BAB 3
METODE PENELITIAN

Penelitian ini adalah penelitian pengklasifikasian *tweet* dengan hasil klasifikasi siap dan tidak siap dari Twitter. Penelitian ini menggunakan algoritma SVM. Pengambilan data untuk penelitian ini berasal dari Twitter yang berkaitan dengan isu resesi tahun 2023 dan diolah menggunakan Google Colaboratory. Tahapan dalam penelitian ini, antara lain yaitu pengambilan data, *data preprocessing*, pelabelan data, *data training*, dan *data testing*. Berikut ini adalah bahan, alat, dan jalan penelitian untuk melakukan klasifikasi *tweet* terkait isu resesi 2023.

3.1 BAHAN DAN ALAT PENELITIAN

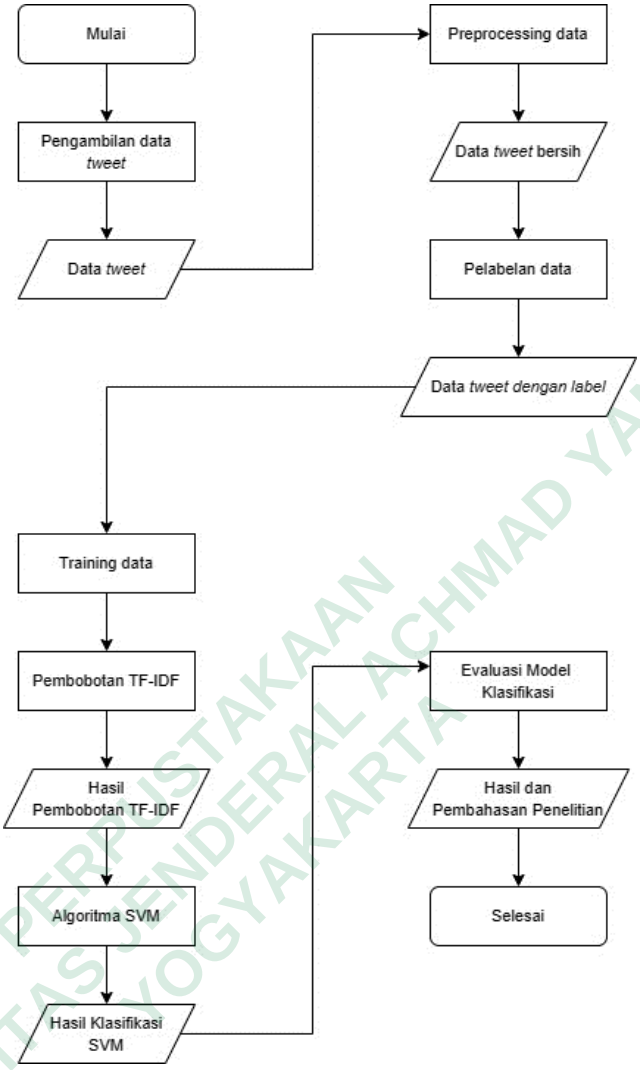
Pada penelitian ini, bahan yang digunakan ialah data *tweet* yang berhubungan dengan isu resesi tahun 2023 yang ada pada media sosial Twitter sebanyak 25.257 *tweet*.

Dan dalam prosesnya, menggunakan beberapa alat yaitu laptop dengan spesifikasi cukup untuk menjalankan sistem operasi dan perangkat lunak pengembangan serta koneksitas Internet. Sistem Operasi dan program-program aplikasi yang dipergunakan dalam dalam pengklasifikasian sentimen *tweet* ini adalah :

1. Sistem Operasi: Windows 11 64 bit.
2. Python version 3.9.13
3. Google Colaboratory
4. Sublime Text 4 Build 4143

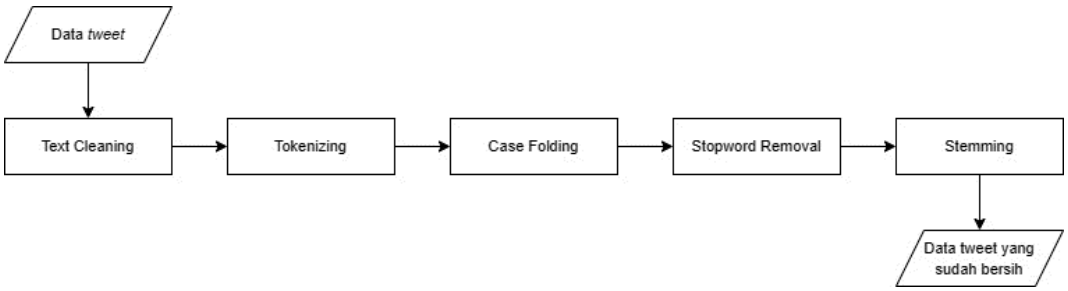
3.2 JALAN PENELITIAN

Penelitian ini menggunakan algoritma SVM untuk melakukan klasifikasi *tweet* terkait isu resesi tahun 2023. Adapun langkah-langkah penelitiannya dapat dilihat pada Gambar 3.1 :



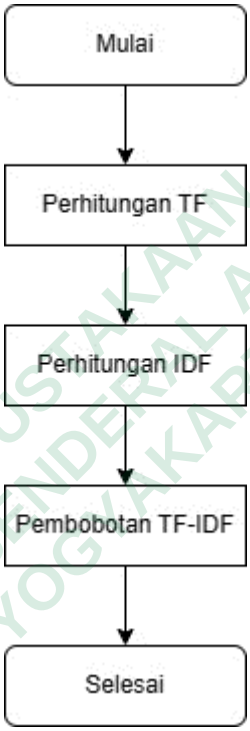
Gambar 3.1 Alur penelitian

Pada gambar 3.1 menampilkan alur penelitian klasifikasi sentimen ini. Dibawah ini merupakan alur untuk tahapan *preprocessing* yang dapat dilihat pada gambar 3.2



Gambar 3.2 Alur tahapan *preprocessing*

Pada gambar 3.2 di atas menampilkan tahapan yang dilakukan pada *preprocessing*, yaitu antara *lain text cleaning, tokenizing, case folding, stopword removal,* dan *stemming*. Setelah tahap *preprocessing* dilakukan, akan menghasilkan data *tweet* yang bersih. Setelah melakukan proses tersebut, dilakukanlah sebuah tahapan untuk perhitungan pembobotan TF-IDF, berikut merupakan alur untuk perhitungan TF-IDF dapat dilihat pada gambar 3.3



