BAB 3 ANALISIS DAN PERANCANGAN

Pada Bab berisi menjelaskan mengenai analisis masalah yang ada ingin diatasi dengan pendekatan dan implementasi, dimulai data *preprocessing* , *Information Gain*, sampai proses klasifikasi.

3.1 Analisis Masalah

Permasalahan yang menjadi fokus utama pada penelitian ini adalah melakukan perbandingan akurasi memakai *Information Gain* atau TF-IDF saat melakukan klasifikasi. Pendeteksian ini dilakukan untuk mengetahui sebuah konten berkata kasar di media sosial dan menggunakan dataset *tweet* yang telah disiapkan.

Dataset yang digunakan didapat dari sebuah situs penyedia database yaitu Kaggle. Dataset yang digunakan berisikan 13169 *tweet* yang telah diberikan pelabelan. Label yang telah tersedia berupa angka 1 berarti ada kata kasar dan angka 0 tidak ada kata kasar. Dimana data dengan pelabelan tidak ada kata kasar terdapat sebanyak 8126 *tweet* dan 5043 data *tweet* dengan pelabelan ada kata kasar.

Dataset akan menggunakan 2 dataset yaitu dataset asli dan dataset pelabelan manual.

Dataset yang ada sudah memiliki label akan dibagi menjadi dataset latih dan dataset uji. Pelatihan dan pengujian dengan rasio pelatihan : pengujian sebesar 90 : 10. Sebelum masuk ke proses klasifikasi, setiap data latih dan uji akan melewati proses text *preprocessing* dan seleksi fitur terlebih dahulu.

Proses text *preprocessing* yang akan dilakukan antara lain *case folding*, *remove emoticon*, *remove unnecessary word*, *remove non alphaNumeric*, *replace slang word*, *stemming*, dan *stop word removal*. Text *preprocessing* dilakukan untuk menghilangkan *noise* pada data *tweet* yang diolah, agar perhitungan ekstraksi fitur agar lebih optimal sehingga menghasilkan analisis sentimen yang akurat.

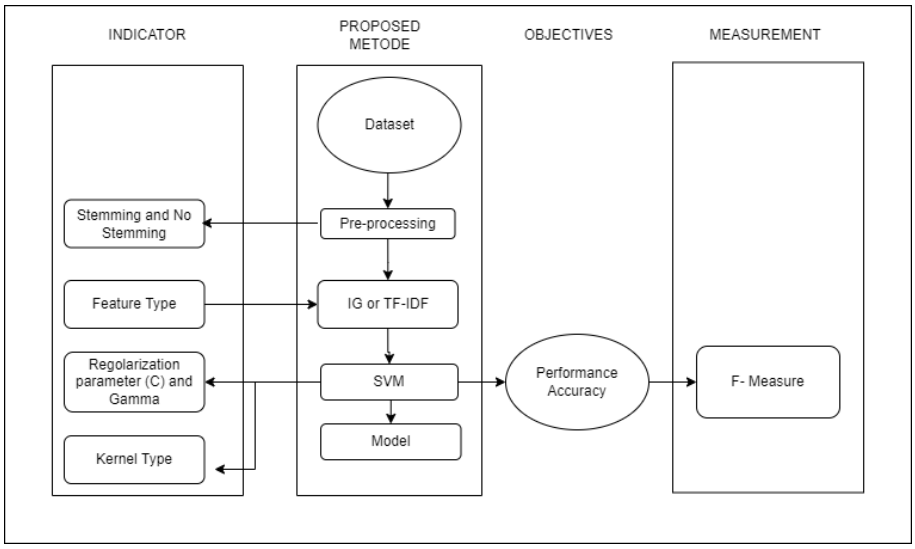
Fitur-fitur yang akan digunakan dari data *tweet* yang sudah bersih antara lain *bag-of-word*, *Information Gain* untuk pembobotan kata, barulah proses latih atau uji dapat dilakukan.

Saat Melakukan klasifikasi akan dibagi menjadi menggunakan *Information Gain* atau tidak. Klasifikasi akan menggunakan SVM dan akan menggunakan *kernel linear , RBF dan polynomial*. Hasil dari penelitian ini akan membandingkan dan

menyimpulkan bahwa *Information Gain* akankah mempengaruhi akurasi dari klasifikasi.

