BAB II

TINJAUAN PUSTAKA

II.1 Perbandingan Penelitian

Berikut ini adalah perbandingan penelitian yang akan dilakukan dengan penelitian-penelitian sebelumnya.

Tabel II.1 Penelitian Terdahulu

No	Nama Peneliti	Judul	Metode	Hasil
1	(Mochamad Tri Anjasmoros, 2020)	Analisis Sentimen Aplikasi Go-Jek Menggunakan Metode *Svm* Dan *Nbc* (Studi Kasus: Komentar Pada *Play Store*)	*Support Vector Machine* dan *Naive Bayesian Clasifier*	*Score accuracy, recall* dan *precision* paling tinggi adalah 0,897 **(89.7%)** dengan data testing sebanyak 50% dan 50% da-ta training dari 2.000 komentar menggunakan metode *SVM* kernel linear
2	(Muzakki et al., 2020)	*Klasifikasi* dan Analisa Sentimen Kuesioner Fasilitas dan Layanan untuk Universitas Qomaruddin Gresik	*Algoritma Naive Bayes*	nilai akurasi untuk *klasifikasi* ketegori fasilitas dan layanan memiliki nilai rata-rata **83.27%** sedangkan untuk *klasifikasi sentimen* memiliki nilai rata-rata **80.67 %** sehingga dapat dikatakan performa sistem cukup stabil

Tabel II.1 Penelitian Terdahulu (lanjutan)

No	Nama Peneliti	Judul	Metode	Hasil
3	(Randhika et al., 2021)	*Implementasi Algoritma Complement* dan *Multinomial Naive Bayes Classifier* Pada *Klasifikasi* Kategori Berita Media *Online*	*Algoritma Complement* dan Multinomial *Naive Bayes Classifier*	Tingkat akurasi kombinasi kedua metode tersebut **90.13%**
4	(Noviriandini et al., 2022)	Klasifikasi *Support Vector Machine* Berbasis *Particle Swarm Optimization* Untuk Analisa Sentimen Pengguna Aplikasi Pedulilindungi	*Support Vector Machine* Berbasis *Particle Swarm Optimization*	Hasil tes dengan nilai akurasi = 93,0% dan nilai AUC = 0,977 **(97.7%).**
5	(Safra et al., 2020)	Analisa Sentimen Persepsi Masyarakat Terhadap Pemindahan Ibukota Baru Di Kalimantan Timur Pada Media Sosial Twitter	*Algoritma Naive Bayes Classifier*	Data yang digunakan sebesar 200 data *tweets*, yang terdiri dari 159 data training dan 41 data testing menghasilkan akurasi sebesar **78%.**
6	(Dwitiyanti et al., 2021)	*Analisis Sentimen Twitter* Kebiasaan *New* Normal	*Algoritma Naive Bayes Classifier*	Aanalisis sentiment *twitter* diperoleh **57%** menghasilkan nilai positif, 35% menghasilkan netral, dan negatif sebesar 8%.

Tabel II.1 Penelitian Terdahulu (lanjutan)

No	Nama Peneliti	Judul	Metode	Hasil
7	(Maulana et al., 2020)	Analisa Sentimen *Cyberbullying* Di Jejaring Sosial *Twitter* Dengan *Algoritma Naive Bayes*	*Algoritma Naive Bayes*	Hasil pengujian dengan data uji real time pada tanggal 12 Mei 2020 pukul 01.00 WIB mendapatkan
8	(Duei Putri et al., 2022)	Analisis Sentimen Kinerja Dewan Perwakilan Rakyat (Dpr) Pada *Twitter* Menggunakan Metode *Naive Bayes Classifier*	Metode *Naive Bayes Classifier*	Hasil dari penelitian ini didapatkan bahwa DPR mendapatkan 95 *tweet* positif dengan polaritas 0.75 atau 75% sentimen positif, 693 *tweet* netral dengan polaritas 0.79 atau 79% sentimen netral dan 758 *tweet* negatif dengan polaritas 0.82 atau 82% sentimen negatif dengan *accuracy score* 0.8 atau 80% berdasarkan data testing sebanyak 20%
9	(Darwis, 2020)	Penerapan *Algoritma Svm* Untuk *Analisis Sentimen* Pada Data *Twitter* Komisi Pemberantasan Korupsi Republik Indonesia	Algoritma Svm	Nilai Akurasi sebesar 82% dan menghasilkan sentimen dengan label negatif lebih besar dengan jumlah 77%, label positif 8% dan label netral 25%. Kata

