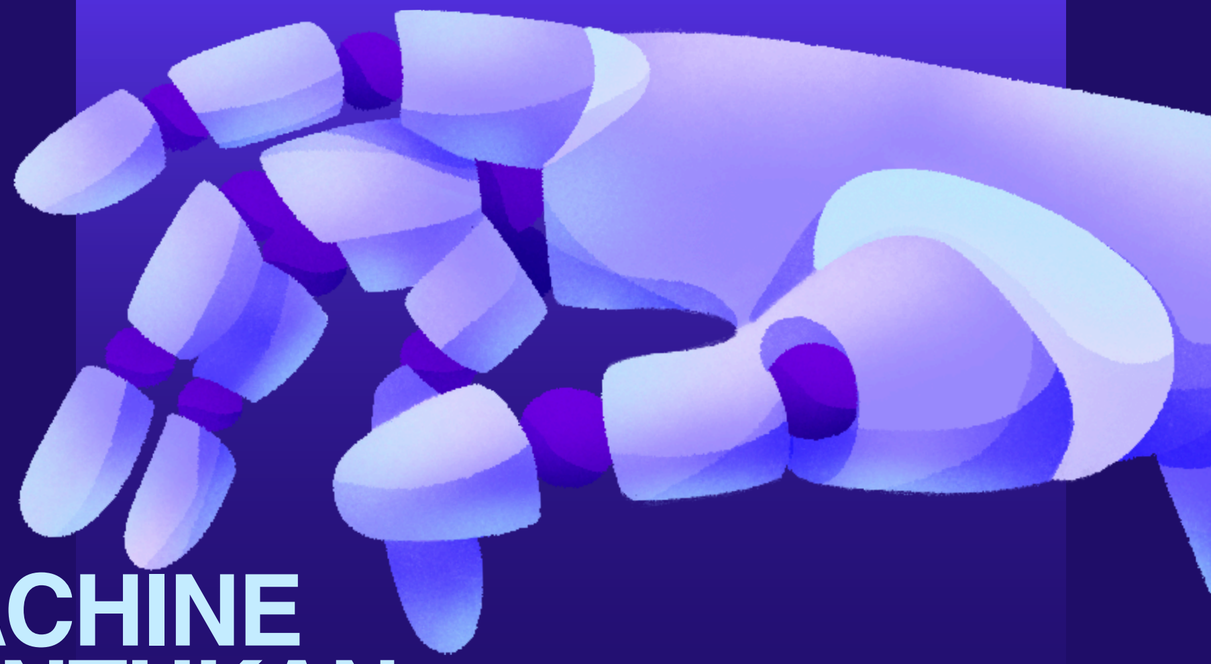
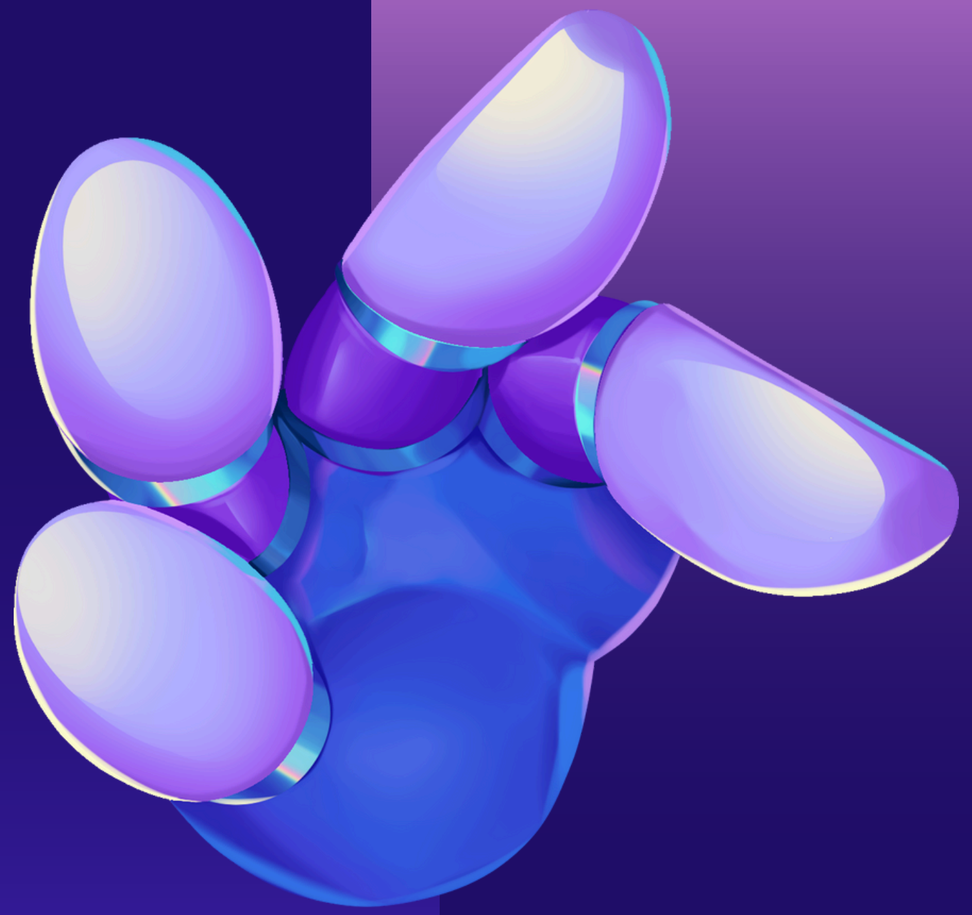


PENGGUNAAN MODEL MACHINE LEARNING DALAM PEMBENTUKAN MODEL KLASIFIKASI SUICIDAL THOUGHTS DENGAN MENGGUNAKAN DATA TWITTER/X

ANGGOTA KELOMPOK

1. Bagus Cipta Pratama (23/516539/PA/22097)
2. M. Akmal Fauzan (23/519741/PA/22303)
3. Daffa Harikhsan (23/513044/PA/21918)
4. Alia Hafida Rahmah (23/520426/PA/22371)
5. Amelia Syahla A. S. (23/512057/PA/21866)



APA YANG AKAN DIBAHAS?

