BAB II

LANDASAN TEORI

2.1 Penelitian Terkait

Pada penelitian yang dilakukan dalam paper [4], penggunaan metode *Support Vector Machine (SVM)* menghasilkan tingkat akurasi yang relatif lebih tinggi dibandingkan dengan metode lain, namun sangat dipengaruhi oleh jumlah dataset, data training, data testing serta jumlah data positif dan negatifnya. Dalam penelitian tersebut dijelaskan bahwa penggunaan metode *SVM* berhasil mengklasifikasikan dokumen dengan baik, hal ini ditunjukan oleh tingginya tingkat akurasi metode yang digunakan. Selanjutya pada penelitian [3] Penggunaan metode *KNN* juga dapat mengklasifikasikan dokumen secara baik, meskipun pada penelitian ini tingkat akurasi yang dihasilkan dengan metode yang digunakan, belum bisa sebanding metode lain pada penelitian sebelumnya. Kemudian Penggunaan metode *KNN* dalam paper [7] menghasilkan tingkat akurasi yang jauh lebih baik dibandingkan metode *Naive Bayes*. Didalam penelitian tersebut diungkapkan bahwa penulis menggunakan data berupa emotikon yang seharusnya dihilangkan dalam proses filtering. Penambahan emotikon ini mengakibatkan meningkatnya tingkat *noisy* data, namun metode *KNN* terbukti lebih tangguh terhadap *noisy* data. Begitu pula pada paper [8] metode Support Vector Machine (SVM) juga memiliki tingkat ketelitian yang lebih baik dibandingkan dengan metode *Naïve Bayes*. Pada paper ini dijelaskan bahwa metode pembobotan yang digunakan tidak mempengaruhi urutan hasil klasifikasi dari dataset yang digunakan.

Berdasarkan dari beberapa penelitian yang sudah pernah dilakukan tersebut, maka dalam penelitian ini metode klasifikasi yang akan digunakan adalah *K-Nearest Neighbor (KNN)* yang mana metode ini terbukti lebih tangguh terhadap data yang memiliki tingkat *noisy* yang relatif tinggi dibandingkan metode lain. Kemudian dengan mengacu pada paper [7] untuk proses evaluasi/ *testing* untuk menguji tingkat akurasi metode tersebut akan digunakan metode *Recall*, *Precision* dan *F-Measure.*

Berikut ini merupakan tabel mengenai penelitian yang terkait yang digunakan sebagai acuan:

Tabel 2.1 Penelitian Terkait

No.	Penulis	Judul	Masalah	Metode	Hasil dan Kontribusi
1	N. Anita, M.K Sabariah, V. Effendy	Analisis Sentimen pada Twitter untuk Mengenai Penggunaan Transportasi Umum Darat Dalam Kota dengan Metode *Support Vector Machine*	Penggunaan metode *Support Vector Machine* sangat bergantung pada jumlah dataset, data training serta testing dan juga komposisi jumlah data positif dan negatif	Dalam paper ini menggunakan metode *Support Vector Machine*. Dimana hasil akurasinya sangat bergantung dengan jumlah dataset, data training serta data testing dan juga komposisi jumlah data postif dan negatif	Penggunaan metode *Support Vector Machine* dalam penelitian ini menghasilkan tingkat akurasi yang relatif tinggi.
2	Y.Y. Luhulima, Marji, L. Muflikhah	*SENTIMENT ANALYSIS* PADA REVIEW BARANG BERBAHASA INDONESIA DENGAN METODE K-NEAREST NEIGHBOR (K-NN)	Didalam paper ini digunakan metode *K-Nearest Neighbor* yang mana metode ini lebih tahan terhadap data *noisy* yang relatif besar namun metode ini menghasilkan tinggkat akurasi yang rendah.	Penggunaan metode *K-Nearest Neighbor* dalam paper ini kurang optimal dikarenakan kemungkinan sistem mengabaikan kata yang memiliki makna yang sama (sinonim)	Analisis yang dilakukan dengan metode *K-Nearest Neighbor* dalam paper ini dapat mengklasifikasikan sentiment dari review barang secara otomatis dengan nilai k yang tepat

