# KLASIFIKASI TEKS DENGAN MENGGUNAKAN ALGORITMA *K-NEAREST NEIGHBOR* PADA KASUS KINERJA PEMERINTAH DI *TWITTER*

**Octaryo Sakti Yudha Prakasa[1],Kemas Muslim Lhaksamana[2]**

[12] Prodi S1 Ilmu Komputasi, Fakultas Informatika, Universitas Telkom
[1]octaryosakti@gmail.com, [2]kemasmuslim@telkomuniversity.acid

Abstrak

Seiring dengan perkembangan teknologi yang begitu cepat dalam hal pengumpulan dan penyimpanan data menyebabkan tumpukan data yang sangat banyak. Dengan adanya kumpulan data yang banyak, maka muncullah suatu kebutuhan untuk bisa memanfaatkan data tersebut. Pemanfaatan data tersebut tentunya bertujuan untuk mendapatkan informasi yang penting dari pola-pola data yang terbentuk.

Data yang dapat digunakan bisa diambil di sosial media salah satunya *twitter. Twitter* adalah salah satu media sosial yang cukup digemari oleh seluruh kalangan, tercatat sekitar 50 juta orang di Indonesia menggunakan *twitter*. Dengan banyaknya pengguna twitter maka data yang dapat dimanfaatkan juga banyak.

Cara untuk mendapatkan informasi dari sebuah data bisa menggunakan metode klasifikasi. Salah satu algoritma dalam klasifikasi adalah algortima *K-Nearest Neighbor Classifier*. Algoritma KNN mempunyai sistem kerja dengan menghitung jarak terdekat dari *record* uji ke *record testing* dengan menggunakan metode *Eucliden Distance*. Hasil dari proses KNN berupa jarak terdekat dari *record* uji ke *record testing* sebanyak K yang diperlukan.

**Kata Kunci : Twitter, *Text Mining*, Klasifikasi, *K-Nearest Neighbor Classifier, Euclidean Distance***

***Abstract***

*With the development of very fast technology in terms of storage and data storage produces very much data. With the existence of large data sets, it appears the need to be able to utilize the data. Utilization of the data is intended to obtain important information from the data patterns formed.*

*Data that can be used in social media one of them twitter. Twitter is one of the social media that is favored by all circles, diesel around 50 million people in Indonesia using twitter. With the twitter then the data can be used also a lot.*

*The way to obtain information from a data can use the classification method. One of the algorithms in the classification is the K-Nearest Neighbors Classifier algorithm. KNN algorithm has a working system by calculating the distance from the test record to record testing using the Eucliden Distance method. The result of the KNN process is the closest distance from the test record to the required K test record.*

***Keywords : Twitter, Text Mining, Klasifikasi, K-Nearest Neighbor Classifier, Euclidean Distance***

## 1. Pendahuluan

### 1.1 Latar Belakang

Perkembangan media sosial di Indonesia cukup sangan pesat pada abad ini. Banyak sekali aplikasi media sosial yang bekembang saat ini salah satunya *twitter. Twitter* adalah salah satu media sosial yang cukup digemari oleh seluruh kalangan, tercatat sekitar 50 juta orang di Indonesia menggunakan *Twitter.* Di *twitter* sendiri selama tahun 2016 telah tercatat 4,1 milyar *tweet* dari pengguna di Indonesia.

Oleh karena itu, dengan banyaknya *tweet* maka informasi yang bisa didapat cukup banyak. Salah satunya adalah informasi pendapat atau opini dari pengguna *tweet*. Pendapat adalah suatu ungkapan perasaan suka atau tidak suka seseorang terhadap suatu objek. Untuk tugas akhir ini saya sebagai penulis memfokuskan hanya kepada pendapat tentang kinerja pemerintah.

Kinerja pemerintah banyak memuai pujian dan kritikan dari masyarakat indonesia pengguna *twitter*, baik terhadap presiden maupun kinerja permerintah secara keseluruhan. Tercatat 1000 ribu *tweet* yang masuk tentang kinerja pemerintah selama 1 bulan. Oleh karena itu penulis melakukan pengklasifikasian terhadap *tweet* tersebut apakah *tweet* tersebut merupakan pujian atau kritikan.

Algoritma klasifikasi yang digunakan adalah algoritma *K-Nearst Neigbhor*. Algoritma *K-nearest neighbor* (k-NN atau KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data

pembelajaran yang jaraknya paling dekat dengan objek tersebut. Pada fase pembelajaran, algoritma ini hanya melakukan penyimpanan vektor-vektor fitur dan klasifikasi dari data pembelajaran. Pada fase klasifikasi, fitur-fitur yang sama dihitung untuk data *testing* (yang kelas tidak diketahui). Jarak dari vektor yang baru ini terhadap seluruh vektor data pembelajaran dihitung, dan sejumlah k buah yang paling dekat diambil. Titik yang baru klasifikasinya diprediksikan termasuk pada klasifikasi terbanyak dari titik-titik tersebut.

Berdasarkan penelitian sebelumya dengan judul jurnal klasifikasi topik berita berbahasa Indonesia menggunakan *Weighted K-Nearest Neighbor* (Setiawan,2018) KNN adalah metode yang cukup baik untuk kalsifikasi teks.