3.2 Kerangka Pemikiran

Berikut Gambar 3.1 merupakan gambaran dari kerangka pemikiran dari metode yang digunakan penelitian ini:



Gambar 3.1 Kerangka pemikiran

Proses dimulai dengan dataset yang berupa *tweet*. Dataset yang ada selanjutnya dilakukan *preprocessing* untuk membersihkan data dari *noise* atau kata yang tidak berpengaruh dalam pendeteksian, setelah data sudah bersih selanjutnya akan dilakukan seleksi fitur.

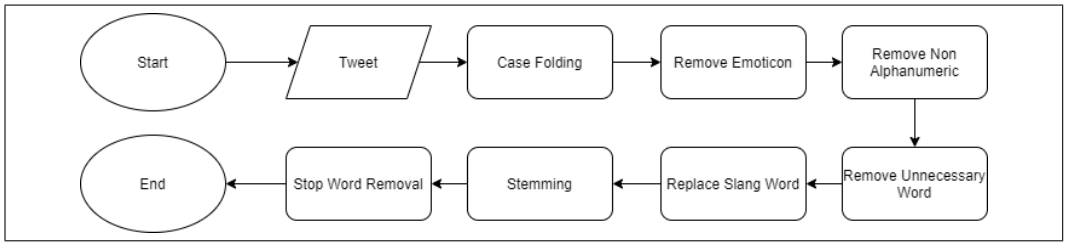
Pada proses seleksi fitur menggunakan *Information Gain* untuk mendapatkan fitur-fitur dari dataset untuk di klasifikasi menggunakan SVM. SVM memiliki berapa *kernel* untuk penelitian ini akan menggunakan *kernel* linear , RBF, *sigmoid* ,dan *polynomial*. Pada setiap *kernel* akan dilakukan penelitian untuk parameter C dan *gamma* dan hasil dari perhitungan tersebut akan dilihat akurasi dan *f-measure* terbaik. Dimana akurasi akan menunjukkan seberapa akurat dalam melakukan pendeteksian kata kasar dan sedangkan pengukuran *f-measure* akan mengukur seberapa stabil model yang telah dibuat.

3.3 Analisis Urutan Proses Global

Pada sub-bab ini akan membahas mengenai urutan pada proses fungsi-fungsi yang ada dalam sistem mendeteksi sebuah konten yang mengandung kata kasar atau.

3.3.1 Preprocessing

Proses *preprocessing* adalah proses dimana semua data akan dibersihkan dari *noise* agar dataset yang dipakai lebih mudah untuk dilakukan klasifikasi.



Gambar 3.2 *Flowchart preprocessing*

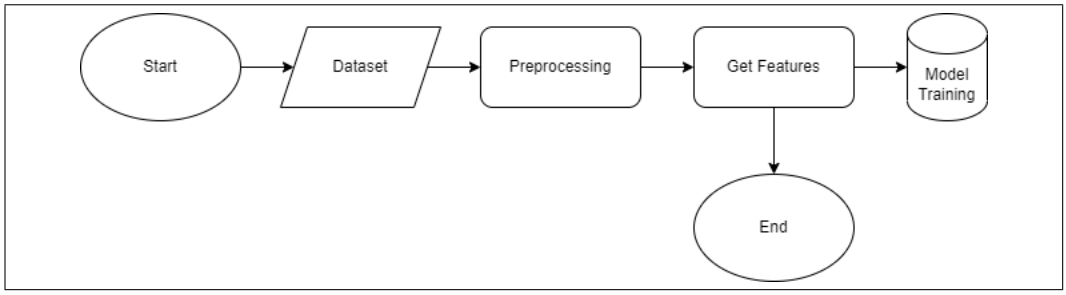
Berikut ini adalah uraian *flowchart* pada Gambar 3.2: yang akan dilakukan penelitian ini:

1. Data *tweet* akan diubah menjadi huruf kecil dengan proses *case folding*
2. Setelah *case folding* dilakukan lalu menghapus *emoticon* dengan cara memakai pustaka RegEX.
3. Menghilangkan semua tanda baca atau semua yang tidak ada hubungannya dengan angka atau huruf atas bantuan pustaka RegEX dengan.
4. Kemudian semua data *tweet* dihilangkan URL dan *mention* menggunakan pustaka RegEX.
5. Data *tweet* yang menggunakan *slang word* akan di ganti menjadi kata baku dibantu Tinjauan Objek 2.3.2.2
6. Lalu data *tweet* akan dicari kata dasar dengan melakukan *stemming* dengan menggunakan pustaka Sastrawi.
7. Menghilangkan kata-kata tidak ada arti pesifik dengan refrensi dataset Tinjauan Objek 2.3.2.3
8. Melakukan kalkulasi setiap kata untuk memberikan bobot menggunakan seleksi fitur dengan metode *Information Gain*.
9. Menyimpan model SVM yang berisi data latih yang akan dilakukan pada proses *testing*.