Gambar 3.3 Alur pembobotan TF-IDF

3.2.1 Pengambilan Data

Pengambilan data dilakukan dengan menggunakan Google Colab dengan teknik snscrape. Data yang diambil merupakan data yang berhubungan dengan Resesi tahun 2023 yang berasal dari twitter. Data *tweet* diambil dengan menggunakan keyword Resesi selama periode 01 September 2022 sampai dengan 31 Januari 2023

```
pip install snscrape
```

Kode di atas merupakan library untuk proses pengambilan data *tweet* dengan menginstall Snscrape menggunakan pip. Setelah terinstall, maka tahap selanjutnya yaitu melakukan pemanggilan librarynya.

```
import snscrape.modules.twitter as sntwitter
import pandas as pd
```

Kode di atas berfungsi untuk pemanggilan library dengan cara mengimport library tersebut. Hal ini dilakukan agar library tersebut dapat digunakan pada tahap selanjutnya.

```
maxTweets = 25000
tweets_list2 = []
for I, tweet in enumerate(sntwitter.TwitterSearchScraper
('Resesi since:2022-09-01 until:2023-01 lang:id').get_items()):
      if i>maxTweets:
            break
            tweets_list2.append([tweet.date, tweet.id,
tweet.rawContent, tweet.user.username])
```

Kode di atas digunakan untuk melakukan pengambilan data *tweet*. Fungsi `maxTweets` yaitu untuk mengatur panjang data yang akan diambil, dengan memasukkan angka maksimal pengambilan data yang diinginkan. Kemudian, fungsi `tweet in enumerate` digunakan untuk mencari kata kunci data beserta periode waktu data *tweet* yang akan diambil. Berikut kode program untuk menyimpan data *tweet* ke dalam csv.

```
tweets_df2.to_csv('dataresesi.csv', sep=',', index=False)
```

Kode di atas digunakan untuk melakukan ekspor data *tweet* kedalam file csv. Hasil data *tweet* yang telah berhasil dieskpor, dapat dilihat pada tabel 3.1

Tabel 3.1 Data *tweet*

No	Data Tweet
1.	Perbaikan kondisi ekonomi nasional di tengah pandemi Covid dan ancaman resesi global menunjukkan bahwa kebijakan pemerintah saat ini sudah dijalankan dg baik. Oleh karenanya, Erick Thohir mengajak semua pihak utk turut menjaga keberlangsungan ekonomi #BangkitBersamaET https://t.co/0e7spkvlzB
2.	Kata org thn 2023 bakal resesi, kyknya bnr deh.. angpao gw di tahun ini jg ikutan resesi. Biasanya angpao 2x UMR, skrng UMR lebih dikit Gw jg bingung dh 2 thn ini ga ada pertemuan keluarga.. pdhl di kelapa gading semua. Dr thn kmrn cmn ambil angpao sama chat no rek BCA doang dh😥
3.	@fromxwz setelah bulan maret resesi (naudzubillah)
4.	RI dipastikan tak resesi!! Berdasarkan ASEAN+3 Macroeconomic Research Office (AMRO) yang memperkirakan ekonomi Indonesia akan tumbuh sebesar 5,0 persen pada 2023, melambat dibandingkan 2022 yang diperkirakan mencapai 5,3 persen. https://t.co/VcGtjXowtu
5.	@nurinuriii Meskipun banyak yg memprediksi thn 2023 sbg tahun resesi besar2an, kutetap berharap banyak orang dapat melaluinya dengan baik. Let's survive together!💪 Harapan lainnya adalah ingin mewujudkan kembali kebiasaan berolahraga dengan rutin demi kesehatan tubuh yang lbh prima😁
6.	@tanyakanrl Cape2 nabung 19 tahun, tahun ke 20 malah resesi😭😭
7.	@OreoOrlando69 Belum sih, masih harus stay dulu yang di sekarang, takut tahun depan jadi resesi :(
8.	Pada 2023, perekonomian Indonesia masih akan tumbuh. Bahkan Indonesia termasuk negara dengan pertumbuhan ekonomi ketiga terbaik. Kalau begitu, perlukah ""overthingking"" terhadap resesi yang katanya akan segera terjadi? #Opini #AdadiKompas https://t.co/it9GAF0uey
9.	Kalo aku, ya kyk td, sejak pandemi sampe skrg, keadaan ekonomiku blom pulih bener. Jd kemungkinannya, saat resesi nanti bakal lebih berat. Huhuhuhu… takut ih! #NabungKriptoDiNOBI#InvestasiLawanResesi#ChillAjaDiNOBI

10.	Kalian khawatir ga sih dg resesi yang di prediksi akan terjadi tahun 2023.. Kalau saya sih sudah persiapkan dari sisi finansial & investasi dong Gimana dengan kalian #NabungKriptoDiNOBI #InvestasiLawanResesi #ChillAjaDiNOBI

Pada tabel 3.1, menunjukkan bahwa data *tweet* yang telah berhasil diambil masih banyak memiliki komponen yang tidak penting dan tidak dibutuhkan dalam proses klasifikasi sentimen. Komponen yang tidak dibutuhkan ini perlu dibersihkan melalui *preprocessing* agar dapat menghasilkan data yang terstruktur.

3.2.2 Preprocessing

Preprocessing merupakan tahapan yang dilakukan untuk membuat data yang tidak terstruktur menjadi terstruktur. Berikut kode program untuk *preprocessing.*

```
import pandas as pd
import numpy as np
import nltk
import string
import emoji
import re
from pandas import DataFrame
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize



def load_data():
data = pd.read_excel('DATARESESI.xlsx',nrows=None,names=
['Datetime','Tweet Id','Text', 'Username'])
return data

data = load_data()

data.head()
```

Kode di atas menampilkan library untuk melakukan *preprocesing* data. Library pandas berfungsi untuk memproses data menjadi terstruktur, library numpy untuk mengolah data yang berbentuk numerik, dan library nltk berfungsi untuk analisis

teks. Selain itu, terdapat library lain seperti, library string yang berfungsi untuk memproses data teks dalam konteks pemrosesan bahasa alami atau Natural Language Processing (NLP), library emoji berfungsi untuk mengelola teks yang mengandung karakter emoji, dan library re atau regex membantu untuk mendefinisikan sebuah pola pencarian.