Untuk itu, dalam tugas akhir ini digunakan metode KNN yang ditujukan untuk mengklasifikasikan atau mengolah *tweet* sehingga akan diketahui klasifikasi dari *tweet* tersebut (positif atau negatif) kinerja pemerintah dengan menggunakan berbagai skenario. Dengan menggunakan metode ini, diharapkan mendapat tingkat akurasi yang cukup tinggi dalam mengklasifikasikan *tweet* tersebut.

## 2. Kajian Pustaka

### 2.1 Klasifikasi Teks

Klasifikasi teks adalah proses pengelompokkan data kedalam kelas yang telah ditetentukan sebelumnya, untuk bisa digunakan memprediksi kelas dari data-data yang kelas belum diketahui. Klasifikasi teks merupakan proses klasifikasi data teks kedalam suatu kelas ataupun kategori yang sebelumnya sudah ditentukan terlebih dahulu.

Terdapat dua jenis klasifikasi teks yaitu *supervised* dan *unsupervised*. Klasifikasi *supervised* adalah proses klasifikasi teks dengan menggunakan metode *learning* pada data teks yang sudah memiliki kelas pada data latih sebagai data untuk pembelajaran. Sementara klasifikasi *unsupervised* adalah metode klasifikasi teks yang tidak memakai label kelas pada data latih untuk menganalisa hubungan antar kedua kata. Dua hal yang penting dalam klasifikasi teks dengan metode *supervised*, yaitu adalah *learning* dan klasifikasi. Tahap *learning* adalah tahap pertama yang perlu dilakukan untuk mengembangkan data latih. Di tahap ini akan didapatkan model *classfier* yang selanjutnya akan digunakan pada tahap klasifikasi. Di tahap kedua akan dilakukan pengklasifikasian terhadap data uji berdasarkan model *classifier* yang sebelumnya sudah diperoleh. Untuk tugas akhir penulis menggunakan metode klasifikasi teks dengan metode *supervised* karena data latih yang digunakan sudah memiliki kelas yang nantinya digunakan untuk pembelajaran.

Beberapa metode klasifikasi adalah sebaagi berikut berikut:

1. *Naïve Bayes*

*Naive bayesian* klasifikasi adalah suatu klasifikasi berpeluang sederhana berdasarkan aplikasi teorema *Bayes* dengan asumsi antar variabel penjelas saling bebas (independen). Dalam hal ini, diasumsikan bahwa kehadiran atau ketiadaan dari suatu kejadian tertentu dari suatu kelompok tidak berhubungan dengan kehadiran atau ketiadaan dari kejadian lainnya (Zhang and Li, 2007).

2. *Support Vector Machine*

*Support vector machine* (SVM) adalah suatu teknik untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi. SVM berada dalam satu kelas dengan *Artificial Neural Network* (ANN) dalam hal fungsi dan kondisi permasalahan yang bisa diselesaikan (Basu,etc.,2003).

3. *K-Nearest Neighbor*

*Algoritma k-nearest neighbor* (k-NN atau KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut (PengYu and GaoYu, 2007).

### 2.2 *Text Mining*

Penambangan teks adalah proses ekstraksi pola berupa informasi dan pengetahuan yang berguna dari sejumlah besar sumber data teks, seperti dokumen Word, PDF, kutipan teks, dll. Jenis masukan untuk penambangan teks ini disebut data tak terstruktur dan merupakan pembeda utama dengan penambangan data yang menggunakan data terstruktur atau basis data sebagai masukan. Penambangan teks dapat dianggap sebagai proses dua tahap yang diawali dengan penerapan struktur terhadap sumber data teks dan dilanjutkan dengan ekstraksi informasi dan pengetahuan yang relevan dari data teks terstruktur ini dengan menggunakan teknik dan alat yang sama dengan penambangan data. Proses yang umum dilakukan oleh penambangan teks di antaranya adalah perangkuman otomatis, kategorisasi dokumen, penggugusan teks, deteksi plagiarisme, dll. Tujuan dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Jadi, sumber data yang digunakan pada *text mining* adalah kumpulan teks yang memiliki format yang tidak terstruktur atau

minimal semi terstruktur. Adapun tugas khusus dari *text mining* antara lain yaitu pengkategorisasian teks (*text categorization*) dan pengelompokan (*text clustering*).

### 2.4 *Text Preprocesing*

*Text Preprocessing* merupakan tahapan awal dalam mengolah data input sebelum memasuki proses tahapan utama dari *metode Latent Semantic Analysis* (LSA). *Preprocessing text* dilakukan untuk tujuan penyeragaman dan kemudahan pembacaan serta proses LSA selanjutnya. *Preprocessin*g terdiri dari beberapa tahapan. Adapun tahapan preprocessing berdasarkan , yaitu: *case folding, tokenizing* / *parsing, filtering, stemming*.

Berikut penjelasan empat tahapan dalam proses preprocessing adalah sebagai berikut:

1. *Case folding*

Merupakan tahapan yang mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf 'a' sampai dengan 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap delimiter (pembatas).

2. *Tokenizing*

Tahap *tokenizing/parsing* adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya. Selain itu, spasi digunakan untuk memisahkan antar kata tersebut.