3.3.2 Proses *Training*

Proses *training* adalah proses pelatihan data sebelum proses uji untuk mendapatkan model yang nantinya model tersebut akan digunakan pada proses

pengujian menggunakan SVM.



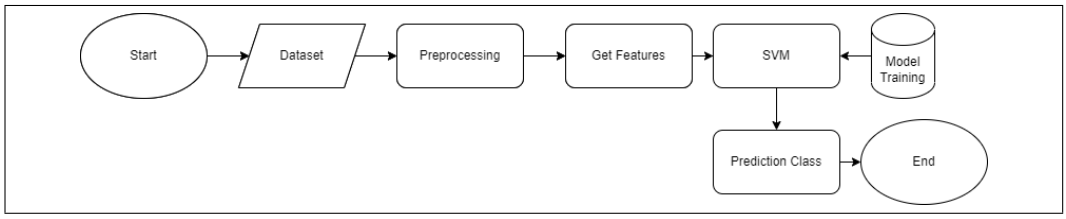
Gambar 3.3 *Flowchart training*

Berikut ini adalah uraian *flowchart* pada Gambar 3.3: yang akan dilakukan penelitian ini:

1. Menyiapkan dataset yaitu data *tweet* yang telah diberi label terlebih dahulu.
2. *Pre-processing* adalah proses dimana semua data akan dibersihkan dari *noise* agar dataset yang dipakai lebih mudah untuk dilakukan klasifikasi.
3. Data akan diberikan bobot pada proses *Get Features*menggunakan *Information Gain* atau TF-IDF.
4. Klasifikasi SVM digunakan melakukan prediksi untuk menentukan data *tweet* ada atau tidak kata kasar.

3.3.3 Proses *Testing*

Proses *testing* akan dilakukan untuk mengetahui hasil dari pengujian.



Gambar 3.4 *Flowchart testing*

Berikut ini adalah uraian *flowchart* pada Gambar 3.4 yang akan dilakukan penelitian ini:

1. Dataset *tweet* yang sudah dibagi dengan data *training* akan digunakan untuk proses pengujian
2. Inisialisasi data dengan menggunakan model SVM yang sudah di dapat dari proses *training*.

3. Setelah melewati proses klasifikasi SVM maka akan dihitung akurasi dan *f-measure*.

3.4 Data Sampling

Dataset diambil dari Kaggle akan menggunakan dan dibagi menjadi dua jenis dataset yaitu dataset asli dan dataset yang sudah diubah dengan melakukan pelabelan secara manual. Dataset hanya menggunakan 2 kolom yaitu *Tweet* dan *Abusive* karena kolom lain hanya akan digunakan apabila ada penelitan khusus mengenai *hate speech*. Dimana penelitan kali ini berfokus kepada pendeteksian kata kasar atau *abusive* saja. Berikut adalah contoh dari data sampling yang akan digunakan dapat dilihat pada Gambar 3.5:

	Tweet	Abusive
0	cowok usaha lacak perhati gue lantas remeh per...	1
1	telat tau edan sarap gue gaul ciga ifla cal licew	1
2	41 kadang pikir percaya tuhan jatuh kali kali ...	0
3	ku tau mata sipit lihat	0
4	kaum cebong kafir lihat dongok dungu haha	1

Gambar 3.5 Contoh data sampling

3.4.1 Dataset Asli

Dataset Asli adalah dataset yang diambil dari Kaggle dan tidak diubah sama sekali dalam penelitian ini untuk menjadi perbandingan dengan dataset pelabelan manual. Dataset asli juga tidak didasarkan teori Subbab 2.1.2.

3.4.2 Dataset Pelabelan Manual

Dataset pelabelan manual adalah dataset yang dibuat dari dataset asli yang akan diperiksa ulang berdasarkan teori kata kasar pada Subbab 2.1.2. Hasil dari dataset pelabelan manual bisa dilihat di pada Lampiran A-1. Berikut langkah melakukan pelabelan manual:

1. Untuk membantu mengetahui daftar kata kasar yang tersedia pada dataset asli. Pembuatan dataset membuat juga daftar kata kasar yang di jelaskan di Subab 2.3.2.4 tentang kamus *abusive* sebagai bantuan kata-kata yang akan diperbarui.