Tabel II.1 Penelitian Terdahulu (lanjutan)

No	Nama Peneliti	Judul	Metode	Hasil
10	(Fahrur Rozi, 2021)	*Analisis Sentimen* Pada *Twitter* Mengenai Pasca Bencana Menggunakan Metode *Naive Bayes* Dengan Fitur *N-Gram*	Metode *Naive Bayes* Dengan Fitur *N-Gram*	Nilai akurasi untuk unigram yaitu sebesar 93.33% dan bigram sebesar 86.67%. Jadi untuk nilai akurasi unigram lebih tinggi daripada bigram

II.2 *Text Mining*

Proses menggali informasi yang dilakukan oleh seorang user yang berinteraksi dengan sekumpulan dokumen dengan menggunakan tools analisis Text mining adalah proses ekstraksi pola (informasi dan pengetahuan yang berguna) dari sejumlah sumber data melalui identifikasi pola yang menarik (Fani et al., 2021)

Dalam konsep *Text Mining* Pekerjaan yang dilakukan secara garis besar adalah penggalian prediktif (*predictive mining*) dan penggalian deskriptif (*descriptive mining*). Pekerjaan *predictive mining* meliputi pengelompokan dokumen ke dalam kategori-kategori, lalu menggunakan informasi tersebut untuk membuat keputusan. Fungsi dari text mining biasanya digunakan dalam klasifikasi dokumen tekstual dimana dokumen-dokumen tersebut akan diklasifikasikan sesuai dengan topik dokumen tersebut. Adanya bantuan dari text mining, maka suatu artikel dapat diketahui jenis kategorinya melalui kata-kata yang terdapat pada artikel tersebut (Sari et al., 2021).

Sebagai contoh kepuasan pelanggan terhadap suatu produk dapat diketahui melalui dokumen komentar pembelian, sehingga komentar pelanggan di pembelian yang akan datang dapat diprediksi. Untuk dapat memperoleh informasi akhir yang berguna bagi pemilik data, *Text Mining* harus melalui tiga tahap yaitu,

1. *Preprocessing*

   Tahap ini merupakan proses awal yang akan mentransformasikan suatu data masukan menjadi data dengan format yang sesuai dan siap untuk dapat diproses

2. Penyusunan Vektor

   Untuk dapat dipahami oleh sistem operasi *text mining*, sebuah vektor representasi atas token-token kata perlu dibuat berdasarkan kemunculan kata tersebut dalam dokumen

3. Ekstraksi Informasi

   Metode ekstraksi informasi yang digunakan pada penelitian ini adalah klasifikasi yang membagi objek ke dalam kategori yang telah ditentukan (*supervised method*). Pendekatan model klasifikasi yang digunakan adalah teknik *statistik/machine learning*. Pendekatan *machine learning* akan dipakai dalam penelitian ini dimana mesin akan mengoperasikan model yang belajar dari contoh dokumen yang terklasifikasi (Munthe et al., 2022).

II.3 Analisis sentimen

Ansalisis Sentiment merupakan proses dalam menganalisis, memahami, dan mengklasifikasi pendapat, evaluasi, penilaian, sikap, dan emosi terhadap suatu entitas tertentu seperti produk, jasa, organisasi, individu, peristiwa, topik, guna mendapatkan informasi. Analisis *mood* pada opini disebut Analisis Sentimen, yang

merujuk kepada analisis secara otomatis mengenai teks yang evaluatif dengan berfokus kepada klasifikasi teks berdasarkan polaritas yang dimiliknya. Klasifikasi data pada kelompok sentimen tertentu (positif atau negatif) dilakukan dengan membangun model probabilitas kemunculan suatu kata dalam dokumen yang telah dikelompokkan sebelumnya menurut Muthia S (Maulana et al., 2020).

Besarnya manfaat dan pengaruh dari *Sentiment Analysis*, menjadikan penelitian ataupun aplikasi mengenai *Sentiment Analysis* berkembang pesat, bahkan perusahaan yang menggunakan *Sentiment Analysis* di Amerika Serikat ada kurang lebih 20 - 30 perusahaan untuk memperoleh informasi tentang sentimen masyarakat terhadap pelayanan perusahaan atau institusi pendidikan tinggi (Muzakki et al., 2020).

