Text Mining

Final Project : Milkha Harlian Ch.

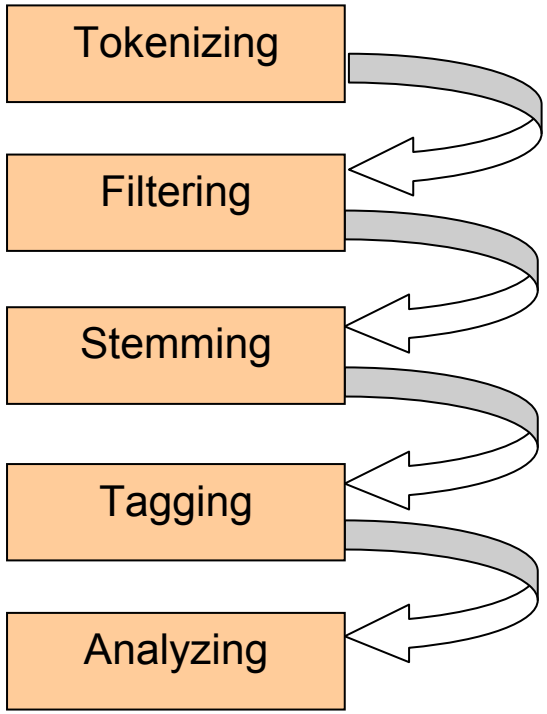
Referensi : Raymond J. Mooney. *CS 391L: Machine Learning Text Categorization*. University of Texas at Austin, 2006

Definisi Text Mining

- *Text mining* memiliki definisi 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

Tahapan dalam Text Mining

- Tahapan yang dilakukan secara umum adalah:

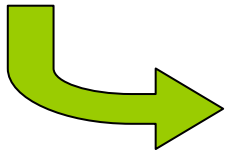


Tokenizing

- Tahap *tokenizing* adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya.
- Contoh dari tahap ini adalah sebagai berikut:

Manajemen pengetahuan adalah sebuah konsep baru di dunia bisnis.

[Teks Input]



manajemen
pengetahuan
adalah
sebuah
konsep
baru
di
dunia
bisnis

[Hasil Token]

Filtering

- Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil token.
- Bisa menggunakan algoritma *stop list* (membuang kata yang kurang penting) atau *word list* (menyimpan kata penting).
- Contoh dari tahap ini adalah sebagai berikut:

manajemen
pengetahuan
adalah
sebuah
konsep
baru
di
dunia
bisnis

[Hasil Token]

manajemen
pengetahuan
konsep
baru
dunia
bisnis

[Hasil Filter]

Stemming

- Tahap *stemming* adalah tahap mencari root kata dari tiap kata hasil filtering.
- Contoh dari tahap ini adalah sebagai berikut:

learning
using
text
mining

[Hasil Filter]

learn
use
text
mine

[Hasil *Stemming*]

Tagging

- Tahap *tagging* adalah tahap mencari bentuk awal/root dari tiap kata lampau atau kata hasil *stemming*.
- Contoh dari tahap ini adalah sebagai berikut:

was
used
stori

[Hasil *Stemming*]