3	M.Y. Nur, D.D. Santika	ANALISIS SENTIMEN PADA DOKUMEN BERBAHASA INDONESIA DENGAN PENDEKATAN *SUPPORT VECTOR MACHINE*	Pada paper ini masalah yang dibahas adalah terdapat ambigu dalam penggunaan kata, tidak adanya intonasi dalam sebuah teks, dan perkembangan bahasa itu sendiri	Penggunaan metode *Support Vector Machine (SVM)* dalam paper ini menunjukan tingginya tingkat ketelitian yang dimiliki metode ini dibandingkan dengan metode *Naïve Bayes*	Metode pembobotan yang digunakan dalam paper ini tidak mempengaruhi urutan hasil klasifikasi dataset yang digunakan
4	Arifin, K. Edy Purnama	CLASSIFICATION OF EMOTIONS IN INDONESIAN TEXTS USING K-NN METHOD	Penelitian ini menggunakan data yang juga mengikut sertakan emotikon didalam datasetnya, sehingga kemungkinanan noisy data lebih besar	Metode *K-Nearest Neighbor* yang digunakan dalam penelitian ini dikarenakan tingkat ketahanan metode ini terhadap data yang memiliki tingkat noisy yang relatif besar	Dalam paper ini disebutkan bahwa metode *KNN* memiliki tingkat akurasi yang lebih tinggi dibandingkan metode *Naive Bayes*, metode ini juga terbukti tangguh terhadap tingginya tingkat data *noisy*

2.2 Studi Pustaka

2.2.1 Twitter

Pada *Microblog data* seperti Twitter, dimana pengguna berinteraksi secara *realtime* serta memberikan opini tentang apa saja. Memberikan suatu kebaruan serta tantangan yang berbeda [1]. Disebut Microblog karena pada situs ini pengguna dapat mengirimkan serta membaca pesan layaknya blog pada umumnya namun hanya terbatas 140 karakter saja yang dapat tampil di halaman profil pengguna. Twitter memiliki format serta karakteristik cara penulisan yang unik menggunakan simbol maupun aturan khusus. Pesan yang ditulisakan pada twitter dikenal dengan sebutan *tweet* [4].

2.2.1.1 Twitter API

A. REST API

REST API menyediakan akses program untuk membaca dan menulis data Twitter . Penulis Tweet baru , membaca profil penulis dan data follower , dan banyak lagi. REST API mengidentifikasi aplikasi Twitter dan pengguna menggunakan OAuth ; tanggapan yang tersedia di JSON .

B. Stream API

Streaming API Streaming memberikan developer akses *latency* rendah ke *stream* global yang Twitter data Tweet . Sebuah implementasi yang tepat dari klien streaming akan mendorong pesan yang menunjukkan Tweet dan acara lainnya telah terjadi , tanpa ada *overhead* yang terkait dengan pengambilan *endpoint* REST.

C. Ads API

Ads Twitter API memungkinkan mitra untuk mengintegrasikan dengan platform iklan Twitter menggunakan iklan mereka sendiri. Mitra yang dipilih memiliki kemampuan untuk menciptakan alat kustom untuk mengelola dan melaksanakan kampanye iklan Twitter.

2.2.2 Sentiment Analysis

Sentiment Analysis merupakan salah satu bagian dari bidang ilmu *opinion mining,* yang memiliki pemahaman proses memahami, mengekstrak serta mengolah data dalam bentuk tekstual dengan tujuan memperoleh informasi [3]. Biasanya sentiment analysis dilakukan untuk mengetahui kecenderungan pasar [3]. Di Amerika Serikat kurang lebih 20-30% perusahaan memfokuskan kinerja mereka pada *sentiment analysis* [8]. Hal ini dikarenakan *sentiment analysis* itu sendiri bertujuan untuk menganalisis apa yang diungkapkan konsumen terhadap produk, jasa, atau organisasi yang mereka miliki, sehingga perusahaan dapat mengetahui bagaimana tanggapan konsumen terhadap apa yang telah mereka berikan.

Sentiment analysis dilakukan dengan berfokus pada pengolahan opini yang mengandung polaritas, yaitu yang membagi sentiment menjadi beberapa kelompok [4]. Sebagai contoh pembagian sentiment menjadi positif dan negatif.

Menurut B.Pang dan L.Lee [9] secara umum *sentiment analysis* terbagi menjadi 2 kategori yaitu:

1. *Coarse-grained* sentiment analysis
2. *Fined-grained* sentiment analysis

2.2.2.1 Coarse-grained sentiment analysis

Pada coarse-grained sentiment analysis, proses klasifikasi dilakukan berdasarkan orientasi sebuah dokumen secara keseluruhan. Orientasi ini dibagi menjadi 3 jenis: negatif, netral dan positif. Namun ada juga yang menjadikan orientasi ini menjadi kontinu atau dapat juga dikatakan bahwa orientasinya bersifat tidak diskrit.

2.2.2.2 Fined-grained sentiment analysis

Banyak dari kalangan peneliti sekarang ini berfokus pada kategori jenis ini. Berbeda dengan metode coarse-grained sentiment analysis, metode ini menggunakan objek yang berupa sebuah kalimat, bukan sebuah dokumen secara keseluruhan.