Sentiment Analysis Pada dasarnya merupakan klasifikasi, tetapi pada kenyataannya tidak semudah proses klasifikasi biasa karena terkait penggunaan bahasa. Terdapat problematis dalam penerapan kata, tidak terdapat adanya intonasi dalam sebuah teks, dan perkembangan dari bahasa itu sendiri (Noviriandini et al., 2022).

II.3.1 Preprocessing

Preprocessing berfungsi mengubah data teks yang tidak terstruktur menjadi suatu data yang terstruktur. Tahap processing ini akan mengurangi teks-teks secara signifikasi yaitu teks yang tidak berpengaruh terhadap dokumen. Dimana penjelasan dari tahap-tahap tersebut adalah sebagai berikut:

1. *Cleaning*

Cleaning adalah proses membersihkan kalimat dari kata yang tidak diperlukan untuk mengurangi noise seperti karakter HTML, RT, ikon emosi, hashtag (#), username (@), url (http://situs.com), email (nama@situs.com), simbol dan tanda baca.

2. *Case Folding*

Case folding adalah proses dimana mengubah semua karakter *uppercase* atau huruf besar menjadi *lowercase* atau huruf kecil.

3. *Tokenisasi*

Tokenisasi adalah memisahkan deretan kata di dalam kalimat, paragraf atau halaman menjadi *token* atau potongan kata tunggal atau *termmed word*.

4. *Normalisasi*

Normalisasi adalah mengganti kata yang mengandung sentimen dari kata tidak baku menjadi kata baku dan yang sesuai dengan sinonim kata-nya, yang telah tersimpan dalam database sinonim.

5. *Stopword Removal*

Stopword removal adalah menghilangkan sejumlah kelas kata penghubung ataupun yang jumlahnya banyak namun tidak mempengaruhi konten dokumen secara keseluruhan sebagai bagian dari *pre-processing*. Tahap ini biasanya dilakukan untuk lebih meningkatkan performa sistem agar sistem bisa secara efektif dimanfaatkan dalam mengolah konten yang benar-benar dianggap penting saja.

6. Stemming

Stemming adalah proses mengubah kata yang memiliki imbuhan menjadi kata dasar. *Stemming* disini menggunakan kamus daftar kata berimbuhan yang memiliki kata dasarnya dengan cara membandingkan kata-kata yang ada dalam dokumen.

7. Convert Negation

Convert Negation adalah Pada tahap ini, setiap tweets yang mengandung kata-kata yang bersifat negasi akan diubah nilai sentimennya. Kata-kata yang bersifat negasi seperti “bukan”, “tidak”, “enggak”, “ga”, “jangan”,“nggak”, “tak”, dan “gak”. *Convert negation* dilakukan jika terdapat kata negasi sebelum kata yang bernilai positif, maka kata tersebut akan diubah nilainya menjadi negatif dan begitupun sebaliknya. Langkah-langkah pada tahap *convert negation* adalah sebagai berikut:

a. Kata yang digunakan adalah hasil dari *stopword removal*.
b. Setiap kata akan diperiksa dan dibandingkan dengan kumpulan kata-kata yang bersifat negasi pada *database*.
c. Jika setelah kata yang bersifat negasi terdapat kata yang termasuk sentimen positif, maka sentimen tersebut akan diubah menjadi negatif.
d. Jika setelah kata yang bersifat negasi terdapat kata yang termasuk sentimen negatif, maka sentimen tersebut akan diubah menjadi positif (Dwitiyanti et al., 2021).

