PROGRAM STUDI SARJANA SISTEM INFORMASI

LAPORAN PROYEK PEMROSESAN BAHASA ALAMI

Analysis Emotion Of Comment Topic Reddit With Multi-label Classification Using SVM Algorithm

Oleh:

12S18027 Christopher A Hutabarat

12S18028 Alda G M Lumban Gaol

12S18032 Sarah H M Siahaan

FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO

INSTITUT TEKNOLOGI DEL

LAGUBOTI

NOVEMBER 2021

DAFTAR ISI

DAFTAR ISI 2
DAFTAR TABEL 3
DAFTAR GAMBAR 4
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Tujuan 2
1.3 Manfaat 2
1.4 Ruang Lingkup 2
BAB 2 ISI 3
2.1 Analisis 3
2.1.1 Analisis Data 3
2.1.2 Analisis Metode 5
2.2 Design 7
2.2.1 Data Preprocessing 7
2.2.2 Feature Extraction 8
2.2.3 Modelling with SVM 8
2.2.4 Evaluation and Result 9
2.3 Implementasi 10
2.3.1 Data Preprocessing 10
2.3.2 Feature Extraction 15
2.3.4 Modelling with SVM 16
2.4 Evaluasi Model 19
BAB 3 PENUTUP 21
3.1 Kesimpulan 21
3.2 Saran 21
3.3 Rencana Kerja 21
3.3.1 Pembagian Tugas 21
3.3.2 Jadwal Kerja 22
REFERENSI 23

DAFTAR TABEL

Table 1 Atribut Dataset 3
Table 2 Pembagian Tugas 21
Table 3 Jadwal Kerja 22

DAFTAR GAMBAR

Gambar 1 Hyperplane SVM..... 6
Gambar 2 Data Linearly Separable..... 6
Gambar 3 Design Analisis..... 7
Gambar 4 Support Vector Machine (SVM)..... 9
Gambar 5 Confusion Matrix..... 9

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Reddit (reddit.com) adalah sebuah situs yang dibangun oleh Alexis Ohanian dan Steve Huffman pada tahun 2005. Reddit merupakan sebuah situs penyedia konten, dimana para penggunanya yang disebut sebagai *redditors* dapat membagikan maupun memberikan komentar terhadap konten-konten yang ada pada reddit. Konten dan komentar tersebut dapat disukai (*upvote*) atau tidak disukai (*downvote*) oleh *redditors* lain. Konten dan komentar *redditors* yang mendapat *downvote* akan diberikan poin karma, dan yang mendapat *upvote* akan mendapatkan penghargaan. Hal ini menjadi pendorong bagi para *redditors* untuk memposting konten yang baik, membuat komentar yang bermanfaat dan juga memberikan umpan balik yang relevan [1].

Salah satu cara yang dapat dilakukan untuk menilai kualitas komentar yang di*posting* pada reddit adalah dengan menggunakan *emotion analysis*. *Emotion analysis* adalah proses mengidentifikasi dan menganalisis emosi yang diungkapkan dalam suatu data tekstual. *Emotion analysis* dalam teks dapat membantu menentukan opini dan maksud afektif dari penulis, serta sikap, evaluasi, dan kecenderungan mereka terhadap berbagai topik [2].

Memahami opini publik dapat membantu kreator dalam mengambil keputusan [3]. Sama halnya dengan *redditors* pada *platform* reddit yang sebaiknya juga memahami opini *redditors* lainnya terhadap konten yang dihasilkan dan komentar yang diberikan. Sehingga peluang *redditors* akan semakin besar dalam menghasilkan konten dan memberikan tanggapan yang ditanggapi dengan aksi *upvote*.

Multilabel classification merupakan sebuah *task* prediktif pada *data mining* yang memungkinkan untuk mengasosiasikan sebuah data kedalam beberapa label. Setiap konten dan komentar yang ada pada reddit akan diberikan label dan bisa saja setiap konten dan komentar memiliki label lebih dari satu. Adapun label pada *dataset* tersebut digunakan untuk mempermudah proses klasifikasi. Terdapat beberapa metode klasifikasi yang dapat digunakan dalam klasifikasi multilabel data, namun pada proyek ini akan digunakan metode klasifikasi *Support Vector Machine* (SVM). Pada dasarnya SVM digunakan untuk menentukan *decision boundary* atau *hyperplane* antar dua kelas, sehingga akan menjadi tantangan apabila diimplementasikan pada *multilabel classification* karena secara native cara kerja pengklasifikasian SVM berbeda dengan klasifikasi multilabel. Sehingga, untuk mengatasi tantangan tersebut dapat digunakan NB-SVM *(Naive Bayes - Suport Vector Machine)* dengan *Logistic Regression* oleh *machine learning open source library* berbasis *python* yaitu Scikit-learn. Strategi ini merupakan pengklasifikasian multi target yang dapat digunakan untuk mendukung klasifikasi multilabel dengan SVM.

Oleh karena itu, tim proyek memutuskan untuk mengangkat proyek dengan judul "*Analysis Emotion Of Comment Topic Reddit With Multi-label Classification Using SVM Algorithm*". Proyek diharapkan dapat berkontribusi memecahkan tantangan *multilabel classification* menggunakan SVM, menganalisis emosi pada komentar reddit dengan *multilabel classification* menggunakan metode *Support Vector Machine* (SVM) *algorithm,* dan melakukan pengujian akurasi terhadap metode tersebut.

1.2 Tujuan

Tujuan dari pelaksanaan proyek ini adalah:

1. Menerapkan metode SVM dalam menganalisis emosi pada komentar di reddit dengan *multilabel classification*
2. Untuk mengetahui bagaimana tingkat akurasi menggunakan metode SVM dalam melakukan *emotion analysis*.

1.3 Manfaat

Manfaat dari pelaksanaan proyek ini adalah:

1. Bagi *redditors*
   Proyek ini diharapkan dapat menjadi bahan evaluasi bagi para *redditors* dalam membagikan konten yang lebih baik.
2. Bagi Mahasiswa
   Proyek ini diharapkan dapat memberi wawasan bagi tim proyek dalam menerapkan pemrosesan bahasa alami tepatnya SVM *algorithm* pada data teks reddit.

1.4 Ruang Lingkup

Ruang lingkup dari pelaksanaan proyek ini adalah:

1. Dataset yang digunakan adalah dataset g0_emotions yang berasal dari Hugging Face yang berisi 58000 komentar reddit dengan label sebanyak 27 kategori.
2. Metode yang digunakan dalam pengerjaan proyek *multilabel classification* adalah *Support Vector Machine* (SVM).

BAB 2
ISI

Bab ini berisikan penjelasan mengenai hasil analisis tim proyek terhadap data dan metode yang digunakan, desain pemrosesan bahasa alami yang disajikan dalam bentuk *flowchart*, implementasi kode program beserta keluarannya, dan hasil evaluasi yang dilakukan.

2.1 Analisis

Sub bab ini berisikan penjelasan mengenai hasil tim proyek terhadap data dan metode yang digunakan.

