

**Multi-Label Emotion Classification  
using *Logistic Regression* Method with  
Dataset GoEmotions**

Proposal Tugas Pemrosesan Bahasa Alami

Oleh :

11S18045 Gorga Siagian

11S18056 Sri Intan Sinaga



11S4037 - Pemrosesan Bahasa Alami  
Fakultas Informatika dan Teknik Elektro  
Institut Teknologi Del  
2021/2022

## DAFTAR ISI

<b>Proposal Tugas Pemrosesan Bahasa Alami Oleh :</b>	<b>1</b>
<b>DAFTAR ISI.....</b>	<b>2</b>
1.3. Tujuan .....	4
1.4. Manfaat Penelitian .....	4
1.5. Ruang Lingkup Penelitian.....	4
<b>BAB 2 PENDAHULUAN.....</b>	<b>5</b>
<b>2.1 Preprocessing Data .....</b>	<b>5</b>
2.1.1. <i>Text Cleaning</i> .....	5
2.1.2. <i>Case Folding</i> .....	5
2.1.3. <i>Tokenization</i> .....	5
2.1.4. Stopwords Removal .....	6
2.1.5. Stemming .....	6
<b>2.2 TF-IDF .....</b>	<b>6</b>
<b>2.3 Logistic Regression .....</b>	<b>6</b>
<b>2.4 Klasifikasi.....</b>	<b>7</b>
<b>2.5 Sosial Media .....</b>	<b>7</b>
<b>BAB 3 Analisis Dan Desain .....</b>	<b>8</b>
<b>3.1. Analisis Data.....</b>	<b>8</b>
3.1.1 Preprocessing Data.....	8
3.1.1.2. Tokenization .....	9
3.1.1.3. Stopword Removal.....	9
3.1.1.4. <i>Stemming</i> .....	9
3.1.1.5. Lemmatization .....	10
3.1.2. TF-IDF .....	10
<b>3.2 Desain.....</b>	<b>10</b>
<b>BAB 4 IMPLEMENTASI DAN PEMBAHASAN.....</b>	<b>13</b>
<b>4.1 Implementasi Pre-processing.....</b>	<b>13</b>
4.1.1 Cleaning Data.....	13
4.1.2 Tokenization .....	13
4.1.3 Stopword Removal.....	14
4.1.4 Stemming .....	14
4.1.5 Lemmatization .....	14
4.1.6 TF-IDF .....	15
4.1.7 <i>Logistic Regression</i> .....	15
<b>BAB 5 Hasil dan Pembahasan.....</b>	<b>17</b>

5.1 Hasil dan Pembahasan .....	17
<b>BAB 6 Penutup .....</b>	<b>19</b>
6.1 Deskripsi Tugas.....	19
6.2 Pembagian Tugas .....	19
6.3 Kesimpulan.....	20
6.4 Saran .....	20
<b>Referensi.....</b>	<b>21</b>

# BAB 1 PENDAHULUAN

## 1.1. Latar Belakang Masalah

Emosi dapat diidentifikasi dari berbagai sumber seperti teks, ekspresi wajah, gambar, pidato, lukisan, lagu, dan sebagainya. Dengan semakin populernya media sosial online, orang-orang suka mengekspresikan emosi mereka atau berbagi acara yang bermakna dengan orang lain di platform jejaring sosial seperti *twitter*, *facebook*, catatan pribadi, blog, novel, email, pesan obrolan, dan berita utama. Emosi adalah perasaan kuat yang berasal dari suasana hati atau interaksi seseorang satu sama lain.

Deteksi emosi di jejaring sosial online menguntungkan banyak aplikasi seperti layanan iklan yang dipersonalisasi, sistem saran, dll. Emosi dapat diidentifikasi dari berbagai sumber seperti teks, ekspresi wajah, gambar, pidato, lukisan, lagu, dll. Deteksi emosi dapat dilakukan dengan berbagai teknik dalam pembelajaran mesin. Teknik deteksi emosi berfokus pada klasifikasi multi-kelas sambil mengabaikan konsistensi beberapa label emosi dalam satu contoh pada komputer melalui pembelajaran mesin (*machine learning*). Komputer dapat memantau emosi pengguna untuk menyarankan musik atau film yang sesuai dalam interaksi komputer manusia. Selain itu, output dari sistem penambangan emosi dapat berfungsi sebagai input ke sistem lain.

Deteksi emosi adalah masalah klasifikasi multi-label yang membutuhkan prediksi beberapa emosi skor dari urutan data yang diberikan. Setiap data urutan yang diberikan dapat memiliki lebih dari satu emosi, sehingga masalahnya dapat diajukan sebagai masalah klasifikasi multi-label daripada sebagai masalah klasifikasi multi-kelas. Pembelajaran mesin dan pembelajaran mendalam digunakan dalam hal ini penelitian untuk memecahkan masalah tersebut. Pada tugas proyek pemrosesan bahasa alami kali ini, kami melakukan implementasi untuk deteksi emosi pada dataset GoEmotion yang disediakan oleh Hugging Face.

Pada dataset tersebut diambil dari komentar reddit, dimana implementasi yang kami lakukan deteksi emosi dari dataset tersebut dalam bentuk emosi positif, negatif, dan netral. Untuk model yang digunakan untuk melakukan klasifikasi deteksi emosi tersebut menggunakan *Logistic Regression* dimana model klasifikasi *Logistic Regression* tersebut dapat memodelkan probabilitas untuk masalah klasifikasi dengan dua kemungkinan hasil. *Logistic Regression* merupakan perpanjangan dari model linear regression untuk masalah klasifikasi.