3.2.2.1 Text Cleaning

Text cleaning merupakan proses untuk membersihkan data mentah dari informasi yang tidak relevan seperti *url*, *emoji*, *hastag*, *mention*, tanda baca, dan simbol. Berikut kode untuk melakukan *text cleaning.*

```
def emoji_remove(tweet):
    if not tweet:
        return ''
    t = tweet.encode('ascii', 'ignore').decode('utf-8')
    return t

def url_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r'http\S+', '', tweet)
    return t

def hashtag_remove(tweet):
    if not tweet:
        return ''
    reg = r'#\S+'
    t = re.sub(reg, ' ', tweet)
    return t

def regex_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r"b'", ' ', tweet)
    return t

def remove_user(tweet):
    if not tweet:
        return ''
    t = re.sub(r'@[^\s]+', '', tweet)
    return t

def number_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r'\d+', '', tweet)
    return t
```

```
def punc_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r'[^\w\s]+', '', tweet)
    return t

def rt_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r'RT\s+', '', tweet)
    return t

def slang_remove(tweet):
    if not tweet:
        return ''
    t = re.sub(r'\\n', ' ', tweet)
    return t

cleaned = []

def clean_text(tweet):
    for i in tweet:
        tweet_raw = re.sub("[\n\r\t\xa0]"," ",str(i)).strip()
        if tweet_raw:
          cleaned.append(url_remove(punc_remove(number_remove
(remove_user(regex_remove(hashtag_remove(rt_remove(emoji_remo
ve(slang_remove(tweet_raw))))))))))
        else:
            cleaned.append("")
clean_text(data["Text"])
```


Kode di atas digunakan untuk melakukan *text cleaning* yaitu menghapus karakter-karakter atau informasi yang tidak diperlukan. Hasil dari tahapan *text cleaning* dapat dilihat pada tabel 3.2

Tabel 3.2 Data hasil proses *text cleaning*

No	Data Tweet
1.	Perbaikan kondisi ekonomi nasional di tengah pandemi Covid dan ancaman resesi global menunjukkan bahwa kebijakan pemerintah saat ini sudah dijalankan dg baikOleh karenanya Erick Thohir mengajak semua pihak utk turut menjaga keberlangsungan ekonomi
2.	Kata org thn 2023 bakal resesi kyknya bnr dehangpao gw di tahun ini jg ikutan resesiBiasanya angpao 2x UMR skrng UMR lebih dikit

	Gw jg bingung dh 2 thn ini ga ada pertemuan keluargapdhl di kelapa gading semuaDr thn kmrn cmn ambil angpao sama chat no rek BCA doang dh
3.	setelah bulan maret resesinaudzubillah
4.	RI dipastikan tak resesiBerdasarkan ASEAN3 Macroeconomic Research OfficeAMRO yang memperkirakan ekonomi Indonesia akan tumbuh sebesar 50 persen pada 2023, melambat dibandingkan 2022 yang diperkirakan mencapai 53 persen
5.	Meskipun banyak yg memprediksi thn 2023 sbg tahun resesi besar2ankutetap berharap banyak orang dapat melaluinya dengan baikLets survive togetherHarapan lainnya adalah ingin mewujudkan kembali kebiasaan berolahraga dengan rutin demi kesehatan tubuh yang lbh prima
6.	Cape2 nabung 19 tahun, tahun ke 20 malah resesi
7.	Belum sih masih harus stay dulu yang di sekarang takut tahun depan jadi resesi
8.	Pada 2023perekonomian Indonesia masih akan tumbuh Bahkan Indonesia termasuk negara dengan pertumbuhan ekonomi ketiga terbaikKalau begitu perlukah overthingking terhadap resesi yang katanya akan segera terjadi
9.	Kalo akuya kyk tdsejak pandemi sampe skrgkeadaan ekonomiku blom pulih benerJd kemungkinannyasaat resesi nanti bakal lebih beratHuhuhuhutakut ih
10.	Kalian khawatir ga sih dg resesi yang di prediksi akan terjadi tahun 2023Kalau saya sih sudah persiapkan dari sisi finansial ampinvestasi dong Gimana dengan kalian

Pada tabel 3.2 di atas menampilkan hasil data *tweet* yang telah bersih dari *url*, *emoji*, *hastag*, *mention*, tanda baca, dan simbol.

3.2.2.2 Tokenizing

Tokenizing merupakan tahapan mengubah kalimat menjadi kata. Berikut kode untuk melakukan *tokenizing*.

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

tokenized = []

def tokenize_text(tweet):
    for tweet in cleaned:
```

```
        tokens = word_tokenize(tweet)
        tokenized.append(tokens)
tokenize_text(data["Text"])
```

Kode di atas menampilkan fungsi untuk melakukan *tokenizing* dengan memecah kalimat menjadi per-kata. Hasil dari *tokenizing* dapat dilihat pada tabel 3.3

Tabel 3.3 Data hasil proses *tokenizing*

No	Data Tweet
1.	Perbaikan, kondisi, ekonomi, nasional, di, tengah, pandemi, Covid, dan, ancaman, resesi, global, menunjukkan, bahwa,kebijakan, pemerintah, saat ini, sudah, dijalankan, dg, baik, Oleh, karenanya, Erick, Thohir, mengajak, semua, pihak, utk, turut, menjaga, keberlangsungan, ekonomi
2.	Kata, org, thn, 2023, bakal, resesi, kyknya, bnr, deh, angpao, gw, di, tahun, ini, jg, ikutan, resesi, Biasanya, angpao, 2x, UMR, skrng, UMR, lebih, dikit, Gw, jg, bingung, dh, 2, thn, ini, ga, ada, pertemuan, keluarga, pdhl, di, kelapa, gading, semua, Dr, thn, kmrn, cmn, ambil, angpao, sama, chat, no, rek, BCA, doang, dh
3.	setelah, bulan, maret, resesi, naudzubillah
4.	RI, dipastikan, tak, resesi, Berdasarkan, ASEAN3, Macroeconomic, Research, Office, AMRO, yang, memperkirakan, ekonomi, Indonesia, akan, tumbuh, sebesar, 50, persen, pada, 2023, melambat, dibandingkan, 2022, yang, diperkirakan, mencapai, 53, persen
5.	Meskipun, banyak, yg, memprediksi, thn, 2023, sbg, tahun, resesi, besar2an, kutetap, berharap, banyak, orang, dapat, melaluinya, dengan, baik, Lets, survive, together, Harapan, lainnya, adalah, ingin, mewujudkan, kembali, kebiasaan, berolahraga, dengan, rutin, demi, kesehatan, tubuh, yang, lbh, prima
6.	Cape2, nabung, 19, tahun, tahun, ke, 20, malah, resesi
7.	Belum, sih, masih, harus, stay, dulu, yang, di, sekarang, takut, tahun, depan, jadi, resesi
8.	Pada, 2023, perekonomian, Indonesia, masih, akan, tumbuh, Bahkan, Indonesia, termasuk, negara, dengan, pertumbuhan, ekonomi, ketiga, terbaik, Kalau, begitu, perlukah, overthingking, terhadap, resesi, yang, katanya, akan, segera, terjadi

9.	Kalo, aku, ya, kyk, td, sejak, pandemi, sampe, skrg, keadaan, ekonomi, ku, blom, pulih, bener, Jd, kemungkinannya, saat, resesi, nanti, bakal, lebih, berat, Huhuhuhu, takut, ih
10.	Kalian, khawatir, ga, sih, dg, resesi, yang, di, prediksi, akan, terjadi, tahun, 2023, Kalau, saya, sih, sudah, persiapkan, dari, sisi, finansial, ampinvestasi, dong, Gimana, dengan, kalian

Pada tabel 3.3 di atas menampilkan data hasil proses *tokenizing* yaitu dengan memecah kalimat menjadi per-kata.

3.2.2.3 Case Folding

Case folding merupakan tahapan untuk menyamakan karakter dalam teks, seperti mengubah huruf besar menjadi huruf kecil (*lowercase*) atau sebaliknya, mengubah huruf kecil menjadi huruf besar (*uppercase*). Hal ini bertujuan untuk memudahkan dalam membandingkan atau mencocokkan kata. Berikut kode program untuk melakukan tahapan *case folding*.