II.3.2 Algoritma *Stemming* Nazief Andriani

Algoritma ini menggunakan beberapa ketentuan morfologi untuk menghilangkan affiks (awalan, imbuhan, akhiran, dll) dari sebuah kata dan

kemudian mencocokannya dalam kamus akar kata (kata dasar). Jadi dasar utama pada algoritma ini yaitu daftar kata dasar. Langkah awal yang dilakukan adalah mengumpulkan daftar kata dasar dalam bahasa Indonesia. Semakin daftarnya lengkap, semakin tinggi pula akurasi algoritma ini. Algoritma ini memiliki tahapan sebagai berikut

1. Cari kata yang akan di-stem di dalam kamus, jika kata tersebut ditemukan maka kata tersebut adalah kata dasar dan algoritma berhenti. Jika tidak ada maka lanjutkan ke langkah-2.
2. Hilangkan *inflectional suffix* (imbuhan infleksional) yaitu (“-lah”, “-kah”, “-tah”, “-ku”, “-mu”, “-nya”).
3. Hapus derivation suffix (imbuhan turunan) yaitu (“-i”, “-an”, atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah-3a.
    a. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k” maka “-k” juga ikut dihapus. Jika kata tersebut telah ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah-3b.
    b. Akhiran yang dihapus (“-i”, “-an”, atau “-kan”) dikembalikan dan lanjut ke langkah-4.
4. Hapus derivation prefix (awalan turunan) yaitu (“be-“, “di-“, “ke-“, “me-“, “pe-“, “se-“, “te-“). Jika pada langkah 3 ada suffix yang dihapus maka pergi ke langkah-4a, jika tidak maka pergi ke langkah-4b.
    a. Periksa tabel kombinasi awalan-akhiran yang tidak diizinkan. Jika ditemukan maka algoritma berhenti, jika tidak pergi ke langkah-4b.

b. Untuk i=1 sampai 3, tentukan tipe awalan kemudian hapus awalan. Jika kata dasar belum ditemukan juga lakukan langkah-5, jika sudah maka algoritma berhenti.

   Catatan: Jika awalan kedua sama dengan awalan pertama maka algoritma berhenti.

c. Lakukan *recording*.

d. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai kata dasar, proses selesai.

Tabel II.2 Tabel Kombinasi Awalan dan Akhiran yang Tidak Diijinkan

Awalan	Akhiran yang tidak diijinkan
be-	-i
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan

Tabel II.3 Tabel Aturan Peluruhan Kata Dasar

Awalan	Peluruhan
berV…	ber-V..| be-rV..
Belajar	bel-ajar
berClerC2	be-ClerC2.. dimana Cl!= {‘r’|’l’}
Awalan	Peluruhan
terV…	ter-V.. | te-rV..
terCer…	ter-Cer… dimana C!=’r’
teClerC2	te-CleC2… dimana Cl!=’r’
me{I|r|w|y}V…	me-{I|r|w|y}V…
mem{b|f|v}…	mem-{b|f|v}…

Tabel II.3 Tabel Aturan Peluruhan Kata Dasar (Lanjutan)

mempe…	m-pe..
mem{r|V|V}…	me-m{r|V|V}…| me-p{r|V|V}…
men{c|d|j|z}…	men-{c|d|j|z}…
menV…	me-nV…| me-tV…
meng{g|h|q|k}…	meng-{g|h|q|k}…
mengV…	meng-V…| meng-kV…
mengeC…	meng-C…
menyV…	me-ny…| men-sV…

Tipe awalan ditentukan melalui langkah-langkah sebagai berikut:

1. Jika awalannya adalah: “di-“, “ke-“, atau “se-“, maka tipe awalannya secara berturut-turut adalah “di-“, “ke-“, atau “se-“.
2. Jika awalannya adalah: “te-“, “me-“, “be-“, atau “pe-“ maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.
3. Jika dua karakter pertama bukan “di-“, “ke-“, “se-“, “te-“, “be-“, “me-“, atau “pe-“ maka berhenti.
4. Jika tipe awalan adalah “none” maka berhenti. Hapus awalan jika ditemukan.

Untuk mengatasi keterbatasan yang ada, maka ditambahkan aturan-aturan dibawah ini Aturan untuk reduplikasi.

a. Jika kedua kata yang dihubungkan penghubung adalah kata yang sama maka *root word* adalah bentuk tunggalnya, contoh: “buku-buku” *root wood*-nya adalah “buku”.
b. Kata lain misalnya “berbalas-balasan”, “bolak-balik”, dan “seolah-olah”. Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Jika

keduanya memiliki *root word*-nya yang sama maka diubah menjadi bentuk tunggal, contoh: kata “berbalas-balasan”, “berbalas” dan “balasan” memiliki *root word* yang sama yaitu “balas”, maka *root wood* “berbalas-balasan” adalah “balas”. Sebaliknya, pada kata “bolak-balik”, “bolak” dan “balik” memiliki *root word* yang berbeda, maka *root word*-nya adalah “bolak-balik”. Tambahan bentuk awalan dan akhiran serta aturannya, sebagai contoh : awalan “mem-“, kata yang diawali dengan awalan “ memp-“ memiliki tipe awalan “mem-“. Tipe awalan “meng-“, kata yang diawali dengan awalan “mengk-“ memiliki tipe awalan “meng-“.