## 1.2. Rumusan Masalah

Berdasarkan Latar Belakang masalah yang telah disampaikan maka rumusan masalah yang diperoleh yaitu :

1. Bagaimana pengklasifikasian emosi berdasarkan dataset GoEmotion menggunakan algoritma *Logistic Regression*?

### **1.3. Tujuan**

Dari Rumusan masalah yang diperoleh ,terdapat tujuan dalam penelitian yaitu:

1. Memperoleh implementasi algoritma *Logistic Regression* dalam klasifikasi emosi berdasarkan pada dataset Go Emotions.

### **1.4. Manfaat Penelitian**

Manfaat penelitian untuk Tugas Pemrosesan Bahasa Alami yaitu :

1. Dapat melakukan implementasi dalam deteksi emosi dalam dataset GoEmotion dengan mengklasifikasikan nya dalam bentuk emosi positif, negatif, dan netral.
2. Dapat memahami pemikiran secara logis terkait menganalisis masalah dengan pengolahan bahasa alami.

### **1.5. Ruang Lingkup Penelitian**

Ruang lingkup penelitian Tugas Pemrosesan Bahasa Alami yaitu :

1. Data yang diolah merupakan data GoEmotions yang telah disediakan oleh Hugging Face.
2. Implementasi ini menerima inputan dalam bentuk kalimat Bahasa inggris dan melakukan normalisasi dalam bentuk huruf.
3. Pemrograman menggunakan bahasa pemrograman *Python*.
4. Fitur ekstraksi yang digunakan adalah TF-IDF.

## BAB 2 PENDAHULUAN

Bab 2 ini membahas mengenai terkait tinjauan pustaka dan langkah-langkah yang dilakukan pada penelitian ini.

### 2.1 Preprocessing Data

Pada tahap ini data disiapkan menjadi data yang siap untuk dianalisis. Data yang diperoleh ditampilkan dalam keadaan apa adanya, sehingga mungkin berupa data yang tidak diinginkan karena merupakan data yang tidak relevan. Data yang tidak relevan tersebut terkadang merupakan data yang sulit ditangani. Oleh karena itu, penghapusan bagian data yang tidak relevan yang sangat disarankan dilakukan dan merupakan tahap yang harus dilewati, sehingga efisiensi sistem dapat terwujud. Berikut struktur yang terdapat pada data tweet yaitu User ID (disimbolkan dengan @), URL, teks, tanggal dan waktu, lokasi, file multimedia (gambar, video dll), emoticon, hashtag (disimbolkan dengan #). Masing-masing bagian struktur tersebut memiliki signifikansi sendiri selama analisis sentimen tetapi ada beberapa data yang tidak memiliki efek signifikan sehingga disarankan untuk menghilangkan data ini .

Nama pengguna dalam *Reddit* , ini untuk memberitahu siapa yang menulis komentar tersebut. Pada proses deteksi teks tidak ada pengaruh yang signifikan dari nama pengguna sehingga dengan menerapkan filter, nama pengguna dikeluarkan dari training data dan testing data. Ada batasan karakter dalam komentar *Reddit* sehingga pengguna menyertakan beberapa tautan URL untuk menjelaskannya dengan lebih baik. URL ini (umumnya dimulai dengan http://) tidak diperlukan selama training data dan testing data karena tidak mengandung informasi berguna di dalamnya yang dapat digunakan selama analisis sentimen.

Tahapan untuk melakukan *preprocessing data* terdapat 5 yaitu *text cleaning, case folding, tokenization, stopword removal, dan stemming*.

#### 2.1.1. Text Cleaning

*Text Cleaning* adalah tahapan karakter dan tanda baca yang tidak perlu dihilangkan pada teks, dimana berfungsi mengurangi *noise* pada dataset. Contoh karakter dihilangkan seperti URL, tag, tanda baca, seperti titik(.), koma(,), dan tanda baca lainnya pada text.

#### 2.1.2. Case Folding

Tahap *Case Folding* yaitu proses menyamakan huruf teks ke dalam huruf kecil, berarti jika terdapat huruf besar di dalam teks maka diubah ke dalam bentuk huruf kecil .

#### 2.1.3. Tokenization

Proses *Tokenization* adalah memecah kalimat menjadi bagian atau kata-kata. Hasil kata dari proses disebut token. proses *tokenization* yang dilakukan seperti menghilangkan tanda baca yang tidak diperlukan. Terdapat beberapa model *tokenization* yang digunakan yaitu *unigram, bigram, trigram, dan ngram*.

#### 2.1.4. Stopwords Removal

Merupakan proses menghilangkan kata-kata yang muncul dalam jumlah banyak tetapi dianggap tidak memiliki arti. Alasan menghapus *stopword* yang berhubungan dengan *text mining* yaitu penggunaannya yang terlalu umum, sehingga pengguna bisa fokus pada kata lain yang jauh lebih penting.

#### 2.1.5. Stemming

*Stemming* adalah proses mengubah kata menjadi kata dasar menurut kaidah bahasa Indonesia yang benar.

### 2.2 TF-IDF

Term Frequency — Inverse Document Frequency atau TF — IDF adalah suatu metode algoritma yang berguna untuk menghitung bobot setiap kata yang umum digunakan. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen. Pada Term Frequency (TF), terdapat beberapa jenis formula yang dapat digunakan :

1. TF biner (binary TF), hanya memperhatikan apakah suatu kata atau term ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak diberi nilai nol (0).
2. TF murni (raw TF), nilai TF diberikan berdasarkan jumlah kemunculan suatu term di dokumen. Contohnya, jika muncul lima (5) kali maka kata tersebut akan bernilai lima (5).
3. TF normalisasi, menggunakan perbandingan antara frekuensi sebuah term dengan nilai maksimum dari keseluruhan atau kumpulan frekuensi term yang ada pada suatu dokumen.
4. TF logaritmik, hal ini untuk menghindari dominansi dokumen yang mengandung sedikit term dalam query, namun mempunyai frekuensi yang tinggi.