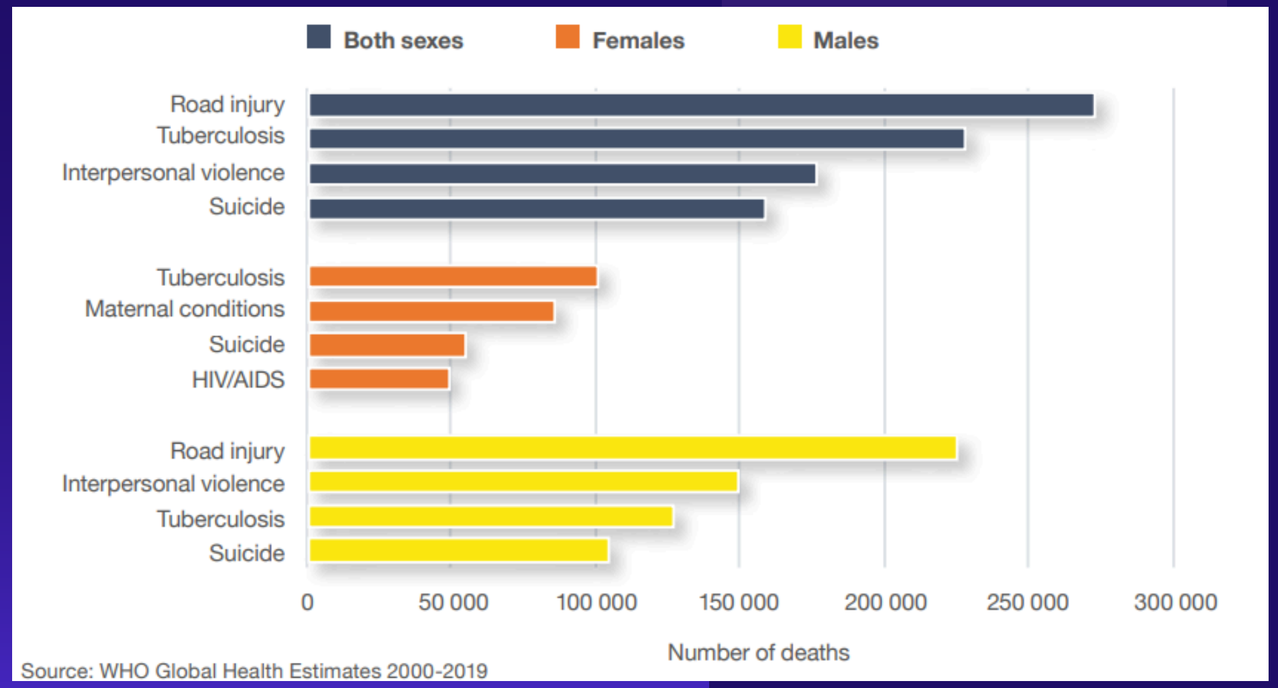
Chapter 1 Latar Belakang

Chapter 2 Metodologi

Chapter 3 Implementasi

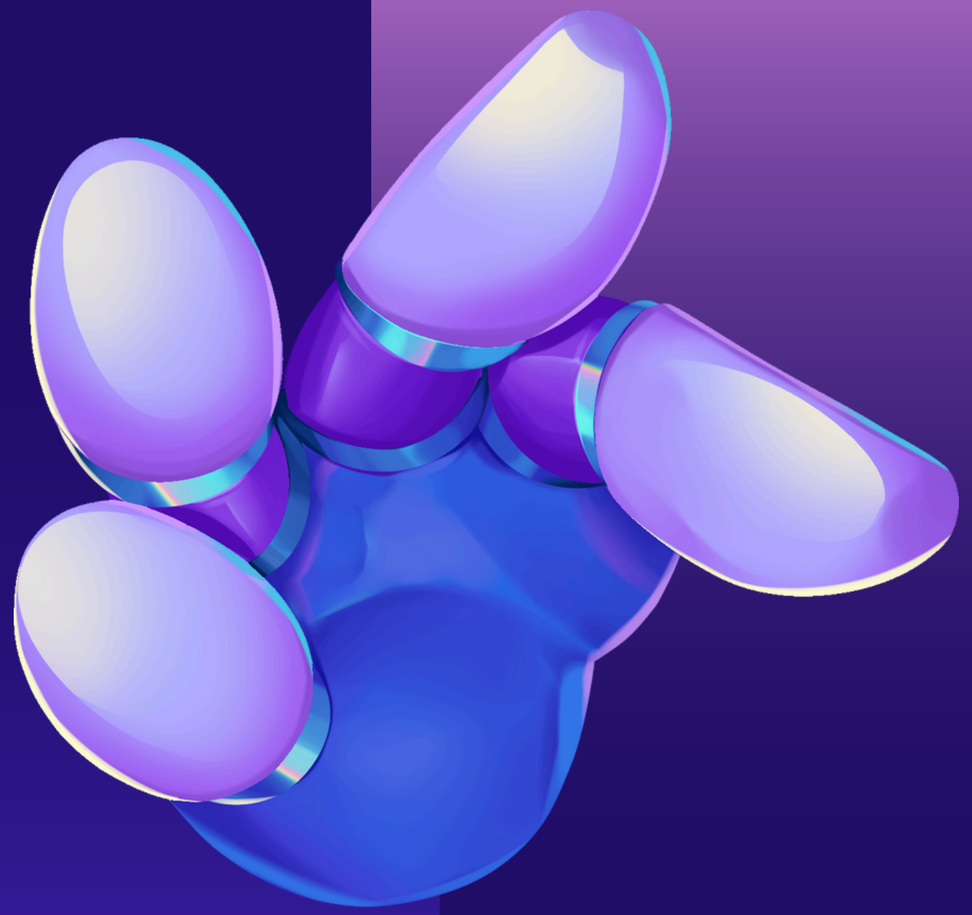
Chapter 4 Kelebihan dan Kekurangan

LATAR BELAKANG



Setiap tahun, 703,000 orang di dunia mengakhiri hidupnya, dengan bunuh diri menjadi penyebab kematian keempat pada usia 15-29 tahun. Di Indonesia, 6544 orang meninggal karena bunuh diri pada 2019, dengan banyak yang mengalami pemikiran bunuh diri atau melukai diri sendiri.