3. *Filtering*

Tahap *filtering* adalah tahap mengambil kata-kata penting dari hasil *tokenizing*. Proses *filtering* dapat menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist / stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag-of-words*. Contoh *stopword* adalah "yang", "dan", "di", "dari" dan lain – lain.(Triawati, 2009).

4. *Stemming*

Teknik *Stemming* diperlukan selain untuk memperkecil jumlah indeks yang berbeda dari suatu dokumen, juga untuk melakukan pengelompokan kata-kata lain yang memiliki kata dasar dan arti yang serupa namun memiliki bentuk atau form yang berbeda karena mendapatkan imbuhan yang berbeda. Sebagai contoh kata bersama, kebersamaan, menyamai, akan distem ke *root word*-nya yaitu "sama".

### 2.5 *Feature Extraction*

*Feature Extraction* adalah cara untuk mengubah *tweet* menjadi nilai yang berupa *vector*, yang bertujuan agar *tweet* dapat diklasifikasikan ke dalam kelas-kelas yang telah dibuat. *Feature ekstraksi* yang penulis gunakan adalah *Binary TF*. *Binary TF* adalah *feature ekstraction* yang hanya memperhatikan apakah suatu kata atau *term* ada/atau tidak dalam suatu *tweet*, jika ada diberi nilai 1 kalau tidak diberi nilai 0 (Toker and Kırmemis, 2007).

### 2.6 *K-Nearest Neighbor*

Algoritma *k-nearest neighbor* (k-NN atau KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut. *K-Nearest Neighbor* berdasarkan konsep '*learning by analogy*'. Data *learning* dideskripsikan dengan atribut numerik n-dimensi. Tiap data *learning* merepresentasikan sebuah titik, yang ditandai dengan c, dalam ruang n-dimensi. Jika sebuah data *query* yang labelnya tidak diketahui diinputkan, maka *K-Nearest Neighbor* akan mencari k buah data *learning* yang jaraknya paling dekat dengan data query dalam ruang n-dimensi.

Jarak antara data *query* dengan data *learning* dihitung dengan cara mengukur jarak antara titik yang merepresentasikan data *query* dengan semua titik yang merepresentasikan data *learning* dengan rumus *Euclidean Distance*. Pada fase klasifikasi, *tweet* sama dihitung untuk *testing* data (klasifikasinya belum diketahui). Jarak dari vektor yang baru ini terhadap seluruh vektor *training sample* dihitung, dan sejumlah k buah yang paling dekat diambil. Titik yang baru klasifikasinya diprediksikan termasuk pada klasifikasi terbanyak dari titik – titik tersebut. Nilai k yang terbaik untuk algoritma ini tergantung pada data; secara umumnya, nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, tetapi membuat batasan antara setiap klasifikasi menjadi lebih kabur. Nilai k yang bagus dapat dipilih dengan optimasi parameter, misalnya dengan menggunakan *cross-validation*.

#### 2.6.1 *Euclidean Distance*

*Euclidean Distance* paling sering digunakan menghitung jarak. Jarak *Euclidean* berfungsi untuk menguji ukuran yang bisa digunakan sebagai interpretasi kedekatan jarak antar dua obyek yang direpresentasikan sebagai berikut:

$$dist = \sum_{i=1}^{p} \sqrt{(x2 - x1)\text{^}2}$$

Keterangan :
*dist* = Jarak
x1 = Data *Training*
x2 = Data *testing*
i = Variable Data
p = Jumlah Atribut

Semakin besar nilai D akan semakin jauh tingkat keserupaan antara kedua individu dan sebaliknya jika nilai D semakin kecil maka akan semakin dekat tingkat keserupaan antar individu tersebut.
Nilai k yang terbaik untuk algoritma ini tergantung pada data. Secara umum, nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, tetapi membuat batasan antara setiap klasifikasi menjadi semakin kabur. Nilai k yang bagus dapat dipilih dengan optimasi parameter, misalnya dengan menggunakan *cross-validation*. Kasus khusus dimana klasifikasi diprediksikan bedasarkan *traning* data yang paling dekat (dengan kata lain, k=1) disebut algoritma *nearest neighbor* (Nikhath and Subrahmanyam, 2016).

**2.7 Evaluasi**

Evaluasi algoritma dilakukan dengan menggunakan *confusion matrix*. *Confusion Matrix* dibuat untuk memetakan kinerja algoritma dalam bentuk tabulasi (Tripathy,etc., 2015). Matriks ini menunjukkan hubungan antara benar tidaknya sebuah data dikategorikan.

*Confusion Matrix* terdiri dari *True positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN). *True positive* merepresentasikan data yang berada pada kelas positif yang diprediksi secara benar oleh algoritma. *False Positive* merepresentasikan data yang seharusnya berada pada kelas positif diprediksi menjadi kelas negatif oleh algoritma. *False Negative* merupakan data yang seharusnya berada di kelas negatif diprediksi menjadi kelas positif oleh algoritma. *True Negative* merupakan data yang berada pada kelas negatif dan diprediksi secara benar oleh algoritma. *Confusion Matrix* dapat dilihat pada tabel berikut:

**Tabel 2.1** *Confusion Matrix*

| | Label atau kelas | |
|---|---|---|
| | Positif | Negatif |
| Positif | *True Positve* | *False Positive* |
| Negatif | *False Negative* | *True Negative* |

Berdasarkan *confusion matrix*, dapat diketahui berbagai parameter pengukuran kinerja algoritma, yaitu *presisi, recall, fl-measure*, dan akurasi.