Contoh:

1. Saya bangga menjadi mahasiswa Udinus (positif)
2. Toilet mahasiswa sangat tidak terawatt (negatif)

Sentiment analysis itu sendiri juga dibagi kedalam 3 subproses besar [10]. Masing-masing dari subproses ini dapat dijadikan topik penelitian secara tersendiri, karena memang dari semua subproses tersebut membutuhkan teknik dan metode yang rumit. Subproses tersebut diantaranya:

1. Subjectivity Classification (menentukan kalimat yang merupakan opini)
   Contoh:
   Mobil itu memliliki 4 roda VS Itu mobil yang sangat keren!
2. Orientation Detection (setelah berhasil diklasifikasikan untuk kategori opini sekarang ditentukan apakah opini tersebut positif, netral ataupun negatif)
   Contoh:
   Mobil itu sangat keren! VS Mobil itu jelek sekali!
3. Opinion Holder and Target Detection (menentukan bagian yang merupakan Opinion Holder dan manakah yang merupakan Target)
   Contoh:
   Budi mengatakan bahwa mobil itu sangat keren.

Masalah yang sering timbul dalam sentiment analysis ini adalah banyaknya penggunaan kata yang tidak sesuai dengan kaidah, sehingga menyebabkan meningkatnya variasi bahasa. Sebagai contoh kata “tidak” memiliki 21 variasi seperti “tdk”, “gak”, “nggak”, “ga”, “kagak”, “nda” dan sebagainya. Pembatasan jumlah karakter sejumlah 140, menjadikan pengguna tidak leluasa mengekspresikan apa yang diinginkan. Jumlah *tweets* yang sangat besar juga membuat pemrosesan secara manual membutuhkan tenaga dan waktu ekstra.

2.2.3 *Preprocessing*

Prepocessing merupakan proses untuk mempersiapkan dan membersikan data dari dataset utuk selanjutnya diklasifikasikan [5]. Proses ini terdiri dari subproses *Cleansing*, *Case Folding* [8], *Tokenizing*, *Filtering*/ *Stopword Removal* serta *Stemming* [5].

2.2.3.1 Cleansing

Proses *cleansing* ini dilakukan untuk membersihkan dokumen dari kata yang tidak diperlukan untuk mengurangi noise. Kata yang dihilangkan adalah

karakter HTML, kata kunci, *emoticon*, *hashtag* (#), *username* (@username), url (http://situs.com), dan email (nama@situs.com) [8].

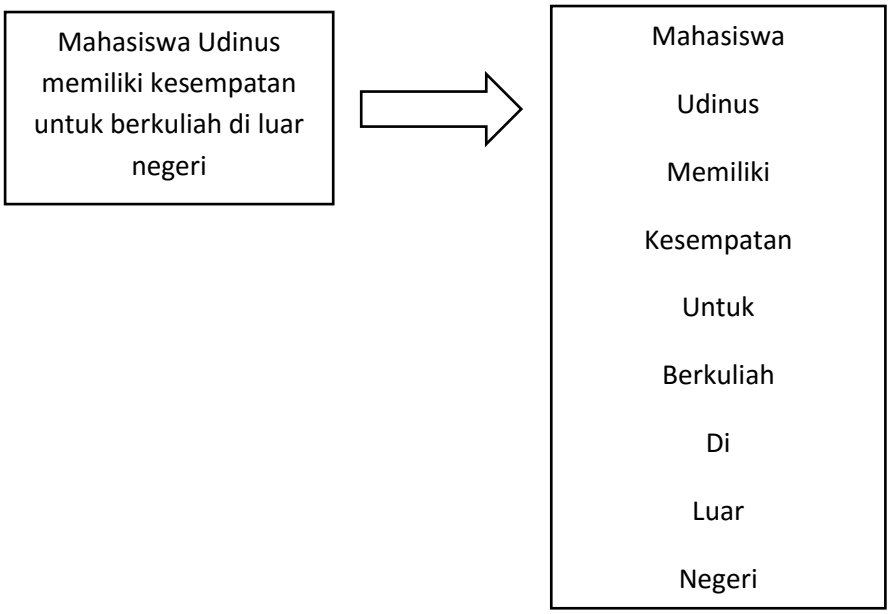
2.2.3.2 Case Folding

Case Folding merupakan proses penyeragaman bentuk huruf serta penghapusan angka dan tanda baca. Dalam kasus ini yang akan dipakai hanyalah huruf latin dari a hingga z [8].

2.2.3.3 Tokenizing

Pada proses *tokenizing* dokumen yang masih berupa kalimat dipecah per kata menjadi beberapa bagian dan secara bersamaan hilangkan semua karakter maupun tanda baca yang ada pada kalimat tersebut, hasil dari proses inilah yang disebut token [5].