2. Pelabelan manual dilakukan dengan menggunakan Microsoft Excel dengan *shortcut* ”CTRL + F” untuk mencari kata kunci yang ingin dicari dari 126 kata

kasar yang ada di kamus *abusive*. Lalu setiap kata kunci akan dibandingkan dengan dataset asli.

3. Jumlah kata yang mengandung kata kasar berjumlah 11140 kata yang tersebar di seluruh kolom *Tweet*.

4. Jika menemukan *tweet* yang tidak sesuai teori Subbab 2.1.2 akan diubah dan diberi alasan mengapa label diberi perubahan.

5. Dataset dengan pelabelan manual berhasil mengubah 89 data *tweet* yang ada di dataset asli, seperti kata ”anjing”, ”komunis”, ”sampah, ”kafir”, dan lain lain. Perubahan data meliputi dari kata kasar menjadi tidak kasar dan kata tidak kasar menjadi kasar.

3.5 Analisis Kasus

Pada bagian ini dilakukan analisis proses secara bertahap.

3.5.1 *Tweet preprocessing*

Berikut adalah contoh data *tweet* yang diambil dari data dengan nomor 1208 untuk sampel data yang akan digunakan untuk *preprocessing*:

USER USER USER Segitu gentarnya dia takut terhadap Kafir...\xf0\x9f\x98\x81”

3.5.1.1 *Case folding*

Proses *case folding* ini menggunakan fungsi yang sudah di sediakan oleh pyton yaitu parameter text di isi kalimat yang ingin diubah ke huruf kecil lalu memanggil lower() untuk mengubah semua dataset menjadi huruf kecil. Data masukan akan berubah menjadi

user user user segitu gentarnya dia takut terhadap kafir...\xf0\x9f\x98\x81”

3.5.1.2 *Remove Emoticon*

Emoji pada dataset berupa hasil convert dari tipe data *byte* seperti Gambar 3.6 supaya tidak mengganggu saat pembobotan maka dilakukan *Remove Emoticon* seperti contoh \xf0\x9f\x98\x81 yang wujud emojinya adalah wajah menyeringai dengan mata yang tersenyum. Semua emoji di dataset yang sudah berubah tipe data menjadi *byte* memiliki pola diawali dengan ”\” maka dari itu ”\” akan dihapus menggunakan fungsi dari tabel 2.5 menggunakan parameter seperti sub(”\”, ” ”,text) untuk mengganti ”\” menjadi spasi. berikut hasil *Remove Emoticon*:

user user user segitu gentarnya dia takut terhadap kafir...”

3.5.1.3 *Remove Unnecessary Word*

Banyak kata yang tidak digunakan seperti ”user”, ”url” dan ”\n” maka harus dihapus. *Remove Unnecessary Word* akan menggunakan fungsi dari tabel 2.5 untuk menghapus kata yang mengandung kata ”user” akan menggunakan parameter seperti sub(’user’,’ ’,text), dan berikut hasil penghapusan kata-kata yang tidak dipakai.

segitu gentarnya dia takut terhadap kafir...”

3.5.1.4 *Remove Non AlphaNumeric*

Berfungsi untuk menghapus tanda baca karena tidak diperlukan dalam proses ini. *Remove Non AlphaNumeric* akan menggunakan fungsi pustaka RegEx untuk menghapus tanda baca yang tidak dipakai.

segitu gentarnya dia takut terhadap kafir

3.5.1.5 *Replace Slang Word*

Proses *Replace Slang Word* akan menggunakan file dari Tinjauan Objek 2.3.2.2 proses ini akan melakukan looping dari kumpulan kata yang terdapat pada kamus *new_kamusalay*. Setiap kata yang ada dalam kamus akan menggubah kata *slang* menjadi kata baku. Sepert contoh kata ”ongkir” menjadi ”ongkos kirim” atau ”ngingetin” menjadi ”mengingatkan” *Replace Slang Word* akan menggunakan fungsi join() dari dari Tabel 2.4 untuk mengganti *slang word*.

