BAB 4
HASIL PENELITIAN

4.1 Ringkasan Hasil Penelitian

Berdasarkan hasil penelitian, pendekatan *baseline* dengan *cosine similarity* cukup efektif untuk mengukur kemiripan kata dalam komentar, tetapi kurang optimal untuk klasifikasi multiclass karena hanya mempertimbangkan kedekatan vektor tanpa mempertimbangkan pola distribusi data yang lebih kompleks. Sebaliknya, metode *Support Vector Machine* (SVM) mampu mempelajari hubungan antarfitur secara lebih mendalam melalui pemisahan *hyperplane* yang optimal, sehingga memberikan performa yang lebih baik, khususnya dalam skenario *multiclass*. Hasil evaluasi menunjukkan model SVM lebih tinggi dibandingkan baseline, menandakan kemampuan model mendeteksi *cyberbullying* secara lebih konsisten dengan tingkat *false positive* yang lebih rendah. Selain itu, proses *pre-processing* terbukti penting dalam meningkatkan kualitas data dan performa model. Temuan ini mendukung bahwa kombinasi TF-IDF dan SVM adalah pendekatan yang lebih unggul daripada *cosine similarity* dalam mendeteksi komentar berpotensi *cyberbullying*.

4.2 Pembahasan Hasil Penelitian

4.2.1 Persiapan Dataset

Persiapan dataset merupakan langkah awal untuk mengumpulkan data teks yang berpotensi mengandung unsur *cyberbullying* dari media sosial. Penelitian ini memanfaatkan dataset publik dari platform *Kaggle* berjudul *Cyberbullying Detection – NLP*, yang memuat 47.692 entri data yang berasal dari *Twitter* dan telah melalui proses *labeling* menurut kategori *cyberbullying*.

Setiap data direpresentasikan dalam bentuk teks mentah *(raw tweet)* dan label kategorikal yang menunjukkan tipe *cyberbullying* yang terjadi. Adapun kategori yang terdapat dalam dataset ini meliputi:

1. *Age*: *cyberbullying* berdasarkan usia,
2. *Gender*: *cyberbullying* berdasarkan jenis kelamin,

3. *Ethnicity*: *cyberbullying* berdasarkan etnis atau ras,
4. *Religion*: *cyberbullying* berdasarkan keyakinan agama,
5. *Other_cyberbullying*: kategori *cyberbullying* lainnya di luar kategori utama,
6. *Not_cyberbullying*: tweet yang tidak mengandung unsur *cyberbullying*.

Dataset ini disediakan dalam format CSV *(Comma-Separated Values)* dengan nama file *cyberbullying*_tweets.csv. Setiap baris dalam file tersebut terdiri atas dua atribut utama seperti dijelaskan dalam tabel 4.1 berikut.

Tabel 4.1 Atribut Dataset

Nama Atribut	Deskripsi
Tweet Text	Isi komentar atau unggahan dari pengguna twitter
Cyberbullying Type	Label klasifikasi berdasarkan jenis *cyberbullying*

Sebelum dilakukan analisis lebih lanjut, peneliti melakukan eksplorasi awal untuk memahami struktur dan kelengkapan dataset. Proses pembacaan file dilakukan menggunakan kode *Python* berikut.

```
import pandas as pd
df = pd.read_csv('cyberbullying_data.csv')
df.info()
```

Output yang diperoleh:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47692 entries, 0 to 47691
Data columns (total 2 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   tweet_text          47692 non-null  object
 1   cyberbullying_type  47692 non-null  object
```

Untuk memberikan gambaran awal, berikut tampilan data yang digunakan pada Tabel 4.2.

Tabel 4.2 Contoh Komentar pada Dataset

No	Data Tweet	Label
1.	6th grader:"What do U do w/these girls who think they're better than everyone?" My reply @ Education.com http://t.co/wF6yWUL #bullying	not_*cyberbullying*

No	Data Tweet	Label
2.	So having rapists in the game is ok. But a joke about gay people we can't have that right. Rape murder extortion all fine but a joke how dare they	gender
3.	Boy, your comment about Journalists wanting to keep churches closed is beneath you. As a Christian woman and human being your bosses filth is brushing off on you. Not at all unbiased and a down right lie. SHAME ON YOU.	religion
4.	(tbh i was surprised to find i was friends with this girl on facebook in the first place because she bullied me in high school lmao)	age
5.	Fuck u bitch RT @tayyoung_: FUCK OBAMA, dumb ass nigger	ethnicity
6.	RT @notwaving: I hate it when I'm trying to board a bus and there's already an asshole on it. https://t.co/Qps29bAaoA	other_*cyberbullying*

Selanjutnya dilakukan pemeriksaan data duplikat untuk menghindari potensi bias dalam pelatihan model. Jumlah data duplikat dapat dicek menggunakan perintah berikut.

```
df.duplicated(subset='content').sum()
np.int64(1675)
```

Hasil menunjukkan 1.675 baris data duplikat, yang kemudian dihapus menggunakan perintah berikut.

```
df = df.drop_duplicates(subset='content')
```

Setelah proses pembersihan, jumlah data akhir yang siap diproses adalah 46.017 baris data.

4.2.2 *Pre-processing*

Tahap *pre-processing* berfokus pada penyiapan data agar lebih bersih, terstruktur, dan sesuai kebutuhan analisis. Proses ini penting untuk meningkatkan kualitas fitur yang dihasilkan dan mendukung akurasi model. Berikut adalah beberapa tahapan yang dilakukan dalam *pre-processing* data:

1. ***Cleaning***

   Teks *cleaning* merupakan proses untuk pembersihan data mentah dari unsur yang tidak relevan misalnya karakter khusus, angka, simbol,

URL, emoji, *mention*, dan *hashtag*, untuk menjaga fokus pada kata-kata bermakna.

Berikut kode untuk yang digunakan untuk proses *cleaning*.

```
import re
def cleaning(text):
    text = re.sub(r'http\S+', '', text)  # Hapus URL
    text = re.sub(r'@\w+', '', text)     # Hapus username
    text = re.sub(r'#\w+', '', text)     # Hapus hashtag
    text = re.sub(r'\d+', '', text)      # Hapus angka
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)  # Hapus tanda
baca
    text = re.sub(r'\s+', ' ', text)     # Hapus spasi berlebih
    text = re.sub(r'[^\x00-\x7F]+', '', text)  # Hapus karakter
non-ASCII
    return text.strip()
```

Setelah proses *cleaning* selesai, data berkurang menjadi 45.650 baris karena beberapa entri kosong dihapus. Hasil data yang telah melalui proses *cleaning* ditampilkan pada tabel 4.3 sebagai berikut.

Tabel 4.3 Hasil Teks *Cleaning*

Komentar	**Setelah *Cleaning***
Kids Love😘♡ @ Mohamad Bin Zayed City مدينة محمد بن زايد http://t.co/0xrOZSNn	Kids Love Mohamad Bin Zayed City
#mkr Promo girls are going to BRING IT in the next round. Yeah! Bring it! Bring your store bought capsicums!	Promo girls are going to BRING IT in the next round Yeah Bring it Bring your store bought capsicums
@sbrew11 If you hadn't noticed, I save my witty replies for sexists with a little more panache. I don't want to get above your reading level	If you hadnt noticed I save my witty replies for sexists with a little more panache I dont want to get above your reading level
Follow News Reports Of #BrandonMcInerney Shooting Death Of #LarryKing Trial http://t.co/WgrutwF #Gay #GLBT #LGBT #Bullying #Murder	Follow News Reports Of Shooting Death Of Trial
@iMrBarfield it starts Thursday. My classes are canceled for tomorrow tho😉.	it starts Thursday My classes are canceled for tomorrow tho

2. *Filtering* Bahasa

Tahap filterisasi bahasa bertujuan memastikan hanya data berbahasa Inggris yang diproses lebih lanjut, agar sesuai dengan kebutuhan model klasifikasi. Proses ini memanfaatkan pustaka *langdetect* untuk mendeteksi bahasa pada setiap entri teks. Langkah *filtering* dilakukan dengan kode berikut.