II.3.3 Membangun index melalui proses *sorting* dan *grouping*

Dalam membangun sebuah index tahap utamanya adalah mengurutkan (sorting) hasil stemming sehingga daftar term tersebut terurut berdasarkan abjad. Term yang sama selanjutnya dikelompokkan (grouping) sehingga menjadi satu dan dihitung frekuensi kemunculannya di tiap-tiap dokumen.

II.3.4 Term Frequency (TF)

TF adalah frekuensi dari kemunculan sebuah term dalam dokumen yang bersangkutan. Semakin besar jumlah kemunculan suatu term (TF tinggi) dalam dokumen, maka semakin besar pula bobotnya atau akan memberikan nilai kesesuaian yang semakin besar. Pada Term Frequency (TF), terdapat beberapa jenis formula yang dapat digunakan (Munthe et al., 2022) :

1. TF biner (binary TF), hanya memperhatikan suatu kata atau term apakah ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak diberi nilai nol (0).

2. TF murni (raw TF), nilai TF diberikan berdasarkan jumlah kemunculan dari suatu term di dokumen. Sebagai contoh, jika muncul lima (5) kali maka kata tersebut akan bernilai lima (5).
3. TF logaritmik, hal ini untuk menghindari dominansi dokumen yang mengandung sedikit term dalam query, namun mempunyai frekuensi yang tinggi.
4. TF normalisasi, menggunakan perbandingan antara frekuensi sebuah term dengan nilai maksimum dari kumpulan frekuensi term atau keseluruhan term yang ada pada suatu dokumen.

II.4 *Multinomial Naive Bayesian Classifier*

Multinomal Naive Bayes Classifier adalah salah satu teknik klasifikasi teks yang paling umum digunakan dalam sentimen analisis. Teknik ini didasarkan pada asumsi bahwa setiap kata dalam sebuah dokumen independen dari kata-kata lainnya.

Nilai probabilitas sebuah dokumen d berada di kelas c dihitung dengan:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \qquad (2.1)$$

1. P(c), adalah *prior probability* dari sebuah dokumen yang terdapat dalam kelas c. Bila term dari sebuah dokumen tidak memberikan petunjuk yang jelas untuk satu kelas dibandingkan dengan kelas lainnya, maka dipilih satu kelas yang memiliki *prior probability* yang tertinggi.
2. $<t_1,t_2, ..., tn_d>$, adalah kumpulan token dalam dokumen d yang merupakan bagian dari *vocabulary* yang digunakan untuk mengklasifikasi dan n_d adalah

jumlah token tersebut didalam dokumen d. Contoh, <t_1,t_2, ..., tn_d> untuk dokumen dengan satu kalimat *Beijing and Taipei join the WTO*, menjadi <*Beijing, Taipei, join, WTO*>, dengan n_d = 4, jika *term and* dan *the* dianggap sebagai *stop words*.

Untuk memperkirakan prior probability P(c) digunakan persamaan sebagai berkut:

$$P(c)\frac{Nc}{N} \qquad (2.2)$$

Keterangan:

1. Nc = Jumlah dari dokumen training dalam kelas c.
2. N = Jumlah keseluruhan dokumen training dari seluruh kelas.

Untuk memperkirakan conditional probability P(t|c) persamaan yang digunakan, yaitu:

$$P(t_k|c) = \frac{T_{c_t}}{\sum t' \in v\, T_{c_{t'}}} \qquad (2.3)$$

Keterangan:

1. T_{c_t} = Jumlah kemunculan term t dalam sebuah dokumen training dari kelas c.
2. $\sum t' \in V\, T_{c_{t'}}$ = Jumlah total dari keseluruhan term yang terdapat dalam sebuah dokumen training dari kelas c.