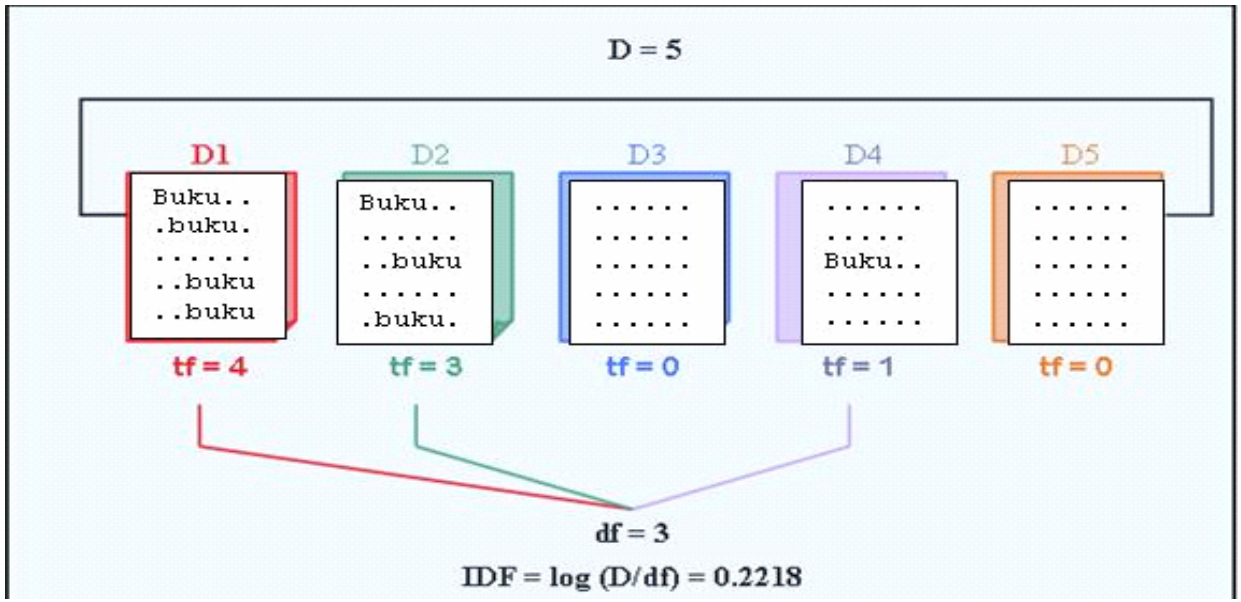
be
use
story

[Hasil *tangging*]

Analyzing

- Tahap Analyzing merupakan tahap penentuan seberapa jauh keterhubungan antar kata-kata antar dokumen yang ada.

Ilustrasi Algoritma Text Mining



D1, D2, D3, D4, D5 = dokumen

tf = banyak kata yang dicari pada sebuah dokumen

D = total dokumen

df = banyak dokumen yang mengandung kata yang dicari

Algoritma TF/IDF

- Formula yang digunakan untuk menghitung bobot (w) masing-masing dokumen terhadap kata kunci adalah
- $W_{d,t} = tf_{d,t} * IDF_t$
- Dimana:
- d = dokumen ke-d
- t = kata ke-t dari kata kunci
- W = bobot dokumen ke-d terhadap kata ke-t
- Setelah bobot (w) masing-masing dokumen diketahui, maka dilakukan proses *sorting*/pengurutan dimana semakin besar nilai w, semakin besar tingkat similaritas dokumen tersebut terhadap kata yang dicari, demikian sebaliknya.

Ilustrasi TF/IDF

Kata kunci (kk) = pengetahuan logistik
Dokumen 1 (D1) = Manajemen transaksi logistik
Dokumen 2 (D2) = Pengetahuan antar individu
Dokumen 3 (D3) = Dalam manajemen pengetahuan terdapat
transfer pengetahuan logistik
Jadi jumlah dokumen (D) = 3

→ Setelah melalui proses filtering, maka kata antar pada dokumen 2 serta kata dalam dan terdapat pada dokumen 3 dihapus

Tabel perhitungan TF-IDF

token	tf				df	D/df	IDF	W			
	kk	D1	D2	D3			log(D/df)	kk	D1	D2	D3
manajemen	0	1	0	1	2	1.5	0.176	0	0.176	0	0.176
transaksi	0	1	0	0	1	3	0.477	0	0.477	0	0
logistik	1	1	0	1	2	1.5	0.176	0.176	0.176	0	0.176
transfer	0	0	0	1	1	3	0.477	0	0	0	0.477
pengetahuan	1	0	1	2	2	1.5	0.176	0.176	0	0.176	0.352
individu	0	0	1	0	1	3	0.477	0	0	0.477	0
Total	**0.352**	**0.829**	**0.653**	**1.181**							

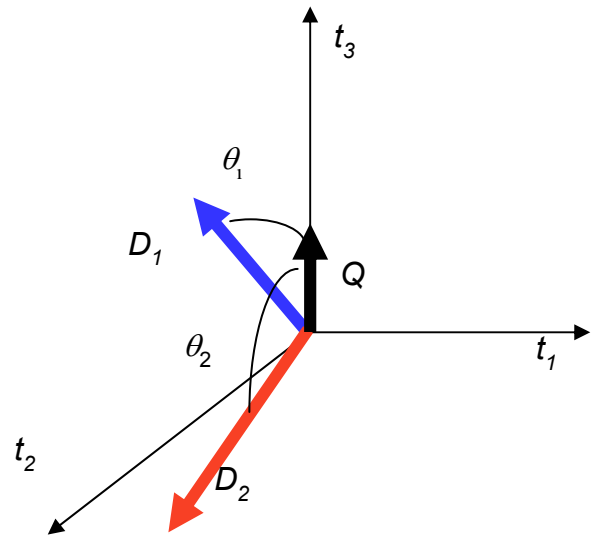
bobot (*w*) untuk D1 = 0.176 + 0 = 0.176
bobot (*w*) untuk D2 = 0 + 0.176 = 0.176
bobot (*w*) untuk D3 = 0.176 + 0.352 = 0.528