```
from langdetect import detect
# Fungsi deteksi bahasa
def detect_lang(text):
    try:
        return detect(str(text))
    except:
        return 'unknown'

# Tambahkan kolom bahasa
df['language'] = df['content'].apply(detect_lang)

# Filter data berbahasa Inggris
df = df[df['language'] == 'en'].reset_index(drop=True)
df = df.drop(columns=['language'])
```

Hasil deteksi menunjukkan mayoritas data berbahasa inggris sebanyak 43.080 baris (sekitar 94,33% dari total 45.650 data). Selain itu, ditemukan data dalam bahasa lain seperti Portugis (339 entri), Jerman (276 entri), Indonesia (183 entri), Afrikaans (177 entri), serta bahasa lain dalam jumlah kecil. Dengan demikian, hanya data berbahasa Inggris yang dipertahankan untuk tahap pemrosesan selanjutnya agar sesuai dengan kebutuhan model klasifikasi.

3. Case Folding

Case folding merupakan tahapan menormalisasikan teks untuk menyeragamkan semua huruf menjadi format huruf kecil *(lowercase)* Berikut adalah kode yang digunakan untuk melakukan proses *case folding*:

```
def case_folding(text):
    return text.lower()
```

Hasil pembersihan melalui proses *case folding* ditampilkan pada Tabel 4.4 berikut.

Tabel 4.4 Hasil *Case Folding*

Cleaning	*Case Folding*
Kids Love Mohamad Bin Zayed City	kids love mohamad bin zayed city
Promo girls are going to BRING IT in the next round Yeah Bring it Bring your store bought capsicums	promo girls are going to bring it in the next round yeah bring it bring your store bought capsicums
If you hadnt noticed I save my witty replies for sexists with a little more panache I dont want to get above your reading level	if you hadnt noticed i save my witty replies for sexists with a little more panache i dont want to get above your reading level
Follow News Reports Of Shooting Death Of Trial	follow news reports of shooting death of trial
it starts Thursday My classes are canceled for tomorrow tho	it starts thursday my classes are canceled for tomorrow tho

4. Normalisasi

Normalisasi merupakan proses penyesuaian bentuk penulisan agar memiliki makna yang sama. Tahap ini penting karena data media sosial sering mengandung singkatan, kesalahan ketik, atau kata gaul.

Langkah pertama dilakukan dengan mendeteksi kata tidak dikenal (*out-of-vocabulary*/OOV) berdasarkan kamus Bahasa Inggris dari pustaka NLTK:

```
nltk.download('words')

# Ambil kata baku dari NLTK
english_words = set(words.words())

# Deteksi kata OOV
unique_words = set(all_text.split())
oov_words = [word for word in unique_words if word not in
english_words]
```

Kata-kata yang terdeteksi sebagai OOV dicocokkan dengan kamus normalisasi eksternal dalam bentuk CSV. Setiap kata yang cocok akan diganti ke bentuk bakunya menggunakan fungsi berikut.

```
df_kamus = pd.read_csv('kamus_normalisasi.csv')
```

```
kamus_normalisasi = dict(zip(df_kamus['kata_asli'],
df_kamus['kata_normalisasi']))

def normalisasi_teks(teks):
    return ' '.join([kamus_normalisasi.get(kata, kata) for
kata in teks.split()])
```

hasil pembersihan setelah melalui tahap normalisasi ditampilkan pada Tabel 4.5 berikut.

Tabel 4.5 Hasil Normalisasi Teks

Sebelum Normalisasi	Setelah Normalisasi
promo girls are going to bring it in the next round yeah bring it bring your store bought capsicums	promotion girls are going to bring it in the next round yeah bring it bring your store bought capsicums
if you hadnt noticed i save my witty replies for sexists with a little more panache i dont want to get above your reading level	if you had not noticed i save my witty replies for sexists with a little more panache i dont want to get above your reading level
follow news reports of shooting death of trial	follow news reports of shooting death of trial
it starts thursday my classes are canceled for tomorrow tho	it starts thursday my classes are cancelled for tomorrow though to

5. Tokenisasi

Tokenisasi merupakan proses memecah teks menjadi satuan kata *(token)*. Proses tokenisasi dalam penelitian ini dilakukan menggunakan fungsi *word_tokenize* dari pustaka NLTK. Berikut implementasi tokenisasi dilakukan dengan kode berikut.

```
from nltk.tokenize import word_tokenize
nltk.download('punkt_tab')
df['token'] = df['normalize'].apply(word_tokenize)
```

Proses ini memecah setiap kalimat menjadi daftar kata yang lebih terstruktur. Contoh hasil tokenisasi dapat dilihat pada Tabel 4.6 berikut:

Tabel 4.6 Contoh Hasil Tokenisasi

Sebelum Tokenisasi	Setelah Tokenisasi
promotion girls are going to bring it in the next round yeah bring it bring your store bought capsicums	[promotion, girls, are, going, to, bring, it, in, the, next, round,

	yeah, bring, it, bring, your, store, bought, capsicums]
if you had not noticed i save my witty replies for sexists with a little more panache i dont want to get above your reading level	[if, you, had, not, noticed, i, save, my, witty, replies, for, sexists, with, a, little, more, panache, i, dont, want, to, get, above, your, reading, level]
follow news reports of shooting death of trial	[follow, news, reports, of, shooting, death, of, trial]
it starts thursday my classes are cancelled for tomorrow though to	[it, starts, thursday, my, classes, are, cancelled, for, tomorrow, though, to]

6. Remove Stopword

Penghapusan *stopword* bertujuan untuk menghapus *noise* serta meningkatkan efisiensi model dalam mengenali kata-kata yang lebih bermakna. Implementasi fungsi penghapusan *stopword* ditunjukkan berikut.