Untuk lebih jelasnya lihat pada gambar 2.1 berikut [7]:



Gambar 2. 1 Proses Tokenisasi

2.2.3.4 Stopword Removal (Filtering)

Stopword Removal merupakan proses menghilangkan kata yang tidak mendeskripsikan sesuatu dalam bahasa Indonesia seperti “di”, ”ke”, “dari”, “yang”, “sedang”, “ini”, dan lain sebagainya. Di dalam *text classification*, kata seperti “tidak”, “bukan”, “tanpa” biasanya tidak termasuk kedalam kata yang

akan dihilangkan. Dalam penerapannya kalimat yang mengandung teks tersebut perlu diubah atau disesuaikan pada proses *preprocessing* [7].

2.2.3.5 Stemming

Proses *Stemming* merupakan proses penghilangan imbuhan yang masih melekat sehingga diperoleh sebuah kata dasar, contohnya: "membaca", "dibaca", "dibacakan" akan dikonversi menjadi kata dasar (*stem*) "baca". Pada penelitian ini proses *Stemming* memiliki 5 aturan [7]:

1. Menghilangkan partikel (*-lah, -kah, -tah,* dan *-pun*).
2. Menghilangkan kata ganti kepemilikan (*-ku, -mu,* dan *-nya*) .
3. Menghilangkan awalan tingkat pertama (*meng-, di-, ter-,* dan *ke-*).
4. Menghilangkan awalan tingkat kedua (*per-,* dan *ber-*).
5. Menghilangkan akhiran (*-i, -kan,* dan *-an*).

2.2.4 *Term Weighting* (Pembobotan)

Proses pembobotan ini dilakukan untuk mendapatkan nilai dari kata (*term*) yang sudah berhasil diekstrak. Umumnya metode yang digunakan untuk melakukan tahap pembobotan ini adalah pembobotan *TF-IDF*.

2.2.4.1 TF-IDF

Metode *TF-IDF* ini merupakan metode pembobotan dalam bentuk sebuah metode pembobotan yang menghbungkan antara *Term Frequency (TF)* dengan *Inverse Document Frequency (IDF)*, yang mana dapat dirumuskan sebagai berikut [3]:

$$w(t,d) = tf\,(t,d) * idf\,, \tag{1}$$

$$idf = log\,(\frac{N}{df}) \tag{2}$$

dimana *tf (t , d)* adalah kemunculan kata *t* pada dokumen *d, N* adalah jumlah dokumen yang ada pada kumpulan dokumen, dan *df* merupakan jumlah dokumen yang mengandung *term t.*

Metode ini berfungsi untuk mencari representasi nilai dari masing-masing dokumen dari sekumpulan data *training (training set)* dimana nantinya akan dibentuk suatu vektor antara dokumen dengan kata (*document with term*) yang kemudian untuk kesamaan antar dokumen dengan *cluster* akan ditentukan oleh sebuah *prototype* vector yang juga disebut juga dengan *cluster centroid* [3].

2.2.4.2 Contoh Perhitungan TF-IDF

A. Menghitung *Term Frequency (tf)*

Term frequency (tf) merupakan frekuensi kemunculan *term (t)* pada *dokumen (d)*.

Terdapat kalimat:

Saya sedang belajar menghitung tf.idf. Tf.idf merupakan frekuensi kemunculan term pada dokumen. Langkah awal perhitungan tersebut adalah menghitung tf, kemudian menghitung df dan idf. Langkah terakhir menghitung nilai tf.idf. Mari kita belajar!

Catatan: tiap kalimat dianggap sebagai dokumen.

Tentukan nilai *tf*!

Penyelesaian:

Tandai masing-masing dokumen:

Saya sedang belajar menghitung tf.idf. Tf.idf merupakan frekuensi kemunculan term pada dokumen. Langkah awal perhitungan tersebut adalah menghitung tf, kemudian menghitung df dan idf. Langkah terakhir menghitung nilai tf.idf. Mari kita belajar!

Tabel *tf*:

Tabel 2.2 Perhitungan *Terms Frequency (tf)*

Term (t)	D1 (dokumen 1)	D2	D3	D4	D5
Akhir	0	0	0	1	0
Awal	0	0	1	0	0
Belajar	1	0	0	0	1
Dokumen	0	1	0	0	0
Frekuensi	0	1	0	0	0
Hitung	1	0	3	1	0
Idf	1	1	1	1	0
Kita	0	0	0	0	1
Langkah	0	0	1	1	0
Muncul	0	1	0	0	0
Saya	1	0	0	0	0
Term	0	1	0	0	0
Tf	1	1	1	1	0

B. Menghitung *Document Frequency (df)*

Document frequency (df) adalah banyaknya dokumen dimana suatu *term (t)* muncul.