Lalu, yang berikutnya atau yang kedua adalah **IDF (Inverse Document Frequency)**. Metode IDF merupakan sebuah perhitungan dari bagaimana term didistribusikan secara luas pada koleksi dokumen yang bersangkutan. Berbeda dengan TF yang semakin sering frekuensi kata muncul maka nilai semakin besar, dalam IDF, **semakin sedikit** frekuensi kata muncul dalam dokumen, maka **makin besar** nilainya.

### 2.3 Logistic Regression

Logistic regression adalah bentuk khusus regresi yang diformulasikan untuk melakukan klasifikasi data ke dalam dua group (prediksi group) dan menjelaskan variabel dependen biner (kategorikal/non-metric) Logistic regression cocok digunakan bila kita ingin memprediksi keanggotaan variabel independen (prediktor) dalam dua grup saja, misal grup orang yang menderita diabetes atau tidak menderita diabetes. Bentuk umum logistic regression:

$$Y = X_1 + X_2 + X_3 + \dots + X_n$$

Dimana  $Y$  = biner non-metric dan  $X_1 + X_2 + X_3 + \dots + X_n$  = non-metric atau metric. Logistic regression tidak memerlukan asumsi normalitas dari variabel dependen. Ketika variabel dependen berbentuk kategorikal biner maka distribusinya adalah binomial. Interpretasi bobot dalam logistic regression berbeda dari interpretasi bobot dalam linear regression, karena hasil dalam logistic regression adalah probabilitas antara 0 dan 1. Bobot tidak lagi mempengaruhi probabilitas secara linier. Jumlah tertimbang ditransformasikan oleh fungsi logistik menjadi probabilitas. Oleh karena itu sangat perlu merumuskan kembali persamaan untuk interpretasi sehingga hanya suku linier yang berada di ruas kanan rumus.

$$\log \left( \frac{P(y = 1)}{1 - P(y = 1)} \right) = \log \left( \frac{P(y = 1)}{P(y = 0)} \right) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

Rumus diatas disebut dengan istilah dalam fungsi  $\log()$  “odds” (probabilitas kejadian dibagi dengan probabilitas tidak ada kejadian) dan dibungkus dalam logaritma yang disebut log odds. Rumus ini menunjukkan bahwa regression models logistik merupakan model linier untuk log odds. Rumus tersebut juga dapat mengetahui bagaimana prediksi berubah ketika salah satu fitur  $x_j$  diubah 1 unit. Untuk melakukan ini, pertama-tama kita dapat menerapkan fungsi  $\exp()$  ke kedua sisi persamaan:

$$\frac{P(y = 1)}{1 - P(y = 1)} = odds = \exp(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p)$$

Logistic regression tidak memerlukan asumsi normalitas dari variabel independen. Selain itu variabel independen dapat bertipe metric maupun non-metric. Tidak seperti *discriminant analysis*, yang bisa digunakan juga untuk prediksi variabel dependen biner, *logistic regression* tidak memerlukan pengecekan keseimbangan matriks variance-covariance di antara 2 grup → bahkan ketika terpenuhi, *logistic regression* lebih disukai.

## 2.4 Klasifikasi

Klasifikasi adalah suatu kegiatan mengelompokkan. Di mana klasifikasi sangat dibutuhkan dalam perpustakaan, karena klasifikasi bertujuan untuk mengelompokkan satu koleksi yang sejenis, yang pengelompokannya berdasarkan judul, pengarang, dan lain sebagainya.

## 2.5 Sosial Media

Media sosial adalah fasilitas jaringan sosial online yang menghubungkan profil pengguna dengan orang lain atau kelompok lainnya sebagai media komunikasi. Media sosial ada dalam berbagai macam bentuk, diantaranya termasuk sosial network, forum internet, weblogs, social blogs, micro blogging, wikis, podcasts, gambar, video, rating, dan bookmark sosial. Salah satu forum sosial adalah Reddit. Sebagai media jejaring sosial, dalam Reddit pengguna bisa menemukan berita terbaru, topik-topik yang sedang tren, dan meme-meme terbaik di internet. Pengguna juga dapat mencari dan menemukan berbagai komunitas dengan topik beraneka ragam. Aktivitas yang dapat dilakukan seperti mengunggah cerita, gambar, dan tautan menarik untuk bahan diskusi dengan mudah.



## BAB 3 Analisis Dan Desain

Pada Bab 3 Proyek PBA ini menjelaskan mengenai Analisis dan Desain pengerjaan proyek PBA yang telah dilakukan

### 3.1. Analisis Data

Pada Analisis Data ini menggunakan sumber data GoEmotion yang telah disediakan oleh *Hugging Face*. Daset GoEmotion ini terdiri dari kumpulan *Reddit Comments* dengan berbahasa Inggris, dimana memiliki kategori 27 label emosi dan label netral. Dataset ini digunakan untuk pendalaman dalam melakukan multi label *emotion classification* dan *multi-class*. Datasets GoEmotion ini berupa data mentah, dimana terdiri dari kolomnya dengan entri bilangan biner 0 atau 1 daripada menggunakan daftar id seperti data-data yang telah disederhanakan. Pada proyek ini data mentah tersebut akan dibagi kedalam data test dan data train.

#### 3.1.1 Preprocessing Data

*Preprocessing Data* adalah sebuah Teknik awal dari data mining dimana mengubah data mentah yang telah dikumpulkan dari berbagai sumber menjadi informasi lebih bersih dan bisa digunakan untuk pengolahan selanjutnya. Pada pengerjaan proyek PBA *Preprocessing Data* yang digunakan adalah *Cleaning Data*, *Tokenization*, *Stopword Removal*, *Stemming*, dan *Lemmatization*.