```
From nltk.corpus import stopwords
stop_words_en = set(stopwords.words('english'))
# Tambahkan ekspresi tidak bermakna
stop_words_en.update(custom_noise)

# Fungsi Filter
def remove_stopwords_en(tokens):
    return [word for word in tokens if word.lower() not in
stop_words_en]
```

Berikut merupakan daftar kata tambahan untuk *custom noise* diantaranya:

'p', 'meh', 'mmm', 'uh', 'ah', 'aha', 'oh', 'eh', 'hmm', 'huh', 'haha', 'hehe', 'hahaha', 'huhu', 'uhu', 'yay', 'wow', 'whoa', 'oops', 'dang', 'geez', 'yea', 'yup', 'nope', 'ya', 'yuh', 'na', 'hm', 'umm', 'uhh', 'aaa', 'ee', 'oo', 'oof', 'grr', 'argh', 'yeet', 'ugh', 'ew', 'yuck', 'bleh', 'brr', 'nah', 'hmm', 'm', 's', 't', 'd', 'k', 'r', 'u', 'ur', 'ya', 'yo', 'w', 'aa', 'ahaha', 'ab', 'ah', 'ab', 'abb', 'abc', 'abd', 'abe', 'abf', 'abg', 'abh', 'abi', 'abj', 'abk', 'abl', 'abm', 'abn', 'abo', 'abp', 'abq', 'abr', 'abs', 'abt', 'abu', 'aca', 'acai', 'ada', 'adf', 'af', 'afa', 'afi', 'afl', 'ag', 'agaknya', 'agh', 'agni', 'ahah', 'aht', 'aj', 'ajay', 'ajc', 'ajim', 'ajit', 'ajokparoghene', 'akc', 'anc', 'ang', 'anut', 'ao', 'aocs', 'aos', 'ap', 'apb', 'apc', 'api', 'apj', 'app', 'aq', 'arffff', 'arh', 'ari', 'aro', 'asd', 'asf', 'asfckzxcs', 'aso', 'ata', 'au', 'awc', 'awh', 'awk', 'awsomesylar', 'aww', 'awww', 'aya', 'ayas', 'ayi', 'baaaad', 'baant', 'bab', 'bai', 'bp', 'br', 'kd', 'lt'

Proses ini memeriksa setiap token, dan menghapus token yang termasuk dalam daftar *stopword*, sehingga data berkurang menjadi 43.033 data. Hasil *remove stopword* ditunjukkan pada tabel 4.6 berikut.

Tabel 4.6 Hasil *Remove Stopword*

Sebelum *Remove Stopword*	**Setelah *Remove Stopword***
[promotion, girls, are, going, to, bring, it, in, the, next, round, yeah, bring, it, bring, your, store, bought, capsicums]	[promotion, girls, going, bring, next, round, yeah, bring, bring, store, bought, capsicums]
[if, you, had, not, noticed, i, save, my, witty, replies, for, sexists, with, a, little, more, panache, i, dont, want, to, get, above, your, reading, level]	[noticed, save, witty, replies, sexists, little, panache, dont, want, get, reading, level]
[follow, news, reports, of, shooting, death, of, trial]	[follow, news, reports, shooting, death, trial]
[it, starts, thursday, my, classes, are, cancelled, for, tomorrow, though, to]	[starts, thursday, classes, cancelled, tomorrow, though]

7. Lemmatization

Tahapan lematisasi dilakukan untuk mengembalikan setiap kata ke bentuk dasarnya *(lemma)* dengan mempertimbangkan struktur gramatikalnya *(Part of Speech).* Pendekatan ini dinilai lebih akurat dibanding *stemming*, karena tetap mempertahankan bentuk kata yang sah secara linguistik.

Proses lemmatisasi diterapkan menggunakan pustaka *WordNet Lemmatizer* dari NLTK, dengan bantuan POS tagging agar hasilnya lebih sesuai konteks. Berikut contoh potongan kode yang digunakan:

```
lemmatizer = WordNetLemmatizer()

def get_wordnet_pos(word):
    tag = pos_tag([word])[0][1][0].upper()
    return {
        'J': wordnet.ADJ, 'N': wordnet.NOUN,
        'V': wordnet.VERB, 'R': wordnet.ADV
    }.get(tag, wordnet.NOUN)