Presisi merupakan parameter untuk mengukur ketepatan dari suatu algoritma. Misalkan untuk menghitung *precision* algoritma dalam memprediksi data berlabel positif, *precision* dihitung berdasarkan rasio jumlah data berlabel positif yang diprediksi secara benar oleh algoritma dengan jumlah data yang diprediksi memiliki label positif oleh algoritma.

$$precision = \frac{TP}{TP + FP}$$

*Recall* merupakan parameter untuk mengukur kelengkapan sebuah algoritma. Misalkan untuk menghitung *recall* algoritma dalam memprediksi data berlabel positif, *recall* dihitung berdasarkan rasio jumlah data berlabel positif yang diprediksi secara benar oleh algoritma dengan jumlah semua data yang berlabel positif pada *dataset*.

$$recall = \frac{TP}{TP + FN}$$

*F-measure* merupakan rata-rata harmonik dari presisi dan *recall*. Nilai tertinggi adalah 1 dan nilai terendah adalah 0.

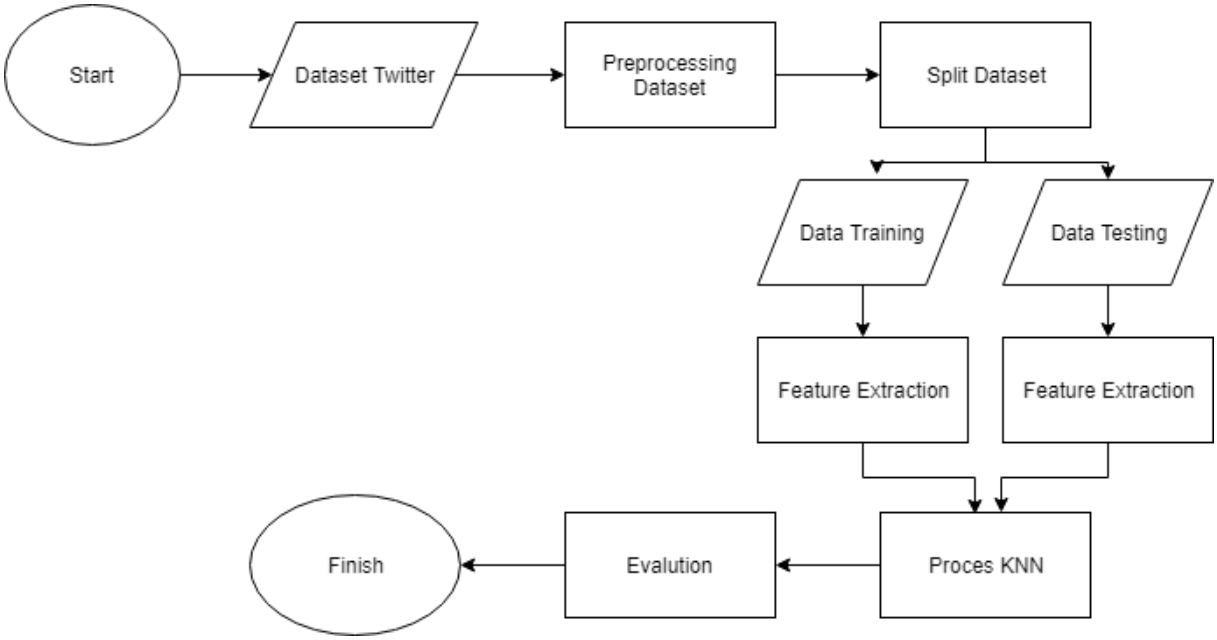
$$F - measure = \frac{2 \times precision \times recall}{precision + recall}$$

Akurasi merupakan perhitungan yang umum digunakan untuk mengevaluasi kinerja dari sebuah algoritma. Akurasi dihitung berdasarkan rasio jumlah data yang diprediksi secara benar oleh algoritma dengan jumlah semua data yang ada pada *dataset.*

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

## 3. Deskripsi Sistem

Rancangan sistem yang dibangun berupa tahapan penelitian yang terstruktur yang bertujuan untuk melakukan klasifikasi *tweet* menggunakan algoritma *K-Nearst Neighbor*. Tahapan dimulai dengan pengambilan *dataset* dan berakhir dengan mendapat akurasi dari berbagai skenario.




**Gambar 3.1** *Flowchart* Perancangan Sistem

### 3.1 Dataset *Twiiter*

Dataset yang digunakan pada penelitian ini adalah *tweet* sebanyak 1000 *tweet*. Dataset didapatkan melalui proses *crawling* dengan menggunakan php. *Dataset* diperlukan untuk membangun sistem klasfikasi menggunakan algoritma *KNN*. Berikut ini adalah sample dari *dataset* yang digunakan.