```
def lowercase(text):
    lower_word = []
    for tokens in tokenized:
        text = ' '.join(tokens)
        text = text.lower()
        lower_word.append(text)
    return lower_word

tokenized_data = tokenize_text(data["tweet"])

lower_data = lowercase(tokenized_data)

lower_data =
pd.Series(lower_data).drop_duplicates().astype(object)

print(lower_data)
```

Kode di atas menampilkan fungsi untuk melakukan tahapan *case folding*. Hasil dari tahapan di atas dapat dilihat pada tabel 3.4

Tabel 3.4 Hasil data proses *lowercase*

No	Data Tweet
1.	perbaikan kondisi ekonomi nasional di tengah pandemi covid dan ancaman resesi global menunjukkan bahwa kebijakan pemerintah saat ini sudah dijalankan dg baikoleh karenanya

	erick thohir mengajak semua pihak utk turut menjaga keberlangsungan ekonomi
2.	kata org thn 2023 bakal resesi kyknya bnr dehangpao gw di tahun ini jg ikutan resesibiasanya angpao 2x umr skrng umr lebih dikit gw jg bingung dh 2 thn ini ga ada pertemuan keluargapdhl di kelapa gading semuadr thn kmrn cmn ambil angpao sama chat no rek bca doang dh
3.	setelah bulan maret resesinaudzubillah
4.	ri dipastikan tak resesiberdasarkan asean3 macroeconomic research officeamro yang memperkirakan ekonomi indonesia akan tumbuh sebesar 50 persen pada 2023, melambat dibandingkan 2022 yang diperkirakan mencapai 53 persen
5.	meskipun banyak yg memprediksi thn 2023 sbg tahun resesi besar2ankutetap berharap banyak orang dapat melaluinya dengan baikLets survive togetherharapan lainnya adalah ingin mewujudkan kembali kebiasaan berolahraga dengan rutin demi kesehatan tubuh yang lbh prima
6.	cape2 nabung 19 tahun, tahun ke 20 malah resesi
7.	belum sih masih harus stay dulu yang di sekarang takut tahun depan jadi resesi
8.	pada 2023perekonomian Indonesia masih akan tumbuh bahkan indonesia termasuk negara dengan pertumbuhan ekonomi ketiga terbaikkalau begitu perlukah overthingking terhadap resesi yang katanya akan segera terjadi
9.	kalo akuya kyk tdsejak pandemi sampe skrgkeadaan ekonomiku blom pulih benerjd kemungkinannyasaat resesi nanti bakal lebih berathuhuhuhutakut ih
10.	kalian khawatir ga sih dg resesi yang di prediksi akan terjadi tahun 2023kalau saya sih sudah persiapkan dari sisi finansial ampinvestasi dong gimana dengan kalian

Tabel 3.4 di atas menampilkan hasil data *tweet* yang telah melalui proses case folding. Pada tahapan ini, semua data tweet diubah menjadi huruf kecil (*lowercase*).

3.2.2.4 Stopword Removal

Stopword removal merupakan tahapan untuk menghapus kata-kata yang tidak memiliki makna penting.

```
pip install sastrawi
```

Kode di atas merupakan installasi library Sastrawi yang akan digunakan untuk melakukan tahapan *stopword removal*. Kode pemanggilan library Sastrawi dan prosesnya dapat dilihat di bawah ini

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory

factory = StopWordRemoverFactory()

stopword = factory.create_stop_word_remover()

stopwords = factory.get_stop_words()

stopwords_str = ", ".join([f"'{word}'" for word in
stopwords])

print(stopwords_str)
```


Kode di atas digunakan menampilkan kata yang tidak memiliki makna penting. Hasil dari proses tersebut dapat dilihat pada tabel 3.5

Tabel 3.5 Hasil kata dari library Sastrawi

Hasil kata yang tidak memiliki makna dari library Sastrawi
'yang', 'untuk', 'pada', 'ke', 'para', 'namun', 'menurut', 'antara', 'dia', 'dua', 'ia', 'seperti', 'jika', 'jika', 'sehingga', 'kembali', 'dan', 'tidak', 'ini', 'karena', 'kepada', 'oleh', 'saat', 'harus', 'sementara', 'setelah', 'belum', 'kami', 'sekitar', 'bagi', 'serta', 'di', 'dari', 'telah', 'sebagai', 'masih', 'hal', 'ketika', 'adalah', 'itu', 'dalam', 'bisa', 'bahwa', 'atau', 'hanya', 'kita', 'dengan', 'akan', 'juga', 'ada', 'mereka', 'sudah', 'saya', 'terhadap', 'secara', 'agar', 'lain', 'anda', 'begitu', 'mengapa', 'kenapa', 'yaitu', 'yakni', 'daripada', 'itulah', 'lagi', 'maka', 'tentang', 'demi', 'dimana', 'kemana', 'pula', 'sambil', 'sebelum', 'sesudah', 'supaya', 'guna', 'kah', 'pun', 'sampai', 'sedangkan', 'selagi', 'sementara', 'tetapi', 'apakah', 'kecuali', 'sebab', 'selain', 'seolah', 'seraya', 'seterusnya', 'tanpa', 'agak', 'boleh', 'dapat', 'dsb', 'dst', 'dll', 'dahulu', 'dulunya', 'anu', 'demikian', 'tapi', 'ingin', 'juga', 'nggak', 'mari', 'nanti', 'melainkan', 'oh', 'ok', 'seharusnya', 'sebetulnya', 'setiap', 'setidaknya', 'sesuatu', 'pasti', 'saja', 'toh', 'ya', 'walau', 'tolong', 'tentu', 'amat', 'apalagi', 'bagaimanapun'