##### 3.1.1.1 Cleaning Data

Pada Dataset yang telah dikumpulkan mempunyai *missing value* atau *noise*, sehingga proses pengumpulan data tidak sempurna dan banyak bagian yang tidak relevan dan hilang. Metode yang digunakan dalam mengatasi *missing value* menggunakan *Cleaning Data*, agar dapat menangani *missing value* atau *noise* yang dimiliki oleh dataset yang dikumpulkan. Teknik *Cleaning Data* yang digunakan dalam penelitian ini adalah sebagai berikut.

##### a. Removal Of Punctuations

*Removal Of Punctuations* merupakan proses dimana sistem menghilangkan tanda baca pada kumpulan dataset yang dimiliki. Kumpulan tanda baca yang dihilangkan yaitu tanda titik(.), tanda koma(,), tanda kurung/kurung kuadrat(), tanda seru(!), dan tanda baca lainnya. Berikut data yang telah dilakukan *Removal Of Punctuations*.

Sebelum Removal Of Punctuations	Sesudah Removal Of Punctuations
>sexuality shouldn't be a grouping category	sexuality shouldn't be a grouping category
You do right, if you don't care then fuck 'em!	You do right if you dont care then fuck em

Tabel 3.1. Removal Of Punctuations

##### b. Case Folding

*Case Folding* merupakan proses dimana mengubah semua data kedalam bentuk huruf kecil, artinya Ketika terdapat bentuk Huruf Kapital dalam data tersebut diubah ke dalam huruf kecil. Tujuan digunakan *Case Folding* agar data tersebut dalam bentuk yang sama yaitu huruf kecil. Berikut contoh penerapan *Case Folding* dapat dilihat pada Tabel 3.2 berikut ini.

Before Case Folding	After Case Folding
You do right if you dont	you do right if you dont

care then fuck em	care then fuck em
Man i love reddit	man i love reddit
NAME was nowhere near them he was by the Fact	name was nowhere near them he was by the fact

**Tabel 3.2. Case Folding**

### 3.1.1.2. Tokenization

Tokenisasi adalah tugas memisahkan deretan kata di dalam kalimat, paragraf atau halaman menjadi token atau potongan kata tunggal atau termmed word. Pada saat bersamaan, tokenisasi juga membuang beberapa karakter tertentu yang dianggap sebagai tanda baca. Contoh penerapan *Tokenization* dapat dilihat pada Tabel 3.3 berikut ini.

Before Tokenization	After Tokenization
sexuality shouldn't be a grouping category	['sexuality', 'shouldn't', 'be', 'a', 'grouping', 'category']
you do right if you dont care then fuck em	['you', 'do', 'right', 'if', 'you', 'dont', 'care', 'then', 'fuck', 'em']
man i love reddit	['man', 'i', 'love', 'reddit']

**Tabel 3.3. Tokenization**

### 3.1.1.3. Stopword Removal

Stopword didefinisikan sebagai term yang tidak berhubungan irrelevant dengan subyek utama dari database meskipun kata tersebut sering kali hadir di dalam dokumen. Berikut ini adalah contoh stopwords dalam bahasa inggris: [i, me, my, myself, we, our, ours, ourselves, you, you're, you've, you'll, you'd, your, yours, yourself, yourselves, he, most, other, some, such, no, nor, not, only, own, same, so, then, too, very, s, t, can, will, just, don, don't, should, should've, now, d, ll, m, o, re, ve, y, ain, aren't, could, couldn't, didn't, didn't]. Contoh penerapan Stopword Removal dapat dilihat pada Tabel 3.4 berikut ini.

Before Stop Removal	After Stop Removal
['sexuality', 'shouldn't', 'be', 'a', 'grouping']	['sexuality', '', 'grouping', 'category']
1. Stemming , 'category']	
['you', 'do', 'right', 'if', 'you', 'dont', 'care', 'then', 'fuck', 'em']	['right', 'dont', 'care', 'fuck', 'em']
['man', 'i', 'love', 'reddit']	['man', 'love', 'reddit']

**Tabel 3.4. Stopword Removal**

### 3.1.1.4. Stemming

Kata-kata yang muncul di dalam dokumen/datasets sering mempunyai banyak varian morfologik. Karena itu, setiap kata yang bukan stop-words direduksi ke bentuk stemmed word term yang cocok. Kata tersebut distem untuk mendapatkan bentuk akarnya dengan menghilangkan awalan atau akhiran. Dengan cara ini, diperoleh kelompok kata yang mempunyai makna serupa tetapi berbeda wujud sintaktis satu dengan lainnya. Kelompok tersebut dapat direpresentasikan oleh satu kata tertentu. Sebagai contoh, kata menyebutkan, tersebut, disebut dapat dikatakan serupa atau satu kelompok dan dapat diwakili oleh satu kata umum sebut Contoh penerapan Stemming dapat dilihat pada Tabel 3.5 berikut ini.

Before Stemming	After Stemming
['sexuality', '', 'grouping', 'category']	['sexual', '', 'group', 'categori']

['right', 'dont', 'care', 'fuck', 'em']	['right', 'dont', 'care', 'fuck', 'em']
['man', 'love', 'reddit']	['man', 'love', 'reddit']
['name', 'nowhere', 'near', 'fact']	['name', 'nowher', 'near', 'fact']

**Tabel 3.5. Stemming**

### 3.1.1.5. Lemmatization

Lemmatization adalah proses pengelompokan bersama bentuk-bentuk infleksi yang berbeda dari sebuah kata sehingga mereka dapat dianalisis sebagai satu item. Lemmatization mirip dengan stemming tetapi membawa konteks pada kata-kata. Jadi itu menghubungkan kata-kata dengan arti yang mirip dengan satu kata Contoh penerapan lemmatization dapat dilihat pada Tabel 3.6 berikut ini.

