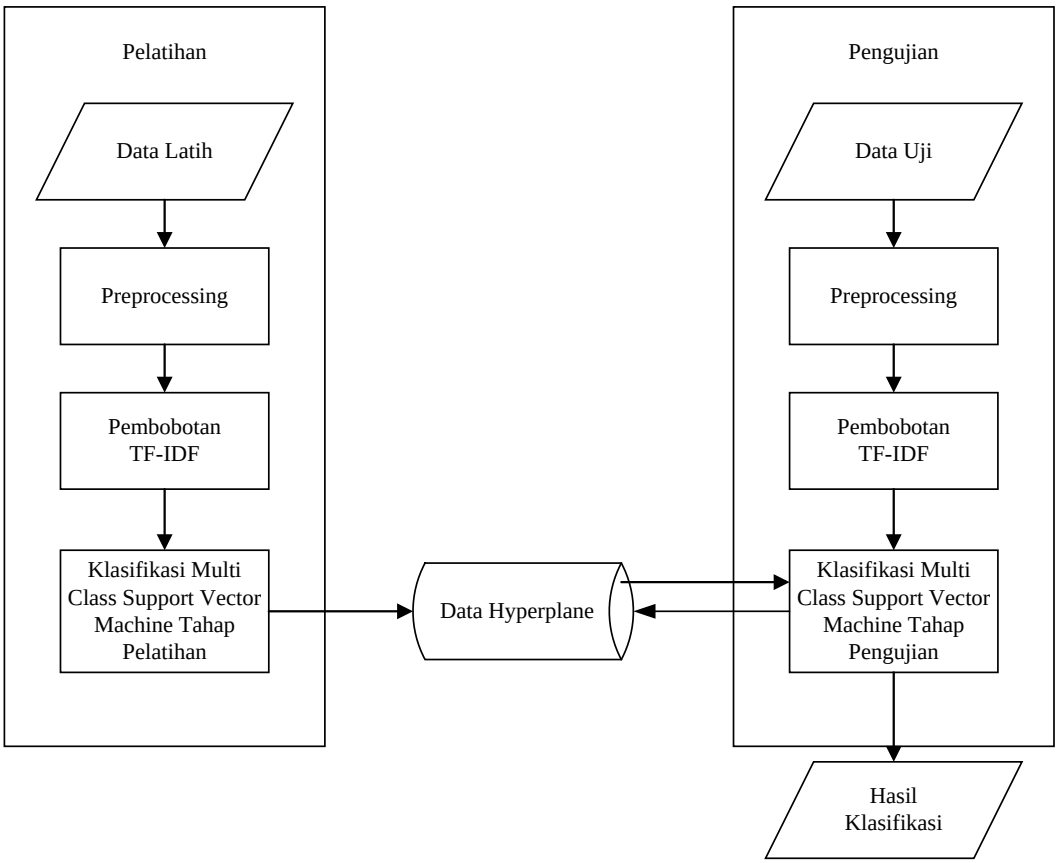
# BAB 3
# ANALISIS DAN PERANCANGAN

## 3.1 Analisis Masalah

Berdasarkan dari perumusan masalah yang sudah didapat, maka dapat dijabarkan secara lebih terperinci tentang masalah yang ditemukan dalam penelitian ini. Masalah yang ditemukan adalah pada penelitian [4] belum menghasilkan tingkat akurasi yang maksimal karena performansi metode klasifikasi yang digunakan kurang baik dalam mengklasifikasikan teks. Solusi yang diberikan adalah mengganti metode tersebut dengan metode *Multi Class Support Vector Machine* karena metode tersebut berhasil mengklasifikasikan 111 dari 120 artikel teks Bahasa Indonesia secara benar sesuai dengan kelas asli ke dalam 6 kelas kategori, yaitu *economy, defense & security, education, health, sports,* dan *politics.* Oleh karena itu, diimplementasikanlah metode *Multi Class Support Vector Machine* untuk mengklasifikasikan emosi pada lirik lagu Bahasa Indonesia.

## 3.2 Analisis Proses

Analisis proses merupakan tahapan untuk menganalisis suatu cara atau menganalisis metode-metode yang digunakan. Tahapan yang digunakan untuk mengetahui kandungan emosi dalam teks pada penelitian ini dibagi menjadi dua tahap, yaitu tahap pelatihan dan pengujian. Berikut gambaran tahapan proses yang akan dilakukan:




**Gambar 3.1 Tahapan Proses Yang Akan Dilakukan**

Berikut adalah penjelasan dari Gambar 3.1:

1. Tahap pertama yang dilakukan adalah proses pelatihan. Dalam proses pelatihan data latih akan melalui tiga proses utama yaitu proses *preprocessing*, pembobotan tf-idf dan klasifikasi *Multi Class Support Vector Machine* tahap pelatihan. Proses *preprocessing* itu sendiri terdiri dari lima proses yaitu *case folding, convert negation, tokenizing, stopword removal* dan *stemming*.
   a. *Case folding* adalah tahapan untuk mengubah semua huruf dalam dokumen menjadi *lowercase* (huruf kecil) semua.
   b. *Convert negatio*n adalah tahapan untuk mengkoversi kata-kata negasi yang terdapat pada suatu kalimat.
   c. *Tokenizing* adalah tahapan untuk memotong *string input* berdasarkan tiap kata yang menyusunnya.

d. *Stopword removal* adalah tahapan untuk menghilangkan kata yang tidak relevan pada hasil *parsing* sebuah dokumen teks dengan cara membandingkannya dengan *stoplist* yang ada.
e. *Stemming* adalah tahapan untuk mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya dengan menggunakan aturan-aturan tertentu.

Setelah proses *preprocessing*, dilakukan proses pembobotan kata menggunakan pembobotan TF-IDF. Proses selanjutnya adalah proses klasifikasi *Multi Class Support Vector* Machine tahap pelatihan. Proses ini dilakukan untuk mendapatkan data *hyperplane* dari data latih yang telah dimasukkan kemudian data *hyperplane* tersebut dimasukkan ke dalam *database*.

2. Tahap kedua yang dilakukan adalah tahap pengujian. Data uji yang dimasukkan melalui proses yang sama seperti data latih yaitu proses *preprocessing* dan pembobotan. Setelah proses pembobotan, data uji akan melalui proses klasifikasi *Multi Class Support Vector* Machine tahap pengujian. Proses ini dilakukan untuk mengklasifikasikan data uji menggunakan data *hyperplane* yang diperoleh pada tahap pelatihan. Hasil dari tahap pengujian ini adalah hasil klasifikasi berupa kategori emosi dari data uji yang telah dimasukkan.

### 3.3 Analisis Data Masukan

Data yang digunakan dalam penelitian ini terdiri dari dua jenis yaitu data latih dan data uji. Berikut data masukan yang akan digunakan:

1. Data latih yang digunakan berasal dari data *International Survey On Emotion Antecedents and Reaction* (ISEAR) yang berbahasa Inggris kemudian diterjemahkan ke dalam bahasa Indonesia tanpa mengurangi maksud dari kalimat-kalimat dalam ISEAR.
2. Data uji yang akan digunakan adalah data lirik lagu berbahasa Indonesia dengan format (.txt).

Berikut adalah sampel data mentah dari latih dan uji yang akan dilakukan dalam penelitian ini:

**Tabel 3.1 Sampel Data Latih**

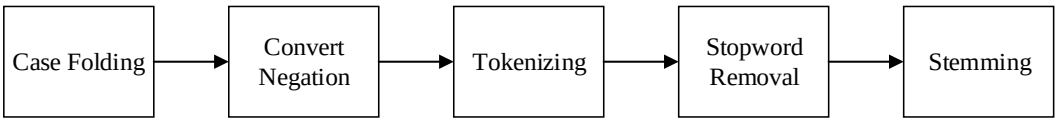
| Data Latih | Kategori | Pernyataan |
|---|---|---|
| P1 | Senang | Ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju, (saya pikir gadis ini tidak menyukai saya), ini adalah emosi senang. |
| P2 | Sedih | Saya merasa sedih ketika ayah saya sakit. |
| P3 | Marah | Saya merasa sangat marah ketika saya diperlakukan tidak adil |
| P4 | Takut | Ketakutan saya muncul dalam bentuk kecemburuan. Saya takut bahwa pacar saya jatuh cinta dengan pria lain, aku takut kehilangan dia. |
| P5 | Bersalah | Saya merasa bersalah ketika saya mengkhianati orang yang saya cintai |

**Tabel 3.2 Sampel Data Uji**

| Data Uji | Kategori | Lirik Lagu |
|---|---|---|
| L1 | ? | Aku juga ingin jatuh cinta seperti yang lainnya<br>Kini saatnya untuk jatuh cinta karna dia tlah nyatakan cinta<br>Tapi Ku takut-takut jatuh cinta<br>Takut-takut patah hatinya<br>Takut-takut jadi gila<br>Karena cinta<br>Takut-takut jatuh cinta<br>Takut-takut patah hatinya<br>Takut-takut jadi gila<br>Karena cinta<br>Namun kini bimbang yang ku rasa<br>Akankah dia trus setia<br>Atau hanya untuk sementara<br>Membuat aku kecewa<br>Tapi Ku takut-takut jatuh cinta<br>Takut-takut patah hatinya<br>Takut-takut jadi gila<br>Karena cinta,<br>Takut-takut jatuh cinta<br>Takut-takut patah hatinya<br>Takut-takut jadi gila<br>Karena cinta |

## 3.4 Analisis Preprocessing

Analisis *preprocessing* merupakan tahap awal dan salah satu langkah yang penting dalam sebuah pengklasifikasian sebuah teks. Adapun tahapan *preprocessing* yang akan dilakukan pada penelitian ini yaitu *case folding, convert negation, tokenizing, stopword removal dan stemming*. Berikut adalah gambaran tahap *preprocessing* yang dapat dilihat pada Gambar 3.2:

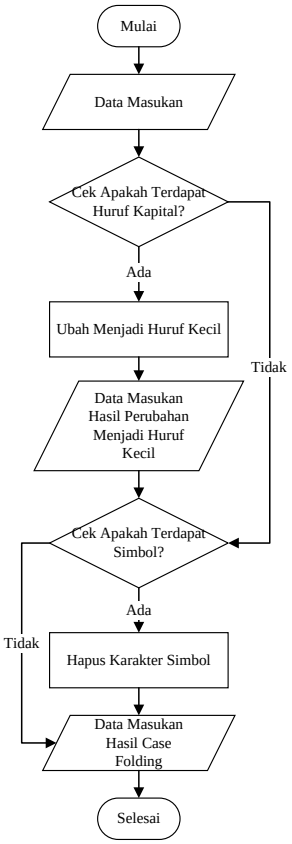



**Gambar 3.2 Tahap *Preprocessing***

*Preprocessing* ini dilakukan pada data latih dan uji yang menjadi data masukan, berikut penjelasan dari tahapan *preprocessing* yang akan dilakukan:

### 3.4.1 Case Folding

Pada tahap ini dilakukan perubahan pada semua huruf dalam dokumen menjadi huruf kecil dan menghilangkan karakter selain a-z, dengan tujuan untuk menyeragamkan karakter dalam dokumen tersebut. Adapun *flowchart* dari tahap *case folding* dapat dilihat pada Gambar 3.3 berikut:




**Gambar 3.3 *Flowchart Case Folding***

Adapun contoh tahap *case folding* diterapkan pada data latih pernyataan pertama (P1) dapat dilihat pada Tabel 3.3 berikut:

**Tabel 3.3 Tahap *Case Folding***

| Pernyataan | Tahap Case Folding | | |
|---|---|---|---|
| | **Ubah Menjadi Huruf Kecil** | **Hapus Karakter Simbol** | **Hasil Case Folding** |
| Ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju, (saya pikir gadis ini tidak menyukai saya), ini adalah emosi senang. | **k**etika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju, (saya pikir gadis ini tidak menyukai saya), ini adalah emosi senang. | ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju**, (**saya pikir gadis ini tidak menyukai saya**),** ini adalah emosi senang**.** | ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju saya pikir gadis ini tidak menyukai saya ini adalah emosi senang |

Berikut ini adalah penjelasan dari Tabel 3.3 di atas:

a. Cek huruf kapital

Data masukan P1 dilakukan pengecekan apakah terdapat huruf kapital atau tidak. Pada P1 terdapat huruf kapital yaitu huruf K pada kata pertama.

b. Ubah menjadi huruf kecil

Huruf K pada kata pertama diubah menjadi huruf kecil menjadi huruf k.

c. Cek karakter simbol

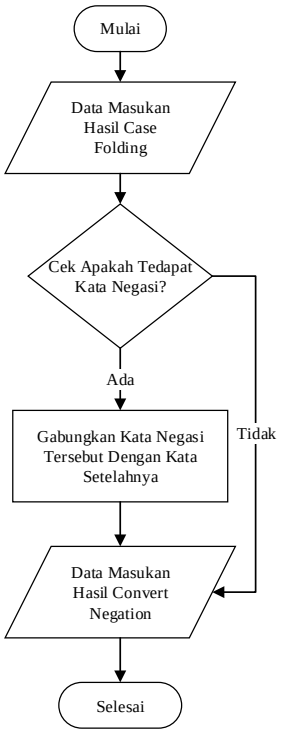
Data hasil dari perubahan menjadi huruf kecil dilakukan pengecekan apakah terdapat karakter simbol atau tidak. Pada P1 terdapat simbol koma (,), kurung buka tutup (()) dan titik (.).

d. Hapus karakter simbol

Karakter simbol koma (,), kurung buka tutup (()) dan titik (.) dihapus dari P1. Data hasil penghapusan karakter simbol disimpan sebagai data hasil *case folding* yang nantinya akan digunakan pada tahap *convert negation*.