Twitter, yang banyak diakses, dapat digunakan untuk mendeteksi perilaku bunuh diri dengan bantuan machine learning, memungkinkan deteksi dini dan pengarahan kepada layanan bantuan seperti hotline bunuh diri atau psikolog.



METHODOLOGY

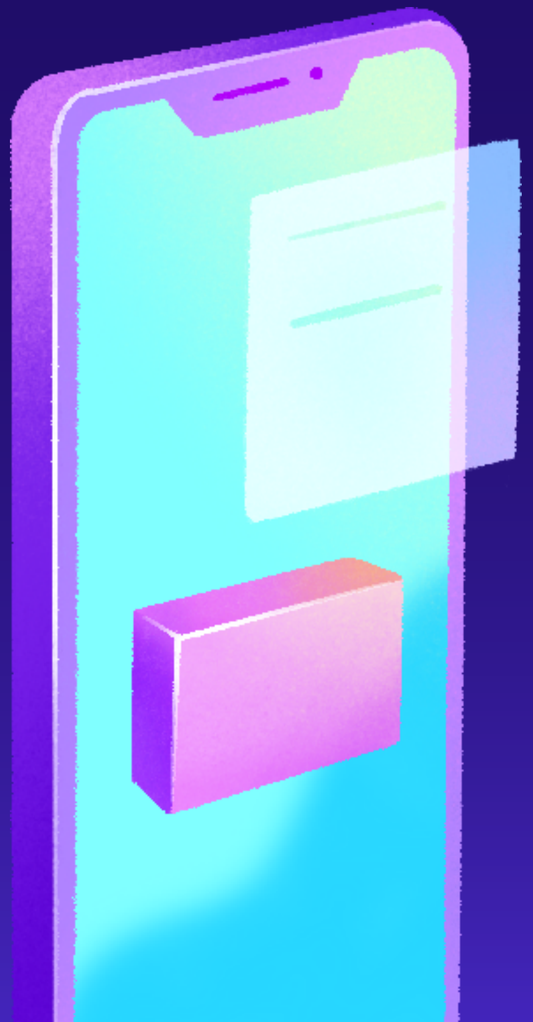
Fase 1 Data Collection

Fase 2 Preprocessing Data

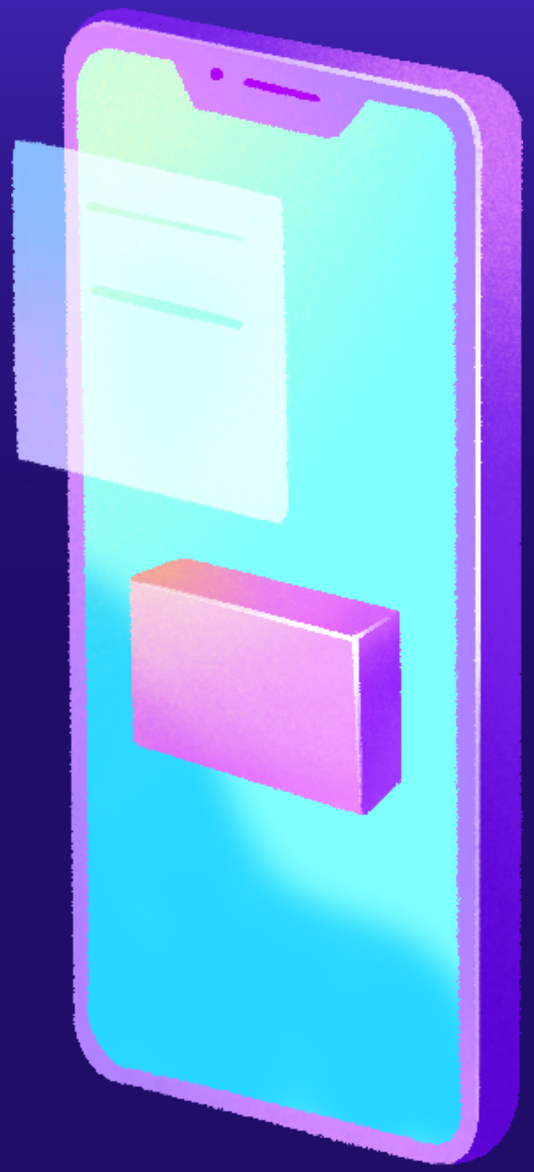
Fase 3 Modelling

Fase 4 Validasi dan Evaluasi

FASE 1
DATA COLLECTION



Dataset yang digunakan oleh kami adalah data tweet dari jejaring sosial X yang dikumpulkan oleh kami menggunakan metode scraping Twitter dengan bahasa pemrograman Python menggunakan kata kunci yang berkaitan dengan bunuh diri seperti “capek hidup”, “depresi”, “nyerah”, dan sebagainya. Kami mendapat sekitar 3000an data tweet dalam bentuk csv yang masih harus di preprocessing lagi.



FASE 2
PREPROCESSING

DATA CLEANING

```
text = re.sub(r'@[A-Za-z0-9_]+', '', text)
text = re.sub(r'#\w+', '', text)
text = re.sub(r'RT[\s]+', '', text)
text = re.sub(r'https?://\S+', '', text)
text = re.sub(r' [^A-Za-z0-9\']', '', text)
text = text.replace("&amp;", "&")
emoji_pattern = re.compile(
 "["
 u"\U0001F600-\U0001F64F"
 u"\U0001F300-\U0001F5FF"
 u"\U0001F680-\U0001F6FF"
 u"\U0001F1E0-\U0001F1FF"
 u"\U00002500-\U00002BEF"
 u"\U00002702-\U000027B0"
 u"\U000024C2-\U0001F251"
 "]+",
 flags=re.UNICODE,
 )
 text = emoji_pattern.sub(r'', text)
```

...

```
text = text.translate(str.maketrans('', '', string.punctuation))
text = re.sub(r'\s+', ' ', text).strip()
```