Before Stemming	After Stemming
['sexual', '', 'group', 'categori']	['sexual', '', 'group', 'categori']
['right', 'dont', 'care', 'fuck', 'em']	['right', 'dont', 'care', 'fuck', 'em']
['man', 'love', 'reddit']	['man', 'love', 'reddit']

**Tabel 3.6. Lemmatization**

### 3.1.2. TF-IDF

*Term Frequency — Inverse Document Frequency* atau TF — IDF adalah suatu metode algoritma yang berguna untuk menghitung bobot setiap kata yang umum digunakan. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen.

### 3.1.3. Analisis Algoritma Logistic Regression

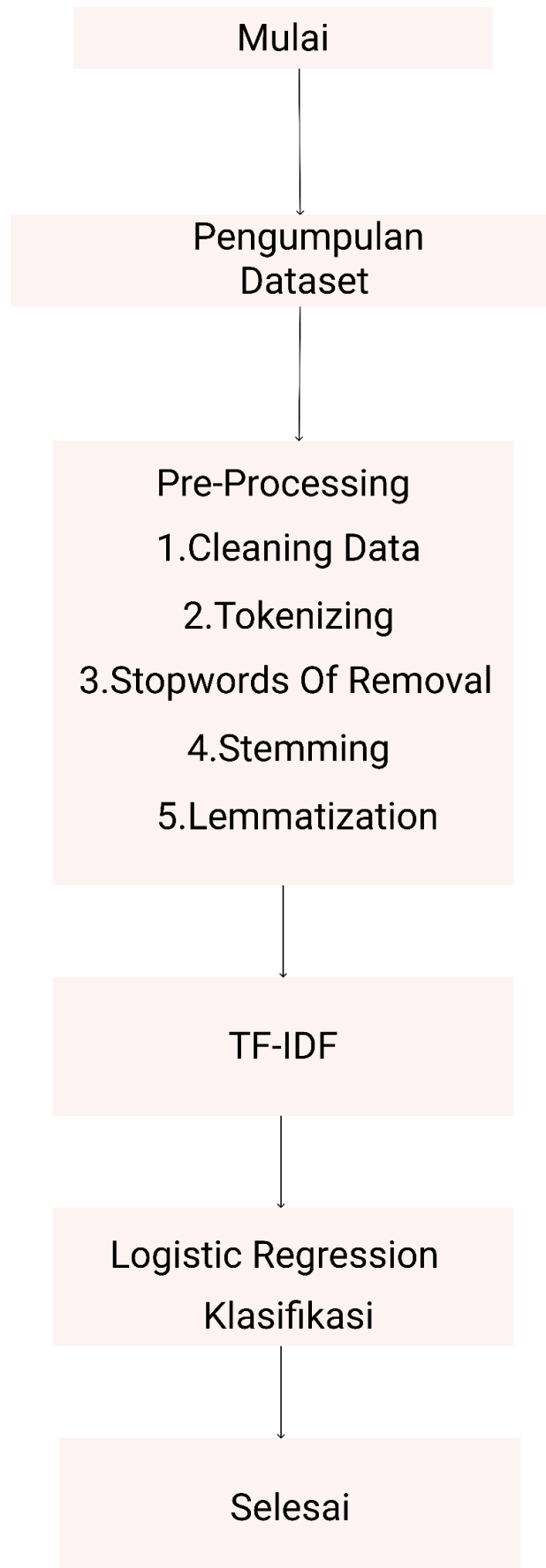
Pada metode pengklasifikasian mengenai *emotion* yang peneliti lakukan yaitu menggunakan metode *Logistic Regression*. Pengklasifikasian ini dilakukan setelah melalui proses dan tahapan yang telah dilakukan sebelumnya

## 3.2 Desain.

Pada bagian desain ini yaitu membuat desain pada implementasi klasifikasi menggunakan Naive Bayes. Proses desain menjelaskan bagaimana melakukan tahapan klasifikasi multi label dengan *Logistic Regression* dengan metode TF-IDF.

### 3.2.1. Desain Proyek Pengerjaan

Berikut merupakan desain flowchart tahapan dalam pengerjaan proyek PBA dalam klasifikasi multi label menggunakan *Logistic Regression* :



**Gambar 3.1 Desain Flowchart Proyek PBA**

Adapun penjelasan dari desain flowchart di atas :

1. Pengumpulan Dataset

Pada pengumpulan dataset ,dikumpulkan dengan menggunakan library datasets ,lalu mengambil data *raw* GoEmotion yang disediakan oleh *Hugging Face*.

2. Pre-Processing Data

Setelah pengumpulan datasets ,tahap selanjutnya melakukan *PreProcessing* pada datasets yang telah diambil diantaranya menghilangkan tanda baca,mengubah semua huruf menjadi huruf kecil,dan lain-lain. Adapun tahapan preprocessing yaitu *cleaning data, stopword of removal, case folding, tokenizing* dan *stemming*.

3. TF-IDF

Pada penerapan TF-IDF yang dibangun menggunakan library yang tersedia .

4. Klasifikasi menggunakan *Logistic Regression*

Selanjutnya melakukan tahap klasifikasi menggunakan model *Logistic Regression*. Untuk label emosi terdiri dari 28 label yaitu *admiration, amusement, anger, annoyance, approval, caring, confusion, curiosity, desire, disappointment, disapproval, disgust, embarrassment, excitement, fear, gratitude, grief, joy, love, nervousness, optimism, pride, realization, relief, remorse, sadness, surprise, neutral*.

5. Hasil klasifikasi dari data uji.

Setelah dilakukan seluruh tahapan, maka akan diperoleh hasil klasifikasi berdasarkan data uji.