Dari soal yang sama pada menghitung *tf*, tentukan nilai *df*:

Tabel 2.3 Perhitungan *Document Frequency (df)*

Term (t)	df
Akhir	1
Awal	1
Belajar	2
Dokumen	1
Frekuensi	1
Hitung	5
Idf	4
Kita	1
Langkah	2
Muncul	1
Saya	1
Term	1
Tf	4

C. Menghitung *Invers Document Frequency (idf)*

Dari soal yang sama dengan perhitungan *df* dengan menggunakan rumus [2] kemudian hitung nilai *idf* (jumlah dokumen = N):

Tabel 2. 4 Perhitungan *Inverse Document Frequency (idf)*

Term (t)	df	idf (log N/*df*)
Akhir	1	0.698970004
Awal	1	0.698970004
Belajar	2	0.397940009
Dokumen	1	0.698970004
Frekuensi	1	0.698970004
Hitung	5	0
Idf	4	0.096910013
Kita	1	0.698970004
Langkah	2	0.397940009
Muncul	1	0.698970004
Saya	1	0.698970004
Term	1	0.698970004
Tf	4	0.096910013

D. Menghitung *tf.idf*

Hasil kali antara *tf* dengan *idf*

Dari soal yang sama dengan perhitungan *df*, hitunglah nilai *tf.idf* (dengan jumlah dokumen = N)

Tabel 2. 5 Perhitungan *tf.idf*

Term (t)	D1 (dokumen 1)	D2	D3	D4	D5	idf (log N/df)	tf.idf				
							D1	D2	D3	D4	D5
Akhir	0	0	0	1	0	0.698970004	0	0	0	0.69897	0
Awal	0	0	1	0	0	0.698970004	0	0	0.69897	0	0
Belajar	1	0	0	0	1	0.397940009	0.39794	0	0	0	0.39794
Dokumen	0	1	0	0	0	0.698970004	0	0.69897	0	0	0
Frekuensi	0	1	0	0	0	0.698970004	0	0.69897	0	0	0
Hitung	1	0	3	1	0	0	0	0	0	0	0
Idf	1	1	1	1	0	0.096910013	0.09691	0.09691	0.09691	0.09691	0
Kita	0	0	0	0	1	0.698970004	0	0	0	0	0.69897
Langkah	0	0	1	1	0	0.397940009	0	0	0.39794	0.39794	0
Muncul	0	1	0	0	0	0.698970004	0	0.69897	0	0	0
Saya	1	0	0	0	0	0.698970004	0.69897	0	0	0	0
Term	0	1	0	0	0	0.698970004	0	0.69897	0	0	0
Tf	1	1	1	1	0	0.096910013	0.09691	0.09691	0.09691	0.09691	0

2.2.5 *K-Nearest Neighbor (KNN)*

Algoritma *K-Nearest Neighbor (KNN)* merupakan salah satu metode yang digunakan dalam proses klasifikasi terhadap sebuah objek berdasarkan data pembelajaran yang memiliki jarak terdekat dengan objek tersebut [6].

Algoritma ini menerapkan konsep "*learning by analogy*", dimana data *learning* dideskripsikan dengan atribut numerik *n*-dimensi. Setiap data *learning* merepresentasikan sebuah titik, yang ditandai dengan *c*, dalam ruang *n*-dimensi [6]. Jika memasukan sebuah data *query* yang tidak diketahui labelnya, maka *K-Nearest Neighbor (KNN)* akan mencari *k* buah data *learning* yang memiliki jarak terdekat dengan *query* dalam ruang *n*-dimensi. Pengukuran jarak antara data *learning* dengan *query* dilakukan dengan mengukur jarak antar titik yang merepresentasikan data *query* dengan semua titik yang merepresentasikan data *learning* dengan rumus *Euclidean Distance* [6].

Pada fase *training*, algoritma ini hanya melakukan peyimpanan vektor-vektor fitur dan klasifikasi data *training sample*. Saat fase klasifikasi, semua fitur yang sama dihitung untuk kemudian dilakukan *testing data* (belum diketahui klasifikasinya) [6]. Jarak dari vektor baru yang terbentuk kemudian dihitung terhadap seluruh vektor *training sample*, dan sejumlah *k* buah yang terdekat diambil. Titik yang baru ini, klasifikasinya diprediksikan termasuk pada klasifikasi terbanyak dari titik-titik tersebut [6].

Nilai *k* yang terbaik untuk algoritma ini tergantung pada data secara umumnya, nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, namun membuat batasan antar setiap klasifikasi akan menjadi lebih kabur. Nilai *k* yang bagus dapat dipilih dengan optimasi parameter, misalnya dengan menggunakan *cross-validation*. Kasus khusus dimana klasifikasi diprediksikan berdasarkan data pembelajaran yang paling dekat (dengan kata lain, $k = 1$) disebut algoritma *nearest neighbor* [6].