**Tabel 3.1** Sampel *Dataset Twitter*

| |
|---|
| @jokowi DPR kerjanya apa sih selain nyinyir kinerja pemerintah tiap hari! #bubarkandpr |
| RT @ebithEBITH: Semoga Bapak Presiden bisa evaluasi kinerja aparat keamanan negri ini... biar rakyat aman. |
| RT @DiaZKillaZ: Disaat situasi kaya gni... Hari ini masi aja ada yg nyinyir2in pemerintahan, kaya yang udah plg tau aja dalem2an kinerja Ã¢â‚¬Â¦ |
| RT @kaka_Diwan: Yth : Bpk Presiden Jokowi, terkait Kasus Mako Brimob dan Bom Surabaya serta bom sidoarjo yang baru terjadi |

| |
|---|
| malam ini. IniÃ¢â‚¬Â¦ |
| @SBYudhoyono Hanya orang paham dan cerdas bisa menilai kinerja mantan atau masih menjabat presiden saÃ¢â‚¬Â¦ https://t.co/mLc9mZifO2 |
| @jokowi Saya dukung 100% dgn dikeluarkannya Perpu ini jika memang dr DPR menghambat kinerja POLRI dlm memberantas TÃ¢â‚¬Â¦ https://t.co/KRed4VFG3V |

### 3.2 *Preprocessing Dataset*

*Preprocessing* adalah proses untuk mengolah data yang belum sesuai dengan bentuk data yang di perlukan untuk proses klasifikasi ini. Untuk tugas akhir ini *preprocessing* yang dipakai adalah *case folding* dan *stopwords removal*. *Case folding* untuk mengubah semua *upper case* menjadi *lower case* dan *stopwords removal* untuk menghapus kata, space dan karakter yang tidak dipakai. Berikut ini adalah data yang telah di *preprocessing*.

**Tabel 3.2** *Preprocessing* Sampel Dataset *Twittter*

| |
|---|
| jokowi dpr kerjanya sih nyinyir kinerja pemerintah bubarkandpr |
| ebithebith semoga presiden evaluasi kinerja aparat keamanan negri biar rakyat aman |
| segenap rakyat bersatu untuk Indonesia yang lebih baik |
| kaka_diwan yth bpk presiden jokowi terkait mako brimob bom surabaya bom sidoarjo malam ini… |
| sbyudhoyono orang paham cerdas menilai kinerja mantan menjabat presiden |
| jokowi dukung dgn dikeluarkannya perpu dr dpr menghambat kinerja polri dlm memberantas t… kredvfgv |

### 3.3 *Split Dataset*

Tujuan dari proses data *split* adalah untuk mendapatkan data *training* dan data *testing*. Untuk pembagian data *training* dan *testing* penulis menggunakan 3 skenario yaitu 80%:20% , 70%:30% dan 60%:40%.

### 3.4 *Feature Extraction*

Pada tahapan *feature extraction tweet* yang telah di *preprocessing* diubah menjadi nilai. Untuk menjadikan *tweet* menjadi nilai maka *feature extraction* yang penulis gunakan adalah *TF binary*. Di dalam proses ini juga *tweet* yang telah *preprocessing* akan dipisah menjadi perkata untuk menjadi atribut di data *training* maupun data *testing*. Yang harus diperhatikan adalah atribut data *training* dan *testing* harus sama agar dapat dilakukan perhitungan jarak pada proses *Distance* KNN. Berikut adalah langkah dalam proses *Feature Extraction.*

1. Langkah pertama yang harus dilakukan adalah membuat atribut pada data *training* yang akan digunakan untuk proses *TF binary*. Atribut didapatkan dari kata-kata yang terdapat pada semua *tweet* data *training*. Tetapi tidak semua kata yang ada pada *tweet* dijadikan atribut, kata yang diambil adalah kata yang berhubungan dengan kinerja pemerintah dan juga kata yang memilki makna diambil salah satu. Dibawah ini adalah contoh dari kata yang terdapat pada *tweet* yang dijadikan atribut.

**Tabel 3.3** Sampel atribut

| Nomor | Kata Unigram |
|---|---|
| 1 | agama |
| 2 | maaf |
| 3 | bahwa |
| 4 | belum |
| 5 | aman |
| 6 | awal |
| 7 | segenap |
| 8 | rakyat |
| 9 | indonesia |
| 10 | gagal |

2. Langkah kedua adalah menyamakan atribut antara data *training* dan data *testing* sehingga tidak terjadi *error* pada saat perhitungan jarak.

3. Langkah ketiga atau langkah terakhir adalah proses *TF Binary*. Di tahap ini *record* yang terdapat pada data *training* dan *testing* di cek satu persatu apakah sama dengan atribut yang telah dibuat. Dari Tabel 3.4 dilihat apakah kata yang ada pada *tweet* tersebut ada di dalam atribut yang talah dibuat pada Tabel 3.3.

**Tabel 3.4** Data *Tweet*

| Nomor | *Tweet* |
|---|---|
| 1 | mbahuyok maaf bahwa belum aman kepada segenap rakyat indonesia apapun suku agamanya |
| 2 | segenap rakyat bersatu untuk Indonesia yang lebih baik |
| 3 | sbyudhoyono orang paham cerdas menilai kinerja mantan menjabat presiden |

Maka hasil yang didapatkan setelah dilakukan proses *feature extraction* dengan menggunakan *TF Binary* yang bila kata yang terdapat pada *tweet* ada maka bernilai 1 sedangkan kalau tidak ada bernilai 0. Hasil pada Tabel 3.5 hanya sebagai contoh saja, sebernanya atribut yang digunakan sebanyak 258 atribut.