## BAB 4 IMPLEMENTASI DAN PEMBAHASAN

Pada Bab 4 ini menjelaskan Implementasi dan pembahasan dari pengerjaan proyek PBA berikut ini :

### 4.1 Implementasi Pre-processing

*Preprocessing Data* adalah sebuah Teknik awal dari data mining dimana mengubah data mentah yang telah dikumpulkan dari berbagai sumber menjadi informasi lebih bersih dan bisa digunakan untuk pengolahan selanjutnya. Pada pengerjaan proyek PBA *Preprocessing Data* yang digunakan adalah *Cleaning Data, Tokenization, Stopword Removal, Stemming, dan Lemmatization*.

#### 4.1.1 Cleaning Data

Implementasi yang dilakukan dalam Cleaning Data yang terdiri dari Punctuational Removal dan Case Folding.

##### 4.1.1.1 Removal Of Punctuation

Salah satu teknik preprocessing teks umum lainnya adalah menghilangkan tanda baca dari data teks. Pada proses ini perlu hati-hati memilih daftar tanda baca untuk mengecualikan tergantung pada kasus penggunaan. Misalnya, `string.punctuation` di python berisi simbol tanda baca berikut: `!"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~`. Kode program yang digunakan untuk menjalankan proses ini terlihat pada *source code* berikut ini.

```
#Data Cleaning

#Removal of Punctuations
import string
string.punctuation
PUNCT_TO_REMOVE = string.punctuation
def remove_punctuation(text):
    return text.translate(str.maketrans('', '',
PUNCT_TO_REMOVE))

feats_train_df['text'] =
feats_train_df['text'].apply(lambda text:
remove_punctuation(text))
feats_train_df.head()
```

##### 4.1.1.2 Case Folding

Pada tahap ini yaitu mengubah semua kalimat pada dataset menjadi huruf kecil. fungsi yang digunakan untuk mengubah semua huruf menjadi huruf kecil dengan menggunakan fungsi `lower()`. Berikut kode program dalam *Case Folding*.

```
#mengubah kalimat ke huruf kecil
feats_train_df['text'] =
feats_train_df['text'].str.lower()
feats_train_df.head()
```

#### 4.1.2 Tokenization

Tahap Tokenisasi yaitu mengubah bentuk kalimat menjadi bentuk kata penyusunnya atau diubah ke dalam bentuk token. Tahapan ini dilakukan dengan menggunakan fungsi `word_tokenize()` yang sebelumnya melakukan import pada library *nlk* python. Berikut Kode program yang digunakan :

```
#Tokenisasi
import nltk
nltk.download('punkt')
```

```
def tokenization(text):
    tokens = nltk.word_tokenize(text)
    return tokens

#tokenization pada dataset
feats_train_df['text'] =
feats_train_df['text'].apply(lambda x: tokenization(x))
feats_train_df.head()
```

### 4.1.3 Stopword Removal

Pada tahap ini menghilangkan kata yang paling sering digunakan pada stopwords bahasa Inggris seperti "i", "you", "their", "to", dll. Pada langkah ini, kita akan hapus kata-kata ini dari seluruh kumpulan data dengan menggunakan pustaka NLTK. Berikut Kode Program yang digunakan :

```
#Stopwords Removal
nltk.download('stopwords')
stopwords = nltk.corpus.stopwords.words('english')

def stopwords_remove(inputs):
    return [item for item in inputs if item not in
stopwords]

feats_train_df['text'] =
feats_train_df['text'].apply(stopwords_remove)
feats_train_df.head()
```

### 4.1.4 Stemming

Stemming adalah proses mereduksi kata-kata infleksi (atau kadang-kadang turunan) menjadi bentuk dasar, dasar, atau akar kata. Misalnya, jika ada dua kata dalam korpus walk dan walking, maka stemming akan membentuk sufiks untuk membuatnya berjalan. Tapi katakanlah dalam contoh lain, kita memiliki dua kata console dan consoling, stemmer akan menghapus sufiks dan menjadikannya consol yang bukan kata bahasa Inggris yang tepat. Ada beberapa jenis algoritma stemming yang tersedia dan salah satu yang terkenal adalah porter stemmer yang banyak digunakan. Kita dapat menggunakan paket nltk untuk hal yang sama.

```
#Stemming
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()
def stemming(text):
    stem_text = [stemmer.stem(word) for word in text]
    return stem_text

feats_train_df['text'] =
feats_train_df['text'].apply(lambda text: stemming(text))
feats_train_df.head()
```

### 4.1.5 Lemmatization

Lemmatization adalah proses pengelompokan bersama bentuk-bentuk infleksi yang berbeda dari sebuah kata sehingga mereka dapat dianalisis sebagai satu item. Lemmatization mirip dengan stemming tetapi membawa konteks pada kata-kata. Jadi, menghubungkan kata-kata dengan arti yang mirip dengan satu kata.

```
#Lematisasi
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')
lemmatizer = WordNetLemmatizer()

def lemmatization(inputs):
    return [lemmatizer.lemmatize(word=x, pos='v') for x in
inputs]

feats_train_df['text'] =
feats_train_df['text'].apply(lemmatization)
feats_train_df.head()
```

#### 4.1.6 TF-IDF

TF-IDF merupakan salah satu metode fitur ekstraksi teks pada dokumen. Dimana sebelum melakukan Klasifikasi multi label pada data yang telah melalui preprocessing maka dilakukan terlebih dahulu ekstraksi dari fitur kata atau TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen. Setelah dataset telah melalui *preprocessing* maka yang dilakukan pertama yaitu menginisiasi data kedalam bentuk X dan Y agar nantinya data tersebut dapat dibagi ke dalam bentuk data Xtrain, Ytrain, Xtest, Ytest. berikut code program dibawah

```
X = feats_train_df["text"]
y = np.asarray(feats_train_df[feats_train_df.columns[1:]])
```