2.1.1 Analisis Data

Proyek ini menggunakan *dataset* yang bersumber dari tautan berikut: https://huggingface.co/datasets/go_emotions. *Dataset* GoEmotions ini berisikan kumpulan data beranotasi manual terbesar yaitu terdiri dari 58.000 *Reddit English comments*, yang dikategorikan kedalam 27 label kategori emosi dan sebuah label netral. *Dataset* GoEmotions ditujukan untuk keperluan pendalaman *multi-class, multi-label emotion classification.* Pada proyek ini data akan dibagi menjadi 20% *data test* dan 80% *data train.*

Table 1 Atribut Dataset

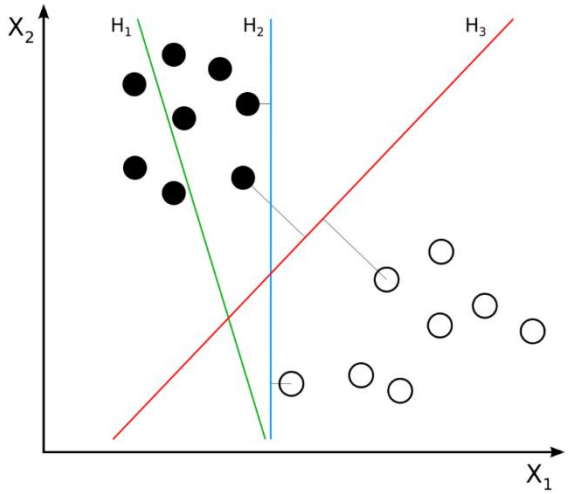
No.	Nama Atribut	Tipe Atribut	Deskripsi
1.	text	string	Teks komentar yang diposting
2.	id	string	Id komentar yang diposting
3.	author	string	Penulis komentar
4.	subreddit	string	Topik khusus dalam komentar
5.	link_id	string	Id link dari komentar
6.	parent_id	string	Id parent dari komentar
7.	created_utc	float	label timestamp dari komentar
8.	rater_id	bool	Id unik dari pembuat catatan
9.	example_very_unclear	boolean	Label yang menandakan bahwa komentar sangat sulit untuk diberi label oleh annotator (pembuat catatan)
10.	admiration	integer	Kategori ekspresi yang menunjukkan kekaguman

11.	amusement	integer	Kategori ekspresi yang menunjukkan sesuatu yang lucu dan menghibur
12.	anger	integer	Kategori ekspresi yang menunjukkan perasaan tidak senang dan antagonisme
13.	annoyance	integer	Kategori ekspresi yang menunjukkan perasaan terganggu dan jengkel
14.	approval	integer	Kategori ekspresi yang mengungkapkan pendapat setuju
15.	caring	integer	Kategori ekspresi yang mengungkapkan kepedulian terhadap orang lain
16.	confusion	integer	Kategori ekspresi yang menunjukkan kebingungan dan ketidakpahaman
17.	curiosity	integer	Kategori ekspresi yang menunjukkan keingintahuan terhadap sesuatu
18.	desire	integer	Kategori ekspresi yang menunjukkan keinginan untuk mengharapkan sesuatu terjadi
19.	disappointment	integer	Kategori ekspresi yang menunjukkan kekecewaan oleh tidak terpenuhinya suatu harapan
20.	disapproval	integer	Kategori ekspresi yang mengungkapkan pendapat yang tidak setuju
21.	disgust	integer	Kategori ekspresi yang menunjukkan rasa jijik oleh sesuatu yang tidak menyenangkan atau menyinggung
22.	embarrassment	integer	Kategori ekspresi yang menunjukkan rasa sadar diri, malu dan canggung
23.	excitement	integer	Kategori ekspresi yang menunjukkan rasa kegembiraan dan antusiasme yang besar
24.	fear	integer	Kategori ekspresi yang menunjukkan rasa takut/khawatir

25.	gratitude	integer	Kategori ekspresi yang menunjukkan rasa bersyukur
25.	grief	integer	Kategori ekspresi yang menunjukkan rasa kesedihan mendalam
27.	joy	integer	Kategori ekspresi yang menunjukkan rasa suka cita
28.	love	integer	Kategori ekspresi yang menunjukkan rasa kasih sayang
29.	nervousness	integer	Kategori ekspresi yang menunjukkan rasa gugup/khawatir/cemas
30.	optimism	integer	Kategori ekspresi yang menunjukkan rasa berharap pada masa depan/keberhasilan sesuatu
31.	pride	integer	Kategori ekspresi yang menunjukkan rasa puas atas pencapaian prestasi
32.	realization	integer	Kategori ekspresi yang menunjukkan rasa menyadari sesuatu
33.	relief	integer	Kategori ekspresi yang menunjukkan rasa lega dari kecemasan
34.	remorse	integer	Kategori ekspresi yang menunjukkan rasa bersalah dan menyesal
35.	sadness	integer	Kategori ekspresi yang menunjukkan sakit, sedih
36.	surprise	integer	Kategori ekspresi yang menunjukkan rasa heran, terkejut oleh sesuatu yang tidak terduga
37.	neutral	integer	Kategori ekspresi yang menunjukkan rasa netral atau biasa saja

2.1.2 Analisis Metode

Support Vector Machine (SVM) adalah kelas algoritma machine learning yang menggunakan fungsi kernel untuk mempelajari decision boundary atau batas keputusan antara dua kelas. Decision boundary ini adalah margin maksimum antara dua kelas, artinya batasan ini akan berjarak sama dari kelas satu dan dua. Penentuan *boundary* yang disebut juga sebagai *hyperplane* akan dilakukan dengan bantuan dari *support vectors*.

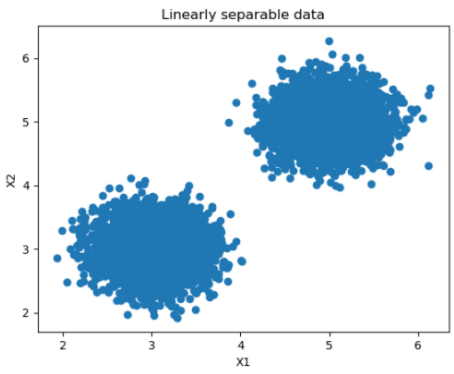


Gambar 1 Hyperplane SVM

Gambar 1 menunjukkan fitur dengan dua dimensi yaitu sumbu X1 dan X2 serta tiga buah garis (*hyperplane*) satu dimensi yaitu H1, H2, dan H3. Selanjutnya H2 dan H3 adalah *decision boundary* karena kedua garis ini dapat memisahkan satu kelas dengan kelas lainnya. Melalui gambar tersebut juga dapat diketahui bahwa H3 adalah *hyperplane* terbaik diantara *hyperplane* yang lainnya, karena H3 memisahkan kedua kelas dengan jarak/margin maksimum yang sama sehingga H3 memiliki margin tertinggi antara kelas dibandingkan dengan *hyperplane* lainnya.