Pada tabel 3.5 di atas menampilkan kata yang akan dihilangkan dari data *tweet.* Berikut kode yang digunakan untuk menghilangkan kata-kata tersebut dari data *tweet*.

```
def removeStopWords(tweet):
    clean_word_list = [word for word in tweet.split() if word
not in stopwords]
    return clean_word_list

stopwords_tweet = lower_data.apply(removeStopWords)

print(stopwords_tweet)
```

Kode di atas menampilkan fungsi untuk melakukan *stopword removal* pada data *tweet.*

3.2.2.5 Stemming

Stemming merupakan tahapan yang dilakukan untuk mengubah kata menjadi bentuk dasar. Berikut kode program proses *stemming*.

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stemmed_wrapper(term):
    return stemmer.stem(term)

term_dict = {}

for document in stopwords_tweet:
    for term in document:
        if term not in term_dict:
            term_dict[term] = " "

print(len(term_dict))
print("------------------------")
for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
    print(term,"=" ,term_dict[term])

print(term_dict)
print("------------------------")

def get_stemmed_term(document):
    return [term_dict[term] for term in document]
```

```
stem_tweet = stopwords_tweet.apply(get_stemmed_term)

print(stem_tweet)
```

Kode di atas menampilkan fungsi untuk mengubah kata menjadi bentuk dasar dengan memanfaatkan library Sastrawi.

3.2.3 Pelabelan data

Pada tahap pelabelan data ini dilakukan secara otomatis dan manual. Pelabelan otomatis memanfaatkan salah satu library python yaitu Textblob. Textblob menyediakan beberapa kegunaan seperti ekstraksi kata, penerjemahan teks, analisis sentimen, dan pelabelan. Sedangkan pelabelan manual dilakukan secara manual dengan tujuan untuk memberikan tingkat akurasi kelas yang lebih maksimal setelah melakukan pelabelan secara otomatis. Pelabelan ini menghasilkan kelas siap dan tidak siap.

```
pip install textblob
```

Kode di atas merupakan library untuk melakukan pelabelan data secara otomatis dengan menginstall Textblob menggunakan pip. Selanjutnya, yaitu proses pelabelan data.

```
import pandas as pd
from textblob import TextBlob

def label_sentiment(text):
    blob = TextBlob(text)
    sentiment = blob.sentiment.polarity
    if sentiment > 0:
        return "SIAP"
    else:
        return "TIDAK SIAP"

data = pd.read_excel('RESESISTEM.xlsx')

teks_column = "text"
label_column = "kelas"
data[label_column] = data[teks_column].apply(label_sentiment)

print(data.head())
```

Kode di atas menampilkan proses untuk melakukan pelabelan data secara otomatis dengan mengimpor file yang telah melalui proses *preprocessing*. Library Pandas digunakan untuk membaca dan memanipulasi data dalam bentuk *dataframe* dan library Textblob digunakan untuk analisis sentimen teks. Fungsi `label_sentiment(text)` menerima teks sebagai argumen dan menggunakan Textblob untuk menganalisis sentimen teks tersebut. Sentimen dihitung menggunakan metode *sentiment.polarity* yang menghasilkan angka antara -1 hingga 1. Berdasarkan nilai sentimen tersebut, fungsi mengembalikan label sentimen yang sesuai `yaitu "SIAP" jika sentimen > 0, "TIDAK SIAP" jika sentimen < 0.` Contoh *tweet* dari pelabelan otomatis tersebut dapat dilihat di bawah ini

Tabel 3.6 Hasil pelabelan otomatis

No.	Kelas	Text
1.	Tidak Siap	gapki optimis industri sawit tahan hadap ancam resesi
2.	Siap	mau dunia resesi dollar kuat orang indonesia bangsa yg kuat alam dollar rakyat biasa tetap makanbeli mobil kok kuatir dng data bgtu mardigu kok kuatir ky duit ribu trilyun aja main politik
3.	Siap	langsung indonesia makasih banyak program bri jadi andal umkm karena banyak nilai resesi indonesia ga gelapgelap banget umkm okay deh moga acara g bal lancar jaya
4.	Tidak Siap	beras nya du timbun kaka siap resesi
5.	Tidak Siap	cacut taliwondo ikat kencang tali satu saudara bangsa tanah air kuat mampu lewat badai resesi
6.	Tidak Siap	makin panik makin rasa resesi ketua kadin
7.	Tidak Siap	resesi beneran agak gue ngeri sepuh kantor gue aja udah cut
8.	Tidak Siap	anjjjj resesi ngeriiiiiii bangettt nyata
9.	Tidak Siap	sekarang takut sih tbh denger berita resesi jd bikin overthinking soal karyawan kontrak kaya gue phk kapan aja
10.	Siap	baru haha hihi trs keinget bakal resesi wkwkwk jd ovt

Pada tabel 3.6 menampilkan hasil *tweet* yang telah melalui proses pelabelan secara otomatis menggunakan library Textblob. Hasil yang didapatkan setelah melalui

proses pelabelan tersebut tidak akurat sehingga dilakukanlah pelabelan secara manual. Contoh tweet dengan melalui proses pelabelan manual dapat dilihat pada tabel 3.7

Tabel 3.7 Hasil pelabelan manual

No.	Kelas	Text
1.	Siap	gapki optimis industri sawit tahan hadap ancam resesi
2.	Siap	mau dunia resesi dollar kuat orang indonesia bangsa yg kuat alam dollar rakyat biasa tetap makanbeli mobil kok kuatir dng data bgtu mardigu kok kuatir ky duit ribu trilyun aja main politik
3.	Siap	langsung indonesia makasih banyak program bri jadi andal umkm karena banyak nilai resesi indonesia ga gelapgelap banget umkm okay deh moga acara g bal lancar jaya
4.	Siap	beras nya du timbun kaka siap resesi
5.	Siap	cacut taliwondo ikat kencang tali satu saudara bangsa tanah air kuat mampu lewat badai resesi
6.	Tidak Siap	makin panik makin rasa resesi ketua kadin
7.	Tidak Siap	resesi beneran agak gue ngeri sepuh kantor gue aja udah cut
8.	Tidak Siap	anjjjj resesi ngeriiiiiii bangettt nyata
9.	Tidak Siap	sekarang takut sih tbh denger berita resesi jd bikin overthinking soal karyawan kontrak kaya gue phk kapan aja
10.	Tidak Siap	baru haha hihi trs keinget bakal resesi wkwkwk jd ovt

Pada tabel 3.7, menampilkan hasil pelabelan secara manual yang menghasilkan kelas lebih akurat. Pada tahap pelabelan ini dari jumlah data *tweet* sebanyak 23.681 didapat data tweet sebanyak 800 tweet dengan rincian 400 *tweet* bernilai siap dan 400 *tweet* bernilai tidak siap. Data tersebut nantinya akan digunakan untuk tahap *training data*.

3.2.4 Training Data

Training data merupakan proses melatih data menggunakan algoritma SVM. Proses pelatihan ini dimulai dengan ekstraksi data menggunakan TF-IDF (Term Frequency-Inverse Document Frequency). Setelah itu, dilakukan pelatihan

data untuk menciptakan model klasifikasi yang dapat digunakan secara otomatis untuk analisis sentimen. Berikut kode library untuk SVM

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

import matplotlib.pyplot as plt
%matplotlib inline
```