**Tabel 3.5** Contoh *TF Binary*

| No | bahwa | belum | aman | segenap | rakyat | indonesia | apapun | suku | agamanya | akan | gagal |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 3.5 Proses KNN

Untuk proses ini data training yang telah melalui proses *preprocessing* dan *feature extraction* digunakan harus sudah mempunyai kelas. Selanjutnya setelah data *training* dan data *testing* telah siap, maka proses *Distance* KNN bisa dilakukan. Prosesnya adalah sebagai berikut:

1. Proses pertama adalah mencari jarak antara *record* data *testing* ke *record* data *training* yang sudah berbentuk *vector* dengan menggunkan rumus *Euclidean Distance*. Rumus *Euclidean Distance* pada masalah ini adalah sebagai berikut:

$$dist = sqrt((test(i,k) - train(j,k))\text{^}2$$

Dimana *test* (i,k ) adalah seluruh *record* dan atribut yang ada pada data *testing* sedangkan *train*(j,k) adalah seluruh *record* dan atribut yang ada pada data *testing*. Tabel 3.6 adalah tabel jarak dari *record tweet* pada tabel 3.5 ke setiap *record tweet* data *training* yang digunakan penulis.

2. Yang kedua setelah mendapatkan jarak dengan menggunakan rumus *Euclidean Distance* selanjutnya adalah menentukan jumlah k atau tetangga berdasarkan jarak pada tabel 3.6. K yang digunakan untuk contoh kali ini adalah 7, jadi dari tabel 3.6 diambil jarak 7 terdekat berdasarkan jumlah k yang ditentukan.

3.Yang ketiga adalah mengurutkan hasil dari nomer 2 secara *ascending* yaitu mengurutkan jarak dari paling kecil ke jarak paling 7 berdasarkan tetangga yang telah ditentukan.

**Tabel 3.6** Hasil dari proses 1-3

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 2.645751 |
| 0 | 0 | 2.645751 |
| 0 | 0 | 2.645751 |
| 0 | 0 | 2.828427 |

4.Yang keempat setelah mendapatkan K yang telah diurutkan secara *ascending* selanjutnya yaitu melakukan klasifikasi. Tabel dibawah ini adalah tabel *tweet* dari tabel 3.6. Jadi untuk data *testing record tweet* 1 dan 2 jarak terdekatnya dengan data *training* adalah *record tweet* 1,2,4,6,7,8 pada data *training* sedangkan untuk data *testing record tweet* 3 jarak terdekatnya adalah dengan 3, 203, 403, 68, 268, 486, 5.

**Tabel 3.7** Tabel *tweet* berdasarkan jarak

| | | |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 2 | 203 |
| 4 | 4 | 403 |
| 6 | 6 | 68 |
| 7 | 7 | 268 |
| 8 | 8 | 468 |
| 10 | 10 | 5 |

**Tabel 3.8** Kelas data *testing*

| *Record Tweet* | Kelas |
|---|---|
| 1 | Postif |
| 2 | Positif |
| 3 | Positif |

Setelah mendapatkan *record tweet testing* berdeketan dengan *record tweet training* selanjutnya yaitu melakukan klasifikasi. Proses klasifikasinya yaitu dengan mengikuti kelas mayoritas dari K sebelumnya. Tabel 3.8 adalah tabel setelah dilakukan pengujian dengan l yang telah ditentukan sebelumnya yaitu 7 didapatkan untuk *record tweet* 1, 2, 3 data *testing* kelasnya adalah kelas bagus karena dari 7 *tweet* pada tabel 3.7 mayoritasnya kelas bagus.

## 4. Pengujian dan Analisis

### 4.1 *Confusion Matrix*

*Confusion matrix* untuk membandingkan hasil prediksi dengan algoritma dengan data *testing* yang sebenarnya. Variabel yang digunakan dalam mengevaluasi algoritma KNN adalah *precision, recall, f1-measure,* dan akurasi.

1. *Confusion matrix* pada data*set* 80%:20% dan k = 3

**Tabel 4.1** *Confusion matrix* pada data*set* 80%:20% dan k = 3

| *Dataset* 80%:20% & k = 3 | | | | |
|---|---|---|---|---|
| Prediksi | | | | |
| Asli | | Positif | Negatif | Total |
| | Positif | 136 | 14 | 150 |
| | Negatif | 8 | 42 | 50 |
| | Total | 144 | 56 | 200 |

Berdasarkan tabel tersebut (Tabel 4.1) dapat diketahui bahwa:

1. *Tweet* positif yang diprediksi secara benar oleh algoritma berjumlah (*True Positive* - TP) 136. Jumlah *tweet* yang salah diprediksi (*False Positive* - FP) adalah 14.
2. *Tweet* negatif yang diprediksi secara benar oleh algoritma berjumlah 42 (*True Negative* - TN). Jumlah *tweet* yang salah diprediksi (*False Negative* - FN) adalah 8.

Dengan demikian, dapat diketahui bahwa hasil perhitungan *precision, recall,* dan *f1-measure* adalah seperti berikut (Tabel 4.2).