Sayangnya, SVM kurang mampu mengatasi klasifikasi multilabel karena decision boundary pada SVM hanya mampu memisahkan dua kelas untuk memastikan margin maksimumnya. Jika kelas lain ditambahkan pada klasifikasi SVM maka jarak antara kelas menjadi tidak maksimal. Namin, keterbatasan tersebut dapat diatasi dengan menggunakan *MultiOutputClassifier* SVM yang disediakan oleh *machine learning open source library* berbasis *python* yaitu Scikit-learn. Penerapan SVM *MultiOutputClassifier* dengan Scikit-learn Scikit-learn menyediakan fungsionalitas *MultiOutputClassifier*, yang mampu mengimplementasikan pengklasifikasi multilabel untuk pengklasifikasi reguler apa pun. Strategi ini merupakan pengklasifikasian multi target yang dapat digunakan untuk mendukung klasifikasi multilabel dengan SVM. Berikut uraian *line scenario*-nya:

1. Bagi dataset secara acak menjadi dua bagian. Asumsikan data dapat dibagi secara linear.

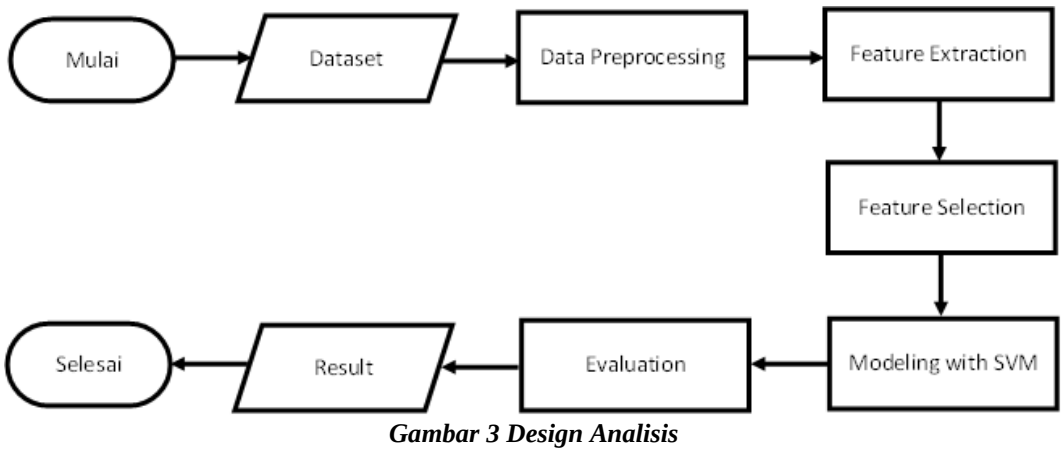


Gambar 2 Data Linearly Separable

2. Lalu, pastikan bahwa semua dependensi diimpor seperti pyplot API dari Matplotlib untuk memvisualisasikan hasil. Dalam hal ini Numpy akan digunakan pada beberapa pemrosesan angka, dan beberapa dependensi sklearn akan diimpor lagi sesuai kebutuhan. Secara khusus *make_blobs* dapat digunakan dalam pembuatan data, *MultiOutputClassifier* untuk pengklasifikasi multilabel, LinearSVC untuk SVM , *train_test_split* untuk membagi data menjadi set pelatihan dan pengujian, hingga *multilabel_confusion_matrix* dan *ConfusionMatrixDisplay* untuk menghasilkan dan memvisualisasikan matriks konfusi.
3. Tentukan opsi-opsi konfigurasi, seperti jumlah sampel yang akan dihasilkan, pusat *cluster*, dan jumlah kelas.
4. Pada tahap ini, telah dihasilkan data yang sesuai dengan spesifikasi diatas. Buatlah array dengan bentuk yang sama untuk label kedua – warna. Inisialisasi secara acak demi kesederhanaan.
5. Gabungkan label pelatihan menjadi satu larik sehingga menghasilkan pemisahan antara data pelatihan dan pengujian.
6. Inisialisasi classifier SVM dan ubah menjadi multilabel.
7. .fit data ke *classifier*, yang artinya proses pelatihan dimulai. Setelah pemasangan selesai, *classifier* terlatih tersedia di *multilabel_classifier*. Kini .predict telah dapat dipanggil untuk menghasilkan prediksi untuk data uji.
8. Bandingkan *y_test* (label kebenaran dasar yang sebenarnya) dan *y_test_pred* (label yang diprediksi) menggunakan matriks konfusi (mengikuti langsung setelah segmen kode).
9. Buatlah matriks konfusi untuk setiap label dengan *multilabel_confusion_matrix*, dan kemudian memplotnya dengan *ConfusionMatrixDisplay* menggunakan Matplotlib.

2.2 Design

Pada sub bab ini dijelaskan desain pemrosesan bahasa alami dari *Analysis Emotion Of Comment Topic Reddit With Multi-label Classification Using SVM* Algorithm yang dapat dilihat pada gambar 3.



Gambar 3 Design Analisis

2.2.1 Data Preprocessing

Pada tahap ini dilakukan pra-pemrosesan data dengan mengolah data pada dataset yang masih terdapat noise, tidak lengkap dan tidak konsisten serta menghilangkan

kata kata yang tidak diperlukan.Teknik pra-pemrosesan data dapat meningkatkan kualitas data, sehingga membantu meningkatkan akurasi dan efisiensi [4].

2.2.1.1 Data Cleaning

Pada tahap ini dilakukan pembersihan terhadap data yang tidak lengkap, *noise*, dan data yang tidak konsisten. *Data cleaning* akan mencoba untuk mengisi data-data yang tidak lengkap, menghaluskan *noise* saat mengidentifikasi *outlier*, dan melakukan perbaikan pada data-data yang tidak konsisten [4].

2.2.1.2 Case Folding

Case folding merupakan proses yang dilakukan untuk menyeragamkan karakter pada data. Proses *case folding* adalah proses mengubah seluruh huruf menjadi huruf kecil. Pada proses ini karakter-karakter 'A'-'Z' yang terdapat pada data diubah kedalam karakter 'a'-'z'. Karakter-karakter selain huruf 'a' sampai 'z' (tanda baca dan angka-angka) akan dihilangkan dari data dan dianggap sebagai delimiter [5].

2.2.1.3 Tokenization

Pada tahap ini dilakukan pemecahan kalimat menjadi kata-kata sehingga membentuk token yang bermakna. Strategi umum yang dilakukan pada tahap tokenizing adalah memotong kata pada white space /spasi dan membuang karakter tanda baca. Tahap tokenizing membagi urutan karakter menjadi kalimat dan kalimat menjadi token.

2.2.1.4 Stopwords Removal

Setelah tahap *tokenization*, maka dilakukan tahap filtering yaitu dengan menghapus kata-kata yang sangat umum (*stopwords removal*). Adapun kata yang termasuk dalam *stopword* contohnya adalah "this", "there", "the", "a", "of" dan lain lain .

2.2.1.5 Stemming

Pada tahap ini dilakukan pembersihan terhadap imbuhan awalan dan akhiran dari sebuah kata [6]. Adapun imbuhan yang dibersihkan yaitu "ss", "ies", "ing", "ed", "ate", "ize", "able", "al", dan lain-lain. Contohnya "*automates", "automatic", "automation*" di *stemming* menjadi "*automat*".