Selanjutnya setelah data dinisiasi ke dalam bentuk X dan Y maka selanjutnya split data train dan test pada data X dan y yang telah diinisiasi. sebelum membuat fitur TF-IDF terlebih dahulu membagi data menjadi rangkaian data *train* dan *test* untuk pelatihan dan menguji kinerja model. Saya akan menggunakan pembagian 80-20 – 80% dari sampel data di set train dan sisanya di set test:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, te
st_size=0.2, random_state=42)
```

Sekarang kita dapat membuat fitur TF-IDF untuk *train* dan set *test*:

```
X_train_tfidf = vetorizar.transform(X_train)
X_test_tfidf = vetorizar.transform(X_test)
```

#### 4.1.7 Logistic Regression



Setelah dilakukan metode fitur ekstraksi TF-IDF maka akan masuk ke model klasifikasi menggunakan *Logistic Regression*. Alasan menggunakan *Logistic Regression*, karena data yang dimiliki sangat banyak, dan menyesuaikan 28 model data yang banyak tersebut dengan set predictor (TF-IDF). Ketika melakukan pelatihan terhadap 28 model dari data yang banyak tersebut memakan banyak waktu pada sistem yang sederhana, sehingga saya menggunakan *Logistic Regression* karena cepat dilatih dengan daya komputasi yang terbatas. Pertama sekali membangun model dari *Logistic Regression*, seperti code program dibawah ini :

```
# Models
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
# Binary Relevance
from sklearn.multiclass import OneVsRestClassifier
lr = LogisticRegression(C=2.0)
clf = OneVsRestClassifier(lr)
# fit model on train data
clf.fit(X_train_tfidf, y_train)
```

Pada kode program diatas saya menggunakan kelas *OneVsRestClassifier* sk-learn untuk menyelesaikan masalah ini sebagai Relevansi Biner atau masalah satu lawan semua. Lalu, cocokkan (*fit*) data train pada 28 model yang dilatih. Selanjutnya Menghitung *accuracy* dan *F1 score* dengan menggunakan *Logistic Regression*, didapat hasil akurasi dan F1

Berikut Kode Program menghitung *F1 score* dan akurasi dari klasifikasi *Logistic Regression*

```
# Performance metric
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score

# make predictions for test set
y_pred = clf.predict(X_test_tfidf)
# evaluate performance
print('F1 score', f1_score(y_test, y_pred, average="micro"))
print('Accuracy', accuracy_score(y_test, y_pred))
```

Terakhir melakukan prediksi dari kalimat coment reddit, saya memprediksi kalimat komen reddit "man love reddit" dan menampilkan tampilan 28 model dalam bentuk array Berikut Kode Program dari menentukan label emosi dari kalimat yang diprediksi :

```
new_sentences = ["man love reddit"]
new_sentence = vectorizer.transform(new_sentences)