def lemmatize_tokens(tokens):
```

```
    return [lemmatizer.lemmatize(token,
get_wordnet_pos(token)) for token in tokens]
```

Tahap ini menyempurnakan hasil *pre-processing* sebelumnya sehingga kata-kata memiliki bentuk dasar yang konsisten. Contoh hasilnya seperti yang terlihat pada Tabel 4.7 berikut:

Tabel 4.7 Hasil Lemmatization

Sebelum *Lemmatization*	**Setelah *Lemmatization***
[promotion, girls, going, bring, next, round, yeah, bring, bring, store, bought, capsicums]	promotion, girl, go, bring, next, round, yeah, bring, bring, store, buy, capsicum
[noticed, save, witty, replies, sexists, little, panache, dont, want, get, reading, level]	notice, save, witty, reply, sexist, little, panache, dont, want, get, read, level
[follow, news, reports, shooting, death, trial]	follow, news, report, shoot, death, trial
[starts, thursday, classes, cancelled, tomorrow, though]	start, thursday, class, cancel, tomorrow, though

Setelah melalui seluruh tahapan *pre-processing* data teks telah disederhanakan dan disiapkan untuk proses ekstraksi fitur. Jumlah data setelah *pre-processing* adalah 43.023 baris data. Keseluruhan hasil *pre-processing* terdapat pada gambar 4.1.

	content	label	cleaning	case_folding	normalize	token	stopword_removed	stemmed	stemmed_text	clean_text	lemma	lemma_text
0	In other words #katandandre, your food was cra...	not_cyberbullying	In other words your food was crapilicious	in other words your food was crapilicious	in other words your food was crapilicious	[in, other, words, your, food, was, crapilicious]	[words, food, crapilicious]	[word, food, crapilicious]	word food crapilicious	word food crapilicious	[word, food, crapilicious]	word food crapilicious
1	Why is #aussietv so white? #MKR #theblock #ImA...	not_cyberbullying	Why is so white	why is so white	why is so white	[why, is, so, white]	[white]	[white]	white	white	[white]	white
2	@XochitlSuckkks a classy whore? Or more red ve...	not_cyberbullying	a classy whore Or more red velvet cupcakes	a classy whore or more red velvet cupcakes	a classy whore or more red velvet cupcakes	[a, classy, whore, or, more, red, velvet, cupc...	[classy, whore, red, velvet, cupcakes]	[classy, whore, red, velvet, cupcake]	classy whore red velvet cupcake	classy whore red velvet cupcake	[classy, whore, red, velvet, cupcake]	classy whore red velvet cupcake
3	@Jason_Gio meh. :P thanks for the heads up, b...	not_cyberbullying	meh P thanks for the heads up but not too conc...	meh p thanks for the heads up but not too conc...	meh p thanks for the heads up but not too con...	[meh, p, thanks, for, the, heads, up, but, not...	[thanks, heads, concerned, another, angry, dud...	[thank, head, concern, another, angry, dude, t...	thank head concern another angry dude twitter	thank head concern another angry dude twitter	[thank, head, concern, another, angry, dude, t...	thank head concern another angry dude twitter
4	@RudhoeEnglish This is an ISIS account pretend...	not_cyberbullying	This is an ISIS account pretending to be a Kur...	this is an isis account pretending to be a kur...	this is an isis account pretending to be a kur...	[this, is, an, isis, account, pretending, to, ...	[isis, account, pretending, kurdish, account, ...	[isi, account, pretend, kurdish, account, like...	isi account pretend kurdish account like islam...	isi account pretend kurdish account like islam...	[isi, account, pretend, kurdish, account, like...	isi account pretend kurdish account like islam...
...	...	...	...	...	...	...	...	...	...	...	...	...
95	@KnightfanNeal #UCFPINKPARTY come on stay aliv...	not_cyberbullying	come on stay alive knights nation This is stil...	come on stay alive knights nation this is stil...	come on stay alive knights nation this is stil...	[come, on, stay, alive, knights, nation, this,...	[come, stay, alive, knights, nation, still, ea...	[come, stay, alive, knight, nation, still, ear...	come stay alive knight nation still early bird...	come stay alive knight nation still early bird...	[come, stay, alive, knight, nation, still, ear...	come stay alive knight nation still early bird...
96	The girls need to learn how to churn #MKR	not_cyberbullying	The girls need to learn how to churn	the girls need to learn how to churn	the girls need to learn how to churn	[the, girls, need, to, learn, how, to, churn]	[girls, need, learn, churn]	[girl, need, learn, churn]	girl need learn churn	girl need learn churn	[girl, need, learn, churn]	girl need learn churn
97	Stick to your day jobs girls #MKR	not_cyberbullying	Stick to your day jobs girls	stick to your day jobs girls	stick to your day jobs girls	[stick, to, your, day, jobs, girls]	[stick, day, jobs, girls]	[stick, day, job, girl]	stick day job girl	stick day job girl	[stick, day, job, girl]	stick day job girl
98	How are You? @edcrossfineart We are well...	not_cyberbullying	How are You We are well	how are you we are well	how are you we are well	[how, are, you, we, are, well]	[well]	[well]	well	well	[well]	well
99	That's crap Lynn and Tony should have stayed #MKR	not_cyberbullying	Thats crap Lynn and Tony should have stayed	thats crap lynn and tony should have stayed	thats crap lynn and tony should have stayed	[thats, crap, lynn, and, tony, should, have, s...	[thats, crap, lynn, tony, stayed]	[thats, crap, lynn, tony, stay]	thats crap lynn tony stay	thats crap lynn tony stay	[thats, crap, lynn, tony, stay]	thats crap lynn tony stay

Gambar 4.1 Hasil *Pre-processing*

4.2.3 *Splitting* Data

Proses pemisahan data dilakukan untuk membagi dataset ke dalam dua subset, yakni data latih dan data uji. Tujuannya adalah agar model dapat dilatih menggunakan sebagian data, dan dievaluasi pada data yang belum pernah dilihat sebelumnya. Pembagian ini dilakukan dengan bantuan fungsi *train_test_split* dari pustaka *scikit-learn*, menggunakan rasio 80% untuk pelatihan dan 20% untuk pengujian. Selain itu, teknik *stratified split* diterapkan guna memastikan distribusi label tetap proporsional pada kedua subset. Berikut kode implementasi ditampilkan sebagai berikut:

```
from sklearn.model_selection import train_test_split

# Fitur dan label
X = df['clean_text']   # Fitur teks
y = df['label']        # Label cyberbullying

# Split data: 80% training, 20% testing
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Proses *splitting* data menghasilkan 34.418 data latih dan 8.605 data uji, dengan distribusi label seperti ditunjukkan pada tabel 4.8 berikut.

Tabel 4.8 Distribusi Data Latih dan Data Uji

Label	Data Latih	Data Uji
religion	6.373	1.593
age	6.355	1.589
gender	6.073	1.518
ethnicity	5.999	1.500
not_cyberbullying	5.261	1.316
other_cyberbullying	4.357	1.089
Total	34.418	8.605

4.2.4 *Ekstraksi* Fitur menggunakan TF-IDF

TF-IDF merupakan teknik yang digunakan untuk merepresentasikan teks dalam bentuk vektor numerik dengan mempertimbangkan seberapa sering suatu

kata muncul dalam dokumen serta seberapa jarang kata tersebut muncul di seluruh korpus. Melalui pendekatan ini, kata-kata yang memiliki frekuensi rendah namun memiliki nilai informasi tinggi akan diberikan bobot lebih besar dibandingkan kata-kata umum yang sering muncul.

Ekstraksi fitur dilakukan menggunakan fungsi *TfidfVectorizer* dari pustaka *scikit-learn* dengan konfigurasi sebagai berikut:

1) *ngram_range*=(1,2) dipilih untuk menangkap konteks kata secara lokal melalui unigram dan bigram. Rentang ini dipilih karena n-gram yang lebih panjang menghasilkan fitur yang jarang muncul, sehingga meningkatkan sparsitas dan risiko *overfitting*. Oleh karena itu, rentang ini dianggap seimbang antara kompleksitas dan informasi yang diperoleh.
2) *sublinear_tf=True* untuk melakukan *scaling log(1 + tf)*, menyeimbangkan pengaruh kata dengan frekuensi tinggi.

Berikut potongan kode implementasinya:

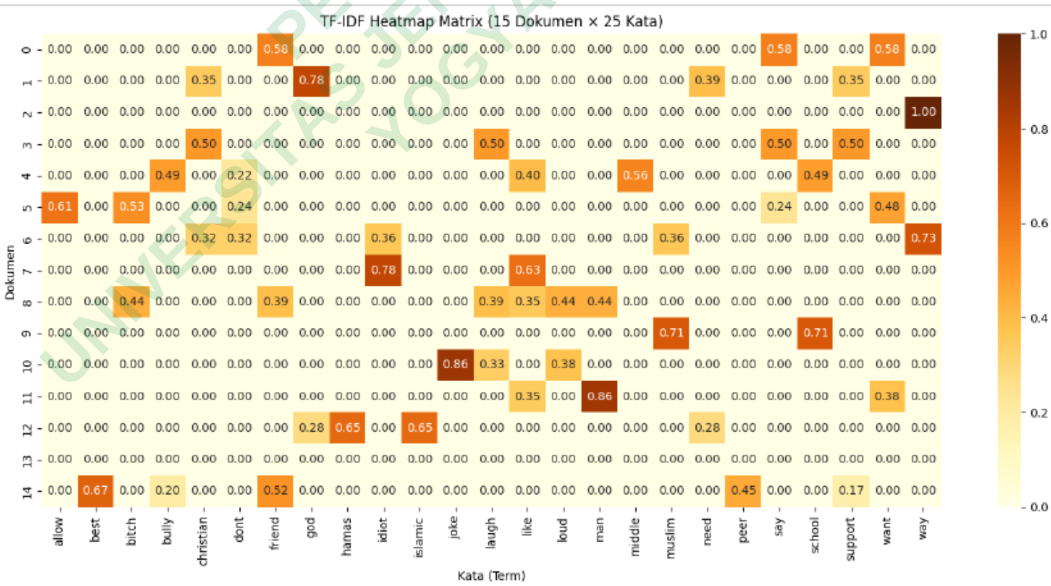
```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(
    ngram_range=(1,2),
    sublinear_tf=True
)

X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

Proses *fit_transform* menghasilkan matriks vektor yang berisi bobot TF-IDF untuk setiap kata di masing-masing dokumen. Struktur hasil *ekstraksi* divisualisasikan pada Gambar 4.2.

abandon	ability	able	abortion	absolute	absolutely	absurd	abuse	abuse bully	abuse woman	...	young	young black	young girl	youth	youtube	youve	ypg	zero
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Gambar 4.2 Hasil TF-IDF *Vectorizer*

Pada gambar 4.3 menampilkan heatmap distribusi bobot TF-IDF dari 25 kata terhadap 15 dokumen. Sumbu horizontal menunjukkan kata-kata penting, sedangkan sumbu vertikal merepresentasikan indeks dokumen. Setiap sel menggambarkan nilai bobot TF-IDF yang diwarnai berdasarkan intensitasnya semakin gelap warnanya, semakin besar nilai TF-IDF kata tersebut pada dokumen terkait.



Gambar 4.3 *Heatmap Matrix* TF-IDF

4.2.5 Klasifikasi Teks *Cyberbullying*

Tahapan klasifikasi untuk memetakan dokumen teks ke dalam kategori yang sesuai. Dalam penelitian ini, digunakan dua pendekatan berbeda sebagai pembanding, yaitu metode *Baseline* berbasis kemiripan kosinus dan metode SVM sebagai pendekatan berbasis *machine learning.*

1. Baseline Classifier

Metode *baseline* dalam penelitian ini menggunakan pendekatan *centroid-based classifier* berbasis *cosine similarity*. Pendekatan ini dipilih karena merupakan metode dasar yang sederhana namun cukup efektif untuk klasifikasi berbasis vektor. Setiap kelas direpresentasikan sebagai vektor pusat (*centroid*) yang dihitung dari seluruh dokumen dalam kelas tersebut setelah direpresentasikan ke dalam bentuk TF-IDF. Kemudian, dokumen uji dibandingkan dengan centroid setiap kelas menggunakan nilai *cosine similarity*, dan diklasifikasikan ke dalam kelas dengan nilai kemiripan tertinggi.

Metode ini bersifat *non-parametrik* karena tidak memerlukan proses pelatihan atau optimasi parameter model, sehingga efisien secara komputasi dan mudah diimplementasikan. Oleh karena itu, pendekatan ini sangat cocok digunakan sebagai acuan awal (*baseline*) untuk mengevaluasi performa model yang lebih kompleks.

Implementasi dari metode ini dimulai dengan inisialisasi objek dan struktur penyimpanan *centroid* untuk setiap kelas sebagai berikut:

```
class TFIDFBaseline:
    def __init__(self):
        self.class_centroids = {}
        self.classes = None
```

Selanjutnya, dilakukan pelatihan model dengan menghitung rata-rata vektor *(centroid)* setiap kelas berdasarkan representasi TF-IDF dokumen latih.

```
 def fit(self, X_train_tfidf, y_train):
        self.classes = np.unique(y_train)
        for cls in self.classes:
            indices = np.where(np.array(y_train) == cls)[0]
```

```
            # pakai mean tanpa convert ke dense besar
            class_matrix = X_train_tfidf[indices]
            centroid = class_matrix.mean(axis=0)
            self.class_centroids[cls] =
np.asarray(centroid).flatten()
```

Menghitung *cosine similarity* dokumen uji terhadap *centroid* semua kelas, lalu memilih kelas dengan nilai tertinggi sebagai prediksi.

```
def predict(self, X_test_tfidf):
        predictions = []
        for i in range(X_test_tfidf.shape[0]):
            vec = X_test_tfidf[i].toarray().flatten()
            similarities = {}
            for cls, centroid in
self.class_centroids.items():
                sim = np.dot(vec, centroid) /
(np.linalg.norm(vec) * np.linalg.norm(centroid) + 1e-10)
                similarities[cls] = sim
            predictions.append(max(similarities,
key=similarities.get))
        return np.array(predictions)
```

2. Klasifikasi menggunakan *Support Vector Machine* (SVM)

Algoritma SVM digunakan dalam penelitian ini karena kemampuannya dalam menangani data berdimensi tinggi, seperti teks. Dengan memanfaatkan kernel linear, model ini dinilai efektif untuk klasifikasi multi kelas. Untuk mengubah teks menjadi bentuk numerik, digunakan metode TF-IDF sebagai teknik representasi fitur agar data dapat diproses dengan optimal oleh SVM.

Berikut adalah implementasi kode klasifikasi menggunakan SVM kernel linear.