**Tabel 4.2** hasil perhitungan *precision, recall,* dan *f1-measure*

| Label | *Precision* | *Recall* | *Fl-measure* |
|---|---|---|---|
| Positif | 0.906666667 | 0.944444444 | 0.925170068 |
| Negatif | 0.84 | 0.75 | 0.79245283 |

2.*Confusion matrix* pada *dataset* 70%:30% dan k = 3

**Tabel 4.3** *Confusion matrix* pada *dataset* 70%:30% dan k = 3

| *Dataset* 70%:30% & k = 3 | | | | |
|---|---|---|---|---|
| Prediksi | | | | |
| Asli | | Positif | Negatif | Total |
| | Positif | 157 | 48 | 205 |
| | Negatif | 41 | 54 | 95 |
| | Total | 198 | 102 | 300 |

Berdasarkan tabel tersebut (Tabel 4.9) dapat diketahui bahwa:

1. *Tweet* positif yang diprediksi secara benar oleh algoritma berjumlah (*True Positive* - TP) 157. Jumlah *tweet* yang salah diprediksi (*False Positive* - FP) adalah 48.
2. *Tweet* negatif yang diprediksi secara benar oleh algoritma berjumlah 54 (*True Negative* - TN). Jumlah *tweet* yang salah diprediksi (*False Negative* - FN) adalah 41.

Dengan demikian, dapat diketahui bahwa hasil perhitungan *precision, recall*, dan *f1-measure* adalah seperti berikut (Tabel 4.10).

**Tabel 4.4** hasil perhitungan *precision, recall,* dan *f1-measure*

| Label | *Precision* | *Recall* | *Fl-measure* |
|---|---|---|---|
| Positif | 0.7658537 | 0.7929293 | 0.7791563 |
| Negatif | 0.5684211 | 0.5294118 | 0.5482234 |

3. *Confusion matrix* pada *dataset* 60%:40% dan k = 3

**Tabel 4.5** *Confusion matrix* pada *dataset* 60%:40% dan k = 3

| *Dataset* 60%:40% & k = 3 | | | | |
|---|---|---|---|---|
| Prediksi | | | | |
| Asli | | Positif | Negatif | Total |

| | | | | |
|---|---|---|---|---|
| | Positif | 144 | 93 | 237 |
| | Negatif | 46 | 117 | 163 |
| | Total | 190 | 210 | 400 |

Berdasarkan tabel tersebut (Tabel 4.17) dapat diketahui bahwa:

1. *Tweet* positif yang diprediksi secara benar oleh algoritma berjumlah (*True Positive* - TP) 144. Jumlah *tweet* yang salah diprediksi (*False Positive* - FP) adalah 93.
2. *Tweet* negatif yang diprediksi secara benar oleh algoritma berjumlah 117 (*True Negative* - TN). Jumlah *tweet* yang salah diprediksi (*False Negative* - FN) adalah 46.

Dengan demikian, dapat diketahui bahwa hasil perhitungan *precision*, *recall*, dan *f1-measure* adalah seperti berikut (Tabel 4.18).

**Tabel 4.6** hasil perhitungan *precision, recall,* dan *f1-measure*

| Label | *Precision* | *Recall* | *Fl-measure* |
|---|---|---|---|
| Positif | 0.6075949 | 0.7578947 | 0.6744731 |
| Negatif | 0.7177914 | 0.5571429 | 0.6273458 |

Berdasarkan tabel-tabel diatas hasil *confusion matrix* terbaik adalah hasil dengan *dataset* 80% data *training* dan 20 % data *testing* dengan k = 3 dengan *precesion* positif 90%, *precesion* negatif 84%, *recall* postif 94%, *recall* negatif 75% dan *fl-measure* positif 92%, *fl-measure* negatif 79%.

### 4.2 Pengujian dan Analisis Pada Skenario

Dari skenario yang telah dibuat pada dilakukan pengujian terhadap data *split* dan jumlah k. Pengujian dilakukan pada jumlah k = 3, k = 5, k=7, k=9 dengan tiap pengujian jumlah digunakan skenario data *split.*

1. Pengujian pada k = 3

Tabel dibawah ini adalah hasil pengujian metode KNN dengan k = 3 dan 3 skenario *split* yaitu 80% data *training* dan 20% data *testing*, 70% data *training* dan 30% data *testing* , 60% data *training* dan 40% data *testing.*

**Tabel 4.19** Perbandinga*n* Accuracy Data Split

| k = 3 | |
|---|---|
| Data *Split* | *Accuracy* |
| 80-20 | 90.00% |
| 70-30 | 73.00% |
| 60-40 | 62.00% |

2. Pengujian pada k = 5

Tabel dibawah ini adalah hasil pengujian metode KNN dengan k = 5 dan 3 skenario *split* yaitu 80% data *training* dan 20% data *testing*, 70% data *training* dan 30% data *testing* , 60% data *training* dan 40% data *testing*

**Tabel 4.19** Perbandinga*n Accuracy* Data *Split*

| k = 5 | |
|---|---|
| Data *Split* | *Accuracy* |
| 80-20 | 90.00% |
| 70-30 | 73.00% |
| 60-40 | 62.00% |

3. Pengujian pada k = 7

Tabel dibawah ini adalah hasil pengujian metode KNN dengan k =7 dan 3 skenario *split* yaitu 80% data *training* dan 20% data *testing*, 70% data *training* dan 30% data *testing*, 60% data *training* dan 40% data *testing*.