Ketepatan algoritma *KNN* ini sangat dipengaruuhi oleh eksistensi fitur-fitur yang tidak relevan, atau jika bobot fitur tersebut tidak setara dengan relevansinya terhadap klasifikasi. Penelitian terhadap algoritma ini sebagian besar membahas bagaimana memilih dan memberi bobot terhadap fitur, agar performa klasifiaksi menjadi lebih baik [6].

K buah data *learning* akan melakukan *voting* untuk menentukan label mayoritas. Label data *query* akan ditentukan berdasarkan label mayoritas dan jika ada lebih dari satu label mayoritas makalabel data *query* dapat dipilih secara acak di antara label-label mayoritas yang ada [6].

Setelah melalui tahap *preprocessing*, setiap dokumen akan menjadi fitur vektor yang memiliki dimensi ke-*m* tahapan penerapan Algoritma *KNN* dapat dituliskan sebagai berikut [7]:

1. Ubah dokumen X dari semua *sample training* kedalam sebuah vektor yang sama (X_1, X_2, X_3, ..., X_m)
2. Hitung *similarity* dari semua *training samples* dan dokumen X. Ambil dokumen ke-*i* dari d_i (d_{i1}, d_{i2}, ..., d_{im}). Sebagai contoh, *similarity* dari SIM (X, d_i) adalah sebagai berikut:

$$cosSim(\,X, di\,) = \frac{\sum_{j=1}^{m} x_j \,.\, d_{ij}}{\sqrt{\left(\sum_{j=1}^{m} x_j\right)^2 .\, \left(\sum_{j=1}^{m} d_{ij}\right)^2}} \qquad (3)$$

3. Pilihlah satu *sample* k yang lebih besar daripada *similarity* N yang diperoleh dari SIM (X, d_i), (i = 1, 2, ..., N). Anggaplah *sample* tersebut sebagai *KNN* dari X. Kemudian hitung probabilitas dari X terhadap rumus berikut:

$$P\left(X, C_j\right) = \sum_{d_{i \in KNN}} SIM(X, d_i).\, y(d_i, C_j) \qquad (4)$$

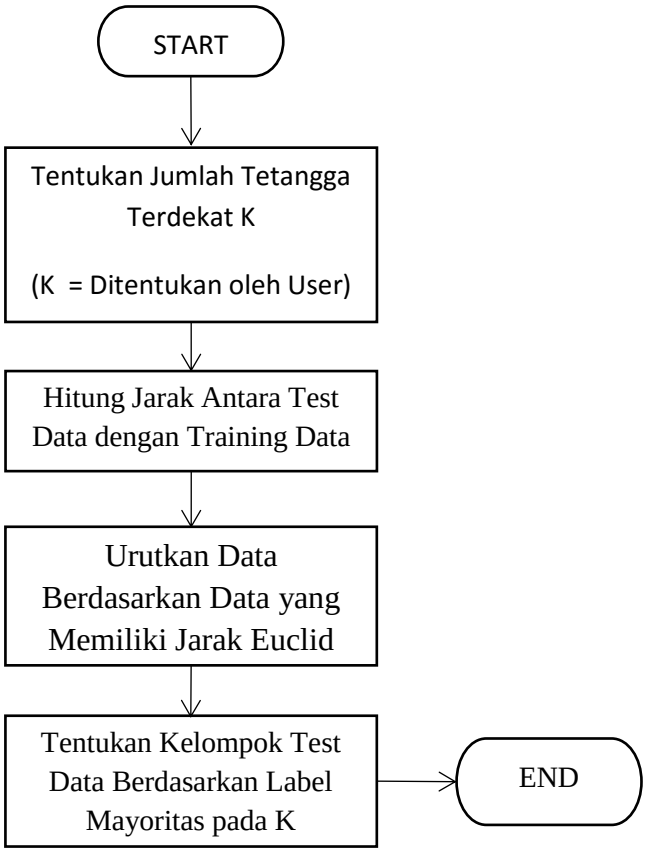
Dimana $y(d_i, C_j)$ merupakan fungsi dari atribut kategori yang mengisi persamaan berikut:

$$y\left(d_i, C_j\right) = \begin{cases} 1, d_{i \in Cj} \\ 0, d_{i \in Cj} \end{cases} \qquad (5)$$

4. Uji dokumen X untuk mencari kategori dengan melihat $P\left(X, C_j\right)$ terbesar.

2.2.5.1 Gambaran Umum Algoritma KNN

Gambaran umum algoritma *KNN* ini dapat dilihat dalam *flowchart* berikut[6]:



Gambar 2.2 Flowchart KNN

2.2.5.2 Konsep Perhitungan Jarak (Euclidean Distance)

Diberikan 2 buah titik *X* dan *Y* dalam sebuah ruang vektor *n*-dimensi dengan $X(x_1, x_2, \ldots, x_n)$ dan $Y(y_1, y_2, \ldots, y_n)$, maka jarak antara *X* dan *Y* dapat dihitung menggunakan persamaan *Euclidean Distance* berikut [6]:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)} \qquad (6)$$

Dimana x dan y merupakan titik pada ruang vektor n dimensi sedangkan x_i dan y_i merupakan besaran skalar untuk dimensi ke i dalam ruang vektor n dimensi.