2.2.1.6 Lemmatization

Pada tahap ini mirip dengan *stemming*, namun dilakukan pencocokan dengan lemma-lemma yang terdapat pada kamus sehingga hasil akan lebih akurat [6]. Adapun imbuhan yang dibersihkan yaitu "-s", "-'s", "am", "are", "is" dan lain-lain.

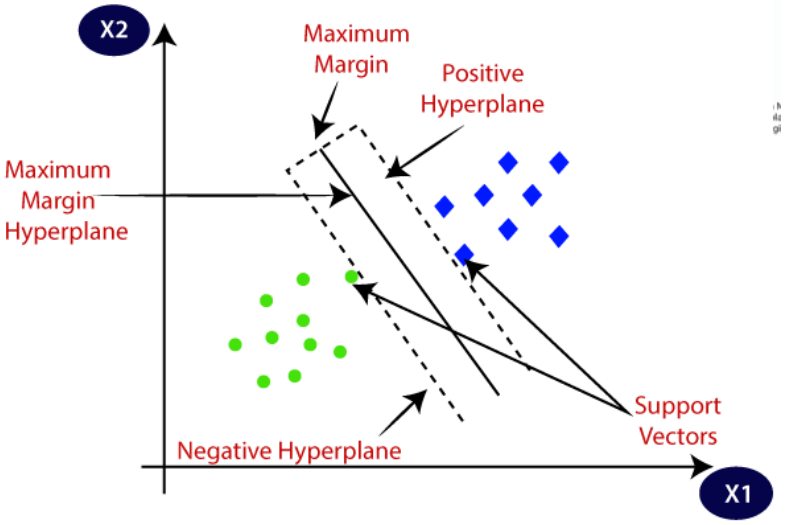
2.2.2 Feature Extraction

Feature Extraction adalah proses mengubah dokumen teks dari format apa pun menjadi daftar fitur yang dapat dengan mudah diproses dengan teknik klasifikasi teks. *Feature extraction* akan sangat mendukung dalam menyediakan fitur yang baik bagi *training model*. Cara yang akan digunakan untuk melatih model tekstual adalah dengan TF IDF Vectorization dan Word2Vec.

2.2.3 Modelling with SVM

Setelah *feature* berhasil diseleksi, maka langkah selanjutnya adalah melakukan *train* pada dataset dengan menggunakan algoritma *Support Vector Machine* (SVM). SVM adalah salah satu metode pada *supervised learning*, modeling dengan

algoritma SVM bekerja dengan membangun *hyperplane* yaitu fungsi pemisah antar kelas dengan margin maksimal. Pengklasifikasian dengan SVM menghasilkan data yang akurat meskipun data yang digunakan kurang seimbang (*imbalance*) [7].



Gambar 4 Support Vector Machine (SVM)

Teknik pemodelan dengan SVM akan menggunakan 4 cara yaitu:

- SVM Model with TF IDF
- SVM Model with Word2Vec
- SVM Model with K-Fold Cross Validation
- Linear SVM Classifier

2.2.4 Evaluation and Result

Setelah *modelling with SVM* selesai, selanjutkan akan dilakukan evaluasi terhadap model yang dibangun menggunakan *confusion matrix. Confusion matrix* akan memberikan perbandingan dari hasil klasifikasi model dengan hasil aktual klasifikasi. Berikut adalah tabel *confusion matrix:*

		PREDICTED	
		TRUE	FALSE
ACTUAL	TRUE	TP	FN
	FALSE	FP	TN

Gambar 5 Confusion Matrix

Hasil klasifikasi dipresentasikan dalam 4 istilah yaitu:

- TP adalah *true positive* atau data positif yang diprediksi benar
- FN adalah *false negatif* atau data positif yang diprediksi sebagai data negatif
- FP adalah *false positif* atau data negatif yang diprediksi sebagai data positif
- TN adalah *true negative* atau data negatif yang diprediksi benar

Pengukuran *performance metrics* dari *confusion matrix* akan dilakukan mulai dari melakukan perhitungan terhadap *accuracy score*, *precision*, *recall* dan *F1-Score.* berikut penjelasannya:

1. *Accuracy score* untuk mengetahui tingkat keakuratan klasifikasi yang dihasilkan model

2. *Precision* untuk mengetahui tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model.

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

3. *Recall* untuk mengetahui tingkat keberhasilan model dalam menemukan kebali suatu informasi.

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

2.3 Implementasi

Sub bab ini berisikan penjelasan mengenai implementasi tim proyek terhadap data dan metode yang digunakan.

2.3.1 Data Preprocessing

Data preprocessing akan meningkatkan kualitas data, sehingga membantu meningkatkan akurasi dan efisiensi. Pada tahap ini akan dilakukan *data cleaning, case folding, tokenization, stopwords removal, stemming,* dan *lemmatization.* Namun sebelumnya, dilakukan *import library, import* dataset, pemeriksaan deskripsi dataset, dan pemeriksaan jumlah *row* dataset. Berikut adalah kode program *import library:*

```
import pandas as pd
from pandas import DataFrame as df
```

Berikut adalah kode program *import dataset:*

```
df_go1 = pd.read_csv("goemotions_1.csv")
df_go2 = pd.read_csv("goemotions_2.csv")
df_go3 = pd.read_csv("goemotions_3.csv")

df_go_emotion = pd.concat([df_go1, df_go2, df_go3])
```

Berikut adalah kode program pemeriksaan deskripsi dataset:

```
df.describe(df_go_emotion)
```

Berikut adalah kode program pemeriksaan jumlah *row* dataset:

```
index = df_go_emotion.index
number_of_rows = len(index)
print(number_of_rows)
```

Output kode program diatas menunjukkan bahwa *dataset* terdiri atas 211225 *rows.*

2.3.1.1 Data Cleaning

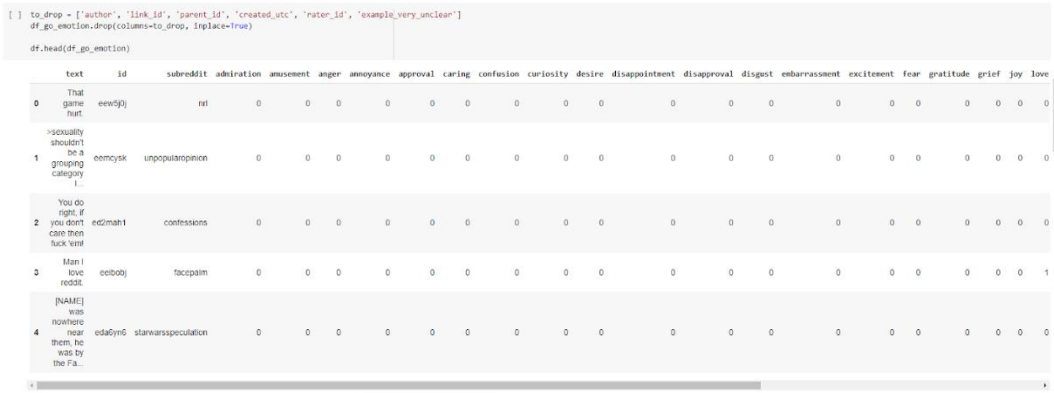
Data cleaning akan mencoba untuk mengisi data-data yang tidak lengkap, menghaluskan *noise* dan melakukan perbaikan pada data-data yang tidak konsisten. Pada tahap ini akan dilakukan *drop column* terhadap kolom yang tidak dibutuhkan, pengecekan terhadap terhadap *data null,* dan identifikasi terhadap *dataset id* yang