**Tabel 4.20** Perbandingan *Accuracy* Data *Split*

| k=7 |
|---|

| Data *Split* | *Accuracy* |
|---|---|
| 80-20 | 86.00% |
| 70-30 | 70.67% |
| 60-40 | 61.50% |

4. Pengujian pada k = 9

Tabel dibawah ini adalah hasil pengujian metode KNN dengan k = 9 dan 3 skenario *split* yaitu 80% data *training* dan 20% data *testing*, 70% data *training* dan 30% data *testing*, 60% data *training* dan 40% data *testing*.

**Tabel 4.21** Perbandingan *Accuracy* Data *Split*

| k = 9 | |
|---|---|
| Data *Split* | *Accuracy* |
| 80-20 | 85.50% |
| 70-30 | 70.33% |
| 60-40 | 60.50% |

Dari tabel diatas dapat lihat bahwa akurasi yang terbaik adalah skenario dengan data *split* 80% data *training* dan 20% data *testing* dan jumlah k = 3, k = 5. Dengan demikian bahwa dapat dianalisa bahwa semakin banyak data *training* yang digunakan maka hasil akurasi yang didapatkan semakin tinggi tetapi untuk penentuan k tidak mementukan akurasi karena belum mencoba untuk k lainya.

## 5. Kesimpulan dan Saran

### 5.1 Kesimpulan

Dari hasil analisis dan pengujian yang telah dilaksanakan maka kesimpulan yang didapat adalah :

1. Skenario pada pengujian dengan perbandingan data *training* dan *testing* 80%:20% memiliki akurasi yang lebih tinggi dari pada yang lain yaitu 90.50%. Sehingga dapat diambil kesimpulan bahwa dalam proses KNN atau lebih tepatnya klasifikasi semakin banyak data *training* yang digunakan maka akurasi yang didapat semakin bagus karena akan banyak proses pembelajaran pada data *training*.

2. Pada proses penelitian ini jumlah k yang ditentukan sangat berpengaruh pada proses KNN. Berdasarkan percobaan yang dilakukan dapat diambil kesimpulan bahwa semakin besar nilai k maka akan memperbesar jumlah kebenaran pada proses KNN ini tetapi tidak menutup kemungkinan untuk k yang tidak dicoba mempunyai akurasi yang lebih kecil ataupun lebih besar.

3. Setiap skenario pengguna *twitter* lebih banyak berpendapat bahwa kinerja pemerintah untuk periode kali bagus di segala aspek dengan menggunakan algoritma KNN.

### 5.1 Saran

Dari hasil analisis dan pengujian yang telah dilaksanakan maka kesimpulan yang didapat adalah :

1. Untuk *hasil preprocessing* yang lebih bagus dan maksimal dapat dikembangkan menggunakan *stemming*. Karena kata bahasa indonesia kebanyakan tidak teratur sehingga susah buat diolah.

2. Sebaiknya untuk *dataset tweet* lebih diperbanyak agar dapat menambah keragaman dan jumlah informasi dari *tweet* yang lain.

## Daftar Pustaka

[1] Ridok Achmad, Latifa Retnani. Klasifikasi Teks Bahasa Indonesia Pada Corpus Tak Seimbang Menggunakan NWKN.

[2] Lu ,Yue.2011.Automatic Construction of a Context-Aware Sentiment Lexicon: An Optimization Approach .
[3] Qiu Guang, Liu Bing. Expanding Domain Sentiment Lexicon through Double Propagation.
[4] Wang ,Xiaolong .Topic Sentiment Analysis in Twitter: A Graph-based Hashtag Sentiment Classification Approach.
[5] Falahah, Adriani Nur . Pengembangan Aplikasi Sentiment Analysis Menggunakan Metode Naïve Bayes.
[6] Toker Gülen, Kırmemiş Öznur. Text Categorization Using K-Nearest Neighbor Classification.
[7] Agarwal Apoorv, Xie Boyi. Sentiment Analysis of Twitter Data.
[8] Pak Alexander, Paroubek Patrick. Twitter as a Corpus for Sentiment Analysis and Opinion Mining .
[9] Kouloumpis Efthymios, Wilson Theresa. Twitter Sentiment Analysis : The Good the Bad and the OMG!.
[10] Rivki Muhammad, Adam Bachtiar. Implementasi Algoritma K-Nearest Neighbor Dalam Pengklasifikasian Follower Twitter Yang Menggunakan Bahasa Indonesia
[11] Guo Gongde, Wang Hui. KNN Model-Based Approach in Classification.
[12] Basu, Walters, Shepherd.Support vector machines for text categorization.
[13] Zhang Haiyi, Li Di. Naïve Bayes Text Classifier.
[14] Setiawan Bagus. Klasifikasi Topik Berita Berbahasa Indonesia menggunakan Weighted K-Nearest Neighbor
[15] Renata Aloysia. Analisis Klasifikasi Opini Pada Jejaring Sosial Twitter Menggunakan Algoritma K-Nearest Neighbor (KNN)