Masalah dari proses perkiraan nilai *conditional probabilities* pada persamaan (2.3) adalah diperoleh nilai nol (0) dari sebuah kombinasi *(term|class)* yang tidak terdapat dalam data *training*. Berdasarkan contoh diatas, bila *term WTO* dalam data training hanya terdapat dalam dokumen China, maka perkiraan untuk kelas-kelas lainnya, misalnya UK, akan bernilai nol (0):

$$P(WTO|UK) = 0 \qquad (2.4)$$

Untuk menghilangkan nilai nol (0), digunakan *add-one* atau *Laplace smoothing.* Proses ini menambahkan nilai satu (1) pada setiap nilai Tct dari perhitungan *conditional probabilities*. Sehingga persamaan untuk *conditional probabilities* menjadi:

$$P(t_k|c) = \frac{T_{c_t} + 1}{(\sum t' \in v\, T_{c_{t'}}) + B'} \qquad (2.5)$$

Keterangan:

B' = jumlah keseluruhan term unik dari seluruh kelas.

Untuk mendapatkan nilai akhir probabilitas pada dokumen data yang diuji, apakah dokumen uji tersebut termasuk dalam kelas negatif, positif atau negatif digunakan persamaan:

$$P = \frac{N_c}{N}\ X\ \frac{T_{c_t} + 1}{(\ t' \in v\, T_{c_{t'}}) + B'} \qquad (2.6)$$

II.5 Evaluasi Sistem dengan *F-score*

Evaluasi sistem pada penelitian ini adalah dengan menerapkan aturan variabel pada Gambar II.1 menggunakan rumus umum perhitungan *precision*, *recall*, dan *F-score* pada persamaan 2.7, 2.8, dan 2.9.

Table II.4 Perhitungan *F-Score*

		Sentimen Manual	
		Relevan	Tidak Relevan
Sentimen Sistem	Relevan	True Positive (TP)	False Positive (FP)
	Tdak Relevan	False Negative (FN)	True Negative (TN)

$$Recall = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad (2.7)$$

$$Precision = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \quad (2.8)$$

$$F\text{-}score = \frac{\boldsymbol{2\ x\ (Precision\ x\ Recall)}}{Precision+Recall} \quad (2.9)$$

Keterangan:

1. *True Positive* (TP) adalah nilai prediksi bernilai benar dan nilai sebenarnya bernilai benar.
2. *False Positive* (FP) adalah nilai prediksi bernilai benar dan nilai sebenarnya bernilai salah.
3. *False Negative* (FN) adalah nilai prediksi bernilai salah dan nilai sebenarnya bernilai salah.
4. *True Negative* (TN) adalah nilai prediksi bernilai salah dan nilai sebenarnya bernilai benar.

II.6 Algoritma *K-Nearest Neighboor*

Algoritma *K-Nearest Neighboor* merupakan salah satu metode klasifikasi yang digunakan untuk pemecahan masalah pada bidang *Data Mining*. Sama halnya dengan beberapa metode lainnya yang ada pada metode klasifikasi, algoritma ini memiliki ciri yaitu dengan pendekatan untuk mencari kasus dengan menghitung kedekatan kasus yang baru dengan kasus yang lama. Adapun teknik yang digunakan yaitu berdasarkan bobot dari sejumlah objek kasus yang ada.

$$Similarity(T,S) = \frac{\sum_{i=1}^{n} f(Ti,Si)*wi)}{wi} \quad (2.10)$$

Keterangan:

T : Kasus Baru

S : Kasus yang ada dalam penyimpanan

n : jumlah attribut dalam setiap kasus

i : attribut individu antara 1 sampai dengan n

f : fungsi similarity atribut i antara kasus T dan Kasus S

w : bobot yang diberikan pada atribut ke-i

Sebagai penjelasan tambahan bahwasanya untuk nilai *Similaritas* atau kesamaan berada di antara nilai 1 dan nilai 0. Yang mana untuk nilai 0 memiliki arti: Kasus Mutlak tidak mirip dan apabila nilai 1 memiliki arti : kasus mutlak memiliki kemiripan. Seperti diketahui bahwasanya dalam *Data Mining* kita bermain dengan istilah Himpunan Data. Dalam *K-Nearest Neighboor* kita juga menggunakan himpunan data.