"RT @ugmuser Mata kuliah TPSC! 😊 #seru https://ugm.ac.id"	"RT Mata kuliah TPSC! 😊 #seru https://ugm.ac.id"
"RT Mata kuliah TPSC! 😊 #seru https://ugm.ac.id"	"RT Mata kuliah TPSC! 😊 https://ugm.ac.id"
"RT Mata kuliah TPSC! 😊 https://ugm.ac.id"	"Mata kuliah TPSC! 😊 https://ugm.ac.id"
"Mata kuliah TPSC! 😊 https://ugm.ac.id"	"Mata kuliah TPSC! 😊"
"Mata kuliah TPSC!"	"Mata kuliah TPSC!"
"Mata kuliah TPSC!"	"Mata kuliah TPSC"

FASE 2
PREPROCESSING

NORMALISASI

tahap ini ditujukan untuk setiap teks yang mengandung singkatan atau kata tidak baku akan digantikan dengan bentuk yang lebih baku. Misalnya, kata seperti "ovt" akan digantikan dengan "overthinking", dan "gak" dengan "tidak".

```
def normalisasi(str_text):
    for key, value in norm.items():
        str_text = re.sub(rf'\b{re.escape(key)}\b', value, str_text)
    return str_text

df['full_text'] = df['full_text'].apply(lambda x: normalisasi(x))
```

FASE 2
PREPROCESSING

SPELL CORRECTION

pada tahap ini dilakukan perbaikan kesalahan ejaan dalam teks dengan memanfaatkan pustaka pyspellchecker. Fungsi correct_spellings pertama-tama membagi teks menjadi kata-kata, kemudian mendeteksi kata-kata yang salah eja dan kemudian memperbaikinya .

```python
def correct_spellings(text):
    spell = SpellChecker()
    corrected_text = []
    misspelled_words = spell.unknown(text.split())

    for word in text.split():
        if word in misspelled_words:
            # Get the correction or keep the original word if None
            correction = spell.correction(word)
            corrected_text.append(correction if correction is not None else word)
        else:
            corrected_text.append(word)

    return " ".join(corrected_text)
```

FASE 2
PREPROCESSING

STOPWORD DELETION

Tahap ini ditujukan untuk menghapus setiap kata yang termasuk dalam daftar stopword, baik bawaan dari Sastrawi maupun tambahan yang didefinisikan pengguna.

```
import Sastrawi
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory,
more_stop_words = ["di", "yang", "dll", "fess", "yuai", "yujiem"]

stop_words = StopWordRemoverFactory().get_stop_words()
stop_words.extend(more_stop_words)

new_array = ArrayDictionary(stop_words)
stop_words_remover_new = StopWordRemover(new_array)

def stopword(str_text):
  str_text = stop_words_remover_new.remove(str_text)
  return str_text

df['full_text'] = df['full_text'].apply(lambda x: stopword(x))
df.head()
```

FASE 2
PREPROCESSING

TOKENIZATION

proses membagi teks menjadi unit-unit kecil

```
tokenized = df['full_text'].apply(lambda x:x.split())
tokenized
```

	full_text
0	[depresi, bukan, sekadar, merasa, sedih, murung]
1	[melakukan, adegan, dewasa, ada, bikin, sedih,...
2	[seringai, baru, merilis, pulang, tersedia, pl...
3	[tahajud, obat, ajaib, depresi, kecemasan, ber...
4	[pernah, terpikirkan, untuk, uninstall, instag...

FASE 2
PREPROCESSING

LEMMATIZATION

mengubah kata menjadi kata dasarnya

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

tokenized = df['full_text'].apply(lambda x: x.split())

lemmatized_text = tokenized.apply(lambda x: ' '.join([stemmer.stem(word) for word in x]))