unik, apabila teridentifikasi tidak akan dilakukan penghapusan terhadap nilai yang sama pada *dataset*. Berikut adalah kode program untuk melakukan *drop column* terhadap kolom yang tidak dibutuhkan:

```
to_drop = ['author', 'link_id', 'parent_id', 'created_utc',
'rater_id', 'example_very_unclear']
df_go_emotion.drop(columns=to_drop, inplace=True)

df.head(df_go_emotion)
```

Output kode program diatas adalah sebagai berikut:



Berikut adalah kode program untuk melakukan pengecekan terhadap terhadap *data null:*

```
df_go_emotion.isna()
```

Output kode program diatas adalah sebagai berikut:

Berikut adalah kode program untuk mengidentifikasi id yang unik:

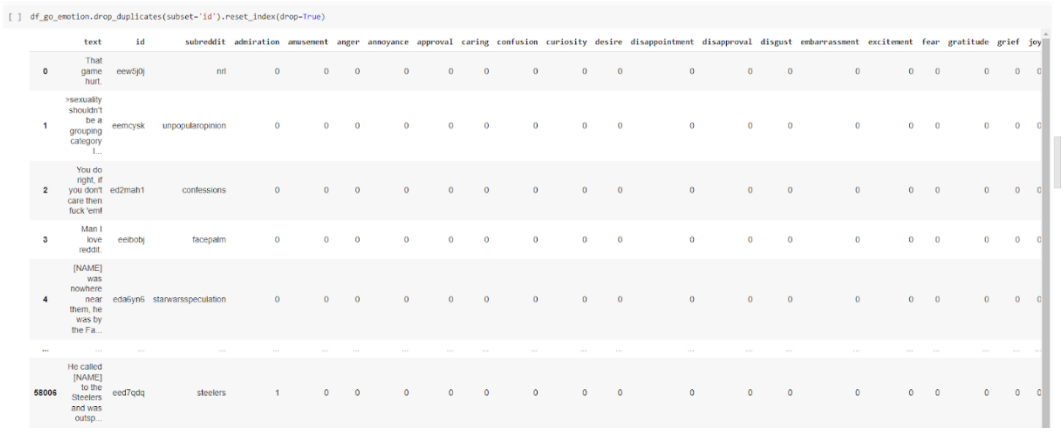
```
df_go_emotion['id'].is_unique
```

Output kode program diatas adalah *False.*

Berikut adalah kode program untuk menghapus nilai yang sama pada *dataset:*

```
df_go_emotion.drop_duplicates(subset='id').reset_index(drop=True)
```

Output kode program sebagai berikut:



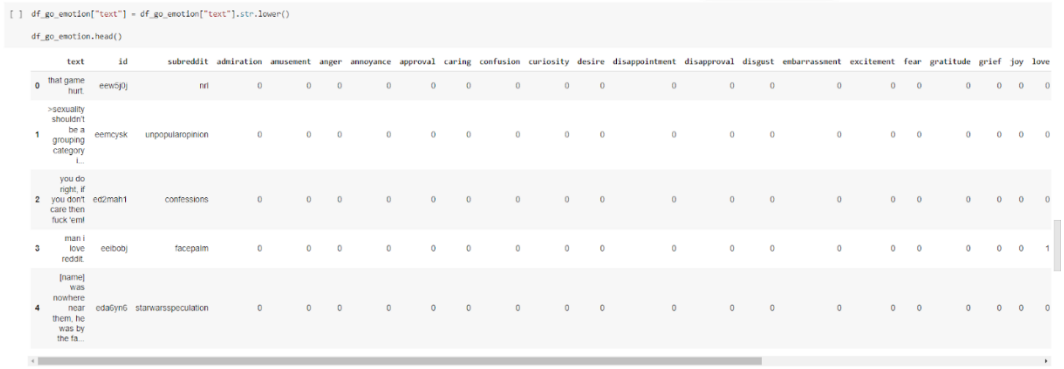
2.3.1.2 Case Folding

Case folding akan mengubah semua teks komentar menjadi *lowercase format.* Berikut adalah kode program untuk melakukan *lowercase formatting*:

```
df_go_emotion["text"] = df_go_emotion["text"].str.lower()

df_go_emotion.head()
```

Output kode program diatas adalah sebagai berikut:



2.3.1.3 Removal of Punctuations

Removal of punctuations akan mengidentifikasi dan menghapus *punctuation* pada teks komentar. *Punctuation* yang dimaksud adalah seperti "!()-[]{};:'"\,<>./?@#$%^&*_~". Berikut adalah kode program untuk melakukan *removal of punctuations*:

```
PUNCT_TO_REMOVE = string.punctuation
def remove_punctuation(text):
    """custom function to remove the punctuation"""
    return text.translate(str.maketrans('', '',
PUNCT_TO_REMOVE))

df_go_emotion["text"] = df_go_emotion["text"].apply(lambda
text: remove_punctuation(text))

df_go_emotion.head()
```

Output kode program diatas adalah sebagai berikut:

```
[ ] PUNCT_TO_REMOVE = string.punctuation
    def remove_punctuation(text):
        """custom function to remove the punctuation"""
        return text.translate(str.maketrans('', '', PUNCT_TO_REMOVE))

    df_go_emotion["text"] = df_go_emotion["text"].apply(lambda text: remove_punctuation(text))

    df_go_emotion.head()
```

	text	id	subreddit	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	disappointment	disapproval	disgust	embarrassment	excitement	fear	gratitude	grief	joy	love
0	that game hurt	eew5j0j	nrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	sexuality shouldn't be a grouping category It...	eemcysk	unpopularopinion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	you do right if you dont care then fuck em	ed2mah1	confessions	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	man i love reddit	eeibobj	facepalm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	name was nowhere near them he was by the falcon	eda6yn6	starwarsspeculation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3.1.4 Stopwords Removal

Stopwords removal akan menghilangkan kata-kata seperti “this”, “there”, “the”, “a”, “of”, “with”, “have”, dan lain sebagainya. Berikut adalah kode program untuk melakukan *stopwords removal:*

```
from nltk.corpus import stopwords
#nltk.download('stopwords') #in case stopwords is undefined

STOPWORDS = set(stopwords.words('english'))
def remove_stopwords(text):
    """custom function to remove the stopwords"""
    return " ".join([word for word in str(text).split() if
word not in STOPWORDS])

df_go_emotion["text"] = df_go_emotion["text"].apply(lambda
text: remove_stopwords(text))

df_go_emotion.head()
```