### 3.4.2 Convert Negation

Pada tahap ini, semua dokumen yang mengandung kata-kata negasi, seperti kata “tidak”, “bukan”, “tak” dan “tanpa”. Kata tersebut akan digabungkan dengan kata setelah kata negasi tersebut. Adapun *flowchart* dari tahap *convert negation* dapat dilihat pada Gambar 3.4 berikut:




**Gambar 3.4 *Flowchart Convert Negation***

Adapun contoh tahap *convert negation* diterapkan pada data latih pernyataan pertama (P1) dapat dilihat pada Tabel 3.4 berikut:

**Tabel 3.4 Tahap *Convert Negation***

| **Tahap *Convert Negation*** | | |
|---|---|---|
| **Hasil *Case Folding*** | **Gabung Kata Negasi dengan Kata Setelahnya** | **Hasil *Convert Negation*** |
| ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju saya pikir gadis ini tidak menyukai saya ini adalah emosi senang | ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju saya pikir gadis ini **tidakmenyukai** saya ini adalah emosi senang | ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju saya pikir gadis ini **tidakmenyukai** saya ini adalah emosi senang |

Berikut ini adalah penjelasan dari Tabel 3.4 di atas:

a. Cek kata negasi

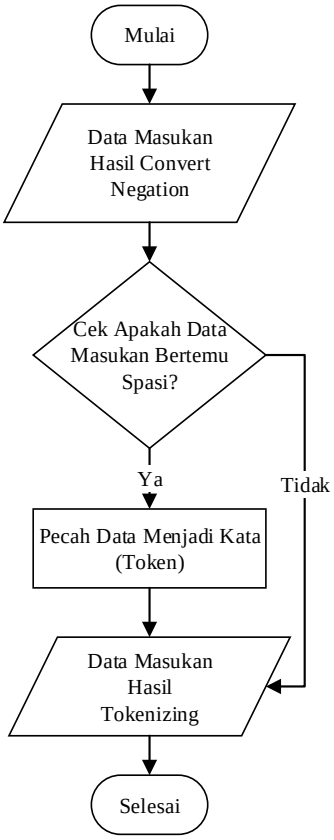
Data latih P1 hasil *case folding* dilakukan pengecekan apakah terdapat kata negasi atau tidak. Pada P1 ditemukan kata negasi yaitu kata tidak.

b. Gabung kata negasi dengan kata setelahnya

Kata negasi tidak digabungkan dengan kata setelahnya yaitu kata menyukai sehingga menjadi kata tidakmenyukai. Data hasil penggabungan kata negasi dengan kata setelahnya tersebut disimpan sebagai data hasil *convert negation* yang nantinya akan digunakan pada tahap *tokenizing*.

### 3.4.3 Tokenizing

Pada tahap ini dilakukan pemotongan kalimat berdasarkan tiap kata yang menyusunnya. Proses ini melakukan penguraian deskripsi yang semula berupa kalimat-kalimat (delimeter kata) menjadi kata-kata. Adapun *flowchart* dari tahap *tokenizing* dapat dilihat pada Gambar 3.5 berikut:




**Gambar 3.5 *Flowchart Tokenizing***

Adapun contoh tahap *tokenizing* diterapkan pada data latih pernyataan pertama (P1) dapat dilihat pada Tabel 3.5 berikut:

**Tabel 3.5 Tahap *Tokenizing***

| **Tahap *Tokenizing*** | | |
|---|---|---|
| **Hasil *Convert Negation*** | **Pecah Data Menjadi Kata** | **Hasil *Tokenizing*** |
| ketika saya bertemu dengan seorang gadis dan memintanya untuk kencan dan dia setuju saya pikir gadis ini tidakmenyukai saya ini adalah emosi senang | ketika<br>saya<br>bertemu<br>dengan<br>seorang<br>gadis<br>dan<br>memintanya<br>untuk<br>kencan<br>dan<br>dia<br>setuju<br>saya<br>pikir<br>gadis<br>ini<br>tidakmenyukai<br>saya<br>ini<br>adalah<br>emosi<br>senang | ketika<br>saya<br>bertemu<br>dengan<br>seorang<br>gadis<br>dan<br>memintanya<br>untuk<br>kencan<br>dan<br>dia<br>setuju<br>saya<br>pikir<br>gadis<br>ini<br>tidakmenyukai<br>saya<br>ini<br>adalah<br>emosi<br>senang |

Berikut ini adalah penjelasan dari Tabel 3.5 di atas:

a. Cek spasi

Data latih P1 hasil *convert negation* dilakukan pengecekan apakah data bertemu spasi atau tidak. Setiap kata pada P1 bertemu spasi.

b. Pecah data menjadi kata

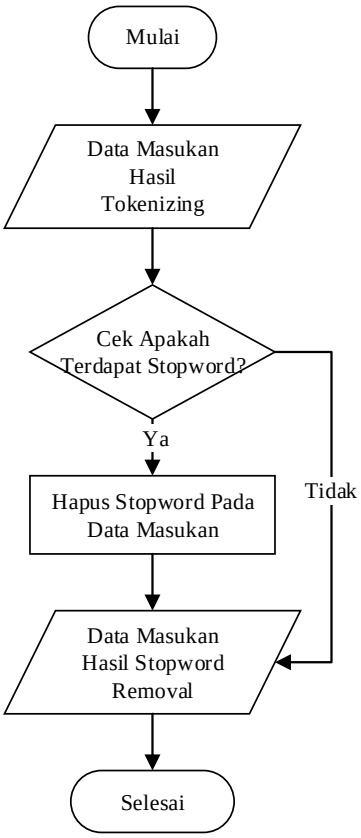
Pecah data menjadi kata setiap bertemu spasi. Data hasil pemecahan data menjadi kata tersebut disimpan sebagai data hasil *tokenizing* yang nantinya akan digunakan pada tahap *stopword removal*.

### 3.4.4 Stopword Removal

Pada tahapan ini dilakukan penghapusan sebuah kata yang dianggap tidak dapat memberikan pengaruh dalam menentukan suatu kategori tertentu dalam suatu dokumen, biasanya berupa kata ganti orang, kata penghubung, penunjuk, dan sebagainya.

Contoh *stopword* yang digunakan seperti kata: aku, yang, ketika, selama, setiap, dan, kita, untuk, saya, dan, ini, telah, bagi, paling, sementara, setelah, kami,

ke, di, sebuah, dalam, belum, bagaimana, agar, bahwa, masihkah dan lain sebagainya. Adapun *flowchart* dari tahap *stopword removal* dapat dilihat pada Gambar 3.6 berikut:




**Gambar 3.6 *Flowchart Stopword Removal***

Adapun contoh tahap *stopword removal* diterapkan pada data latih pernyataan pertama (P1) dapat dilihat pada Tabel 3.6 berikut:

**Tabel 3.6 Tahap *Stopword Removal***

| **Tahap *Stopword Removal*** | | |
|---|---|---|
| **Hasil *Tokenizing*** | **Hapus *Stopword*** | **Hasil *Stopword Removal*** |
| ketika | **ketika** | bertemu |
| saya | **saya** | gadis |
| bertemu | bertemu | memintanya |
| dengan | **dengan** | kencan |
| seorang | **seorang** | setuju |
| gadis | gadis | pikir |
| dan | **dan** | gadis |
| memintanya | memintanya | tidakmenyukai |
| untuk | **untuk** | emosi |
| kencan | kencan | senang |
| dan | **dan** | |
| dia | **dia** | |
| setuju | setuju | |
| saya | **saya** | |
| pikir | pikir | |
| gadis | gadis | |
| ini | **ini** | |
| tidakmenyukai | tidakmenyukai | |
| saya | **saya** | |
| ini | **ini** | |
| adalah | **adalah** | |
| emosi | emosi | |
| senang | senang | |

Berikut adalah penjelasan dari Tabel 3.6 di atas:

a. Cek *stopword*

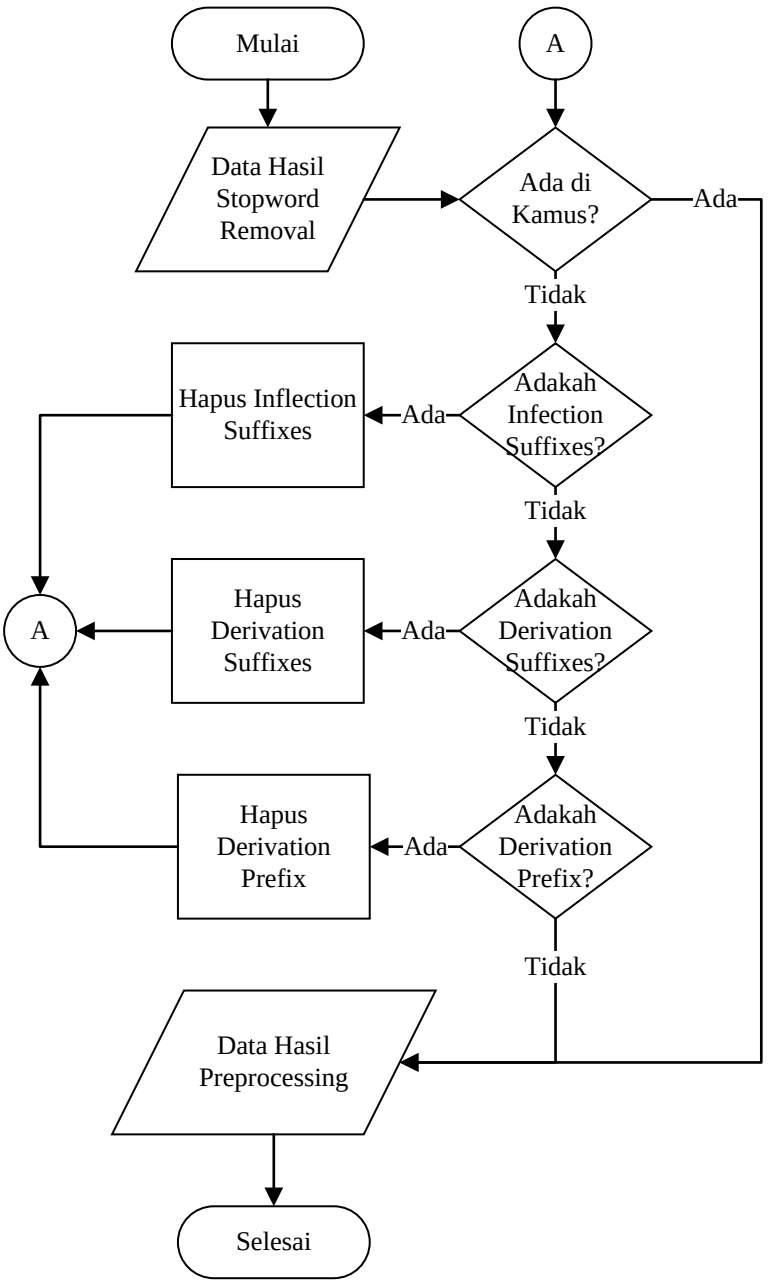
Data latih P1 hasil *tokenizing* dilakukan pengecekan apakah terdapat *stopword* atau tidak. Pada P1 terdapat *stopword* yaitu ketika, saya, dengan, seorang, dan, untuk, dia, ini dan adalah.

b. Hapus *stopword*

*Stopword* ketika, saya, dengan, seorang, dan, untuk, dia, ini dan adalah dihapus dari P1. Data hasil penghapusan *stopword* dari P1 tersebut disimpan sebagai data hasil *stopword removal* yang nantinya akan digunakan pada tahap *stemming*.

### 3.4.5 Stemming

Pada proses ini dilakukan pengolahan kata hasil dari *stopword removal* menjadi kata dasar yaitu dengan menghilangkan imbuhan awalan atau akhiran. Seperti kata “kehilangan” diubah menjadi kata “hilang”. Adapun *flowchart* dari tahap *stemming* dapat dilihat pada Gambar 3.7 berikut:




**Gambar 3.7 *Flowchart Stemming***

Adapun contoh tahap *stemming* diterapkan pada data latih pernyataan pertama (P1) dapat dilihat pada Tabel 3.7 berikut:

**Tabel 3.7 Tahap *Stemming***

| **Tahap *Stemming*** | |
|---|---|
| **Hasil *Stopword Removal*** | **Hasil Stemming** |
| bertemu | temu |
| gadis | gadis |
| memintanya | minta |
| kencan | kencan |
| setuju | setuju |
| pikir | pikir |
| gadis | gadis |
| tidakmenyukai | tidakmenyukai |
| emosi | emosi |
| senang | senang |

Berikut merupakan penjelasan dari setiap urutan proses yang dilalui kata bertemu pada tahap *stemming* hingga ditemukan kata dasarnya:

a. Cari kata dalam kamus kata dasar.