```
from sklearn.svm import SVC

svm_model = SVC(kernel='linear', C=1.0, random_state=42)
svm_model.fit(X_train_tfidf, y_train)
y_pred_svm_linear = svm_model.predict(X_test_tfidf)
```

Parameter C=1.0 digunakan sebagai nilai default untuk mengatur *trade-off* antara kompleksitas model dan kesalahan klasifikasi. Semakin kecil nilai C, semakin besar toleransi terhadap kesalahan, sebaliknya nilai C

yang besar akan berusaha meminimalkan kesalahan pada data latih namun berisiko terhadap *overfitting*.

4.2.6 Evaluasi Model

Evaluasi model dilakukan untuk menilai sejauh mana metode klasifikasi mampu mengenali teks yang mengandung unsur *cyberbullying* secara akurat. Penelitian ini menguji dua pendekatan klasifikasi yang berbeda guna membandingkan performanya dalam mendeteksi cyberbullying, yaitu:

1. **Evaluasi *Baseline Classifier***

   Evaluasi *baseline classifier* dilakukan menggunakan data uji sebanyak 8.605 entri. Model *baseline* berbasis *centroid cosine similarity* menunjukkan hasil evaluasi rata-rata sebagai berikut:

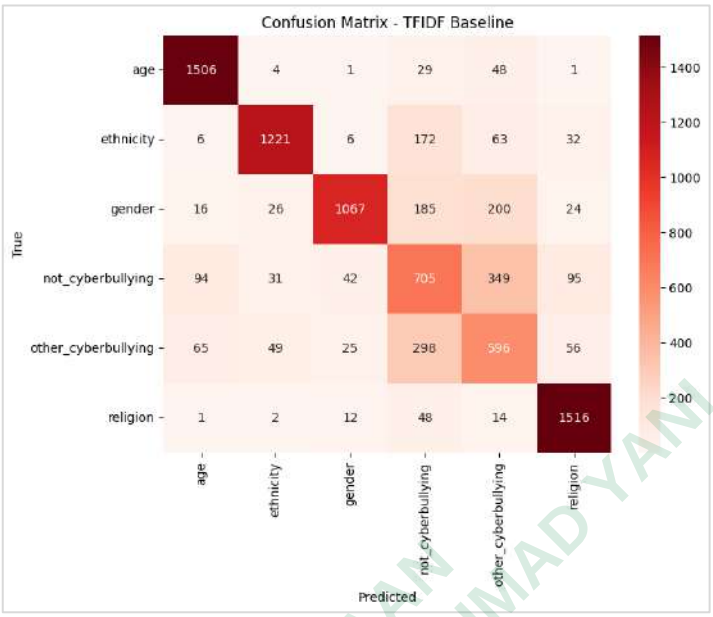
   - Akurasi : 76,8%
   - *Precision* : 78,4%
   - *Recall* : 76,8%
   - *F1-score* : 77,2%

   Tabel 4.9 berikut menampilkan ringkasan hasil *classification report* untuk masing-masing kelas.

Tabel 4.9 *Classification report* Model *Cosine similarity*

Label	***Precision***	***Recall***	***F1-score***
Age	0.89	0.95	0.92
Ethnicity	0.92	0.81	0.86
Gender	0.93	0.70	0.80
Not_*cyberbullying*	0.49	0.54	0.51
Other_*cyberbullying*	0.47	0.55	0.51
religion	0.88	0.95	0.91

Untuk melihat evaluasi, *confusion matrix baseline classifier* ditampilkan pada gambar 4.4 berikut.



Gambar 4.4 Visualisasi *Confusion Matrix Baseline*

Sebagai contoh, berikut perhitungan terhadap label *gender* berdasarkan *confusion matrix*, nilai evaluasi dihitung sebagai berikut:

1) *True Positive* (TP), prediksi benar untuk label *gender* = 1.067.
2) *False Positive* (FP), prediksi salah sebagai *gender* = 86.
3) *False Negative* (FN), label *gender* salah prediksi ke label lain = 451.
4) *True Negative* (TN) Jumlah seluruh data selain *gender* tidak diklasifikasikan sebagai gender = 7.004

Berdasarkan data tersebut, diperoleh metrik evaluasi untuk label *gender* sebagai berikut:

1) $Accuracy = \frac{TP+TN}{Total} = \frac{1.067+7.004}{8.605} = \frac{8.071}{8.605} \approx 93{,}7\%$
2) $Precision = \frac{TP}{TP+FP} = \frac{1.067}{1.067+86} = \frac{1.067}{1.153} \approx 92{,}5\%$
3) $Recall = \frac{(TP)}{(TP+FN)} = \frac{1.067}{1.067+451} = \frac{1.067}{1.518} \approx 70{,}2\%$
4) $F1 - Score = 2 \times \frac{(Precision\ x\ Recall)}{(Precision + Recall)} = 2 \times \frac{0.925 \times 0.702}{0.925+0.702} \approx 79{,}8\%$

Nilai-nilai tersebut konsisten dengan hasil yang ditampilkan pada Tabel 4.9 untuk label *gender*, yang menunjukkan bahwa model cukup baik dalam mendeteksi komentar dengan label *gender*, meskipun masih memiliki

kelemahan dalam mendeteksi seluruh komentar yang seharusnya termasuk kategori tersebut (recall rendah).

Secara Keseluruhan, hasil *baseline* menunjukkan performa cukup baik dalam mendeteksi kemiripan teks, namun masih menghasilkan prediksi salah terutama pada kelas yang memiliki kata-kata ambigu atau konteks mirip (misalnya antara *other_cyberbullying* dan *not_cyberbullying*). Hal ini terjadi karena metode *centroid* hanya mengandalkan kedekatan rata-rata tanpa mempelajari batas pemisah optimal antar kelas.

2. Evaluasi SVM

Evaluasi menggunakan model *Support Vector Machine* (SVM) dengan kernel linear dilakukan menggunakan data uji sebanyak 8.605 entri. Model ini menghasilkan kinerja yang lebih baik dibandingkan baseline. Hasil evaluasi rata-rata ditunjukkan sebagai berikut:

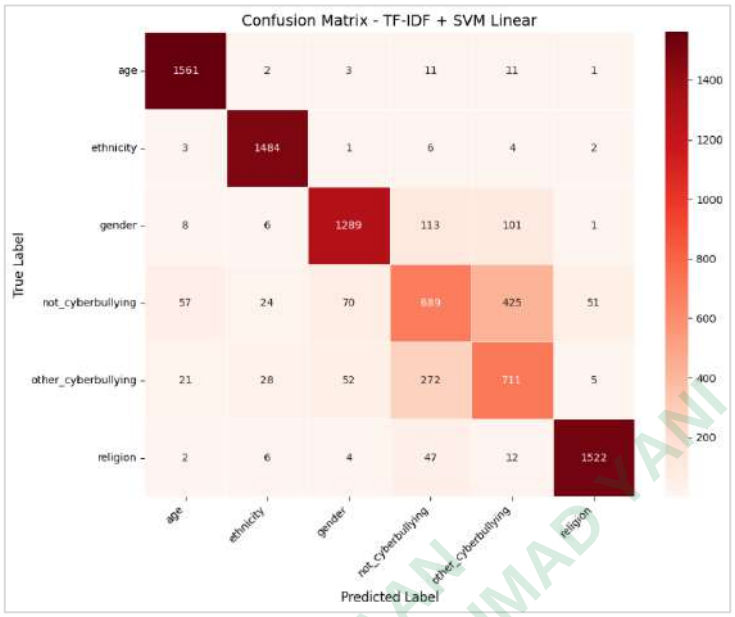
1) Akurasi : 84,32%
2) *Precision* : 84,35%
3) *Recall* : 84,32%
4) *F1-score* : 84,22%

Tabel 4.10 berikut merangkum *classification report* model SVM pada masing-masing label.

Tabel 4.10 *Classification Report* Model SVM

Label	***Precision***	***Recall***	***F1-score***
Age	0.94	0.98	0.96
Ethnicity	0.96	0.99	0.97
Gender	0.91	0.85	0.88
Not_*cyberbullying*	0.61	0.52	0.56
Other_*cyberbullying*	0.56	0.65	0.60
religion	0.96	0.96	0.96

Visualisasi *confusion matrix* tersebut dapat dilihat pada Gambar 4.5 untuk memberikan gambaran yang lebih jelas mengenai performa model.