predicted_sentences = clf.predict(new_sentence)
print(predicted_sentences)
```

## BAB 5 Hasil dan Pembahasan

Pada Bab 5 ini menjelaskan Hasil yang didapatkan selama pengerjaan proyek PBA

### 5.1 Hasil dan Pembahasan

Paada implementasi menggunakan metode *Logistic Regression* ini diperoleh dengan menggunakan hasil klasifikasi yang berisikan informasi hasil klasifikasi *precision*, *recall*, *F1-score*, dan *accuracy*. hasil klasifikasi tersebut dapat dilihat di Gambar 5.1 dan 5.2 di bawah ini .



F1 score 0.24216735430634345  
Accuracy 0.14212332820452125

**Gambar 5.1 Hasil Klasifikasi *F1-score (micro avg)* dan *Accuracy***

	precision	recall	f1-score
0	0.67	0.29	0.40
1	0.58	0.31	0.40
2	0.53	0.11	0.18
3	0.36	0.02	0.04
4	0.58	0.03	0.06
5	0.40	0.03	0.06
6	0.53	0.03	0.06
7	0.68	0.04	0.07
8	0.45	0.07	0.12
9	0.53	0.02	0.04
10	0.46	0.01	0.02
11	0.58	0.08	0.13
12	0.63	0.06	0.11
13	0.58	0.06	0.11
14	0.59	0.19	0.28
15	0.89	0.72	0.79
16	0.15	0.02	0.03
17	0.50	0.11	0.18
18	0.65	0.39	0.48
19	0.35	0.03	0.06
20	0.57	0.17	0.26
21	0.69	0.03	0.07
22	0.51	0.02	0.04
23	0.00	0.00	0.00
24	0.47	0.15	0.23
25	0.53	0.13	0.20
26	0.46	0.08	0.14
27	0.53	0.17	0.25

**Gambar 5.2 Hasil Klasifikasi *precision* dan *recall* ,dan *F1-Score***

Pada Gambar diatas menampilkan hasil klasifikasi yang diperoleh,pada **Gambar 5.1** diperoleh nilai dari *F1-Score* dalam *average micro* dan hasil *accuracy* yang menunjukkan keberhasilan dari mode *Logistic Regression* dalam melakukan prediksi atau klasifikasi data,terlihat *accuracy* yang didapat adalah **0.14212332820452125**.Sedangkan pada **Gambar 5.2** menampilkan nilai dari *precision*, *recall*, *F1-score* dari 28 model label yang ada.Pada gambar **5.2** tersebut nilai *precision* tertinggi didapatkan pada label ke 15(disapproval) sebesar 0,89 . *Precision* adalah rasio pengamatan positif yang diprediksi dengan benar dengan total pengamatan positif yang diprediksi.Selanjutnya nilai *recall* tertinggi pada **Gambar 5.2** yaitu pada label 15(disapproval) sebesar 0,72 .*Recall* adalah rasio pengamatan positif yang diprediksi dengan benar dengan semua pengamatan di kelas yang sebenarnya.Begitu juga sebaliknya pada nilai *F1-score* tertinggi diapatkan pada label 15 (disapproval) sebesar 0,79,*F1-Score* adalah rata-rata tertimbang dari *Precision* dan *Recall*. Oleh karena itu, skor ini memperhitungkan positif palsu dan negatif palsu.

Pada **Gambar 5.1** menampilkan nilai *accuracy* yang didapatkan pada klasifikasi menggunakan *Logistic Regression* yaitu **0.14212332820452125**.Hasil yang didapatkan *accuracy* nya cukup rendah,sehingga ketika hasil akurasi yang didapatkan rendah maka Ketika melakukan klasifikasi emosi pada salah satu reddit comments terdapat beberapa tidak kecocokan dari label emosi yang ditemukan terhadap label emosi sebenarnya.Misalnya seperti pada table di bawah ini :

Teks	Label Sebenarnya	Label Hasil Prediksi
Man I Love Reddit	love	Neutral

**Tabel 5.1 Hasil Prediksi**

Seperti ditammpilkan pada tabel diatas ketika mencari label emosi pada *Reddit comment* "Man I Love Reddit"pada label sebenarnya yaitu *love* ternyata hasil prediksi yang didapatkan adalah label emosi *neutral*,sehingga tidak sesuai yang didapatkan. Untuk mendapatkan akurasi yang tinggi sebuah model harus mampu melakukan prediksi dan klasifikasi pada model yang ada.

## BAB 6 Penutup

Pada Bab 6 menjelaskan mengenai deskripsi tugas pengerjaan proyek, pembagian tugas dari masing-masing anggota kelompok, kesimpulan dan saran dari pengerjaan proyek PBA.

### 6.1 Deskripsi Tugas

Berikut merupakan deskripsi tugas selama pengerjaan proyek PBA berlangsung :

No	Pekerjaan	Keterangan
1	Studi Literatur	Melakukan studi literatur terkait penelitian yang berasal dari jurnal dan <i>paper</i> .
2	Pengumpulan Data	Melakukan pengumpulan data berupa kalimat-kalimat berbahasa Indonesia dengan teknik <i>scraping</i> dari beberapa buku, jurnal dan <i>website</i> .
3	Analisis Pengolahan Data	Melakukan analisis mengenai pengolahan data yang berupa <i>preprocessing</i> yang dilakukan pada data
4	Analisis Algoritma	Melakukan analisis mengenai algoritma yang digunakan untuk melakukan normalisasi.
5	Implementasi	Melakukan implementasi.
6	Pembahasan, Kesimpulan dan Saran	Membuat penjelasan dari pengujian model yang telah dilakukan serta membuat kesimpulan akhir dari tugas akhir dan saran untuk penelitian selanjutnya.

Table 6.1 Deskripsi Tugas

### 6.2 Pembagian Tugas

Berikut merupakan Tabel Pembagian Tugas dari masing-masing anggota kelompok :

No	Nama	Pekerjaan						
		1	2	3	4	5	6	7
1	Gorga Siagian	✓	✓	✓	✓	✓	✓	✓
2	Sri Intan Sinaga	✓						

Tabel 6.2 Pembagian Tugas

### 6.3 Kesimpulan

Kesimpulan yang didapatkan dari hasil pengerjaan proyek PBA yaitu :

- a. Pada pengerjaan proyek PBA telah melakukan implementasi menggunakan algoritma *Logistic Regression* dalam klasifikasi multi-label emosi pada dataset GoEmotion dan menggunakan TF-IDF sebagai metode fitur ekstraksi data teks .
- b. Hasil *accuracy* yang diperoleh pada pengerjaan proyek PBA menggunakan *Logistic Regression* dan TF-IDF yaitu sebesar **0.14212332820452125**

### 6.4 Saran

Saran yang dapat diberikan dalam pengerjaan proyek PBA ini yaitu :

- a. Hasil akurasi yang didapatkan masih cukup rendah ,sehingga terdapat ketidaksesuai dari hasil prediksi pada label emosi yang didapatkan .Diharapkan pada penelitian selanjutnya dapat membangun model yang lebih baik dan metode fitur ekstraksi data tek yang lebih baik juga.
- b. Pada pengerjaan atau penelitian selanjutnya diharapkan menggunakan metode fitur ekstraksi data teks lainnya seperti *Word2Vec* atau lainnya.
- c. Pada pengerjaan atau penelitian mendatang diharapkan melakukan kerjasama yang baik anatr anggota kelompok .
- d. Pada pengerjaan atau penelitian yang akan datang diharapkan membandingkan model klasifikasi terhadap *Logistic Regression*

## Referensi

- [1] Gangadhara Rao Kommu, M.Trupthi, Suresh Pabboju, "A novel approach for multilabel classification using probabilistic classifiers", 2014 International Conference on Advances in Engineering and Technology Research (ICAETR), pp. 1-8, Aug. 2014
- [2] Jun Huang,Guorong Li, Shuhui Wang, Weigang Zhang,QingmingHuang,"Group Sensitive Classifier Chains For Multilabel Classification",2015 IEEE International Conference on Multimedia and Expo (ICME), pp. 1-6, 2015.
- [3] Feng Q in,Xian Juan,Tang Ze-Kai Cheng"Applications of Apri-ori Algorithm in Multi label Classification using probabilistic classifiers",13 Proceedings of the 2013 International Conference on Computational and Information Sciences (ICCIS), pp.717-720,June, 2013.
- [4] Ronaldo C.Prati,"Fuzzy rule classifiers for multi-label Classification",2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp.1-8, Aug.2015.
- [5] Yuichiro Kase, Takao Miura,"Multi-Label Classification Using Labelled Association",2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM),pp.90-95, Aug.2015.