df['lemmatized_text'] = lemmatized_text

print(df[['full_text', 'lemmatized_text']].head())
```

FASE 2
PREPROCESSING

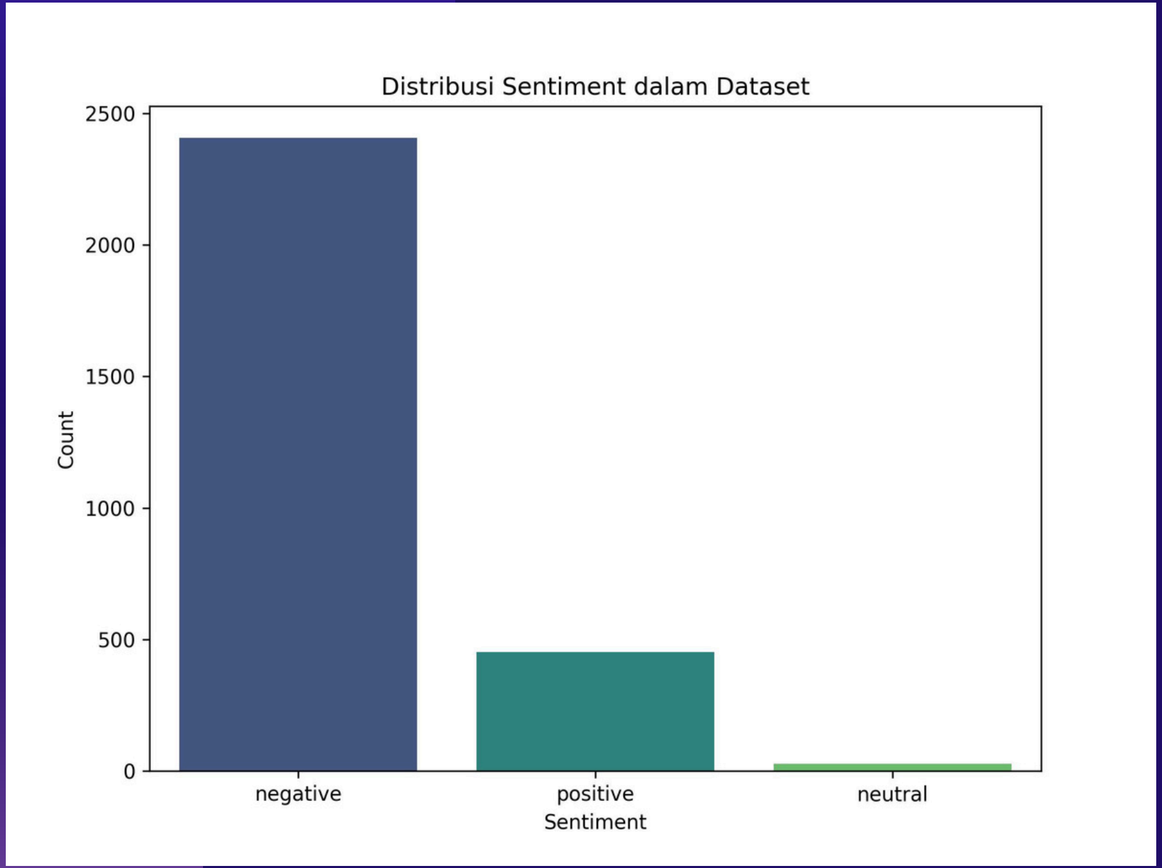
LABELING

menggunakan model Distilbert. Labeling bertujuan untuk melabeli kalimat kalimat yang termasuk neutral, positive, maupun negative.

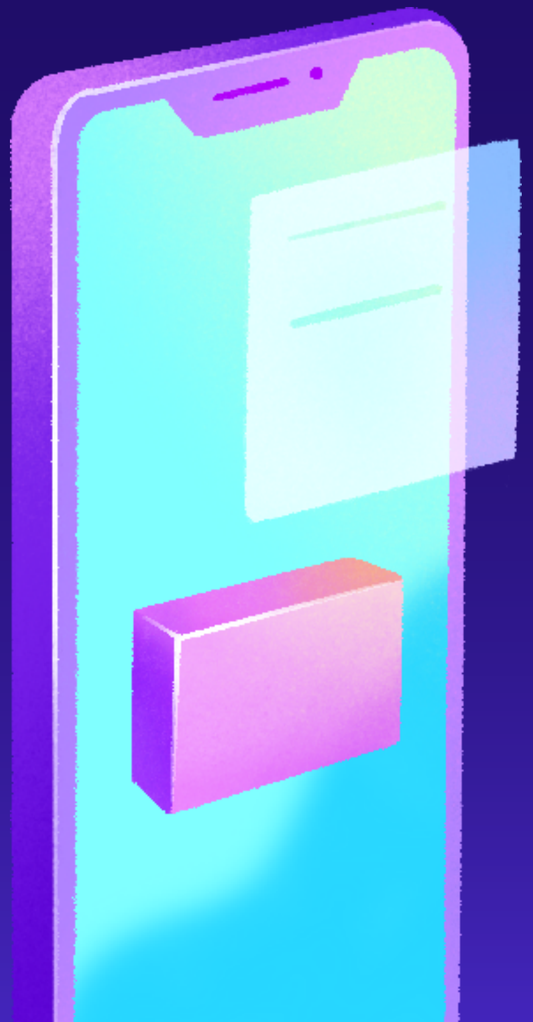
```
# Load model directly
from transformers import AutoTokenizer, AutoModelForSequenceClassification