Gambar 4.5 *Visualisasi Confusion Matrix* SVM

Model SVM menunjukkan performa sangat baik pada label *age*, *ethnicity*, dan *religion*, dengan *F1-score* di atas 0,95. Hal ini menunjukkan bahwa model mampu mengenali pola kata yang eksplisit dan khas pada kategori tersebut. Untuk label *gender*, meskipun *precision* cukup tinggi (0,91), *recall* lebih rendah (0,85), menandakan bahwa sebagian data dengan konteks *gender* tidak berhasil dikenali secara konsisten, dikarenakan penggunaan bahasa yang tidak langsung atau *sarkastik*.

Sementara itu, performa model paling rendah terlihat pada kategori *not_cyberbullying* dan *other_cyberbullying*, dengan *F1-score* masing-masing sebesar 0,56 dan 0,60. Rendahnya nilai ini menunjukkan bahwa model kesulitan membedakan teks netral dengan teks perundungan yang tidak memiliki pola *eksplisit*. Berdasarkan *confusion matrix,* banyak data *not_cyberbullying* terklasifikasi sebagai *other_cyberbullying*, dan sebaliknya. Hal ini mengindikasikan adanya tumpang tindih representasi antar kelas yang disebabkan oleh keterbatasan TF-IDF dalam menangkap makna kontekstual.

Berikut adalah evaluasi kinerja model terhadap label *gender* berdasarkan *confusion matrix*:

1) True Positive (TP) = 1.289 (prediksi benar sebagai gender)

2) False Positive (FP) = 130 (prediksi gender dari kelas lain: age, ethnicity, not_cyberbullying, other_cyberbullying, religion)
3) False Negative (FN) = 229 (label gender yang diprediksi salah ke kelas lain)
4) True Negative (TN) = 8.605 - TP - FP - FN = 6.957

Dengan demikian diperoleh matrik evaluasi:

1) $Accuracy = \frac{TP+TN}{Total} = \frac{1.289+6.957}{8.605} = \frac{8.246}{8.605} \approx 95{,}8\%$
2) $Precision = \frac{TP}{TP+FP} = \frac{1.289}{1.289+130} = \frac{1.289}{1.419} \approx 90\%$
3) $Recall = \frac{(TP)}{(TP+FN)} = \frac{1.289}{1.289+229} = \frac{1.289}{1.518} \approx 84{,}9\%$
4) $F1 - Score = 2 \times \frac{(Precision\ x\ Recall)}{(Precision + Recall)} = 2 \times \frac{0.90 \times 0.849}{0.90+0.849} \approx 87\%$

Hasil *precision* yang tinggi menunjukkan bahwa sebagian besar prediksi untuk kelas gender adalah benar, dan recall yang meningkat menunjukkan kemampuan model dalam mengenali lebih banyak data gender yang sebenarnya.

3. Analisis Perbandingan

Berdasarkan hasil evaluasi pada dua pendekatan klasifikasi, yaitu *baseline classifier* berbasis centroid dengan *cosine similarity* dan *Support Vector Machine* (SVM) linear, dengan hasil perbandingan kinerja model keduanya pada tabel 4.11.

Tabel 4.11 Perbandingan Kinerja Model

Metode	Akurasi	*Precision*	*Recall*	*F1-Score*
Baseline	76,8%	78,4%	76,8%	77,2%
Linear SVM	84,32%	84,35%	84,32%	84,22%

Dari hasil evaluasi tersebut, dapat disimpulkan bahwa model SVM menunjukkan performa yang lebih unggul dibandingkan pendekatan baseline. Peningkatan yang konsisten pada seluruh metrik evaluasi menunjukkan bahwa SVM tidak hanya lebih akurat dalam memprediksi kelas, tetapi juga lebih andal dalam mengenali variasi teks yang mengandung unsur *cyberbullying*. Hal ini disebabkan oleh kemampuan

SVM untuk membentuk batas klasifikasi yang optimal pada ruang fitur berdimensi tinggi, sehingga mampu menangkap kompleksitas bahasa alami dan konteks sosial dalam data teks media sosial.

Namun demikian, evaluasi mendalam mengungkapkan bahwa model SVM masih mengalami kesulitan dalam mengklasifikasikan dua kategori tertentu, yaitu *not_cyberbullying* dan *other_cyberbullying*, yang menunjukkan nilai *F1-score* rendah. Hal ini menjadi temuan penting dalam penelitian ini karena kedua kelas tersebut memiliki karakteristik yang lebih kompleks dan ambigu. Kategori *not_cyberbullying* mencakup seluruh konten netral yang tidak mengandung unsur perundungan, sehingga sangat bervariasi dan tidak mudah dipetakan ke dalam pola tertentu. Sementara itu, *other_cyberbullying* merupakan kelas generik tanpa kata kunci eksplisit seperti halnya kelas *gender*, *religion*, atau *ethnicity*, sehingga sering terjadi tumpang tindih klasifikasi.

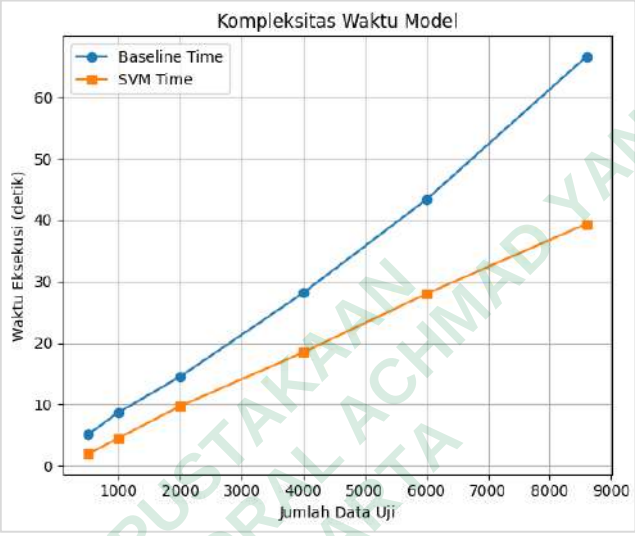
Rendahnya performa pada kedua kategori tersebut mengindikasikan keterbatasan representasi fitur berbasis TF-IDF dalam menangkap makna implisit, konteks sarkastik, atau nuansa emosional dalam teks. Dengan demikian, meskipun SVM berhasil mengungguli *baseline* secara keseluruhan, tantangan utama dalam klasifikasi tetap ada pada teks-teks dengan konteks *ambigu* dan representasi linguistik yang tidak eksplisit.