sebegitu gentarnya dia takut terhadap kafir

3.5.1.6 *Stemming*

Stemming adalah proses mengubah sebuah kata menjadi kata dasar seperti contoh pada kata ”sebegitu” terdapat kata depan ”se”. Semua kata awalan dan imbuhan akan dilakukan penghapusan untuk mendapatkan kata dasar Proses *stemming* akan dibantu pustaka Sastrawi di Tabel 2.6 dengan menggunakan fungsi stem() untuk mencari kata dasar dari bahasa Indonesia. Berikut adalah langkah untuk proses *stemming*:

1. Sastrawi akan mencari suatu kata pada kamus, jika menemukan kata dasar maka algoritma akan berhenti.

2. Menghilangkan Inflection Suffixes(Akhiran) akhiran seperti : “-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”.
3. Menghilangkan Derivation Suffixes(imbuhan turunan) imbuhan seperti : ”-i”, ”-kan”, atau ”-an”.
4. Menghilangkan Derivation Prefix(awalan turunan) awalan seperti : ”be-”, ”di-”, ”ke-”, ”me-”, ”pe-”, ”se-” dan ”te-”.
5. Bila dari ke 4 langkah di atas masih belum menemukan kata dasar, maka akan dicek pada tabel ambiguitas.
6. Bila semua proses gagal, maka akan mengembalikan kata asal.

begitu gentar dia takut hadap kafir

3.5.1.7 *Stop Word Removal*

Proses *Stop Word Removal* akan menggunakan file dari Tinjauan Objek 2.3.2.3 proses ini akan melakukan looping dari kumpulan kata yang terdapat pada kamus Tinjauan Objek 2.3.2.3 Setiap kata yang ada dalam kamus akan dihapus yang dianggap tidak memberikan pengaruh terhadap kalimat. Seperti ”begitu”, ”dia”, dan yang lainnya. Berikut adalah langkah untuk proses *Replace Stopwords*:

1. Fungsi join() akan mencacah kalimat menjadi kata-kata.
2. Setelah data sudah menjadi sebuah kata dan akan dibandingkan dengan Tinjauan Objek 2.3.2.3
3. Jika sebuah kata tersebut termasuk dalam Tinjauan Objek 2.3.2.3 akan dihapus.

gentar takut hadap kafir

3.5.2 *TF-IDF*

TF-IDF dengan model unigram akan menghasilkan nilai bobot setiap kata unigram dengan frekuensi kemunculan kata tersebut pada tingkat dokumen dan tingkat seluruh dokumen. Kata-kata yang sudah terpotong ini akan menghasilkan nilai bobot pada setiap katanya. Kata-kata yang digunakan berasal hasil *preprocessing*.

Dalam proses TF-IDF, dibutuhkan untuk membuat sebuah array yang berisikan kata-kata yang ada di dalam dataset yang diberi nama corpus. Dengan adanya corpus ini dapat membantu untuk melakukan perhitungan jumlah kata dalam dokumen. Berikut asumsi data latih pada Tabel 3.1:

Tabel 3.1 Asumsi kemunculan kata

Kata	**Tweet 1**	**Tweet 2**	**Tweet 3**	**Tweet 4**	**Jumlah**
gentar	1	1	0	1	3
takut	1	0	0	1	2
hadap	1	1	0	0	2
kafir	1	0	1	1	3

Setelah diketahui jumlah kemunculan sebuah kata. TF-IDF dapat dihitung sebagai berikut:

Tabel 3.2 Contoh perhitungan bobot TF, IDF pada data latih

Kata	**TF**	**IDF**	**TF-IDF**
gentar	1 / 4 = 0.25	log (4 / 3) = 0.125	0.25 x 0.125 = 0.0313
takut	1 / 4 = 0.25	log (4 / 2) = 0.301	0.25 x 0.301 = 0.0753
hadap	1 / 4 = 0.25	log (4 / 2) = 0.301	0.25 x 0.301 = 0.0753
kafir	1 /4 = 0.25	log (4 / 3) = 0.125	0.25 x 0.125 = 0.313

Persamaan 2.1 untuk menghitung TF. TF didapat dari kemungkinan pada jumlah kata yang muncul dengan jumlah kata pada sebuah kalimat. dari kemuncu lalu Persamaan 2.2 dihitung untuk mendapatkan IDF. IDF perhitungan setiap kata yang muncul akan dibagi dengan jumlah dokumen pada setiap kata yang muncul. Jika sudah mendapat nilai TF dan IDF baru bisa masuk ke Persamaan 2.3 untuk menghitung TF-IDF dengan TF dikali IDF akan menghasilkan nilai dari TF-IDF.

