keputusan biner, dimana masing-masing memiliki satu jawaban yang tepat, baik Yes atau No. Setiap jawaban menerangkan jumlah keputusan tiap tipe. Misalnya a adalah jumlah dimana sistem menghasilkan Yes, dan Yes merupakan jawaban yang tepat. Dan c adalah jumlah ketika sistem menjawab Yes, padahal jawaban tepatnya adalah No.

Tabel 2-1 : Tabel kontigensi untuk kumpulan keputusan biner

	PREDICTED CLASS		
ACTUAL CLASS		**Class = Yes**	**Class = No**
	Class = Yes	A	b
	Class = No	C	d

Berikut ini dua pengukuran penting untuk keefektifan yang didapat dari tabel kontingensi:

- ***Recall*** dihitung untuk mengevaluasi seberapa besar *coverage* suatu model dalam memprediksi suatu kelas tertentu. *Recall* didapat dari jumlah kategori yang tepat diklasifikasi dibagi total jumlah kategori benar yang seharusnya :

$$R = \frac{a}{a + b} \tag{2.1}$$

- ***Precision*** dihitung untuk mengevaluasi seberapa baik ketepatan model dapat memprediksi suatu kelas. *Presicion* didapat dari jumlah kategori yang tepat diklasifikasi dibagi jumlah kategori seluruhnya yang harus diklasifikasi :

$$P = \frac{a}{a + c} \tag{2.2}$$

Pengukuran yang efektif dapat melakukan kesalahan apabila pemeriksaan recall dan precision dilakukan sendiri. Precision mungkin dikorbankan untuk

mendapat recall yang tinggi. Untuk menyimpulkan dan membuat pengukuran yang composite, digunakan *F-measure* sebagai kriteria evaluasi. *F-Measure* merupakan matriks gabungan kombinasi terbaik antara *recall* dan *presicion* :

$$F\text{-}Measure = \frac{2a}{2a + b} \tag{2.3}$$

2.5.4 Estimasi Akurasi

Ukuran lain yang umumnya digunakan pada mesin learning adalah akurasi (A) dan *error* (E) tidak digunakan secara luas pada klasifikasi teks. Akurasi digunakan untuk menghitung seberapa akurat model yang telah dihasilkan dapat memprediksi data. Sesuai tabel kontingensi pada tabel 2.1, akurasi dan error didefinisikan sebagai berikut :

$$\mathrm{A} = \frac{a + d}{a + b + c + d} \tag{2.4}$$

$$\mathrm{E} = \frac{b + c}{a + b + c + d} = 1 - \mathrm{A} \tag{2.5}$$