Analisa TF/IDF

- Dari contoh studi kasus di atas, dapat diketahui bahwa nilai bobot (w) dari **D1** dan **D2** adalah **sama**.
- Apabila diurutkan maka proses *sorting* juga tidak akan dapat mengurutkan secara tepat, karena nilai w keduanya sama.
- Untuk mengatasi hal ini, digunakan algoritma dari *vector-space model*

Vector Space Model

- Ide dari metode ini adalah dengan menghitung nilai cosinus sudut dari dua vektor, yaitu W dari tiap dokumen dan W dari kata kunci



t = kata di *database*

D = dokumen

Q = kata kunci

Rumus: CosSim(***dj***, ***q***) =

$$\frac{\vec{d_j} \cdot \vec{q}}{\left|\vec{d_j}\right| \cdot \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t}(w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^{\ 2} \cdot \sum_{i=1}^{t} w_{iq}^{\ 2}}}$$

Tabel perhitungan *Vector-Space Model*

token	kk	D1	D2	D3	Kk*D1	Kk*D2	kk*D3
manajemen	0	0.031	0	0.031	0	0	0
transaksi	0	0.228	0	0	0	0	0
logistik	0.031	0.031	0	0.031	0.031	0	0.031
transfer	0	0	0	0.228	0	0	0
pengetahuan	0.031	0	0.031	0.124	0	0.031	0.062
individu	0	0	0.228	0	0	0	0
	Sqrt(kk)	Sqrt(Di)			Sum(kk dot Di)		
	0.249	**0.539**	**0.509**	**0.643**	**0.031**	**0.031**	**0.093**

Perhitungan

- **Sqrt(kk)** = Sqrt($\sum_{j=1}^{n} kk_j^2$)
  dimana j = kata di database
- Misalnya untuk Sqrt(kk) = Sqrt($\sum_{j=1}^{n} kk_j^2$)

  = $\sqrt{(0+0+0.031+0+0.031+0)}$

  = $\sqrt{0.062}$ = 0.249
- **Sqrt(D_i)** = Sqrt($\sum_{j=1}^{n} D_{i,j}^2$)
  dimana j = kata di database
- Misalnya untuk Sqrt(D_2) = Sqrt($\sum_{j=1}^{n} D_{2,j}^2$)

  = $\sqrt{(0+0+0+0+0.031+0.228)}$

  = $\sqrt{0.259}$ = 0.509

Perhitungan

- **Sum(kk dot D_i) =** $\sum_{j=1}^{n} kk_j D_{i,j}$

  dimana j = kata di database
- Misalnya untuk Sum(kk dot D_3) = $\sum_{j=1}^{n} kk_j D_{3,j}$

  = 0 + 0 + 0.031 + 0 + 0.062 + 0

  = 0.093

Perhitungan

- Selanjutnya menghitung nilai Cosinus sudut antara vektor kata kunci dengan tiap dokumen dengan rumus:

  Cosine (D_i) = sum(kk dot D_i) / [sqrt(kk) * sqrt(D_i))]
- Misalnya untuk D_3 maka:
- **Cosine (D_3)**= sum(kk dot D_3) / [sqrt(kk) * sqrt(D_3))]

  = 0.093 / [0.249 * 0.643]

  = 0.581

Analisa Vector Space Model

- Demikian juga untuk Cosine dari D_1 dan D_2. Sehingga hasil yang diperoleh untuk ketiga dokumen di atas adalah seperti berikut ini.
- **Tabel 2.3** Tabel hasil *Vector-Space Model*

	D1	D2	D3
Cosine	**0.231**	**0.245**	**0.581**
	Rank 3	Rank 2	Rank 1

- Dari hasil akhir (Cosine) maka dapat diketahui bahwa document 3 (D3) memiliki tingkat similaritas tertinggi kemudian disusul dengan D2 lalu D1.