3.5.3 *Information Gain*

Information Gain akan menunjukkan seberapa banyak informasi ada atau tidaknya sebuah *term* memberikan kontribusi dalam membuat keputusan klasifikasi secara benar atau salah. Kata-kata yang digunakan berasal dari hasil *preprocessing*. Untuk memudahkan penjelasan pada Tabel 3.3 kelas 1 akan diubah menjadi kelas kasar dan kelas 0 akan diubah kelas tidak kasar. Berikut contoh *tweet* untuk mencari tahu relasi kata takut:

Tabel 3.3 Contoh relasi kata takut antara *tweet*

Tweet	**Kelas**	**Len**
anjing komen mention cebong budug **takut** anjing	Kasar	7
gentar **takut** hadap kafir	Tidak Kasar	4
banci berani balas dasar **takut**	Kasar	5

Untuk menghitung persamaannya harus mengetahui relasi kata ”takut” jadi untuk memudahkan akan menggunakan *confusional matrix* pada Tabel 2.1 untuk memudahkan perhitungan.

Tabel 3.4 Tabel perhitungan *Confusion Matrix* untuk *Information Gain*

	ec = ekasar = 1	ec = etidak kasar = 0
et = etakut = 1	2	1
et = etakut = 0	10	3

Berikut penjelasan hasil dari menggunakan *confusional matrix* pada Tabel 3.4:

1. Kata ”takut” yang mengandung kata kasar ada 2 kata.
2. Kata ”takut” yang tidak mengandung kata kasar ada 1 kata.
3. Kata ”takut” yang mengandung kata kasar ada 2 kata.
4. Kata ”takut” yang tidak mengandung kata kasar ada 1 kata.

Setelah mendapat hasil dari *confusional matrix* selanjutnya bisa memasukkannyake Persamaan 2.4

$$
\begin{aligned}
I(takut, kasar) = & \frac{2}{16}\log_2\left(\frac{16\cdot 2}{(2+1)\cdot(2+10)}\right) \quad + \frac{10}{16}\log_2\left(\frac{16\cdot 10}{(10+2)\cdot(10+3)}\right) \\
& + \frac{1}{16}\log_2\left(\frac{16\cdot 1}{(1+2)\cdot(1+3)}\right) \quad + \frac{3}{16}\log_2\left(\frac{16\cdot 3}{(3+10)\cdot(3+1)}\right) \\
& = 0.006
\end{aligned}
$$

3.5.4 *Support Vector Machine*

Proses ini akan melakukan klasifikasi dengan *Support Vector Machine*, proses ini akan menghasilkan model prediksi pada data pelatihan dan akan dipakai untuk memprediksi kata kasar atau bukan pada data pengujian.

Pada penelitian ini SVM akan menggunakan pustaka bernama SVC yang telah disediakan oleh pustaka Scikit-Learn pada Tabel 2.6. data masukan adalah data yang didapatkan melewati ekstraksi fitur menggunakan TF-IDF dan sudah memiliki bobot. Berikut masukan untuk klasifikasi SVM:

```
  (0, 7760)     0.6615524290955168
  (0, 8682)     0.31967636986735404
  (0, 9829)     0.31990602147351355
  (0, 7167)     0.4812486318996767
  (0, 5388)     0.3552679183741884
  (1, 6619)     0.2134071838589522
  (1, 9516)     0.2134071838589522
  (1, 11855)    0.2134071838589522
  (1, 10352)    0.39620099943862
  (1, 1186)     0.2915386013965746
  (1, 489)      0.3162518140184277
  (1, 6884)     0.3471258507605307
  (1, 7901)     0.49696611941102387
  (1, 6493)     0.39232033288260953
  (2, 8063)     0.46031027927227797
  (2, 9938)     0.6888648275567667
  (2, 1632)     0.5599818712703686
  (3, 829)      0.11854532135071356
  (3, 6820)     0.13865471780373412
  (3, 11379)    0.195477177482908
  (3, 7989)     0.2314249655294431
  (3, 910)      0.2314249655294431
  (3, 2724)     0.2217342582574783
  (3, 4135)     0.2217342582574783
  (3, 11415)    0.15401982215130414
```