Output kode program diatas adalah sebagai berikut:

```
[ ] from nltk.corpus import stopwords
    nltk.download('stopwords') #in case stopwords is undefined

    STOPWORDS = set(stopwords.words('english'))
    def remove_stopwords(text):
        """custom function to remove the stopwords"""
        return " ".join([word for word in str(text).split() if word not in STOPWORDS])

    df_go_emotion["text"] = df_go_emotion["text"].apply(lambda text: remove_stopwords(text))

    df_go_emotion.head()

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

	text	id	subreddit	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	disappointment	disapproval	disgust	embarrassment	excitement	fear	gratitude	grief	joy	love
0	game hurt	eew5j0j	nrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	sexuality shouldnt grouping category makes di...	eemcysk	unpopularopinion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	right dont care fuck em	ed2mah1	confessions	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	man love reddit	eeibobj	facepalm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	name nowhere near falcon	eda6yn6	starwarsspeculation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3.1.5 Stemming

Stemming akan menghilangkan imbuhan pada setiap kata untuk memperoleh *stem* atau kata dasar pada teks komentar. Berikut adalah kode program untuk melakukan *stemming*:

```
from nltk.stem.porter import PorterStemmer

stemmer = PorterStemmer()
def stem_words(text):
    return " ".join([stemmer.stem(word) for word in
text.split()])

df_go_emotion["text"] = df_go_emotion["text"].apply(lambda
text: stem_words(text))

df_go_emotion.head()
```

Output kode program diatas adalah sebagai berikut:

	text	id	subreddit	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	disappointment	disapproval	disgust	embarrassment	excitement	fear	gratitude	grief	joy	love
0	game hurt	eew5j0j	nrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	sexual shouldn't group categori make differ ot...	eemcysk	unpopularopinion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	right dont care fuck em	ed2mah1	confessions	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	man love reddit	eelbobj	facepalm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	name nowher near falcon	eda6yn6	starwarsspeculation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3.1.6 Lemmatization

Lemmatization akan mengubah kata menjadi bentuk dasarnya sesuai dengan kata-kata yang terdapat dalam kamus. Berikut adalah kode program untuk melakukan *lemmatization*:

```
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')

lemmatizer = WordNetLemmatizer()
def lemmatize_words(text):
    return " ".join([lemmatizer.lemmatize(word) for word in
text.split()])

df_go_emotion["text"] = df_go_emotion["text"].apply(lambda
text: lemmatize_words(text))

df_go_emotion.head()
```

Output kode program diatas adalah sebagai berikut:

```
[ ] from nltk.stem import WordNetLemmatizer
    nltk.download('wordnet')

    lemmatizer = WordNetLemmatizer()
    def lemmatize_words(text):
        return " ".join([lemmatizer.lemmatize(word) for word in text.split()])

    df_go_emotion["text"] = df_go_emotion["text"].apply(lambda text: lemmatize_words(text))

    df_go_emotion.head()
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

	text	id	subreddit	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	disappointment	disapproval	disgust	embarrassment	excitement	fear	gratitude	grief	joy	love
0	game hurt	eew5j0j	nrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	sexual shouldn't group categori make differ ot...	eemcysk	unpopularopinion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	right dont care fuck em	ed2mah1	confessions	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	man love reddit	eeibobj	facepalm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	name nowher near falcon	eda6yn6	starwarsspeculation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

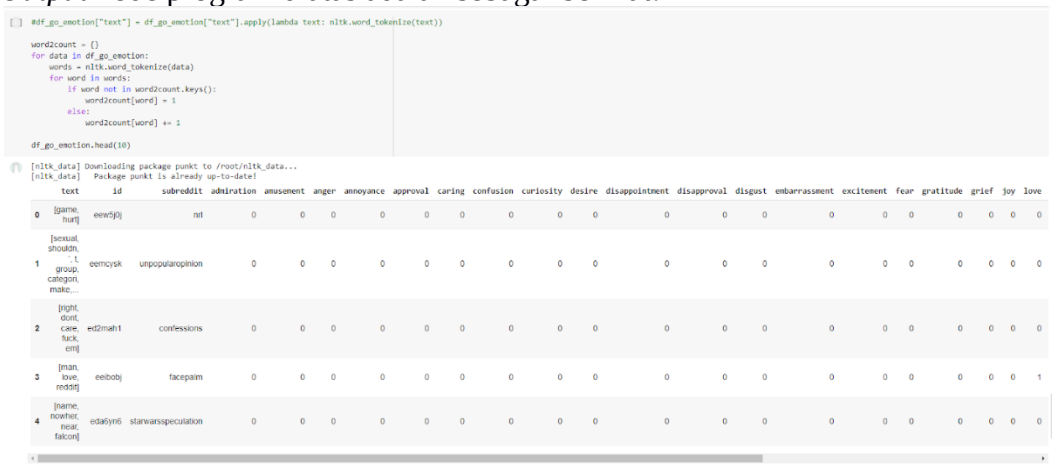
2.3.1.7 Tokenization

Tokenization akan memecah teks komentar menjadi unit yang relevan, seperti *words*, *symbol*, *phares*, ataupun bentuk token lainnya. Berikut adalah kode program untuk melakukan *tokenization*:

```
word2count = {}
for data in df_go_emotion:
    words = nltk.word_tokenize(data)
    for word in words:
        if word not in word2count.keys():
            word2count[word] = 1
        else:
            word2count[word] += 1

df_go_emotion.head(10)
```

Output kode program diatas adalah sebagai berikut:



```
[ ] #df_go_emotion["text"] = df_go_emotion["text"].apply(lambda text: nltk.word_tokenize(text))

    word2count = {}
    for data in df_go_emotion:
        words = nltk.word_tokenize(data)
        for word in words:
            if word not in word2count.keys():
                word2count[word] = 1
            else:
                word2count[word] += 1

    df_go_emotion.head(10)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

	text	id	subreddit	admiration	amusement	anger	annoyance	approval	caring	confusion	curiosity	desire	disappointment	disapproval	disgust	embarrassment	excitement	fear	gratitude	grief	joy	love
0	[game, hurt]	eew5j0j	nrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	[sexual, shouldn, ', t, group, categori, make,...	eemcysk	unpopularopinion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	[right, dont, care, fuck, em]	ed2mah1	confessions	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	[man, love, reddit]	eeibobj	facepalm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	[name, nowher, near, falcon]	eda6yn6	starwarsspeculation	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.3.2 Feature Extraction

Feature extraction akan dilakukan terhadap teks komentar menggunakan TF IDF. Berikut adalah kode program untuk melakukan *feature extraction*:

```
# tf-idf
from sklearn.feature_extraction.text import CountVectorizer
```

```
from sklearn.feature_extraction.text import TfidfTransformer

def do_nothing(tokens):
    return tokens

count_vect = CountVectorizer(tokenizer=do_nothing,
lowercase=False)
df_go_emotion_tf =
count_vect.fit_transform(df_go_emotion["text"])
tfidf_transformer = TfidfTransformer()
df_go_emotion_tfidf =
tfidf_transformer.fit_transform(df_go_emotion_tf)

print(df_go_emotion_tfidf)
```

Output kode program diatas merepresentasikan *document index*, *word vector index*, dan *tf idf score* sebagai berikut:

2.3.4 Modelling with SVM

Modeling with SVM, merupakan tahap pemodelan terhadap *sentiment analysis* yang akan dilakukan. Dalam menerapkan SVM *algorithm*, tahapan yang dilakukan adalah sebagai berikut:

1. Membagikan dataset menjadi 80% data *train* dan 20% data *test*:

```
from sklearn.model_selection import train_test_split

df = df_go_emotion

train, test = train_test_split(df, test_size=0.2)

to_drop = ['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']
```

```
test.drop(columns=to_drop, inplace=True)

train.to_csv('go_emotion_train.csv', index=False)
test.to_csv('go_emotion_test.csv', index=False)
```

2. Menerapkan fungsi tokenisasi terhadap data yang digunakan:

```
import re, string
re_tok =
re.compile(f'([{string.punctuation}‘’¨«»®´·º½¾¿¡§£₤’])')

def tokenize(s):
    return re_tok.sub(r' \1 ', s).split()
```

3. Menerapkan teknik TF-IDF pada model SVM:

```
n = train.shape[0]
vec = TfidfVectorizer(ngram_range=(1,2), tokenizer=tokenize,
               min_df=3, max_df=0.9, strip_accents='unicode',
use_idf=1,
               smooth_idf=1, sublinear_tf=1)
trn_term_doc = vec.fit_transform(train[COMMENT])

test_term_doc = vec.transform(test[COMMENT])
```

Output dari TF-IDF berupa sparse matrix dapat dilihat dengan menggunakan fungsi berikut:

```
trn_term_doc, test_term_doc
```

Berikut adalah *output* dari kode program TF-IDF diatas:

```
(<168980x227321 sparse matrix of type '<class 'numpy.float64'>'
        with 4905348 stored elements in Compressed Sparse Row format>,
 <42245x227321 sparse matrix of type '<class 'numpy.float64'>'
        with 1161728 stored elements in Compressed Sparse Row format>)
```

4. Menerapkan pemodelan dengan NB-SVM dan Logistic Regression:

```
def pr(y_i, y):
    p = x[y==y_i].sum(0)
    return (p+1) / ((y==y_i).sum()+1)

x = trn_term_doc
test_x = test_term_doc

#Modeling
def get_mdl(y):
    y = y.values
    r = np.log(pr(1,y) / pr(0,y))
    m = LogisticRegression(C=4, dual=True, solver='liblinear')
    x_nb = x.multiply(r)
    return m.fit(x_nb, y), r

preds = np.zeros((len(test), len(label_cols)))
```

```
for i, j in enumerate(label_cols):
    print('fit', j)
    m,r = get_mdl(train[j])
    preds[:,i] = m.predict_proba(test_x.multiply(r))[:,1]
```

Kode program pemodelan dengan NB-SVM dan *Logistic Regression* diatas menghasilkan nilai *fit* dari setiap label data *test* ke data *train* berdasarkan label_cols.

5. Setelah model berhasil dibangun, maka label data dapat di assign sebagai berikut:

```
submid = pd.DataFrame({'id': subm["id"]})
submission = pd.concat([submid, pd.DataFrame(preds, columns =
label_cols)], axis=1)

submission.loc[submission['admiration'] > 0.5, 'admiration'] = 1
submission.loc[submission['amusement'] > 0.5, 'amusement'] = 1
submission.loc[submission['anger'] > 0.5, 'anger'] = 1
submission.loc[submission['annoyance'] > 0.5, 'annoyance'] = 1
submission.loc[submission['approval'] > 0.5, 'approval'] = 1
submission.loc[submission['caring'] > 0.5, 'caring'] = 1
submission.loc[submission['confusion'] > 0.5, 'confusion'] = 1
submission.loc[submission['curiosity'] > 0.5, 'curiosity'] = 1
submission.loc[submission['desire'] > 0.5, 'desire'] = 1
submission.loc[submission['disappointment'] > 0.5,
'disappointment'] = 1
submission.loc[submission['disapproval'] > 0.5, 'disapproval'] =
1
submission.loc[submission['disgust'] > 0.5, 'disgust'] = 1
submission.loc[submission['embarrassment'] > 0.5,
'embarrassment'] = 1
submission.loc[submission['excitement'] > 0.5, 'excitement'] = 1
submission.loc[submission['fear'] > 0.5, 'fear'] = 1
submission.loc[submission['gratitude'] > 0.5, 'gratitude'] = 1
submission.loc[submission['grief'] > 0.5, 'grief'] = 1
submission.loc[submission['joy'] > 0.5, 'joy'] = 1
submission.loc[submission['love'] > 0.5, 'love'] = 1
submission.loc[submission['nervousness'] > 0.5, 'nervousness'] =
1
submission.loc[submission['optimism'] > 0.5, 'optimism'] = 1
submission.loc[submission['pride'] > 0.5, 'pride'] = 1
submission.loc[submission['realization'] > 0.5, 'realization'] =
1
submission.loc[submission['relief'] > 0.5, 'relief'] = 1
submission.loc[submission['remorse'] > 0.5, 'remorse'] = 1
submission.loc[submission['sadness'] > 0.5, 'sadness'] = 1
submission.loc[submission['surprise'] > 0.5, 'surprise'] = 1
submission.loc[submission['neutral'] > 0.5, 'neutral'] = 1

submission.loc[submission['admiration'] < 0.5, 'admiration'] = 0
submission.loc[submission['amusement'] < 0.5, 'amusement'] = 0
submission.loc[submission['anger'] < 0.5, 'anger'] = 0
submission.loc[submission['annoyance'] < 0.5, 'annoyance'] = 0
submission.loc[submission['approval'] < 0.5, 'approval'] = 0
submission.loc[submission['caring'] < 0.5, 'caring'] = 0
submission.loc[submission['confusion'] < 0.5, 'confusion'] = 0
```