tokenizer = AutoTokenizer.from_pretrained("lxyuan/distilbert-base-multilingual-cased-sentiments-student")
model = AutoModelForSequenceClassification.from_pretrained("lxyuan/distilbert-base-multilingual-cased-sentiments-student")
```

HASIL LABELLING

Setelah melakukan labelling , dari dataset yang ada didapat mayoritas data sejumlah 2407 tweet negative , 452 tweet positive dan 27 tweet netral



FASE 3
FINE TUNE DAN MODELLING



Random Forest🌳:

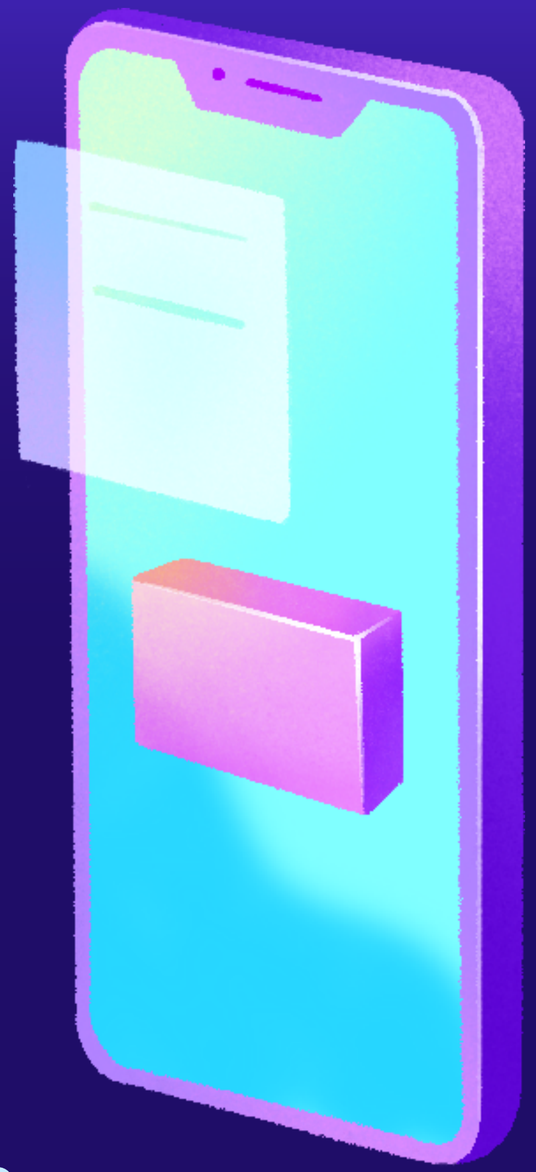
- Menggabungkan banyak **pohon keputusan (decision trees) untuk membuat prediksi**.
- Efektif dalam menangani data terstruktur dan dapat mengurangi overfitting.

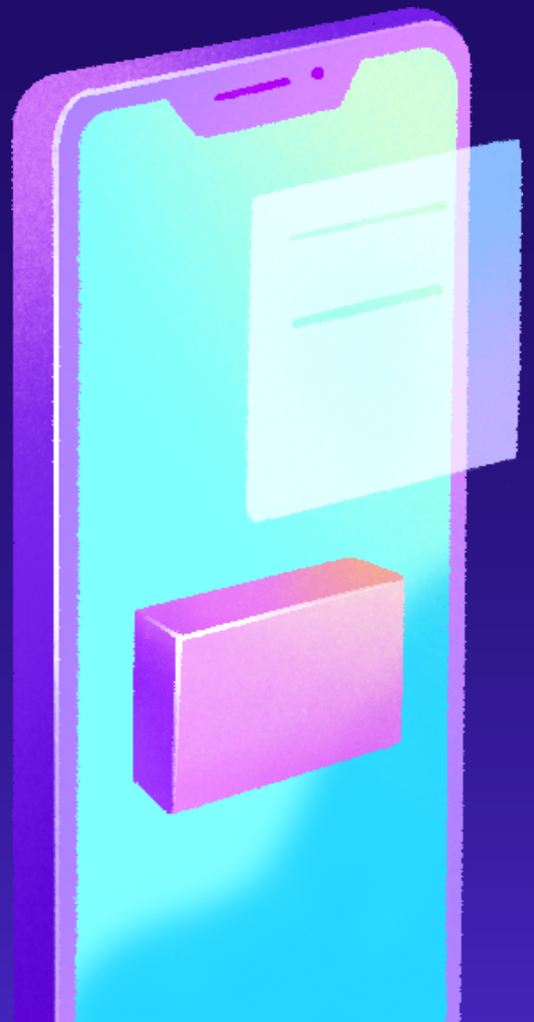
SVM (Support Vector Machine)🤖:

- Pendekatan matematis untuk **menemukan garis atau batas (hyperplane) terbaik yang memisahkan data** ke dalam kategori yang berbeda.
- Baik untuk dataset kecil hingga menengah.

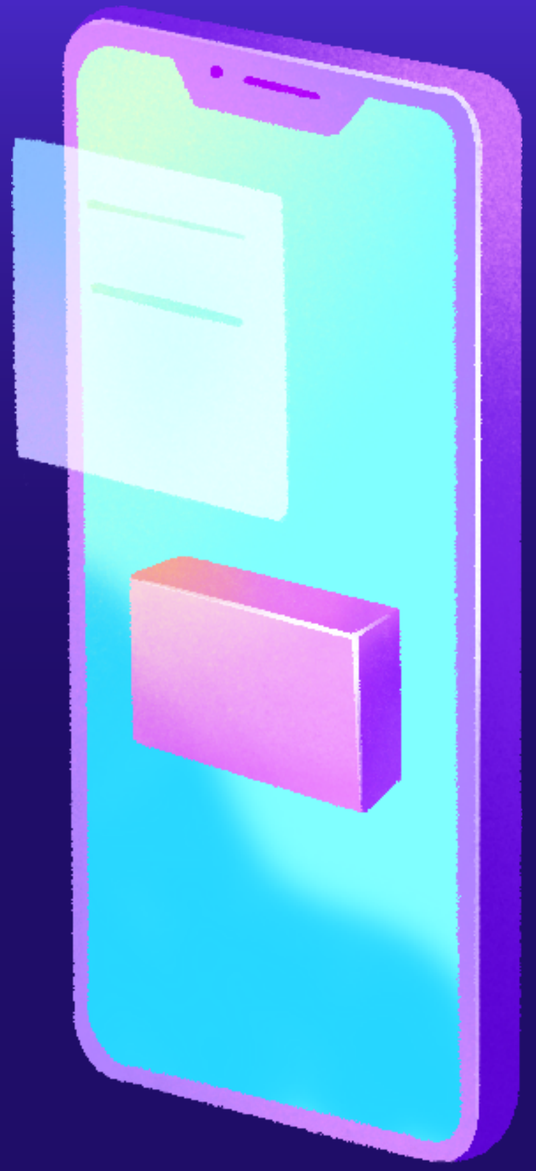
IndoBERT📚:

- Model berbasis Transformer yang menggunakan jaringan saraf dalam (deep learning) dan mekanisme atensi (attention mechanism) untuk **memahami konteks teks secara menyeluruh**.
- Unggul dalam memahami hubungan antar kata serta konteks kalimat secara mendalam





FASE 4
VALIDASI DAN AKURASI



MODEL SUPPORT VECTOR MACHINE

Model SVM sebenarnya kurang cocok digunakan untuk NLP. Kami mendapatnya test accuracy hanya sebesar 0.8480. Hasil ini belum cukup akurat untuk digunakan dalam sistem NLP. Model ini juga masih memiliki kelemahan dalam membedakan kalimat neutral dan positive/negative

```
Support Vector Machine Results:
Validation Accuracy: 0.8270
Test Accuracy: 0.8408

Classification Report (SVM):
              precision    recall  f1-score   support

    Negative       0.86      0.97      0.91       241
     Neutral       0.00      0.00      0.00         3
    Positive       0.53      0.20      0.29        45