Gambar 3.6 Data masukan

Pada Gambar 3.6 dapat dilihat ada 3 kolom pada hasil setiap baris keluaran saat melakukan proses ekstraksi fitur. Pada kolom pertama mengartikan urutan dokumen tersebut, lalu pada kolom kedua mengartikan nomor dari urutan fitur kata, dan yang terakhir adalah sebuah value atau bobot kata pada dokumen tersebut. Berikut adalah contoh perhitungan klasifikasi pada data pelatihan dengan 5 data:

Tabel 3.5 Data pelatihan

Fitur	F1	F2	F3	F4	F5	Class
D1	0.0313	0.0753	0.6616	0.0313	0.3163	0
D2	0.0753	0.0313	0.3163	0.6616	0.0313	1
D3	0.2217	0.3163	0.0753	0.3163	0.6616	1
D4	0.0313	0.6616	0.2217	0.0313	0.3163	0
D5	0.6616	0.2217	0.0313	0.2217	0.0313	1

Tabel 3.5 dapat dilihat pada setiap fitur kata akan memiliki bobot atau value untuk dilakukan klasifikasi menggunakan SVM. Ada juga kelas yang berisikan angka 0 dan 1, dimana angka 0 menunjukkan bahwa kalimat ini berisikan berkata kasar dan

0 berarti tidak ada kata kasar. Tahap selanjutnya adalah menghitung *kernel* dengan menggunakan *kernel* linear . Berikut adalah contoh perhitungan dengan nilai *kernel* linear dengan menggunakan data pelatihan Tabel 3.1:

$$
\begin{aligned}
&K(D1,D1) = D1D1 = (0.0313*0.0313)+(0.0753*0.0753)+(0.6616*0.6616)\\
&+(0.0313*0.0313)+(0.3163*0.3163) = 0.54538972\\
&K(D1,D2) = D1D2 = (0.0313*0.0753)+(0.0753*0.0313)+(0.6616*0.3163)\\
&+(0.0313*0.6616)+(0.3163*0.0313) = -0.24458613\\
&K(D1,D3) = D1D3 = (0.0313*0.2217)+(0.0753*0.3162)+(0.6616*0.0753)\\
&+(0.0313*0.3163)+(0.3163*0.6616) = -0.29973182\\
&K(D1,D4) = D1D4 = (0.0313*0.0313)+(0.0753*0.6616)+(0.6616*0.2217)\\
&+(0.0313*0.0313)+(0.3163*0.3162) = -0.29846864\\
&K(D1,D5) = D1D5 = (0.0313*0.6616)+(0.0753*0.2217)+(0.6616*0.0313)\\
&+(0.0313*0.2217)+(0.3163*0.0313) = 0.07494957
\end{aligned}
$$

Semua data akan dihitung demikian seperti perhitungan di atas, sehingga akan menghasilkan Tabel 3.6:

Tabel 3.6 Skenario pengujian parameter dan *kernel* SVM

Dokumen	D1	D2	D3	D4	D5
D1	0.54538972	0.24458613	0.29973182	0.29846864	0.07494957
D2	0.24458613	0.54538972	0.28038062	0.12379382	0.21431429
D3	0.29973182	0.28038062	0.69256367	0.45192925	0.30996694
D4	0.29846864	0.12379382	0.45192925	0.58880727	0.19116028
D5	0.07494957	0.21431429	0.30996694	0.19116028	0.53797572

Setelah selesai melakukan perhitungan menggunakan dengan *kernel* linear , selanjutnya data tersebut akan dimasukkan ke dalam persamaan linear untuk mencari nilai alpha (a) dan bias (b). Berikut adalah persamaan linear dari data pelatihan :

$$(1)a1+(-1)a2+(-1)a3+(-1)a4+(1)a5+(0)b=0$$
$$D1=(0.54538972)\alpha^1-(0.24458613)\alpha^2-(0.29973182)\alpha^3-(0.29846864)\alpha^4+(0.07494957)\alpha^5+b=0$$
$$D2=(0.24458613)\alpha^1-(0.54538972)\alpha^2-(0.28038062)\alpha^3-(0.12379382)\alpha^4+(0.21431429)\alpha^5+b=1$$
$$D3=(0.29973182)\alpha^1-(0.28038062)\alpha^2-(0.69256367)\alpha^3-(0.45192925)\alpha^4+(0.30996694)\alpha^5+b=1$$
$$D4=(0.29846864)\alpha^1-(0.12379382)\alpha^2-(0.45192925)\alpha^3-(0.58880727)\alpha^4+(0.19116028)\alpha^5+b=1$$
$$D5=(0.07494957)\alpha^1-(0.21431429)\alpha^2-(0.30996694)\alpha^3-(0.19116028)\alpha^4+(0.53797572)\alpha^5+b=0$$
$$=0.094$$