Kata bertemu tidak terdapat dalam kamus kata dasar sehingga kata bertemu akan menuju proses selanjutnya yaitu pengecekan *inflection suffixes*.

b. Cek *inflection suffixes*

Pada kata bertemu tidak memiliki akhiran -kah, -lah, -ku, -mu atau -nya maka kata bertemu akan langsung menuju proses selanjutnya yaitu pengecekan *derivation suffixes*.

c. Cek *derivation suffixes*

Pada kata bertemu tidak memiliki akhiran -i, -an atau -kan, maka kata bertemu akan langsung menuju proses penghapusan *derivation prefix*.

d. Cek *derivation prefix*

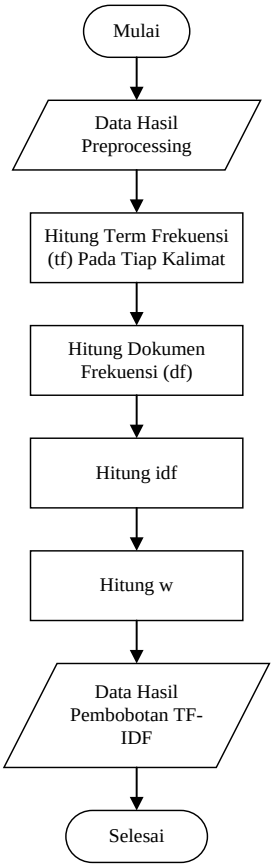
Kata bertemu memiliki tipe awalan ber- sehingga awalan ber- tersebut dihapus sehingga menjadi kata temu. Kata temu tersebut dicek apakah terdapat di dalam kamus atau tidak, ternyata kata temu ada di dalam kamus sehingga kata temu tersebut menjadi kata dasar dari kata temu dan disimpan sebagai hasil dari proses *stemming*.

Hal yang sama dilakukan pada kata gadis, memintanya, kencan, setuju, pikir, tidakmenyukai, emosi dan senang sehingga ditemukan kata dasar gadis, minta, kencan, setuju, pikir, tidakmenyukai, emosi dengan menggunakan aturan yang ada pada algoritma Nazief & Adriani. Tahap *stemming* ini merupakan tahap terakhir dalam proses *preprocessing*, kata dasar yang dihasilkan pada tahap *stemming* ini

dijadikan sebagai data hasil *preprocessing* yang nantinya akan digunakan pada tahap pembobotan TF-IDF.

## 3.5 Analisis Pembobotan TF-IDF

Dalam pengenalan emosi pada teks, pembobotan kata digunakan untuk mendapatkan suatu topik atau *keyword* dari kumpulan teks. Salah satu metode pembobotan adalah *Term Frequency-Inverse Document Frequency* (TF-IDF). Nilai bobot suatu kata (*term*) menyatakan kepentingan bobot tersebut dalam merepresentasikan emosi. Pada penelitian ini pembobotan diperoleh dari jumlah kemunculan *term* dalam satu dokumen *term frequency* (tf) dan sebuah kata di dalam kumpulan dokumen atau jumlah kemunculan *term* dalam koleksi dokumen *inverse document frequency* (idf). Nilai idf sebuah *term* (kata) dapat dihitung menggunakan persamaan (2.1). Untuk menghitung bobot (W) masing-masing dokumen terhadap setiap *term* (kata) dapat menggunakan persamaan (2.2). Adapun *flowchart* dari proses pembobotan tf-idf dapat dilihat pada Gambar 3.8 berikut:




**Gambar 3.8 *Flowchart* Pembobotan TF-IDF**

Berikut adalah data latih yang sudah melewati tahapan *preprocessing* dapat dilihat pada Tabel 3.8:

**Tabel 3.8 Data Latih Setelah *Preprocessing***

| Pernyataan | Kata |
|---|---|
| P1 | temu gadis minta kencan setuju pikir gadis tidakmenyukai emosi senang |
| P2 | sedih ayah sakit |
| P3 | marah laku tidakadil |
| P4 | takut muncul bentuk cemburu takut pacar jatuh cinta pria takut hilang |
| P5 | salah khianat cinta |

Contoh perhitungan pembobotan TF-IDF akan diterapkan pada kata temu dalam pernyataan pertama (P1).

a. Menghitung *Term Frequency* (tf)

Data hasil *preprocessing* dilakukan perhitungan kemunculan kata temu (*term frequency* (tf)) pada setiap dokumen. Dokumen pada contoh kasus ini adalah P1, P2, P3, P4, dan P5.

**Tabel 3.9 *Term Frequency* Kata Cinta**

| Kata | Term Frequency (tf) | | | | |
|---|---|---|---|---|---|
| | P1 | P2 | P3 | P4 | P5 |
| temu | 1 | 0 | 0 | 0 | 0 |

Dari Tabel 3.9 dijelaskan bahwa kata takterduga pada pernyataan pertama (P1) muncul sebanyak satu kali, sedangkan pada P2, P3, P4, dan P5 kata temu tidak muncul sama sekali (0).

b. Menghitung *Document Frequency* (df)

Nilai *document frequency* (df) didapatkan dari jumlah dokumen yang mengandung kata temu yaitu 1 pada dokumen P1 saja.

c. Menghitung *Inverse Document Frequency* (idf)

Nilai *inverse document frequency* (idf) kata temu didapatkan dengan cara dihitung menggunakan persamaan (2.1):

$$\text{IDF} = \log\left(\frac{\text{D}}{\text{df}}\right) = \log\left(\frac{5}{1}\right) = \log(5) = 0.699$$

D merupakan jumlah dokumen, dimana pada kasus ini jumlah dokumennya ada 5 yaitu P1, P2, P3, P4, dan P5. Kemudian nilai df kata temu sudah didapatkan pada langkah sebelumnya. Jadi nilai idf dari kata temu adalah 0.699.

d. Menghitung W

Menghitung bobot (w) kata temu pada setiap pernyataan dapat dihitung menggunakan persamaan (2.2). Berikut perhitungan w untuk setiap dokumen pernyataan:

$$w_{P1} = tf_{P1} \times idf = 1 \times 0.699 = 0.699$$
$$w_{P2} = tf_{P2} \times idf = 0 \times 0.699 = 0$$
$$w_{P3} = tf_{P3} \times idf = 0 \times 0.699 = 0$$
$$w_{P4} = tf_{P4} \times idf = 0 \times 0.699 = 0$$
$$w_{P5} = tf_{P5} \times idf = 0 \times 0.699 = 0$$

Dari perhitungan w yang telah dilakukan di atas, diperoleh nilai w untuk P1 adalah 0.699 serta w untuk P2, P3, P4, dan P5 adalah 0.

Berikut hasil perhitungan pembobotan TF-IDF keseluruhan kata data latih hasil *preprocessing* dapat dilihat pada Tabel 3.10:

**Tabel 3.10 Hasil Perhitungan Pembobotan TF-IDF Data Latih**

| Kata | tf | | | | | df | idf | w | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P1 | P2 | P3 | P4 | P5 | | | P1 | P2 | P3 | P4 | P5 |
| temu | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| gadis | 2 | 0 | 0 | 0 | 0 | 1 | 0.699 | 1.398 | 0 | 0 | 0 | 0 |
| minta | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| kencan | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| setuju | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| pikir | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| tidakmenyukai | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| emosi | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| senang | 1 | 0 | 0 | 0 | 0 | 1 | 0.699 | 0.699 | 0 | 0 | 0 | 0 |
| sedih | 0 | 1 | 0 | 0 | 0 | 1 | 0.699 | 0 | 0.699 | 0 | 0 | 0 |
| ayah | 0 | 1 | 0 | 0 | 0 | 1 | 0.699 | 0 | 0.699 | 0 | 0 | 0 |
| sakit | 0 | 1 | 0 | 0 | 0 | 1 | 0.699 | 0 | 0.699 | 0 | 0 | 0 |
| marah | 0 | 0 | 1 | 0 | 0 | 1 | 0.699 | 0 | 0 | 0.699 | 0 | 0 |
| laku | 0 | 0 | 1 | 0 | 0 | 1 | 0.699 | 0 | 0 | 0.699 | 0 | 0 |
| tidakadil | 0 | 0 | 1 | 0 | 0 | 1 | 0.699 | 0 | 0 | 0.699 | 0 | 0 |
| takut | 0 | 0 | 0 | 3 | 0 | 1 | 0.699 | 0 | 0 | 0 | 2.097 | 0 |
| muncul | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| bentuk | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| cemburu | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| pacar | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| jatuh | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| cinta | 0 | 0 | 0 | 1 | 1 | 2 | 0.398 | 0 | 0 | 0 | 0.398 | 0.398 |
| pria | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| hilang | 0 | 0 | 0 | 1 | 0 | 1 | 0.699 | 0 | 0 | 0 | 0.699 | 0 |
| salah | 0 | 0 | 0 | 0 | 1 | 1 | 0.699 | 0 | 0 | 0 | 0 | 0.699 |
| khianat | 0 | 0 | 0 | 0 | 1 | 1 | 0.699 | 0 | 0 | 0 | 0 | 0.699 |

## 3.6 Analisis Metode Multi Class Support Vector Machine

Tahap klasifikasi menggunakan metode *Multi Class Support Vector Machine* terbagi menjadi dua yaitu tahap pelatihan dan tahap pengujian. Tahap pelatihan dilakukan untuk menghasilkan model fitur yang nantinya akan digunakan sebagai acuan perhitungan untuk mengklasifikasikan lirik lagu ke dalam lima kelas senang, sedih, marah, takut atau bersalah.

Data masukan akan diubah menjadi data vektor. Format data *input* untuk klasifikasi SVM pada penelitian ini adalah 1 1: 0.699. Masukan pertama +1 atau -1 menyatakan dua label kelas awalan yang diberikan. Angka 1 sebelum tanda ":" menyatakan indeks kata dan angka 0.699 setelah tanda ":" menyatakan bobot tf-idf dari kata tersebut. Sebagai contoh dalam pengubahan data teks dalam sebuah kalimat penyataan menjadi data vektor dapat dilihat pada Tabel 3.11 berikut:

**Tabel 3.11 Pengubahan Data Kalimat Pernyataan Menjadi Data Vektor**

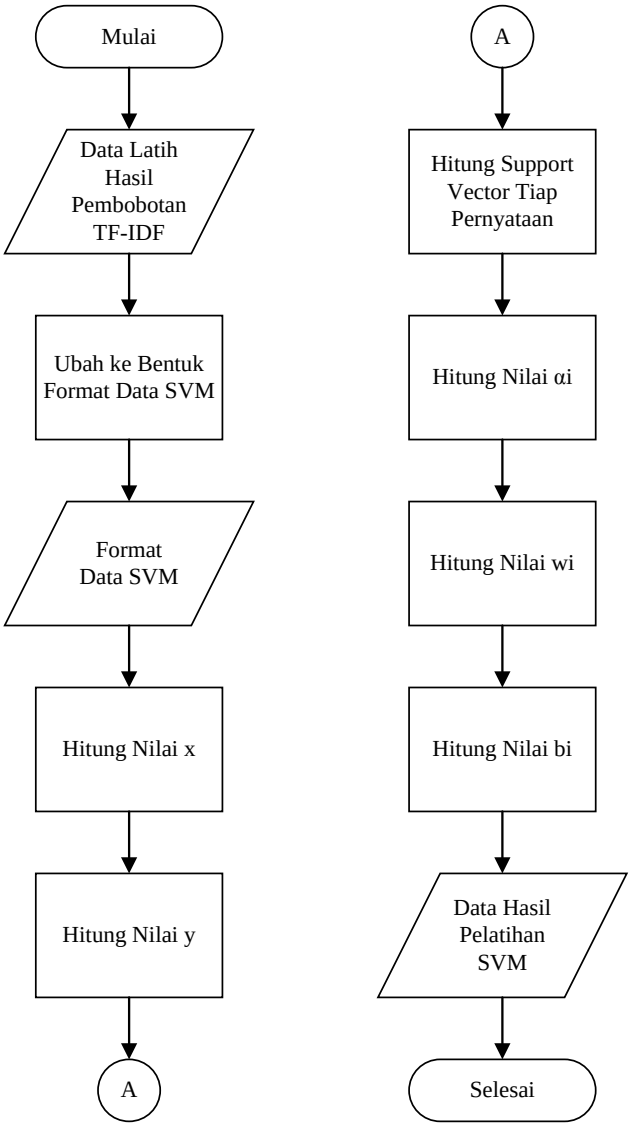
| | |
|---|---|
| **Pernyataan** | temu gadis minta kencan setuju pikir gadis tidakmenyukai emosi senang |
| **Vektor** | [1 1:0.699 2:1.398 3:0.699 4:0.699 5:0.699 6:0.699 7:0.699 8:0.699 9:0.699] |

Begitu seterusnya sehingga semua kata dalam kalimat terwakili oleh format data vektor. Untuk kata yang sama muncul lebih dari sekali dalam sebuah kalimat akan diwakili oleh sebuh data vektor saja dengan nilai tf-idf yang bersesuaian.