```
submission.loc[submission['curiosity'] < 0.5, 'curiosity'] = 0
submission.loc[submission['desire'] < 0.5, 'desire'] = 0
submission.loc[submission['disappointment'] < 0.5,
'disappointment'] = 0
submission.loc[submission['disapproval'] < 0.5, 'disapproval'] =
0
submission.loc[submission['disgust'] < 0.5, 'disgust'] = 0
submission.loc[submission['embarrassment'] < 0.5,
'embarrassment'] = 0
submission.loc[submission['excitement'] < 0.5, 'excitement'] = 0
submission.loc[submission['fear'] < 0.5, 'fear'] = 0
submission.loc[submission['gratitude'] < 0.5, 'gratitude'] = 0
submission.loc[submission['grief'] < 0.5, 'grief'] = 0
submission.loc[submission['joy'] < 0.5, 'joy'] = 0
submission.loc[submission['love'] < 0.5, 'love'] = 0
submission.loc[submission['nervousness'] < 0.5, 'nervousness'] =
0
submission.loc[submission['optimism'] < 0.5, 'optimism'] = 0
submission.loc[submission['pride'] < 0.5, 'pride'] = 0
submission.loc[submission['realization'] < 0.5, 'realization'] =
0
submission.loc[submission['relief'] < 0.5, 'relief'] = 0
submission.loc[submission['remorse'] < 0.5, 'remorse'] = 0
submission.loc[submission['sadness'] < 0.5, 'sadness'] = 0
submission.loc[submission['surprise'] < 0.5, 'surprise'] = 0
submission.loc[submission['neutral'] < 0.5, 'neutral'] = 0

submission.to_csv('go_emotion_svm_result.csv', index=False)
```

2.4 Evaluasi Model

Pada sub bab ini berisikan hasil evaluasi kuantitatif yang dilakukan berdasarkan hasil modeling SVM *multilabel classification* menggunakan NB-SVM *(Naive Bayes - Suport Vector Machine)* dengan *Logistic Regression* oleh *machine learning open source library* berbasis *python.* Tahapan yang dilakukan adalah sebagai berikut:

1. Melakukan perhitungan confusion matrix terhadap 27 label emosi yang akan diprediksi. Kode program ini akan menghasilkan keluaran berupa matrix korelasi antara nilai hasil prediksi dan nilai aktual.

```
from sklearn.metrics import confusion_matrix

cf_matrix = confusion_matrix(y_test.values.argmax(axis=1),
y_pred.values.argmax(axis=1))

print(cf_matrix)
```

2. Melakukan perhitungan akurasi dari hasil prediksi menggunakan fungsi `accuracy_score()`. Keluaran dari kode ini adalah nilai akurasi dari model NB-SVM yang telah dibuat.

```
from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, y_pred)
```

```
print(accuracy)
```

Output :
0.04918925316605515

BAB 3
PENUTUP

3.1 Kesimpulan

Proyek ini melakukan analisis emosi terhadap topik komentar pada *platform* reddit dengan *multi-label classification* menggunakan algoritma SVM. *Data set* yang semula terbagi atas 3 dokumen berekstensi .csv digabungkan sehingga terdiri atas 211225 *rows.* Kemudian data dibersihkan dengan dilakukan pengecekan terhadap data *null,* menghapus nilai yang sama/*ununique,* dan memisahkan atribut yang tidak dibutuhkan. Setelah itu dilanjutkan dengan *text processing* dengan melakukan *case folding, removal of punctuation, stopwords removal, stemming,*dan *lemmatization.* Dilanjutkan kembali dengan melakukan *feature extraction* terhadap teks komentar menggunakan *TF IDF* dan *modeling.* Pada tahap modeling dataset dibagi menajdi 80% *data train* dan 20% data test, dan model dibangun dengan NB-SVM *(Naive Bayes - Suport Vector Machine)* dengan *Logistic Regression* oleh *machine learning open source library* berbasis *python* untuk menangani pengkalsifikasian multilabel. Namun, meskipun NB-SVM dapat menyelesaikan pengkalsifikasian multilabel tetapi nilai akurasi dari model yang dibangun sangat rendah yaitu hanya 0.04918925316605515. Rendahnya nilai akurasi ini disebabkan oleh label yang digunakan berjumlah cukup banyak yaitu 27 label, sehingga tingkat akurasi model yang dihasilkan sangat rendah.

3.2 Saran

Sub bab ini berisikan saran yang dapat dijadikan bahan pertimbangan dalam melakukan "*Analysis Emotion Of Comment Topic Reddit With Multi-label Classification Using SVM Algorithm*":

1. Pemodelan *muti-label classification* sebaiknya tidak diimplementasikan menggunakan SVM *algorithm* dikarenakan algoritma ini secara native tidak dapat menangani klasifikasi multi label.
2. Sebaiknya jumlah label dari dataset yang digunakan lebih diperhatikan, karena jumlah label yang sangat banyak akan mempengaruhi tingkat akurasi dari model yang dibangun.
3. Pada studi kasus multi label classification menggunakan 5-6 label, NB-SVM masih cocok untuk digunakan seperti yang telah dilakukan oleh Jeremy Howard pada *Toxic Comment Classification Challenge* menghasilkan nilai akurasi sebesar 97,72% [8]. Namun untuk studi kasus dengan jumlah label yang lebih banyak seperti pada penelitian ini tidak dianjurkan.

3.3 Rencana Kerja

3.3.1 Pembagian Tugas

Table 2 Pembagian Tugas

NIM	NAMA	KONTRIBUSI PEMBAGIAN TUGAS
12S18027	Christopher A Hutabarat	33,33%
12S18028	Alda G M Lumban Gaol	33,33%

12S18032	Sarah H M Siahaan	33,33%

Berikut tugas yang dilakukan oleh tim proyek:

- Data Analyst

  Mengidentifikasi dan menafsirkan data model kemudian melakukan analisis terhadap strategi yang efisien untuk diimplementasikan pada proyek.

- Programmer

  Merancang dan mengimplementasikan *code* untuk membangun sistem sesuai hasil analisis *data analyst.*

3.3.2 Jadwal Kerja

Table 3 Jadwal Kerja

No	Task	Minggu ke-			
		1	2	3	4
1.	Membuat Proposal Proyek - Analisis Latar Belakang - Pemilihan Metode - Presentasi Proposal				
2.	Desain - Data Preprocessing - Feature Extraction - Feature Selection - Modelling with SVM				
3.	Implementasi - Exploratory Data Analysis (EDA) - Data Preprocessing - Feature Extraction - Feature Selection - Modelling with SVM				
4.	Evaluasi Akurasi Model				
5.	Laporan Akhir dan Presentasi Akhir				

REFERENSI

[1] Anderson, "Ask me anything: what is Reddit?," *Library Hi Tech News*, vol. 32, pp. 8-11, 2015.

[2] S. Aman and S. S. , "Identifying expressions of emotion in text," in *International Conference on Text, Speech and Dialogue*, Springer, Berlin, Heidelberg, 2007.

[3] J. K. Rout, "A model for sentiment and emotion analysis of unstructured social media text," *Electron Commer Res* , no. 18, pp. 181-199, 2017.

[4] J. Han, M. Kamber and J. Pei, Data Mining Concpets and Techniques, Waltham: Morgan Kaufmann Publishers is an imprint of Elsevier, 2012.

[5] U. Hasanah, "An experimental study of text preprocessing techniques for automatic short answer grading in Indonesian," in *2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE)*, Yogyakarta, 2018.

[6] U. M, "Formation of SQL from Natural Language Query using NLP," in *International Conference on Computational Intelligence in Data Science (ICCIDS)*, Chennai, 2019.

[7] Y. Tang, "SVMs Modeling for Highly Imbalanced Classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 1, pp. 281-288, 2008.

[8] J. Howard, "NB-SVM strong linear baseline. Retrieved," Kaggle, 2018. [Online]. Available: https://www.kaggle.com/jhoward/nb-svm-strong-linear-baseline. [Accessed 29 November 2021].