Kode di atas mengimpor algoritma SVM yang akan digunakan untuk pelatihan model klasifikasi. Berikut ini merupakan kode program untuk memanggil *data training.*

```
df = pd.read_excel ('training.xlsx')
df=pd.DataFrame(df)
df=df.dropna()
df
```

Kode di atas digunakan untuk memanggil file data training berupa file excel yang telah diberi label dan skor. Hasil dari pemanggilan file tersebut dapat dilihat pada gambar 3.4

	NO	Text	Kelas	Skor
0	0	bgst duduk tetiba mikirin resesi ekonomi global	TIDAK SIAP	0
1	1	resesi is real	TIDAK SIAP	0
2	2	after effect nya pandemi emang luar biasa sih ...	TIDAK SIAP	0
3	3	ga kuat tho serius resesi ta	TIDAK SIAP	0
4	4	gue overthinking tetang resesi tiap hari wak	TIDAK SIAP	0
...	...	...	...	...
795	796	ayo semangat kerja kerja kerja taun depan rese...	SIAP	1
796	797	gass hadap resesi cocok tanam	SIAP	1
797	798	resesi aja siap arah nya masa depan negara	SIAP	1
798	799	emang resesi udah mulai	SIAP	1
799	800	memang tahun resesi sih siap	SIAP	1

800 rows × 4 columns

Gambar 3.4 Hasil pemanggilan *data training*

Gambar 3.4 di atas merupakan hasil dari pemanggilan *data training* yang telah dilabeli secara manual. Dari *data training* tersebut diambil beberapa dokumen yang akan digunakan untuk menghitung pembobotan TF-IDF.

```
from sklearn.feature_extraction.text import TfidfVectorizer

d1 = "gini lulus era resesi susah dapet kerja"
d2 = "yg gw takutin bangkrut nyusul uber berapa ratus juta
driver driver yg fokus gojek nambah anggur ae dah isu tahun
depan resesi ekonomi"
d3 = "udah siap buat tahun baru siap resesi"
d4 = "jadi lebih siap pede hadap resesi"

vect = TfidfVectorizer()
X = vect.fit_transform([d1, d2, d3, d4])

X.toarray()
```

Kode di atas menampilkan contoh dokumen yang akan dibobot dengan TF-IDF. Perhitungan pembobotan kata dilakukan secara otomatis menggunakan google colab. Dengan memanfaatkan library Scikit-Learn ntuk melakukan vektorisasi teks berdasarkan skema pembobotan TF-IDF. Menggunakan metode `fit_transform()` untuk menghitung dan membangun model TF-IDF berdasarkan data teks yang diberikan yang dapat dilihat pada d1, d2, d3, d4. Hasil dari perhitungan TF-IDF secara otomatis dapat dilihat pada gambar 3.5

```
[(0.0, 'ae'),
 (0.0, 'anggur'),
 (0.0, 'bangkrut'),
 (0.0, 'baru'),
 (0.0, 'berapa'),
 (0.0, 'buat'),
 (0.0, 'dah'),
 (0.0, 'dapet'),
 (0.0, 'depan'),
 (0.0, 'driver'),
 (0.0, 'ekonomi'),
 (0.0, 'era'),
 (0.0, 'fokus'),
 (0.0, 'gini'),
 (0.0, 'gojek'),
 (0.0, 'gw'),
 (0.45203489051046103, 'hadap'),
 (0.0, 'isu'),
 (0.45203489051046103, 'jadi'),
 (0.0, 'juta'),
 (0.0, 'kerja'),
 (0.45203489051046103, 'lebih'),
 (0.0, 'lulus'),
 (0.0, 'nambah'),
 (0.0, 'nyusul'),
 (0.45203489051046103, 'pede'),
 (0.0, 'ratus'),
 (0.2358905582496689, 'resesi'),
 (0.3563894998080638, 'siap'),
 (0.0, 'susah'),
 (0.0, 'tahun'),
 (0.0, 'takutin'),
 (0.0, 'uber'),
 (0.0, 'udah'),
 (0.0, 'yg')]
```

Gambar 3.5 Hasil perhitungan TF-IDF

Pada gambar 3.5 di atas merupakan hasil dari perhitungan TF-IDF secara otomatis pada google colab. Setelah proses tersebut, dilanjutkan dengan perhitungan *Confusion Matrix* pada *data training* yang dapat dilihat di bawah ini

```
from sklearn.model_selection import ShuffleSplit
from sklearn.metrics import accuracy_score, f1_score,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
X = df.Text
y = df.Kelas

ss = ShuffleSplit(n_splits=10, test_size=0.2)
sm = SMOTE()

accs = []
f1s = []
cms = []

for train_index, test_index in ss.split(X):
```

```
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    X_train_vect = vect.fit_transform(X_train)
    X_test_vect = vect.transform(X_test)

    X_train_res, y_train_res = sm.fit_resample(X_train_vect,
y_train)

    SVM.fit(X_train_res, y_train_res)
    y_pred = SVM.predict(X_test_vect)

    accs.append(accuracy_score(y_test, y_pred))
    f1s.append(f1_score(y_test, y_pred, average='weighted'))
    cms.append(confusion_matrix(y_test, y_pred))

print("\nAverage accuracy cross-folds: {:.2f}%".format(sum(accs)
/ len(accs) * 100))
print("\nAverage F1 score cross-folds: {:.2f}%".format(sum(f1s)
/ len(f1s) * 100))
print("\nAverage Confusion Matrix cross-folds: \n
{}".format(sum(cms) / len(cms)))

y_pred
```