2.2.5.3 Contoh Perhitungan KNN

Terdiri dari 2 atribut dengan skala kuantitatif yaitu X1 dan X2 serta 2 kelas yaitu baik dan buruk. Jika terdapat data baru dengan nilai X1=3 dan X2=7 [6]

Tabel 2. 6 Contoh Soal

X1	X2	Y
7	7	Buruk
7	4	Buruk
3	4	Baik
1	4	Baik

Langkah-langkah:

1. Tentukan parameter K = jumlah tetangga terdekat. Misalkan ditetapkan K = 3
2. Hitung jarak antara data baru dengan semua data training

Tabel 2.7 Perhitungan Jarak Baru

X1	X2	Kuadrat jarak dengan data baru (3,7)
7	7	$(7-3)^2 + (7-7)^2 = 16$
7	4	$(7-3)^2 + (4-7)^2 = 25$
3	4	$(3-3)^2 + (4-7)^2 = 9$
1	4	$(7-3)^2 + (7-7)^2 = 13$

3. Urutkan jarak tersebut dan tetapkan tetangga terdekat berdasarkan jarak minimum ke-K

Tabel 2.8 Penentuan Jarak Terdekat

X1	X2	Kuadrat jarak dengan data baru (3,7)	Peringkat jarak minimum	Termasuk 3 tetangga terdekat
7	7	$(7-3)^2 + (7-7)^2 = 16$	3	Ya
7	4	$(7-3)^2 + (4-7)^2 = 25$	4	Tidak
3	4	$(3-3)^2 + (4-7)^2 = 9$	1	Ya
1	4	$(7-3)^2 + (7-7)^2 = 13$	2	Ya

4. Periksa kelas dari tetangga terdekat

Tabel 2.9 Penentuan Kelas

X1	X2	Kuadrat jarak dengan data baru (3,7)	Peringkat jarak minimum	Termasuk 3 tetangga terdekat	Y = kelas tetangga terdekat
7	7	$(7-3)^2 + (7-7)^2 = 16$	3	Ya	Buruk
7	4	$(7-3)^2 + (4-7)^2 = 25$	4	Tidak	-
3	4	$(3-3)^2 + (4-7)^2 = 9$	1	Ya	Baik
1	4	$(7-3)^2 + (7-7)^2 = 13$	2	Ya	Baik

2.2.6 Recall and Precision

Evaluasi dari kemiripan sebuah dokumen dapat dilakukan berdasarkan *recall*, *precision*, dan *F-measure*. Didalam hasil klasifikasi (kelas prediksi I), ada kemungkian dimana hasil klasifikasi tersebut memang benar seharusnya diklasifikasikan sebagai (kelas I) atau bukan, atau bisa jadi malah hasil klasifikasi tersebut seharusnya berada pada (kelas II) [6]. Oleh karena itu parameter tersebut akan digunakan sebagai penghitungan parameter evaluasi, yang mana [13]:

1. *Precision*, adalah kemampuan untuk mengambil *topranked* dokumen terambil yang relevan,
2. *Recall*, adalah sebagian dari dokumen relevan yang terambil.

Sedangkan *F-measure* merupakan hasil representasi keseluruhan sistem dan dimatematikakan dengan menggabungkan hasil dari *recall* dengan *precision.*

2.2.7 *R* Programming

R merupkan bahasa pemrograman komputasi statisik dan grafik. Bahasa pemrograman ini merupakan implementasi dari bahasa pemrograman *S* yang dikembangkan oleh John Chambers di *Bell Laboratories* (AT&T) yang sekarang lebih dikenal sebagai *Lucent Technologies*.

R itu sendiri memliliki beragam jenis model statistik, yaitu:

1. *Linear* dan *nonlinear modeling*,
2. *Classical statistical tests*,
3. *Time series analysis*,
4. *Classification*,
5. *Clustering*,
6. Dan lain sebagainya.