4. Analisis Kompleksitas dan Kompatibilas

Pengujian kompleksitas waktu dilakukan dengan mengukur durasi prediksi kedua model terhadap enam skenario jumlah data uji yaitu 500, 1.000, 2.000, 4.000, 6.000, dan 8.605 data. Hasil pengukuran waktu ditampilkan pada Gambar 4.6.

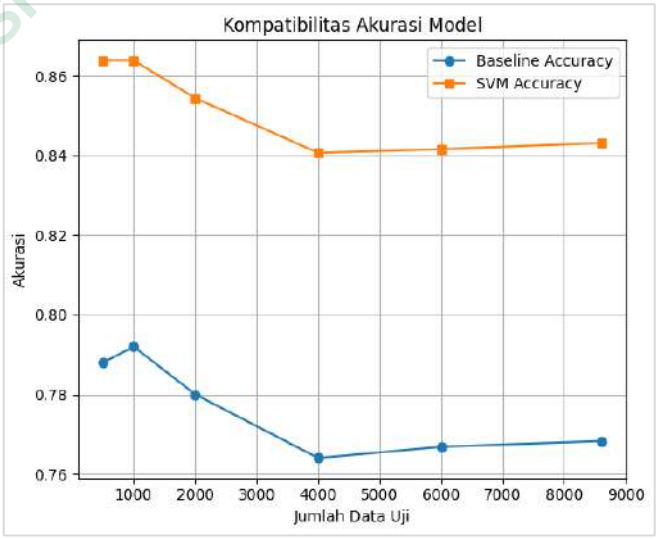
Gambar 4.6 menunjukkan bahwa waktu eksekusi model baseline meningkat secara signifikan seiring bertambahnya jumlah data uji. Hal ini disebabkan karena proses prediksi model baseline memerlukan perhitungan kemiripan vektor (cosine similarity) antara setiap data uji dengan centroid dari masing-masing kelas, yang berarti proses komputasi dilakukan berulang kali untuk setiap data. Sebaliknya, model SVM menunjukkan

waktu prediksi yang jauh lebih efisien dan pertumbuhannya lebih linear. Hal ini karena proses prediksi pada SVM hanya memanfaatkan fungsi keputusan *(decision function)* yang telah dibentuk pada tahap pelatihan, tanpa memerlukan perhitungan jarak antar vektor secara eksplisit terhadap semua kelas.



Gambar 4.6 Grafik Perbandingan Kompleksitas Waktu

Dengan demikian, model SVM terbukti lebih unggul dalam hal efisiensi waktu pengujian, menjadikannya lebih layak diterapkan pada skenario dengan jumlah data uji yang besar.



Gambar 4.7 Grafik Perbandingan Akurasi

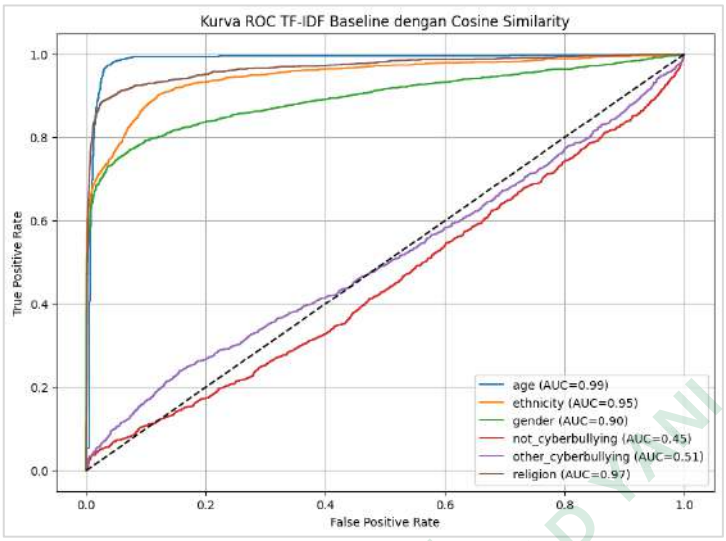
Gambar 4.7 menampilkan grafik kompatibilitas model terhadap akurasi prediksi pada variasi jumlah data uji yang sama. Model SVM menujukkan akurasi lebih tinggi dibandingkan *baseline*. Namun, terjadi penurunan akurasi SVM dari 86% menjadi 84% ketika jumlah data uji bertambah, yang mengindikasikan adanya potensi overfitting ringan terhadap data latih. Namun setelah titik 4.000 data, akurasi SVM cenderung stabil, dengan fluktuasi kecil menuju 84,3% pada 8.605 data uji.

Sementara itu, model baseline memiliki akurasi yang relatif stabil namun rendah, berkisar antara 76% hingga 78%. model ini tidak terlalu sensitif terhadap volume data karena menggunakan strategi sederhana berbasis kemiripan vektor rata-rata, yang tidak secara aktif belajar pola klasifikasi dari data.

Perbadaan ini menunjukkan bahwa pendekatan berbasis SVM lebih mampu memodelkan kompleksitas semantik dan konteks linguistik pada data teks sosial media, dibandingkan pendekatan berbasis representasi statis *(centroid cosine similarity)*.

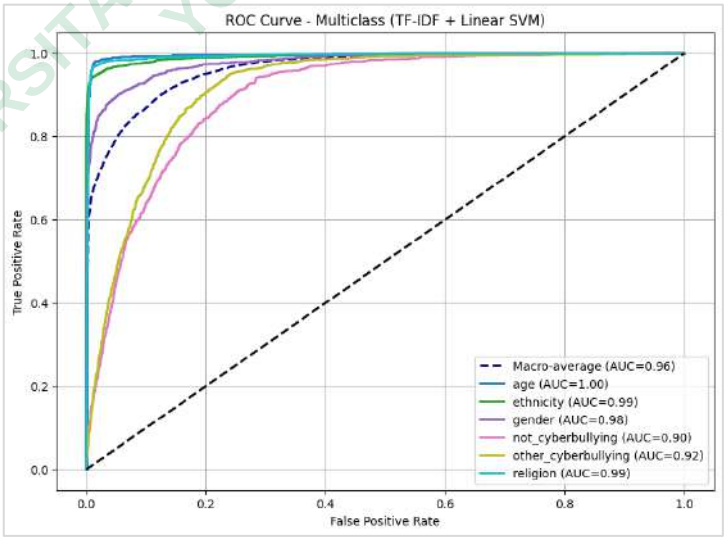
8. Kurva ROC

Kurva ROC *(Receiver Operating Characteristic)* digunakan untuk menilai performa model klasifikasi dalam membedakan kelas positif dan negatif pada setiap kategori. Grafik ini menggambarkan hubungan antara *true positive rate* (sensitivitas) dan *false positive rate* (1 - spesifisitas), sehingga memberikan gambaran mengenai *trade-off* antara deteksi yang benar dan kesalahan klasifikasi.



Gambar 4.8 Kurva ROC *Baseline*

Pada Gambar 4.8, model *baseline* dengan *cosine similarity* terlihat mampu memisahkan kelas *age*, *religion, dan ethnicity* dengan nilai AUC tinggi (di atas 0,9). Namun, AUC untuk not_*cyberbullying* dan other_*cyberbullying* cenderung rendah karena kontennya lebih bervariasi dan mirip satu sama lain. Ini menandakan pendekatan *baseline* cukup efektif untuk kelas berpola jelas, tetapi kurang optimal menangani kelas yang ambigu.



Gambar 4.9 Kurva ROC SVM

Sementara itu, pada Gambar 4.11, model SVM menunjukkan AUC yang lebih tinggi dan stabil di semua kelas. Nilai tertinggi tercapai pada

kelas age (1,00), ethnicity (0,99), religion (0,99), gender (0,98), not_*cyberbullying* (0,90), dan other_*cyberbullying* (0,92).

Namun, meskipun nilai AUC tergolong tinggi, terdapat perbedaan dengan metrik *F1-score*, khususnya pada *not_cyberbullying* yang hanya mencapai 0,56. Hal ini mengindikasikan bahwa AUC menggambarkan kemampuan pemisahan kelas secara keseluruhan, tanpa bergantung pada satu *threshold*. *F1-score* mengukur performa prediksi aktual pada *threshold* default (biasanya 0,5). Model dapat mengenali pola umum, namun belum optimal dalam menetapkan ambang batas klasifikasi yang tepat untuk data ambigu. Secara keseluruhan, SVM memberikan pemisahan yang lebih baik antara teks *cyberbullying* dan non-*cyberbullying*, termasuk pada kategori dengan konteks tumpang tindih, sehingga performanya lebih baik.