Kode di atas menampilkan program untuk perhitungan *Confusion Matrix data training* dengan menggunakan teknik *Cross-Validation*. Perhitungan ini menghasilkan informasi perbandingan hasil klasifikasi yang dilakukan pada sistem, sehingga dapat diketahui hasil dari kinerja pada model yang telah dibuat. Tahap selanjutnya yaitu menyimpan file dengan format pickle.

```
import os
import pickle
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import LabelEncoder

x = df.Text
y = df.Kelas

Encoder = LabelEncoder()
```

```
Train_Y = Encoder.fit_transform(y)

Tfidf_vect = TfidfVectorizer()
Tfidf_vect.fit(x)
Train_X_Tfidf = Tfidf_vect.transform(x)

SVM = SVC(C= 1.0, kernel='linear', degree=3, gamma='auto')
text_clf=SVM.fit(Train_X_Tfidf,Train_Y)

files = open('SVM_classifier.pickle', 'wb')
pickle.dump(text_clf, files)
files.close()

print('SELESAI!')
```

Kode di atas digunakan untuk menyimpan file ke dalam format pickle yang kemudian digunakan untuk tahap klasifikasi SVM.

3.2.5 Algoritma Support Vector Machine

Pada tahap ini, dilakukan perhitungan klasifikasi SVM dengan menggunakan file yang telah disimpan ke dalam format pickle. Berikut kode pemanggilan SVM untuk klasifikasi

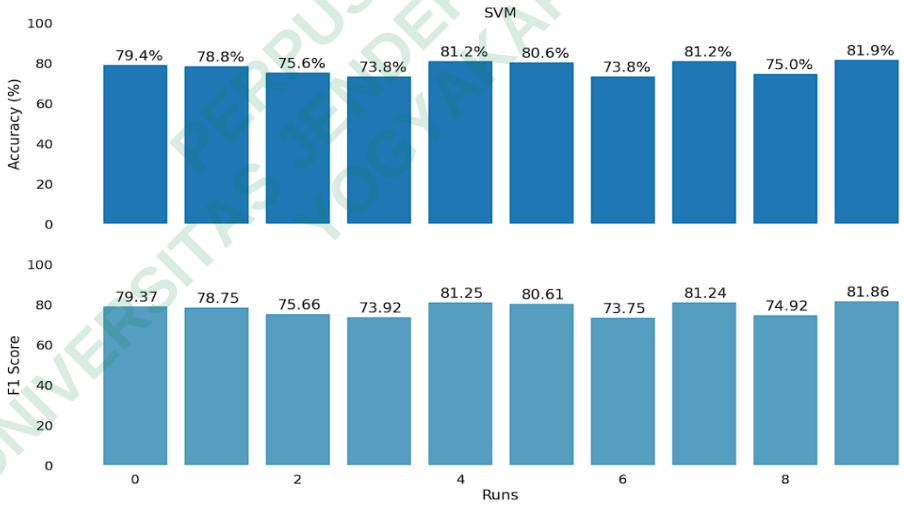
```
model = open('SVM_classifier.pickle', 'rb')
svm_classifier = pickle.load(model)
svm_classifier
```

Kode di atas digunakan untuk melakukan pemanggilan SVM dalam format pickel. Selanjutnya yaitu melakukan visualisasi untuk menampilkan grafik SVM. Berikut kode program untuk menampilkan grafik SVM

```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(16,9))

acc_scores = [round(a * 100, 1) for a in accs]
f1_scores = [round(f * 100, 2) for f in f1s]

x1 = np.arange(len(acc_scores))
x2 = np.arange(len(f1_scores))

ax1.bar(x1, acc_scores)
ax2.bar(x2, f1_scores, color='#559ebf')
```

```
for i, v in enumerate(list(zip(acc_scores, f1_scores))):
    ax1.text(i - 0.25, v[0] + 2, str(v[0]) + '%')
    ax2.text(i - 0.25, v[1] + 2, str(v[1]))

ax1.set_ylabel('Accuracy (%)')
ax1.set_title('SVM')
ax1.set_ylim([0, 100])

ax2.set_ylabel('F1 Score')
ax2.set_xlabel('Runs')
ax2.set_ylim([0, 100])

sns.despine(bottom=True, left=True)

plt.show()
```

Kode di atas menampilkan program untuk menampilkan grafik SVM dengan menggunakan nilai *accuracy* dan *f1-score*. Hasil dari kode tersebut dapat dilihat pada gambar 3.6



Gambar 3.6 Grafik akurasi SVM

Gambar 3.6 menampilkan grafik *accuracy* dan *f1-score*. Untuk mendapatkan hasil akurasi yang tinggi dilakukan pengujian sebanyak 10 kali. Menghasilkan nilai paling tinggi yaitu *accuracy* sebesar 81.9% dan *f1-score* sebesar 81.6% pada pengujian ke-10.

3.2.6 Testing Data

Testing data merupakan tahapan untuk menguji performa model yang telah dilatih pada data pelatihan. *Data testing* adalah data yang tidak digunakan selama proses pelatihan model dan digunakan untuk mengevaluasi seberapa baik model dapat melakukan prediksi pada data yang belum pernah dilihat sebelumnya.

```
df = pd.read_excel ('testing.xlsx')
df=pd.DataFrame(df)
df=df.dropna()
df
```

Kode di atas melakukan pemanggilan *data testing* berupa file excel yang akan digunakan untuk menguji model. Hasil pemanggilan *data testing* dapat dilihat pada gambar 3.7

	NO	Text	Kelas	Skor
0	0	zaman sekarang semua tuh susah bahkan beberapa...	TIDAK SIAP	0
1	1	resesi seram kok kalo siap	TIDAK SIAP	0
2	2	anjir ni beneran mau resesi ekonomi g sih	TIDAK SIAP	0
3	3	jangan sampe deh indonesia kena resesi	TIDAK SIAP	0
4	4	sangat rugi bgt klo ad resesi huhu	TIDAK SIAP	0
...	...	...	...	...
195	195	smgt bntar resesi	SIAP	1
196	196	bersiapsiap nih buat resesi	SIAP	1
197	197	aku sdm rendah siap hadap resesi mas	SIAP	1
198	198	inget resesi inget resesi	SIAP	1
199	199	semangaatttt kabar resesi taun depan	SIAP	1