Selain itu *R* juga memlikiki teknik komputasi grafik yang mana *Source Code* tersedia secara bebas dibawah lisensi *GNU General Public License*, dan versi *pre-compiled binary* tersedia untuk berbagai sistem operasi. *R* biasanya menggunakan *command line interface* untuk mengeksekusi perintah, namun beberapa *graphical user interface* juga tersedia untuk digunakan bersama *R*.

2.2.8 PHP

PHP: Hypertext Preprocessor adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memrogram situs web dinamis. PHP dapat digunakan untuk membangun sebuah CMS.

Pada awalnya PHP merupakan kependekan dari Personal Home Page (Situs personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu

itu PHP masih bernama Form Interpreted (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari web.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini, interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang PHP: Hypertext Preprocessing.

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

Versi terbaru dari bahasa pemograman PHP adalah versi 5.6.4 yang resmi dirilis pada tanggal 18 Desember 2014.

2.2.9 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: database management system) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL.

Tidak sama dengan proyek-proyek seperti Apache, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh

penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

2.2.10 XAMPP

XAMPP merupakan sebuah perangkat lunak (*free software*) bebas, yang mendukung untuk banyak sistem operasi, yang merupakan kompilasi dari beberapa program. Fungsi XAMPP sendiri adalah sebagai server yang berdiri sendiri (localhost), yang terdiri beberapa program antara lain : Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP sendiri merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam GNU General Public License dan bebas, merupakan web server yang mudah untuk digunakan yang dapat menampilkan halaman web yang dinamis.

2.2.11 Sastrawi Library

Sastrawi merupakan library PHP yang dikembangkan khusus untuk proses *stemming* teks berbahasa Indonesia, dalam penggunaannya library ini harus diinstal menggunakan *composer*.

Contoh:

“Rakyat memenuhi halaman gedung untuk menyuarakan isi hatinya”.

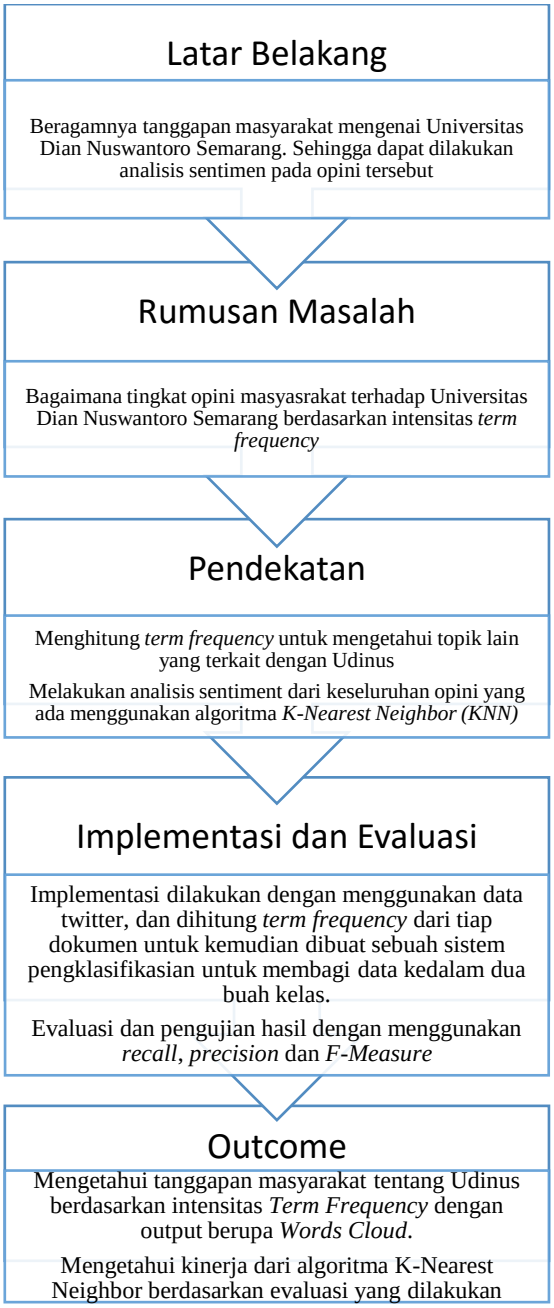
Jika menggunakan query seperti ini:

```
SELECT * FROM posts WHERE content LIKE ‘%suara%’
```

Maka tidak akan didapatkan teks yang diinginkan, bahkan penggunaan Algoritma Fuzzy *full-text-search* juga membutuhkan sebuah *stemmer* untuk meningkatkan hasil pencariannya. Kemampuan library ini akan merubah dokumen utuh tersebut kedalam sebuah kalimat dengan menghilangkan imbuhan yang melekat menjadi seperti berikut:

“rakyat penuh halaman gedung suara isi hati”.

2.3 Kerangka Pemikiran



Gambar 2.3 Skema Kerangka Pemikiran