Dari persamaan linear diatas, didapatkan hasil seperti berikut:

Tabel 3.7 Hasil persamaan linear

a1	a2	a3	a4	a5	b
-0.85731403	-1.203758849	-1.977132061	1.642198431	0.943072546	-0.000049

Setelah melakukan perhitungan menggunakan persamaan linear untuk mendapatkan nilai alpha dan bias, maka dapat melakukan klasifikasi untuk menguji data pada data uji. Data uji diasumsikan dengan data *tweet* B1. Proses perhitungan akan sama dengan proses pelatihan, dengan menggunakan *kernel* linear untuk melakukan perhitungan SVM. Berikut adalah contoh data uji pada:

Tabel 3.8 Data uji

Fitur	F1	F2	F3	F4	F5	class
B1	0.0753	0.2217	0.3163	0.0313	0.6616	?

$$K(D1,D1) = D1D1 = (0.0753 * 0.0313) + (0.2217 * 0.0753) + (0.3163 * 0.6616) + (0.0313 * 0.0313) + (0.6616 * 0.3163) = 0.54538972$$
$$K(D1,D2) = D1D2 = (0.0753 * 0.0753) + (0.2217 * 0.0313) + (0.3163 * 0.3163) + (0.0313 * 0.6616) + (0.6616 * 0.0313) = -0.24458613$$
$$K(D1,D3) = D1D3 = (0.0753 * 0.2217) + (0.2217 * 0.3162) + (0.3163 * 0.0753) + (0.0313 * 0.3163) + (0.6616 * 0.6616) = -0.29973182$$
$$K(D1,D4) = D1D4 = (0.0753 * 0.0313) + (0.2217 * 0.6616) + (0.3163 * 0.2217) + (0.0313 * 0.0313) + (0.6616 * 0.3162) = -0.29846864$$
$$K(D1,D5) = D1D5 = (0.0753 * 0.6616) + (0.2217 * 0.2217) + (0.3163 * 0.0313) + (0.0313 * 0.2217) + (0.6616 * 0.0313) = 0.07494957$$

Fitur	D1	D2	D3	D4	D5
B1	0.43855875	0.15407115	0.55822769	0.42933493	0.13651685

$$f(x) = sign \sum^{T} i = 1K(B1,Di) + b = sign((-0.85731403 * 1 * 0.43855875 + -0.000049) + (-1.203758849 * -1 * 0.15407115 + -0.000049) + (-1.977132061 * -1 * 0.55822769 + -0.000049) + (1.642198431 * -1 * 0.42933493 + -0.000049) + (0.943072546 * 1 * 0.13651685 + -0.000049)) = 1$$

3.5.5 *Evaluation Matrix*

Setelah melakukan proses *testing* diasumsikan mendapatkan hasil klasifikasi seperti Tabel 3.9:

Tabel 3.9 Hasil evaluasi matrik

	Actual Value (Yes)	**Actual Value (No)**
Predicted (Yes)	3567	534
Predicted (No)	683	1690

Setelah diketahui evaluasi matrik, selanjutnya menggunakan Persamaan 2.13 untuk menghitung akurasi. Perhitungan akan sebagai berikut:

$$Accuracy = \frac{3567 + 1690}{6197} = 0.85 \times 100\% = 85\%$$

Untuk melakukan perhitungan untuk mendapat *f-measure* memerlukan nilai dari *precision* dan *recall* dahulu dari Persamaan 2.14 dan 2.15

$$Precision = \frac{3567}{3567 + 534} = 0.87 \times 100\% = 87\%$$

$$Recall = \frac{3567}{3567 + 683} = 0.84 \times 100\% = 84\%$$

Setelah mendapat hasil dari *precision* dan *recall* selanjutnya dapat dilakukan perhitungan untuk mendapatkan *f-measure* dengan persamaan 2.16.

$$F - Measure = 2 \times \frac{0.87 \times 0.84}{0.87 + 0.84} = 0.85 \times 100\% = 85\%$$