200 rows × 4 columns

Gambar 3.7 Hasil pemanggilan *data testing*

Gambar 3.7 di atas menampilkan hasil pemanggilan *data testing* yang berisi data *tweet* dengan label siap dan tidak siap serta telah diberi skor. *Data testing* ini berisi 100 data *tweet* berlabel siap dan 100 data *tweet* berlabel tidak siap. Data-data ini digunakan untuk tahap perhitungan *Confusion Matrix pada data testing*. Berikut kode perhitungannya

```
df.loc[(df['Label'] == 'SIAP') & (df['Predict'] == 'SIAP'),
'Keterangan'] = 'TP'
df.loc[(df['Label'] == 'SIAP') & (df['Predict'] == 'TIDAK SIAP'),
'Keterangan'] = 'FP'
df.loc[(df['Label'] == 'TIDAK SIAP') & (df['Predict'] == 'TIDAK
SIAP'), 'Keterangan'] = 'TN'
df.loc[(df['Label'] == 'TIDAK SIAP') & (df['Predict'] == 'SIAP'),
'Keterangan'] = 'FN'
```

Kode di atas menampilkan perhitungan Confusion Matrix pada *data testing* dengan menggunakan TP (True Positive), FN (False Negative), TN (True Negative), dan FP (False Positive). Hasil dari perhitungan tersebut dapat dilihat pada gambar 3.8

Keterangan	No	Text	Label	Predict
FN	31	31	31	31
FP	25	25	25	25
TN	69	69	69	69
TP	75	75	75	75

Gambar 3.8 Hasil perhitungan *Confusion Matrix Data Testing*

Pada gambar 3.8 menampilkan hasil perhitungan TP sebesar 75, FN sebesar 31, FP sebesar 25, dan TN sebesar 69. Tahap selanjutnya ialah menghitung *accuracy data testing*. Berikut kode yang digunakan untuk menghitung *accuracy data testing*

```
accuracy = (TP+TN)/(TP+FP+FN+TN)
print('Accuracy =', accuracy)

>>>Accuracy = 0.72
```

Kode di atas menghitung *accuracy* pada *data testing* yang menghasilkan nilai sebesar 72%. Berikut kode untuk menghitung *precission*

```
precission = (TP) / (TP+FP)
print('Precission =', precission)
```

```
>>>Precission = 0.75
```

Kode di atas menampilkan perhitungan *precission* yang menghasilkan nilai sebesar 75%. Berikut kode untuk menghitung *recall*

```
recall = (TP) / (TP + FN)
print('Recall =', recall)

>>>Recall = 0.7075471698113207
```

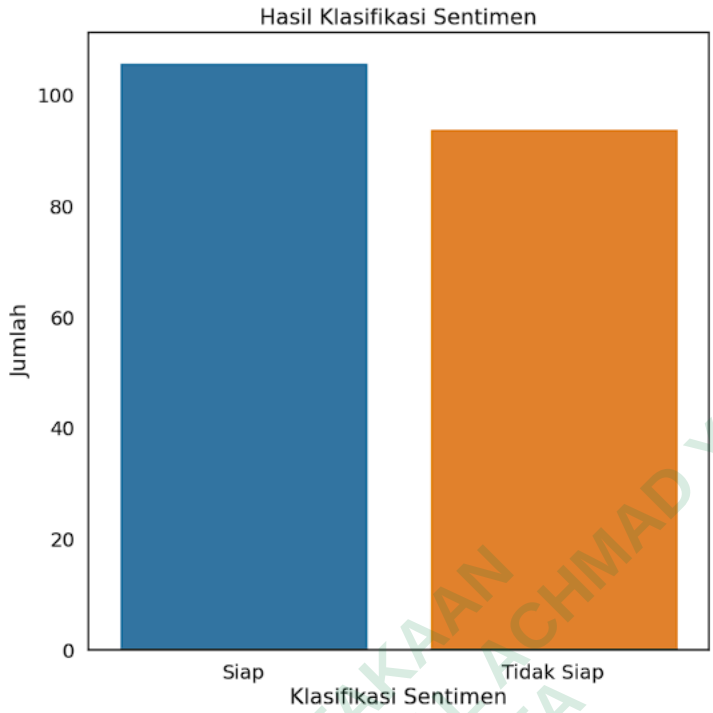
Kode di atas menampilkan perhitungan *recall* yang menghasilkan nilai sebesar 71%. Berikut kode untuk menghitung *f1-score*

```
F1_Score = 2 * (precission*recall) / (precission + recall)
print('F1-Score =', F1_Score)

>>>F1-Score = 0.7281553398058253
```

Kode di atas menampilkan perhitungan *f1-score* pada *data testing* yang menghasilkan nilai sebesar 73%. Berikut kode untuk menampilkan grafik dari *data testing*

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots(figsize=(8, 8))
sentimen = ['Siap','Tidak Siap']
jumlah_sentimen = [106,94]
sns.barplot(x=sentimen,y=jumlah_sentimen)
ax.set_ylabel('Jumlah')
ax.set_xlabel('Klasifikasi Sentimen')
ax.set_title('Hasil Klasifikasi Sentimen')
plt.show()
```

Kode di atas menampilkan grafik pada *data testing*. Tampilan grafik *data testing* dapat dilihat pada gambar 3.9



Gambar 3.9 Grafik *data testing*

Gambar 3.9 di atas menampilkan grafik klasifikasi pada *data testing* dengan diagram berwarna biru menunjukkan sentimen siap sebanyak 106 dan diagram berwarna oranye menunjukkan sentimen tidak siap sebanyak 94.

3.2.7 Klasifikasi

Tahap klasifikasi ini menampilkan hasil perhitungan data menggunakan algoritma SVM. Data yang digunakan ialah data bersih setelah melewati tahapan *preprocessing* sebanyak 23.681 data. Kode untuk menampilkan grafik dari klasifikasi sentimen, berikut kode programnya

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.rcParams["figure.figsize"] = [8,8]
plt.rcParams["figure.autolayout"] = True
x = ['SIAP', 'TIDAK SIAP',]
y = [13347,10334]
percentage = [13347,10334]
ax = sns.barplot(x=x, y=y)
patches = ax.patches
for i in range(len(patches)):
    x = patches[i].get_x() + patches[i].get_width()/2
    y = patches[i].get_height()+.5
    ax.annotate('{:}'.format(percentage[i]), (x, y),
ha='center')

plt.title('Klasifikasi Sentimen \n')
plt.xlabel('Jenis Klasifikasi')
plt.ylabel('Jumlah')
plt.show()
```

Pada gambar 3.38 di atas merupakan kode untuk menampilkan grafik berupa bar dengan persamaan x= siap dan tidak siap, sedangkan persamaan y=jumlah sentimen siap sebanyak 13.347 dan jumlah sentimen tidak siap sebanyak 10.334.