    accuracy                           0.84       289
   macro avg       0.46      0.39      0.40       289
weighted avg       0.80      0.84      0.81       289
```

MODEL RANDOM FOREST

Model Random Forest juga kesusahan dalam membedakan kalimat neutral. Kami memperkirakan model kami bias dalam mengenali kalimat negative, karena banyaknya data yang negative. Sehingga hanya memperoleh akurasi dengan model Random Forest sebesar 0.84

	precision	recall	f1-score	support
Negative	0.85	0.99	0.92	241
Neutral	0.00	0.00	0.00	3
Positive	0.60	0.13	0.22	45
accuracy			0.84	289
macro avg	0.48	0.37	0.38	289
weighted avg	0.80	0.84	0.80	289

MODEL INDO-BERT

Model ini paling bagus diantara model lain yang kami coba. Model Indo-Bert bagus dalam membangun model untuk NLP. Namun, kelemahan pada topik kami yaitu model kesulitan dalam membedakan neutral dengan negative.

	precision	recall	f1-score	support
Negative	0.91	0.94	0.93	484
Neutral	0.00	0.00	0.00	5
Positive	0.60	0.52	0.56	91
accuracy			0.87	580
macro avg	0.50	0.49	0.49	580
weighted avg	0.85	0.87	0.86	580

	Subject	Category	pred
0	ringan tugas bu jangan tekan diri terlalu kera...	neutral	negative
1	gue udah capek liat negara koar koar hidup mak...	negative	negative
2	hening panjang jalan pulang lelah minta tetap ...	negative	negative
3	capek hidup banten kalau korupsi leceh pindah ...	negative	negative
4	paruh hidup hilang mikat orang paruh lain hila...	negative	negative
..	...	...	...
281	kesi yh udah gantung diri pny kaki	negative	negative
282	1 depresi arti kurang ibadahamp syukur padahal...	negative	negative
283	eiser nim jangan laku itu ya kalau gantung dir...	negative	negative
284	sampai bubar sih ingin liat orangorang keluarg...	negative	negative