### 3.6.1 Analisis Pelatihan Multi Class Support Vector Machine

Pada tahap pelatihan metode *Multi Class Support Vector Machine* ini akan dicari fungsi pemisah *(hyperplane)* yang akan digunakan sebagai model dalam mengklasifikasikan lirik lagu. Pada kasus *Multi Class Support Vector Machine One Against All* jumlah *hyperplane* yang terbentuk yaitu sebanyak *k* atau jumlah kelas. Jadi untuk kasus lima kelas seperti kasus pada penelitian ini, maka akan terbentuk lima *hyperplane*. Pada *hyperplane* pertama, data latih dalam kelas 1 (senang) diberi label +1, dan data dalam kelas 2 (sedih), 3 (marah), 4 (takut), dan 5 (bersalah) diberi label -1. Pada *hyperplane* kedua, data latih dalam kelas 2 (sedih) diberi label +1, dan data dalam kelas 1 (senang), 3 (marah), 4 (takut), dan 5 (bersalah) diberi label -1. Pada *hyperplane* ketiga, data latih dalam kelas 3 (marah) diberi label +1, dan data dalam kelas 1 (senang), 2 (sedih), 4 (takut), dan 5 (bersalah) diberi label -1.

Pada *hyperplane* keempat, data latih dalam kelas 4 (takut) diberi label +1, dan data dalam kelas 1 (senang), 2 (sedih), 3 (marah), dan 5 (bersalah) diberi label -1. Pada *hyperplane* kelima, data latih dalam kelas 5 (bersalah) diberi label +1, dan data dalam kelas 1 (senang), 2 (sedih), 3 (marah), dan 4 (takut) diberi label -1. Sehingga dalam setiap pembentukan *hyperplane*, kelas yang awalnya ada 5 kelas menjadi 2 kelas. Berikut adalah *flowchart* dari tahap pelatihan metode *Multi Class Support Vector Machines One Against All* untuk pembentukan setiap *hyperplane* ditunjukkan pada Gambar 3.9 berikut:




**Gambar 3.9 Tahap Pelatihan *Multi Class Support Vector Machine***

Data latih yang digunakan adalah data latih yang telah melalui tahap *preprocessing* dan pembobotan tf-idf, data latih tersebut dapat dilihat pada Tabel 3.12 berikut:

**Tabel 3.12 Data Tahap Pelatihan Kelas 1**

| Pernyataan | Kata | Label |
|---|---|---|
| P1 | temu gadis minta kencan setuju pikir gadis tidakmenyukai emosi senang | 1 |
| P2 | sedih ayah sakit | -1 |
| P3 | marah laku tidakadil | -1 |
| P4 | takut muncul bentuk cemburu takut pacar jatuh cinta pria takut hilang | -1 |
| P5 | salah khianat cinta | -1 |

Data latih dalam kelas 1 (senang) diberi label +1, dan data dalam kelas 2 (sedih), 3 (marah), 4 (takut), dan 5 (bersalah) diberi label -1. Dari Tabel 3.11 didapat hasil pembobotan tf-idf diketahui bobot setiap kata (*term*) dari semua pernyataan. Selanjutnya data tersebut diubah formatnya ke dalam format data SVM, dapat dilihat pada Tabel 3.13 berikut:

**Tabel 3.13 Format Data SVM**

| Kata | Pernyataan | | | | |
|---|---|---|---|---|---|
| | P1 | P2 | P3 | P4 | P5 |
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
| temu | 0.699 | 0 | 0 | 0 | 0 |
| gadis | 1.398 | 0 | 0 | 0 | 0 |
| minta | 0.699 | 0 | 0 | 0 | 0 |
| kencan | 0.699 | 0 | 0 | 0 | 0 |
| setuju | 0.699 | 0 | 0 | 0 | 0 |
| pikir | 0.699 | 0 | 0 | 0 | 0 |
| tidakmenyukai | 0.699 | 0 | 0 | 0 | 0 |
| emosi | 0.699 | 0 | 0 | 0 | 0 |
| senang | 0.699 | 0 | 0 | 0 | 0 |
| sedih | 0 | 0.699 | 0 | 0 | 0 |
| ayah | 0 | 0.699 | 0 | 0 | 0 |
| sakit | 0 | 0.699 | 0 | 0 | 0 |
| marah | 0 | 0 | 0.699 | 0 | 0 |
| laku | 0 | 0 | 0.699 | 0 | 0 |
| tidakadil | 0 | 0 | 0.699 | 0 | 0 |
| takut | 0 | 0 | 0 | 2.097 | 0 |
| muncul | 0 | 0 | 0 | 0.699 | 0 |
| bentuk | 0 | 0 | 0 | 0.699 | 0 |
| cemburu | 0 | 0 | 0 | 0.699 | 0 |
| pacar | 0 | 0 | 0 | 0.699 | 0 |
| jatuh | 0 | 0 | 0 | 0.699 | 0 |
| cinta | 0 | 0 | 0 | 0.398 | 0.398 |
| pria | 0 | 0 | 0 | 0.699 | 0 |
| hilang | 0 | 0 | 0 | 0.699 | 0 |
| salah | 0 | 0 | 0 | 0 | 0.699 |
| khianat | 0 | 0 | 0 | 0 | 0.699 |

Untuk label dalam setiap pernyataan dapat dilihat pada Tabel 3.14 berikut:

**Tabel 3.14 Label Data SVM**

| **Pernyataan** | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| **Label(y)** | 1 | -1 | -1 | -1 | -1 |

Kemudian tahap selanjutnya adalah melakukan perhitungan kernel $x_i x_j^T$ dimana $i, j = 1, .., n$. Untuk $x_1$ adalah seluruh nilai yang diambil dari kolom P1, $x_2$ = P2 sampai $x_5$ = P5. Sedemikian sehingga setiap penyataan untuk nilai $x_1$ dapat diubah sesuai hasil pembobotan tf-idf menjadi bentuk sebagai berikut, sebagai contoh bentuk pada pernyataan pertama:

$x_1$ = [0.699 1.398 0.699 0.699 0.699 0.699 0.699 0.699 0.699 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

Kemudian data vektor setiap pernyataan sesuai dengan hasil pembobotan TF-IDF pada setiap pernyataannya. Dilakukan perhitungan $x_i x_j^T$ pada setiap pernyataan, sebagai contoh dilakukan perhitungan pada pernyataan pertama sebagai berikut:

$x_1 x_1^T$ = [0.699 1.398 0.699 0.699 0.699 0.699 0.699 0.699 0.699 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [0.699 1.398 0.699 0.699 0.699 0.699 0.699 0.699 0.699 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] $^T$

= 0.489 + 1.954 + 0.489 + 0.489 + 0.489 + 0.489 + 0.489 + 0.489 + 0.489

= 5.863

Setelah dilakukan perhitungan pada seluruh pernyataan, maka matriks yang terbentuk dari hasil perhitungan $x_i x_j^T$ adalah sebagai berikut:

$$x_1 x_1^T = \begin{bmatrix} 5.863 & 0 & 0 & 0 & 0 \\ 0 & 1.466 & 0 & 0 & 0 \\ 0 & 0 & 1.466 & 0 & 0 \\ 0 & 0 & 0 & 7.796 & 0.158 \\ 0 & 0 & 0 & 0.158 & 1.136 \end{bmatrix}$$

Kemudian tahap selanjutnya adalah melakukan perhitungan $y_i y_j^T$ dimana $i, j = 1, .., n$. Untuk nilai $y$ adalah nilai dari label yang diberikan. Matriks yang terbentuk dari hasil perhitungan $y_i y_j^T$ adalah hasilnya sebagai berikut:

$$y_1 y_1^T = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Kemudian tahap selanjutnya adalah mencari nilai $a_i$, didapatkan dengan menggunakan persamaan (2.23). Proses mendapatkan nilai $a_i$ diawali dengan mengubah setiap pernyataan menjadi nilai vektor (*support vector*) = $\binom{x}{y}$. Nilai x didapatkan menggunakan persamaan (2.21) sehingga didapatkan untuk nilai x pada setiap pernyataan pada Tabel 3.15 berikut:

**Tabel 3.15 Nilai x Setiap Pernyataan**

| **Pernyataan** | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| **X** | 5.863 | 1.466 | 1.466 | 8.134 | 1.294 |

Nilai y didapatkan menggunakan persamaan (2.22), sehingga didapatkan untuk nilai x pada setiap pernyataan pada Tabel 3.15 berikut:

**Tabel 3.16 Nilai y Setiap Pernyataan**

| **Pernyataan** | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| **Y** | -3 | 3 | 3 | 3 | 3 |

Setelah nilai x dan y didapatkan, substitusikan nilai tersebut ke persamaan (2.20), sebagai contoh perhitungan dilakukan pada pernyataan pertama (P1), proses perhitungan sebagai berikut:

$$\sqrt{(5.863)^2+(-3)^2} = \sqrt{34.377 + 9} = \sqrt{43.377} = 6.586 > 2$$

$$\varphi_1 \binom{x}{y} = \binom{4 - y + |x - y|}{4 - x + |x - y|} = \binom{4 - (-3) + |5.863 - (-3)|}{4 - 5.863 + |5.863 - (-3)|} = \binom{7 + 8.863}{-1.863 + 8.863} = \binom{15.863}{7}$$

Setelah dilakukan perhitungan terhadap seluruh pernyataan, maka didapatkan hasilnya pada Tabel 3.17 sebagai berikut:

**Tabel 3.17 *Support Vector* Setiap Pernyataan**

| **Pernyataan** | **P1** | **P2** | **P3** | **P4** | **P5** |
|---|---|---|---|---|---|
| **Support Vector** | $\begin{bmatrix} 15.863 \\ 7 \end{bmatrix}$ | $\begin{bmatrix} -0.534 \\ 1 \end{bmatrix}$ | $\begin{bmatrix} -0.534 \\ 1 \end{bmatrix}$ | $\begin{bmatrix} 6.134 \\ 1 \end{bmatrix}$ | $\begin{bmatrix} -0.706 \\ 1 \end{bmatrix}$ |

Setelah itu masing-masing *support vector* diberi nilai bias 1. Untuk mendapatkan jarak tegak lurus yang optimal dengan mempertimbangkan vektor positif, serta

membantu mendapatkan nilai b atau nilai *hyperplane* [25], hasilnya dapat dilihat pada Tabel 3.18 berikut:

**Tabel 3.18 *Support Vector* Bias**

| Pernyataan | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| **Support Vector Bias** | $\begin{bmatrix}15.863\\7\\1\end{bmatrix}$ | $\begin{bmatrix}-0.534\\1\\1\end{bmatrix}$ | $\begin{bmatrix}-0.534\\1\\1\end{bmatrix}$ | $\begin{bmatrix}6.134\\1\\1\end{bmatrix}$ | $\begin{bmatrix}-0.706\\1\\1\end{bmatrix}$ |

Kemudian kalikan setiap kalimat menggunakan persamaan (2.23), sebagai contoh dilakukan perhitungan pada pernyataan pertama (P1) berikut:

$$a_1\begin{bmatrix}15.863\\7\\1\end{bmatrix}^T * \begin{bmatrix}15.863\\7\\1\end{bmatrix} = 301.642a_1$$

Setelah dilakukan perhitungan pada seluruh pernyataan. Kemudian cari parameter $a_i$ menggunakan persamaan (2.24), dengan cara substitusikan nilai hasil dari perhitungan menggunakan persamaan (2.23), sehingga bentuknya dapat dilihat sebagai berikut:

$$301.641a_1 - 0.474a_2 - 0.474a_3 + 105.312a_4 - 3.199a_5 = 1$$
$$-0.474a_1 + 2.285a_2 + 2.285a_3 - 1.277a_4 + 2.377a_5 = -1$$
$$-0.474a_1 + 2.285a_2 + 2.285a_3 - 1.277a_4 + 2.377a_5 = -1$$
$$105.312a_1 - 1.277a_2 - 1.277a_3 + 39.631a_4 - 2.331a_5 = -1$$
$$-3.199a_1 + 2.377a_2 + 2.377a_3 - 2.331a_4 + 2.498a_5 = -1$$

Sedemikian sehingga didapatlah nilai untuk $a_1$ sampai $a_5$ berikut:

$a_1$ = 0.278, $a_2$ = 0.001, $a_3$ = -0.810, $a_4$ = -0.790, $a_5$ = -0.011

Setelah didapatkan nilai $a_i$ masukkan ke persamaan (2.24) sesuai persamaan (2.25) seperti berikut:

$$\widetilde{W} = 0.278\begin{bmatrix}15.863\\7\\1\end{bmatrix} + 0.001\begin{bmatrix}-0.534\\1\\1\end{bmatrix} - 0.810\begin{bmatrix}-0.534\\1\\1\end{bmatrix} - 0.790\begin{bmatrix}6.134\\1\\1\end{bmatrix} - 0.011\begin{bmatrix}-0.706\\1\\1\end{bmatrix}$$

$$= \begin{bmatrix}4.406\\1.944\\0.278\end{bmatrix} + \begin{bmatrix}-0.001\\0.001\\0.001\end{bmatrix} - \begin{bmatrix}-0.433\\0.810\\0.810\end{bmatrix} - \begin{bmatrix}4.847\\0.790\\0.790\end{bmatrix} - \begin{bmatrix}-0.008\\0.011\\0.011\end{bmatrix}$$

$$= \begin{bmatrix}-0.001\\0.334\\-1.332\end{bmatrix}$$

Kemudian hasil yang didapatkan melalui perhitungan dengan menggunakan persamaan (2.25), digunakan persamaan (2.26) dengan hasil yang didapatkan:

$w^1 = \begin{bmatrix} -0.001 \\ 0.334 \end{bmatrix}$ dan $b^1 = -1.332$

Langkah-langkah untuk menemukan *hyperplane* kedua hingga kelima sama saja seperti menentukan *hyperplane* pertama. Berikut ini merupakan nilai *hyperplane* untuk kelas kedua hingga kelima:

$w^2 = \begin{bmatrix} 0.011 \\ 0.340 \end{bmatrix}$ dan $b^2 = -1.332$

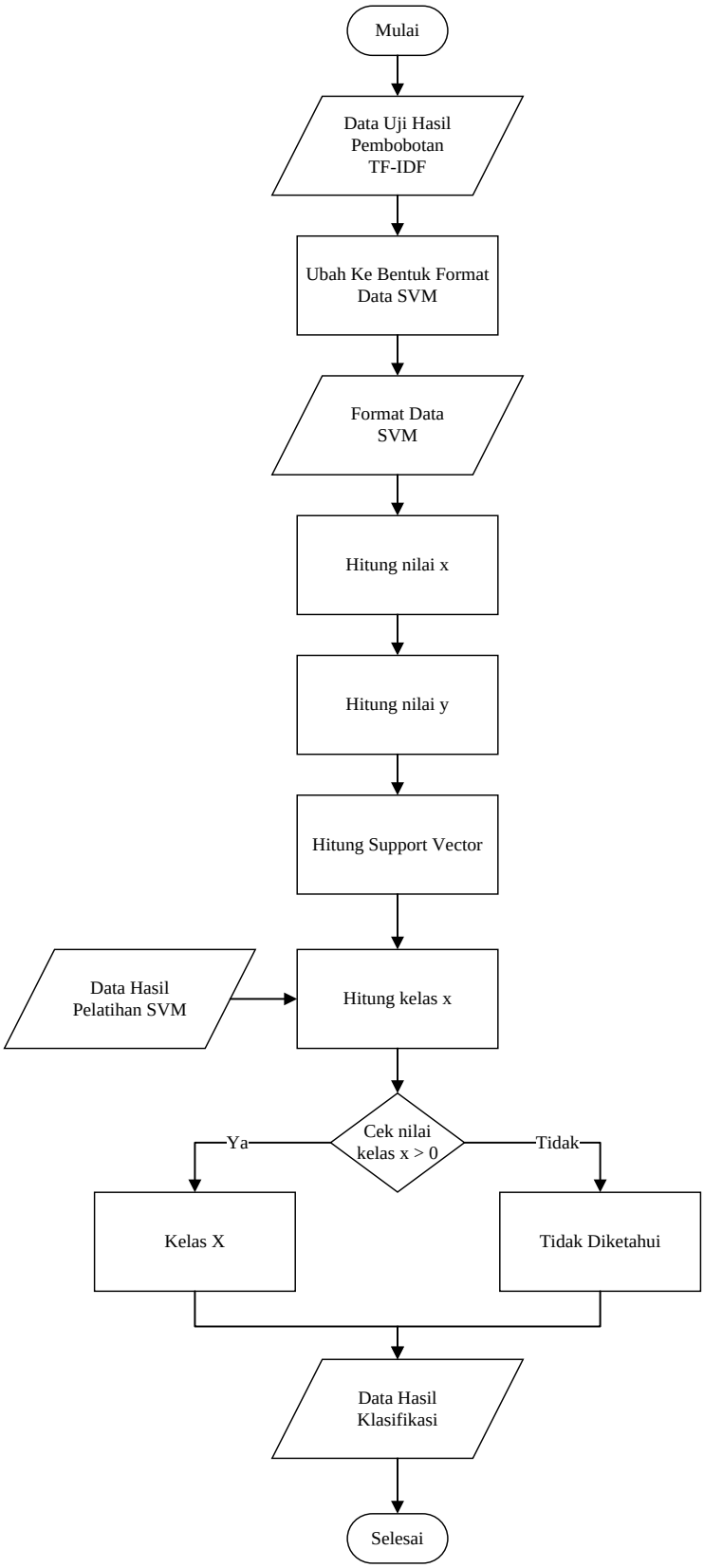
$w^3 = \begin{bmatrix} 0.001 \\ 0.332 \end{bmatrix}$ dan $b^3 = -1.334$

$w^4 = \begin{bmatrix} -0.001 \\ -0.334 \end{bmatrix}$ dan $b^4 = -1.332$

$w^5 = \begin{bmatrix} -0.001 \\ 0.334 \end{bmatrix}$ dan $b^5 = -1.332$

### 3.6.2 Analisis Pengujian Multi Class Support Vector Machine

Setelah mendapatkan nilai *hyperplane* pertama hingga kelima, selanjutnya dapat menentukan data uji masuk ke dalam kategori senang, sedih, marah, takut atau bersalah dengan cara menghitung nilai *hyperplane* paling besar. Kelas i dengan hasil perhitungan *hyperplane* terbesar adalah kelas data dari data uji. Adapun *flowchart* klasifikasi data uji menggunakan metode *Multi Class Support Vector Machine* pada tahap pengujian ditunjukkan pada Gambar 3.10 berikut:




**Gambar 3.10 Flowchart Pengujian *Multi Class Support Vector Machine***

Sebagai data uji dilakukan pada lirik lagu (L1) dengan *support vector* (130.264,4). Nilai vektor tersebut disubsitusikan ke dalam persamaan (2.27) berikut:

$$kelas\ x = \arg\max_{k=1\ldots5}([w^1]^T.\varphi(x) + b^1, [w^2]^T.\varphi(x) + b^2, [w^3]^T.\varphi(x) + b^3, [w^4]^T.\varphi(x) + b^4, [w^5]^T.\varphi(x) + b^5)$$

Nilai $w^1$ dan $b^1$ didapatkan dari hasil pelatihan yang telah dilakukan sebelumnya dimana angka 1 menunjukkan indeks kelas 1 yaitu senang, angka 2 menunjukkan indeks kelas 2 yaitu sedih, angka 3 menunjukkan indeks kelas 3 yaitu marah angka 4 menunjukkan indeks kelas 4 yaitu takut, dan angka 5 menunjukkan indeks kelas 5 yaitu bersalah.

$$kelas\ x = \arg\max_{k=1\ldots5}\left(\begin{bmatrix}-0.001\\0.334\end{bmatrix}^T.\begin{bmatrix}130.264\\4\end{bmatrix} - 1.332, \begin{bmatrix}0.011\\0.340\end{bmatrix}^T.\begin{bmatrix}130.264\\4\end{bmatrix} - 1.332, \begin{bmatrix}0.001\\0.332\end{bmatrix}^T.\begin{bmatrix}130.264\\4\end{bmatrix} - 1.334, \begin{bmatrix}-0.001\\-0.334\end{bmatrix}^T.\begin{bmatrix}130.264\\4\end{bmatrix} - 1.332, \begin{bmatrix}-0.001\\0.334\end{bmatrix}^T.\begin{bmatrix}130.264\\4\end{bmatrix} - 1.332\right)$$

$$kelas\ x = \arg\max_{k=1\ldots5}(1.268 - 1.332, 2.855 - 1.332, 1.399 - 1.334, 1.268 - 1.332, 1.268 - 1.332)$$

$$kelas\ x = \arg\max_{k=1\ldots5}(-0.065, 1.523, 0.065, -0.065, -0.065)$$

$$kelas\ x = 1.523$$

Nilai *hyperplane* terbesar adalah 1.523 dimana nilai *hyperplane* tersebut merupakan nilai *hyperplane* kelas 2. Berarti L1 termasuk lirik lagu dengan kategori sedih.

### 3.7 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional yang dilakukan dibagi menjadi tiga tahap, yaitu analisis kebutuhan perangkat keras, analisis kebutuhan perangkat lunak, dan analisis pengguna.

### 3.7.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras pada penelitian ini merupakan kebutuhan perangkat keras yang digunakan dalam pembangunan perangkat lunak ini. Adapun perangkat keras yang digunakan sebagai berikut:

a. *Processor* 2.50 GHz
b. RAM 4.00 GB
c. Layar dengan resolusi 1366 x 768 piksel
d. *Keyboard*
e. *Mouse*

### 3.7.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak pada penelitian ini merupakan *tools* yang digunakan dalam pembangunan perangkat lunak ini. *Tools* yang digunakan adalah sebagai berikut:

a. Sistem Operasi Microsoft Windows 8 *Single Language*
b. Microsoft Visual Studio 2012

### 3.7.3 Analisis Pengguna

Karakteristik pengguna yang dapat menjalankan perangkat lunak yang akan dibangun hanya terdapat satu jenis pengguna yaitu seorang penguji. Penguji bekerja untuk menjalankan serta mengetahui hasil dari perangkat lunak yang dijalankan. Adapun spesifikasi pengguna yang dibutuhkan adalah sebagai berikut:

1. Menguasai penggunaan komputer.
2. Mengerti secara teknis *tools* dan perangkat lunak pendukung dalam menjalankan perangkat lunak yang telah dibangun.
3. Mengerti tahapan-tahapan dalam menjalankan perangkat lunak yang telah dibangun.
4. Memahami proses dan kebutuhan dalam menggunakan perangkat lunak yang telah dibangun.

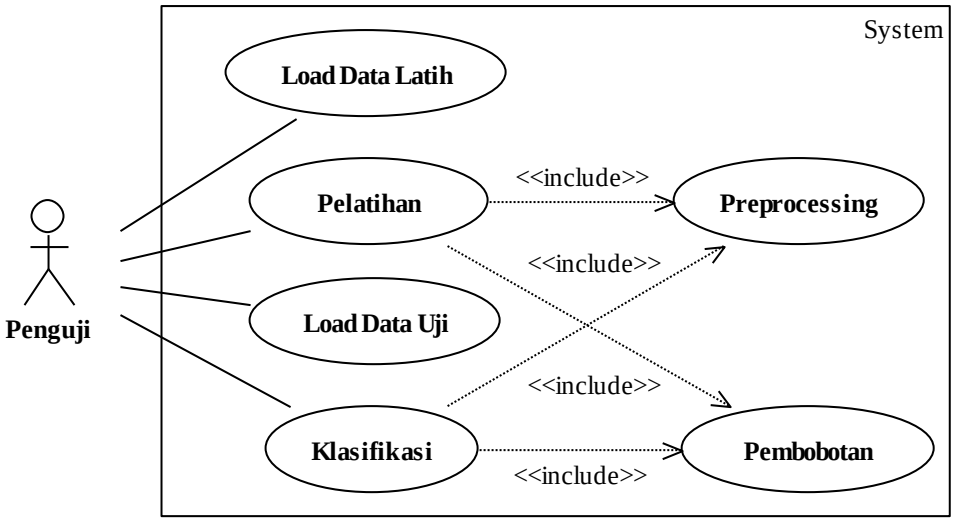
## 3.8 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk menganalisis proses yang diterapkan dalam perangkat lunak yang dibangun dan menjelaskan kebutuhan yang

diperlukan. Perangkat lunak yang dibangun dimodelkan menggunakan UML (*Unified Modelling Language*). Bagian-bagian yang dilakukan dalam pemodelan tersebut antara lain sebagai berikut:

### 3.8.1 Use Case Diagram

*Use case diagram* adalah gambaran umum dari perancangan sistem yang dibuat untuk membantu pengguna dalam mendapatkan pemahaman yang utuh tentang sistem yang dibangun. *Use case diagram* pada penelitian ini terdiri dari satu aktor dan 6 *use case*. Berikut ini adalah *use case diagram* dari sistem yang akan dibuat dapat dilihat pada Gambar 3.11 berikut:




**Gambar 3.11 Use Case Diagram**

Berikut ini merupakan deskripsi aktor yang ditunjukkan pada Tabel 3.19:

**Tabel 3.19 Deskripsi Aktor**

| Aktor | Deskripsi |
|---|---|
| Penguji | Penguji dapat melakukan load data latih, load data uji, pelatihan, dan klasifikasi. |

Berikut ini merupakan deskripsi *use case* yang ditunjukkan pada Tabel 3.20:

**Tabel 3.20 Deskripsi *Use Case***

| No | *Use Case* | Deskripsi |
|---|---|---|
| 1 | Load Data Latih | Fungsionalitas ini digunakan penguji untuk load data latih. |
| 2 | Load Data Uji | Fungsionalitas ini digunakan penguji untuk load data uji. |
| 3 | *Preprocessing* | Fungsionalitas ini digunakan untuk memecahkan data dokumen sehingga layak untuk diproses pada tahap selanjutnya. |
| 4 | Pembobotan | Fungsionalitas ini digunakan untuk memberi nilai bobot pada tiap *term* (kata). |
| 5 | Pelatihan | Fungsionalitas ini digunakan penguji untuk mendapatkan model fitur dari data latih. |
| 6 | Klasifikasi | Fungsionalitas ini digunakan penguji untuk mengklasifikasikan data uji ke dalam kategori senang, sedih, marah, takut atau bersalah. |

### 3.8.2 Use Case Skenario

Pada bagian ini akan dijelaskan skenario untuk setiap *use case* yang menggambarkan urutan interaksi aktor dengan *use case* tersebut dari awal sampai akhir. Berikut *use case scenario* dari masing-masing *use case*:

**Tabel 3.21 *Use Case* Skenario Load Data Latih**

| **Nama *Use Case*** | Load Data Latih |
|---|---|
| **Tujuan** | *Load* data latih. |
| **Deskripsi** | Fungsionalitas ini digunakan penguji untuk load data berupa folder data latih. |
| **Aktor** | Penguji |
| ***Related Case*** | - |
| **Kondisi Sukses** | Data latih berhasil di*load*. |
| **Kondisi Gagal** | Data latih gagal di*load*. |
| ***Include Case*** | - |
| ***Trigger*** | Memilih menu *load* data latih |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu load data latih. | |
| | 2. Menampilkan halaman *file directory*. |
| 3. Memilih folder data latih. | |
| | 4. Menampilkan pesan data latih berhasil di*load*. |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu load data latih. | |
| | 2. Gagal menampilkan pesan data latih berhasil di*load*. |

**Tabel 3.22 *Use Case* Skenario Load Data Uji**

| **Nama *Use Case*** | Load Data Uji |
|---|---|
| **Tujuan** | *Load* data uji |
| **Deskripsi** | Fungsionalitas ini digunakan penguji untuk *load* data berupa dokumen data uji. |
| **Aktor** | Penguji |
| ***Related Case*** | - |
| **Kondisi Sukses** | Data uji berhasil di*load*. |
| **Kondisi Gagal** | Data uji gagal di*load*. |
| ***Include Case*** | - |
| ***Trigger*** | Memilih menu load data uji. |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu load data uji. | |
| | 2. Menampilkan halaman *file directory*. |
| 3. Memilih dokumen data uji. | |
| | 4. Menampilkan pesan data uji berhasil di*load.* |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu load data uji. | |
| | *2.* Gagal menampilkan pesan data uji berhasil di*load*. |

**Tabel 3.23 *Use Case* Skenario Preprocessing**

| **Nama *Use Case*** | *Preprocessing* |
|---|---|
| **Tujuan** | Preproses data. |
| **Deskripsi** | Fungsionalitas ini digunakan untuk memecahkan data dokumen, sehingga layak untuk diproses selanjutnya. |
| **Aktor** | - |
| ***Related Case*** | Pelatihan, Pengujian |
| **Kondisi Sukses** | Data berhasil dipreproses. |
| **Kondisi Gagal** | Data gagal dipreproses. |
| ***Include Case*** | - |
| ***Trigger*** | Terdapat hasil *load* data. |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| | 1. Melakukan proses *case folding*. |
| | 2. Melakukan proses *convert negation*. |
| | 3. Melakukan proses *tokenizing*. |
| | 4. Melakukan proses *stopword removal*. |
| | 5. Data berhasil dipreproses. |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| | 1. Data gagal dipreproses. |

**Tabel 3.24 *Use Case* Skenario Pembobotan**

| **Nama *Use Case*** | Pembobotan |
|---|---|
| **Tujuan** | Memberi nilai bobot pada tiap kata. |
| **Deskripsi** | Fungsionalitas ini digunakan untuk memberi nilai bobot pada tiap *term* (kata). |
| **Aktor** | - |
| ***Related Case*** | Pelatihan, Pengujian |
| **Kondisi Sukses** | Data hasil *preprocessing* berhasil diberi nilai bobot. |
| **Kondisi Gagal** | Data hasil *preprocessing* gagal diberi nilai bobot. |
| ***Include Case*** | |
| ***Trigger*** | Terdapat data hasil proses *preprocessing*. |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| | 1. Menghitung *term frequency*. |
| | 2. Menghitung *document frequency*. |
| | 3. Menghitung *inverse document frequency*. |
| | 4. Menghitung bobot. |
| | 5. Data hasil *preprocessing* berhasil diberi nilai bobot |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| | 1. Data hasil *preprocessing* gagal diberi nilai bobot. |

**Tabel 3.25 *Use Case* Skenario Pelatihan**

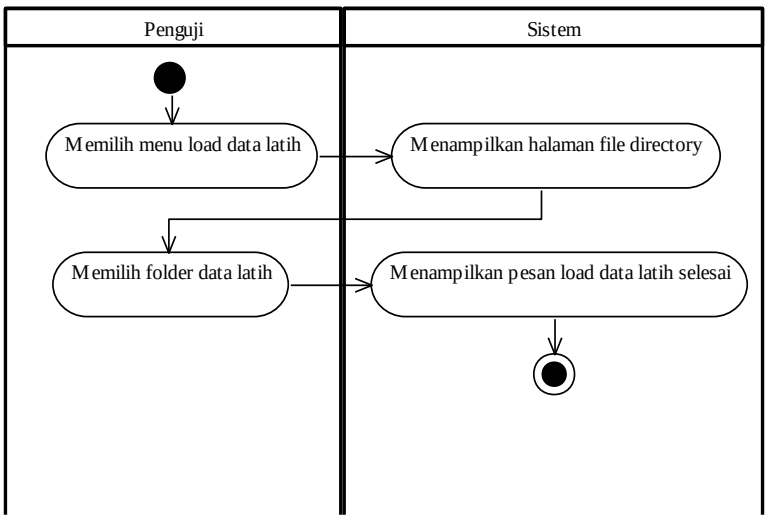
| **Nama *Use Case*** | Pelatihan |
|---|---|
| **Tujuan** | Membentuk model fitur. |
| **Deskripsi** | Fungsionalitas ini digunakan penguji untuk membentuk model fitur. |
| **Aktor** | Penguji |
| ***Related Case*** | - |
| **Kondisi Sukses** | Menampilkan hasil pelatihan berupa model fitur. |
| **Kondisi Gagal** | Gagal menampilkan hasil pelatihan berupa model fitur. |
| ***Include Case*** | *Preprocessing*, Pembobotan |
| ***Trigger*** | Terdapat data hasil pembobotan data latih. |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu pelatihan. | |
| | 2. Melakukan proses *preprocessing* data latih. |
| | 3. Melakukan proses pembobotan data latih. |
| | 4. Melakukan proses pembentukan model fitur. |
| | 5. Menampilkan hasil pelatihan berupa model fitur. |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu pelatihan | |
| | 2. Gagal menampilkan hasil pelatihan berupa model fitur. |

**Tabel 3.26 *Use Case* Skenario Klasifikasi**

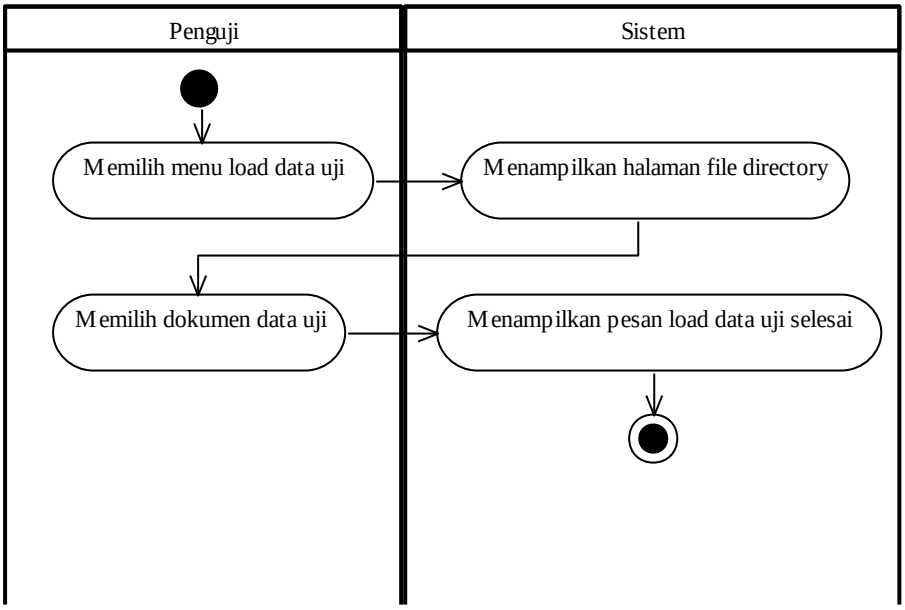
| **Nama *Use Case*** | Klasifikasi |
|---|---|
| **Tujuan** | Mengklasifikasikan data uji ke dalam kategori senang, sedih, marah, takut atau bersalah. |
| **Deskripsi** | Fungsionalitas ini digunakan penguji untuk mengklasifikasikan data uji ke dalam kategori senang, sedih, marah, takut atau bersalah. |
| **Aktor** | Penguji |
| ***Related Case*** | - |
| **Kondisi Sukses** | Menampilkan hasil kategori dari data uji. |
| **Kondisi Gagal** | Gagal menampilkan hasil kategori dari data uji. |
| ***Include Case*** | *Preprocessing*, Pembobotan |
| ***Trigger*** | Terdapat data hasil pembobotan data uji. |
| **Skenario Utama** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu klasifikasi. | |
| | 2. Melakukan proses *preprocessing* data uji. |
| | 3. Melakukan proses pembobotan data uji. |
| | 4. Melakukan proses klasifikasi data uji. |
| | 5. Menampilkan hasil kategori dari data uji. |
| **Skenario Alternatif** | |
| **Aksi aktor** | **Reaksi Sistem** |
| 1. Memilih menu klasifikasi. | |
| | 2. Gagal menampilkan hasil kategori dari data uji. |

### 3.8.3 Activity Diagram

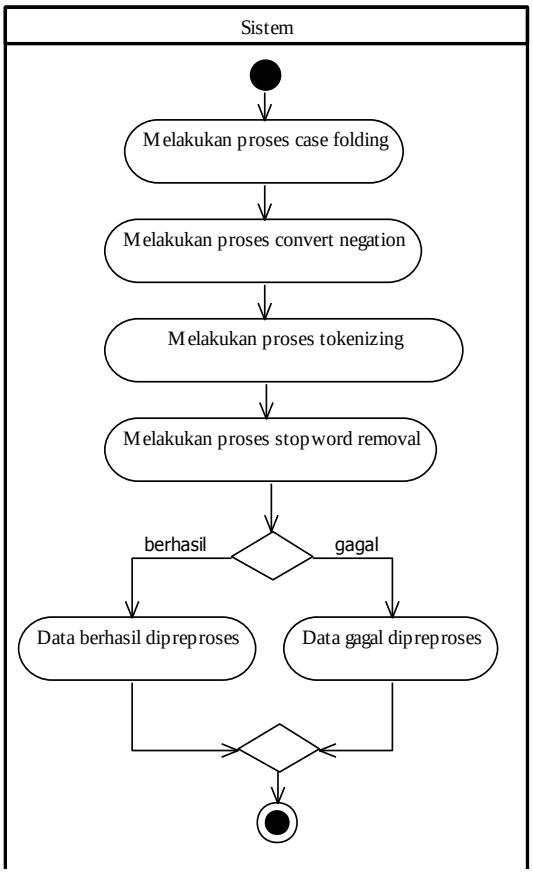
*Activity diagram* pada penelitian ini menjelaskan tentang alur kerja tahapan-tahapan aktivitas dari *use case* yang akan dibangun. Berikut merupakan *activity diagram* dari masing-masing *use case* skenario:



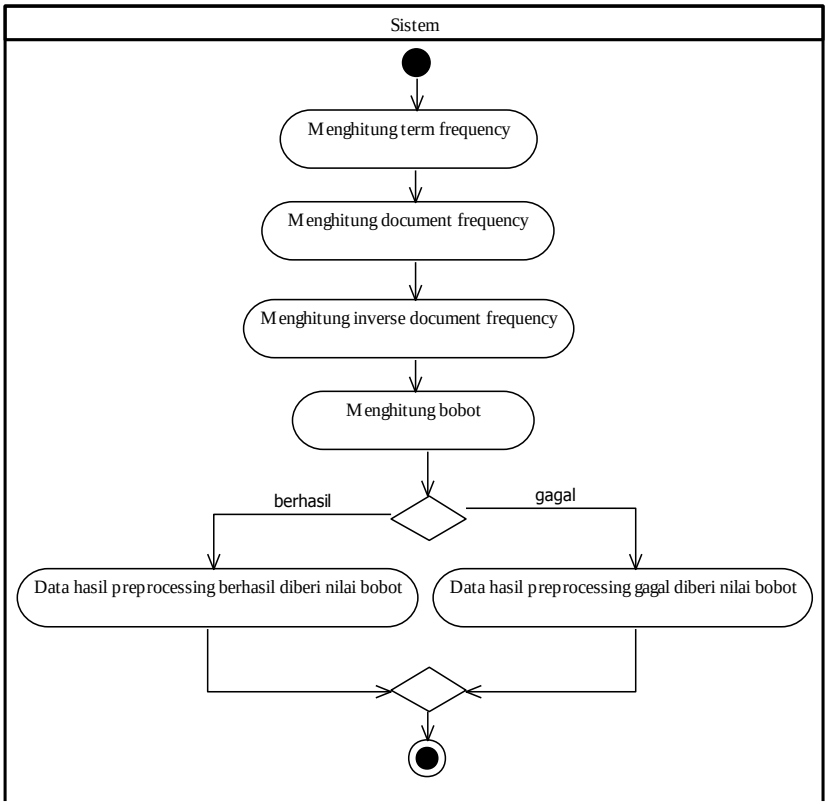

**Gambar 3.12 *Activity Diagram Load* Data Latih**



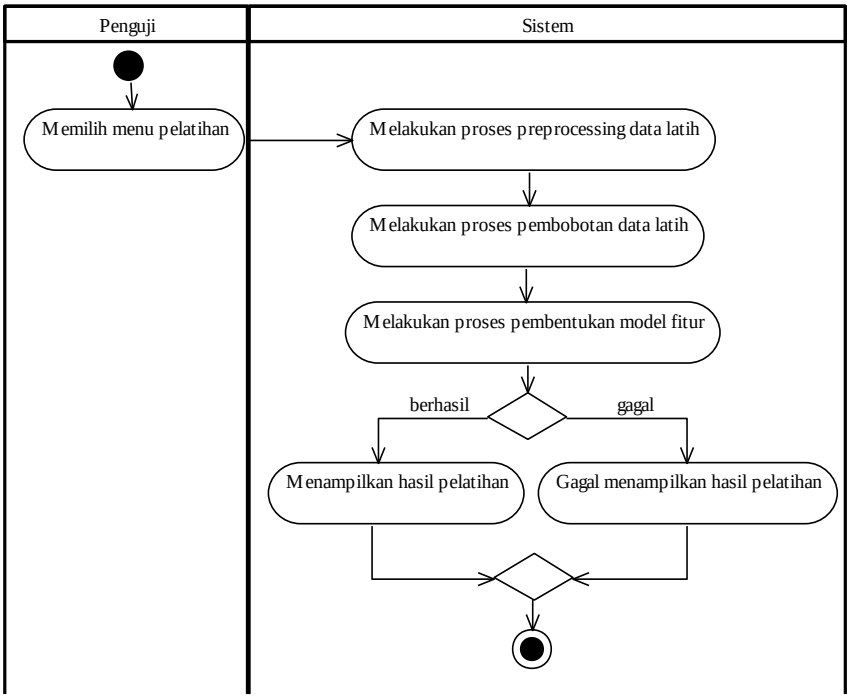

**Gambar 3.13 *Activity Diagram Load* Data Uji**



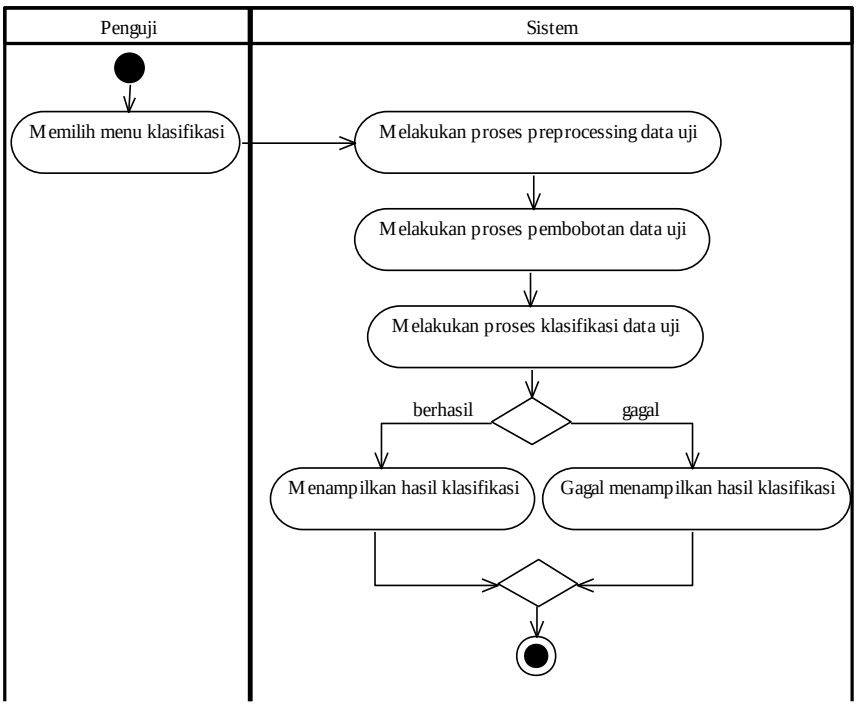

**Gambar 3.14 *Activity Diagram Preprocessing***




**Gambar 3.15 *Activity Diagram* Pembobotan**



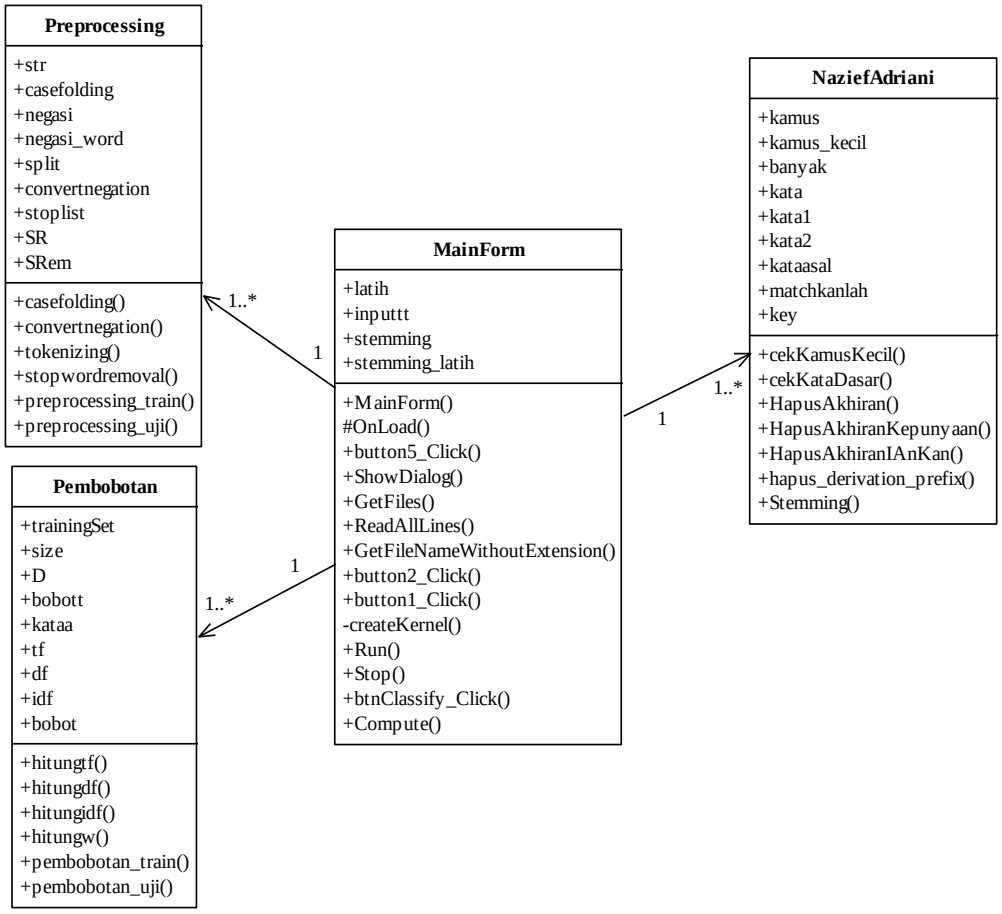

**Gambar 3.16 *Activity Diagram* Pelatihan**




**Gambar 3.17 *Activity Diagram* Klasifikasi**

### 3.8.4 Class Diagram

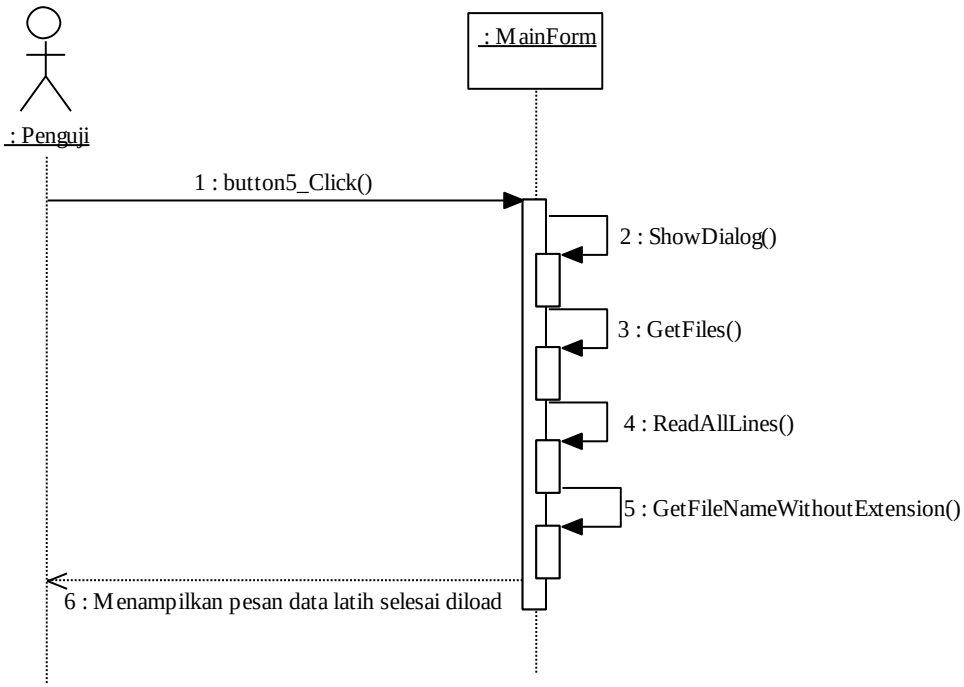
*Class Diagram* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari desain berorientasi objek. *Class diagram* yang terdapat pada penelitian ini dapat dilihat pada Gambar 3.18 berikut:



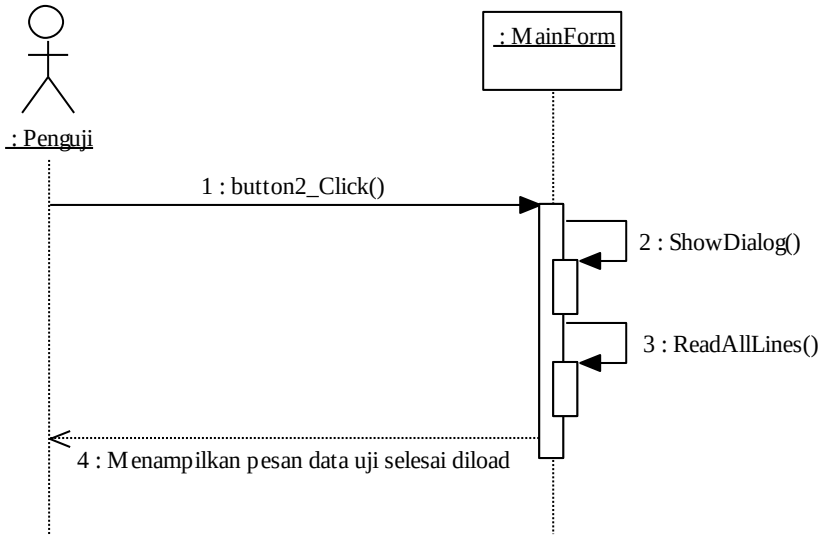

**Gambar 3.18 *Class Diagram***

### 3.8.5 Sequence Diagra*m*

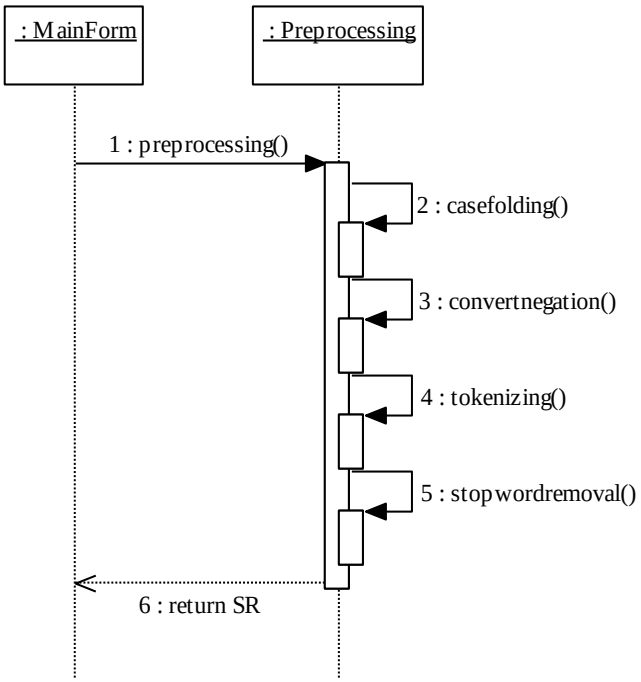
*Sequence diagram* menggambarkan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Berikut adalah *sequence diagram* pada penelitian ini:



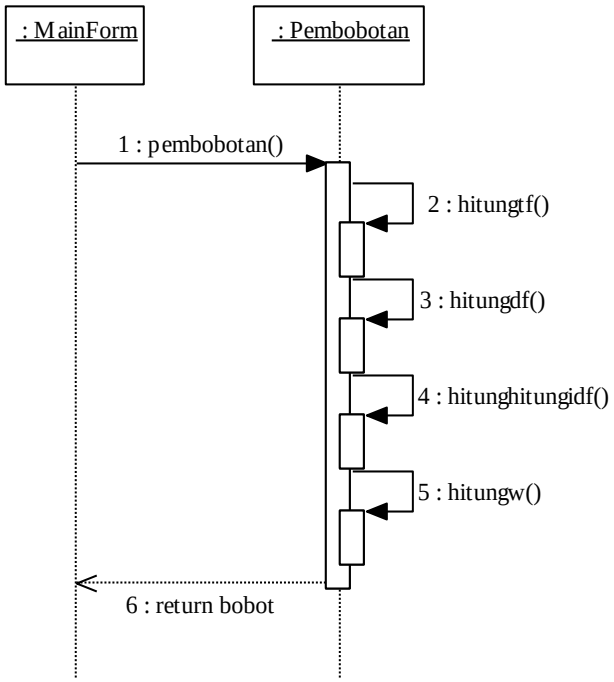

**Gambar 3.19 *Sequence Diagram Load* Data Latih**



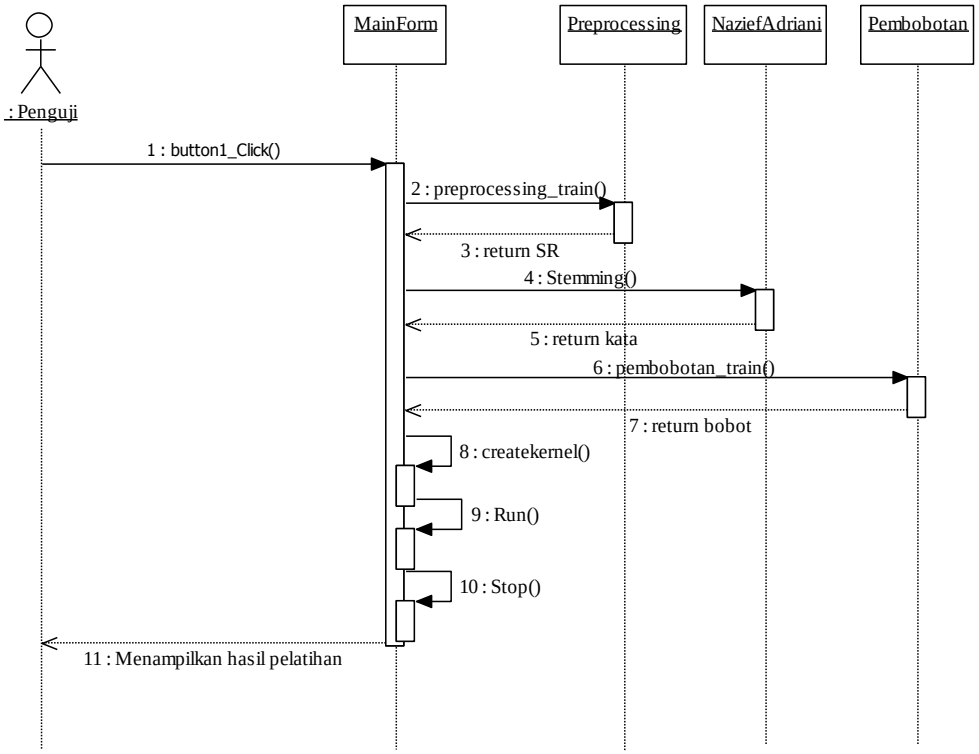

**Gambar 3.20 *Sequence Diagram Load* Data Uji**



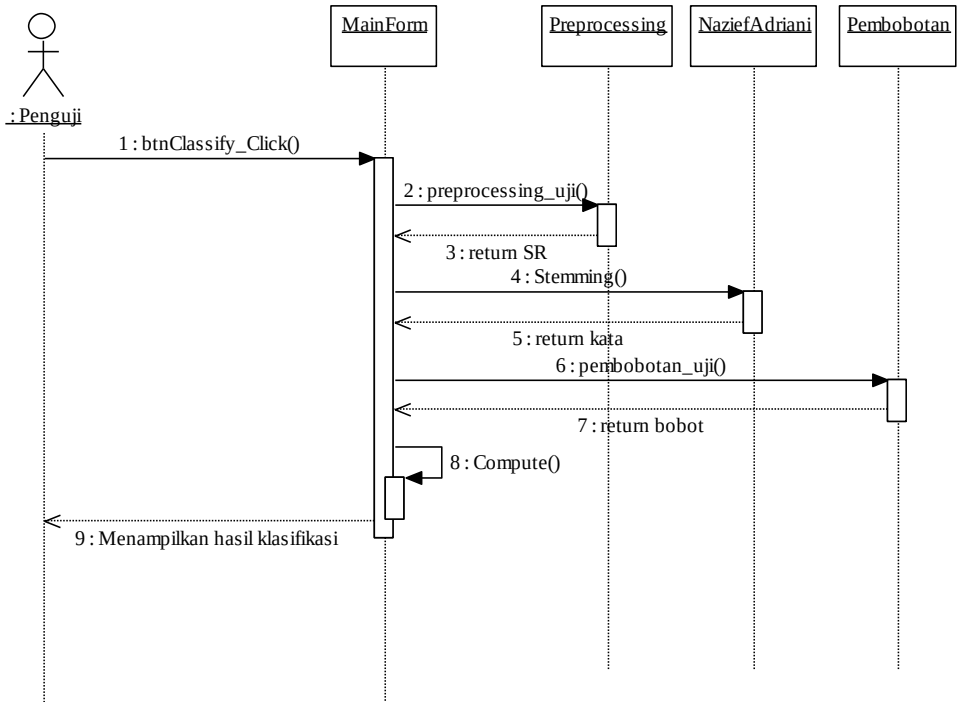

**Gambar 3.21** ***Sequence Diagram Preprocessing***




**Gambar 3.22** ***Sequence Diagra*****m Pembobotan**




**Gambar 3.23 *Sequence Diagram* Pelatihan**



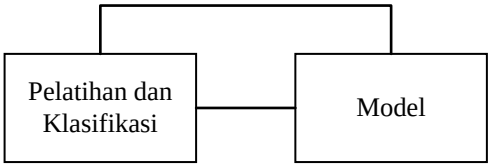

**Gambar 3.24 *Sequence Diagram* Klasifikasi**

## 3.9 Perancangan Sistem

Pada bagian ini akan dijelaskan perancangan sistem untuk mengimplementasikan hasil analisis sebelumnya, dimana sistem yang akan dibuat akan digunakan untuk mensimulasikan metode yang digunakan. Perancangan sistem yang dibuat terdiri dari perancangan struktur menu, perancangan antarmuka, perancangan pesan dan jaringan semantik.

### 3.9.1 Perancangan Struktur Menu

Perancangan struktur menu menjelaskan gambar jalur pemakaian perangkat lunak sehingga perangkat lunak yang dibangun mudah dipahami dan digunakan. Adapun perancangan struktur menu yang akan diterapkan pada perangkat lunak yang dibangun dapat dilihat pada Gambar 3.25 berikut:

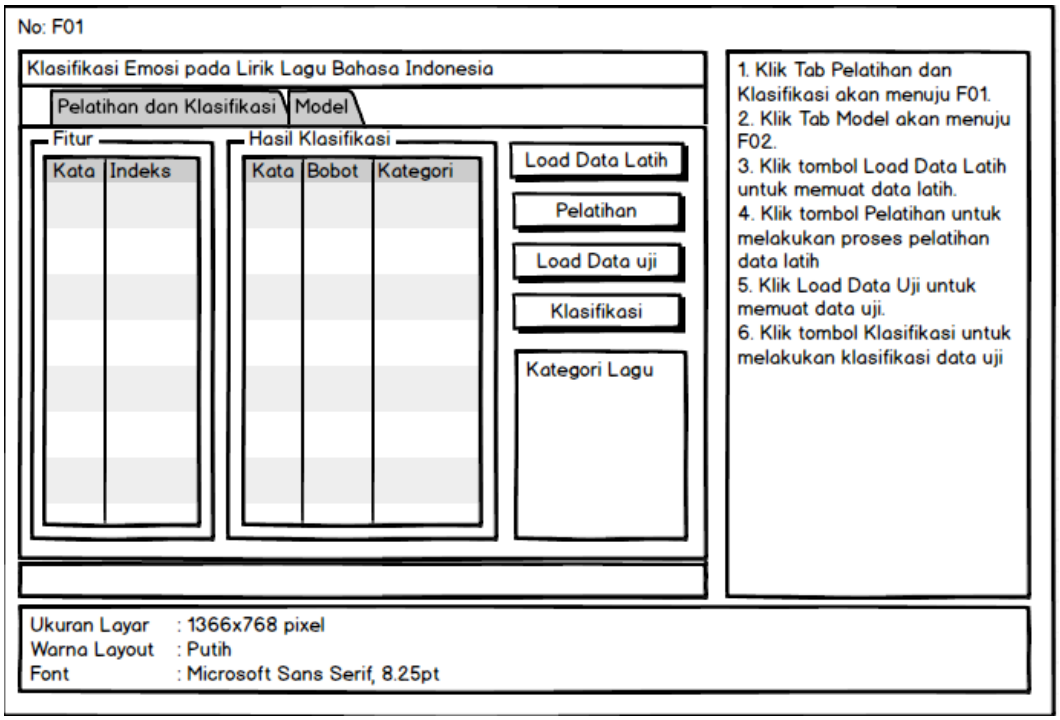



**Gambar 3.25 Struktur Menu**

### 3.9.2 Perancangan Antarmuka

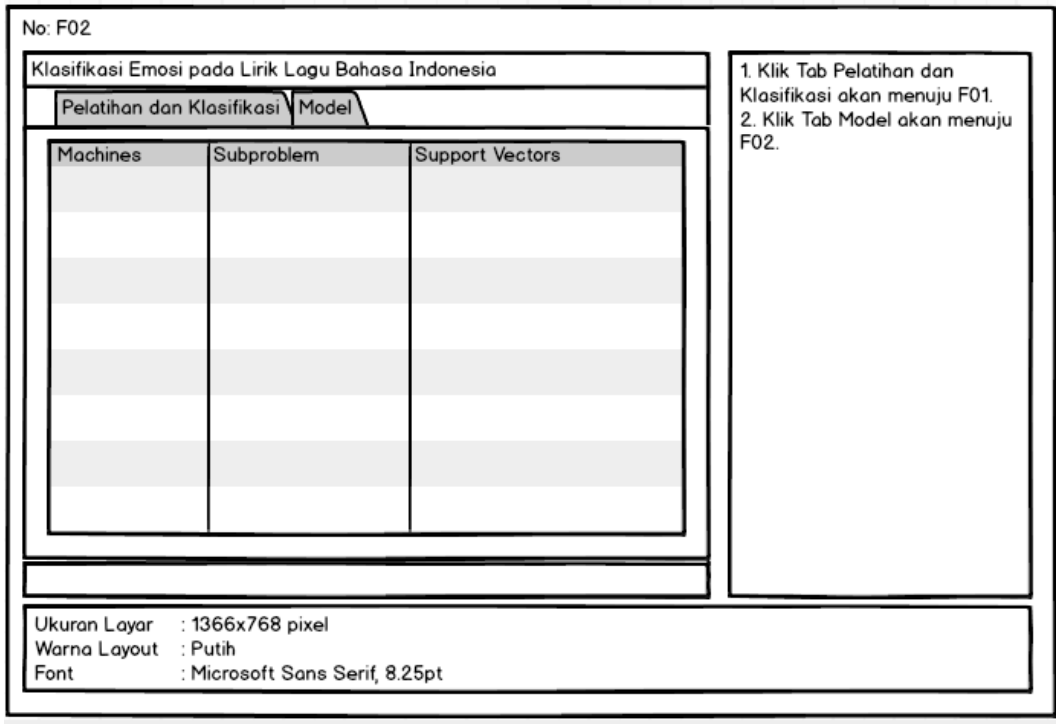
Perancangan antarmuka merupakan suatu bentuk tampilan dari perangkat lunak yang dibangun untuk kebutuhan *interface* dengan pengguna. Berikut adalah perancangan antarmuka yang akan dibangun:

1. F01 - Form Pelatihan dan Klasifikasi




**Gambar 3.26 Perancangan Antarmuka F01**

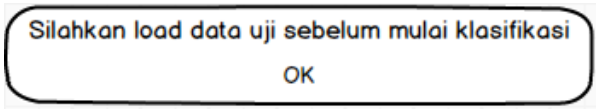
2. F02 – Form Model




**Gambar 3.27 Perancangan Antarmuka F02**

### 3.9.3 Perancangan Pesan

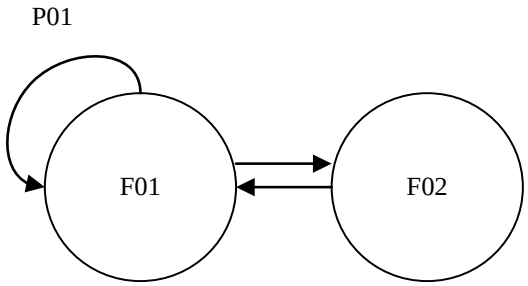
Perancangan pesan merupakan validasi dan pesan yang akan disampaikan oleh perangkat lunak dalam keadaan-keadaan tertentu. Perancangan pesan yang terdapat pada perangkat lunak yang dibangun ditunjukkan pada Gambar 3.28 berikut:




**Gambar 3.28 Perancangan Pesan P01**

### 3.9.4 Jaringan Semantik

Jaringan semantik menggambarkan keterhubungan dari navigasi menu dari satu antarmuka ke antarmuka lain. Jaringan semantik yang terbentuk pada perangkat lunak yang dibangun dapat dilihat pada Gambar 3.29 berikut:




**Gambar 3.29 Jaringan Semantik**