285	hidup derita tahan hidup arti temu makna pende...	negative	negative

[286 rows x 3 columns]

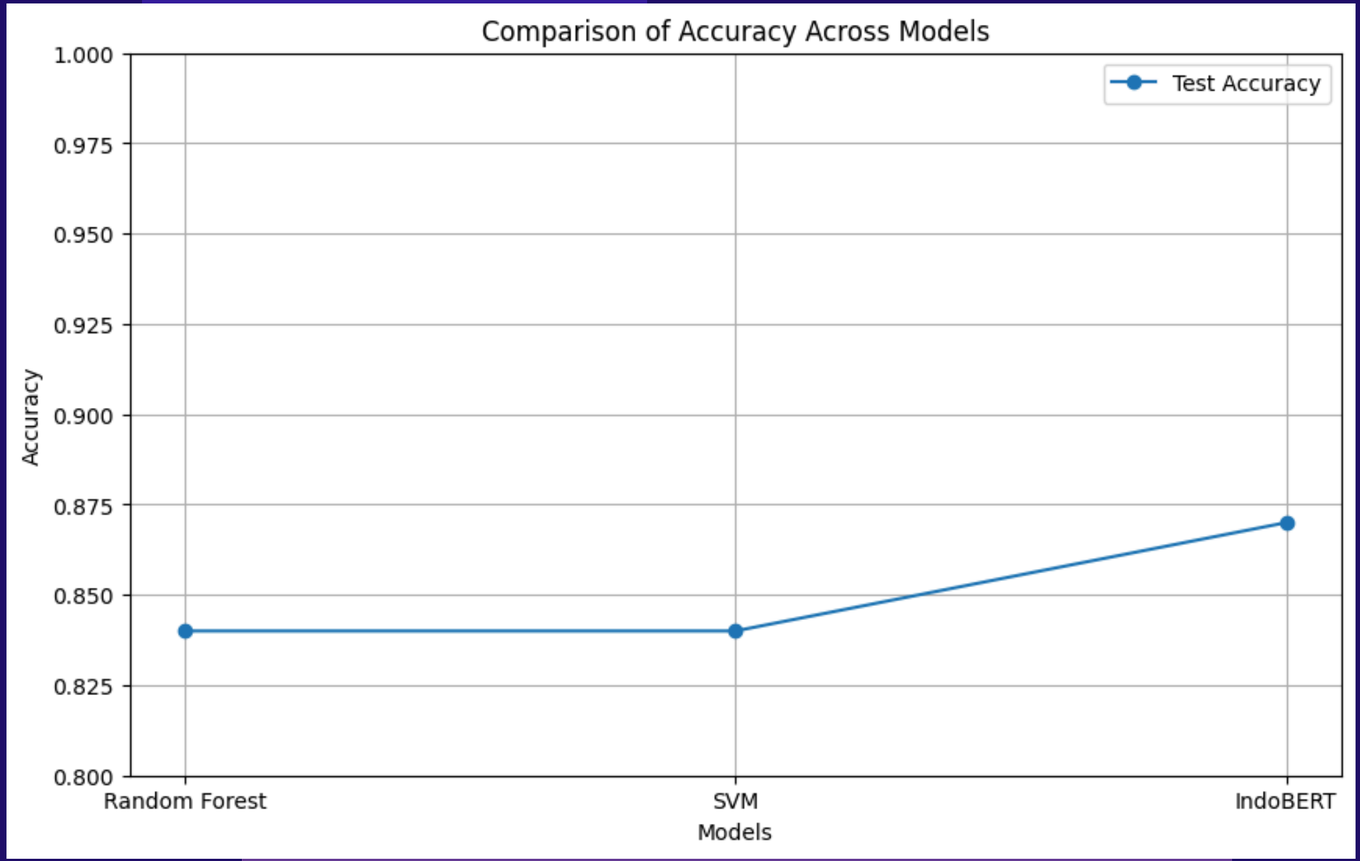
IMPLEMENTASI

Link Google Colab:

https://colab.research.google.com/drive/1xgsFLlj5MxSutfGUnPW6Q2izh-MRwj4E?usp=sharing

ANALISIS

Berdasarkan grafik, IndoBERT unggul dengan Test Accuracy sebesar 87%, lebih tinggi dibandingkan Random Forest dan SVM yang mencapai 84%. Hal ini menunjukkan keunggulan IndoBERT dalam memahami konteks teks yang lebih kompleks, sehingga mampu memberikan performa lebih baik pada data pengujian dibandingkan model konvensional.



KELEBIHAN DAN KEKURANGAN

KELEBIHAN PENELITIAN KAMI

- Berfungsi dalam aspek kesehatan mental terutama untuk pengguna media sosial X
- Data kami original
- Menguji Indo-Bert untuk mengolah data kami, membuktikan dan mengevaluasi penelitian yang ada sebelumnya

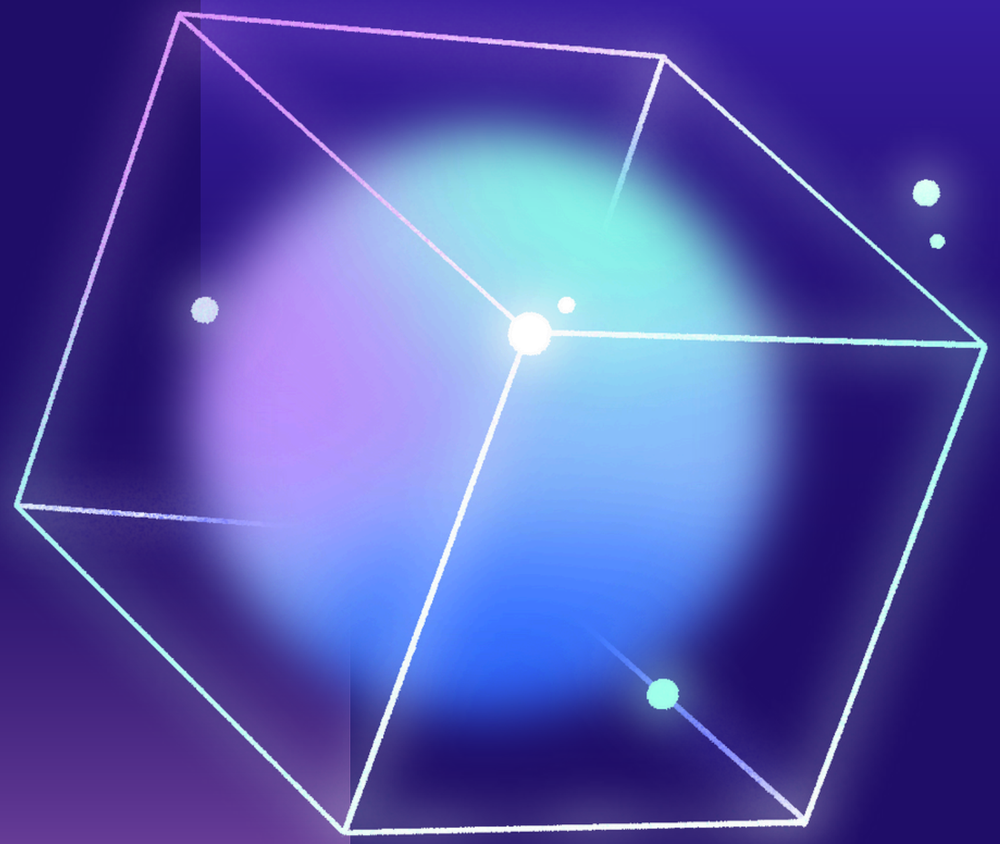
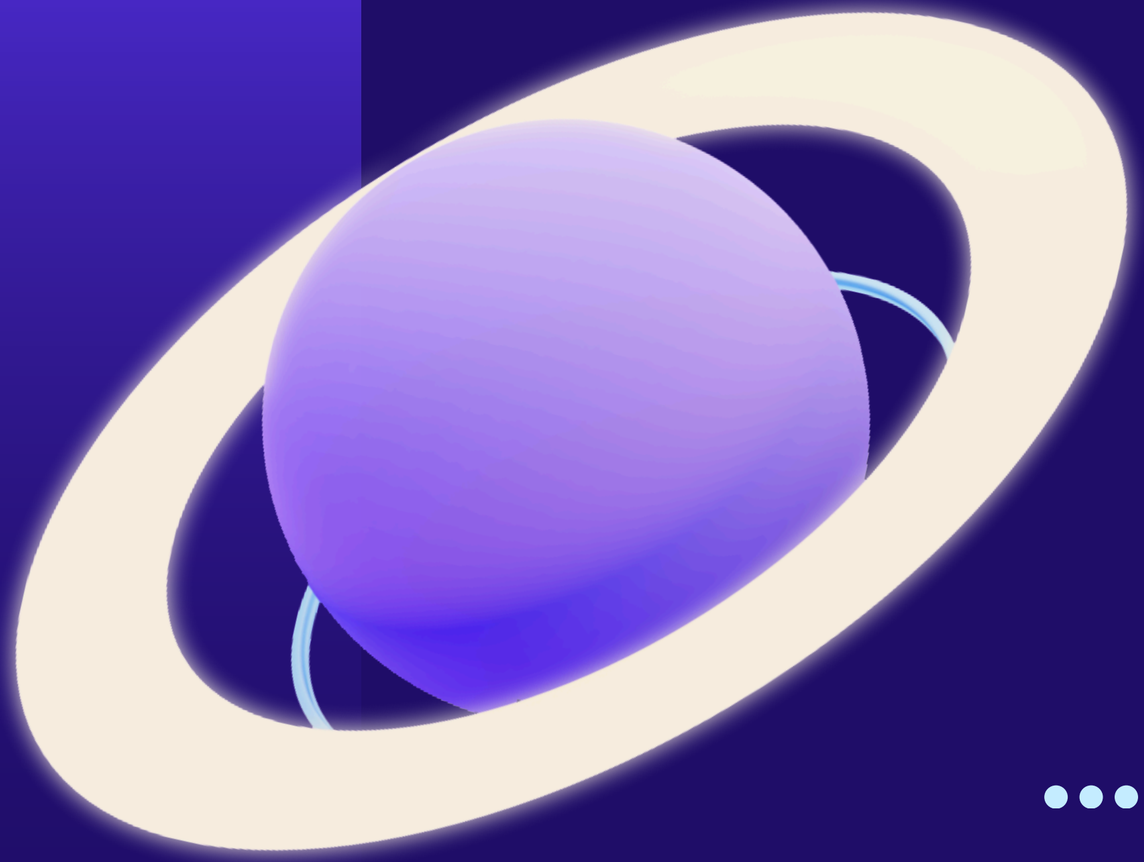
KEKURANGAN

- Keterbatasan data scrapping kami hanya ada sekitar 3000 kalimat yang di scrapping dari X
- Banyak data negative sangat banyak, sehingga imbalance dataset menyebabkan model menjadi bias
- Kami belum sempat deploy ke website

KESIMPULAN

Hasil analisis menunjukkan bahwa metode IndoBERT dengan akurasi 87% efektif untuk mendeteksi tweet yang mengandung kecenderungan bunuh diri. Kelebihannya mencakup fokus pada kesehatan mental, penggunaan data original, serta pemrosesan dan evaluasi data dengan IndoBERT.

Namun, penelitian ini memiliki keterbatasan, seperti jumlah data yang hanya sekitar 3000 kalimat dan ketidakseimbangan dataset akibat dominasi data negatif, yang dapat menyebabkan model menjadi bias. Dengan penambahan data dan perbaikan keseimbangan dataset, performa model dapat dioptimalkan lebih lanjut.



TERIMAKASIH