ELT(EXTRACT, LOAD, TRANSFORM) PROCESS

“Analisis Sentimen Komentar YouTube Warga Indonesia terhadap Danantara”

By :

Al Fitra Nur Ramadhani

202210370311264

Data, Information, and Knowledge B

BACKGROUND

- Media sosial kini telah menjadi platform utama bagi masyarakat untuk mengekspresikan opini, emosi, dan tren yang berkembang. Salah satu topik yang sedang menarik perhatian publik Indonesia adalah Danantara Indonesia, Badan Pengelola Investasi Daya Anagata Nusantara, yang memiliki peran strategis dalam mengonsolidasikan dan mengoptimalkan investasi pemerintah untuk mendukung pertumbuhan ekonomi nasional.

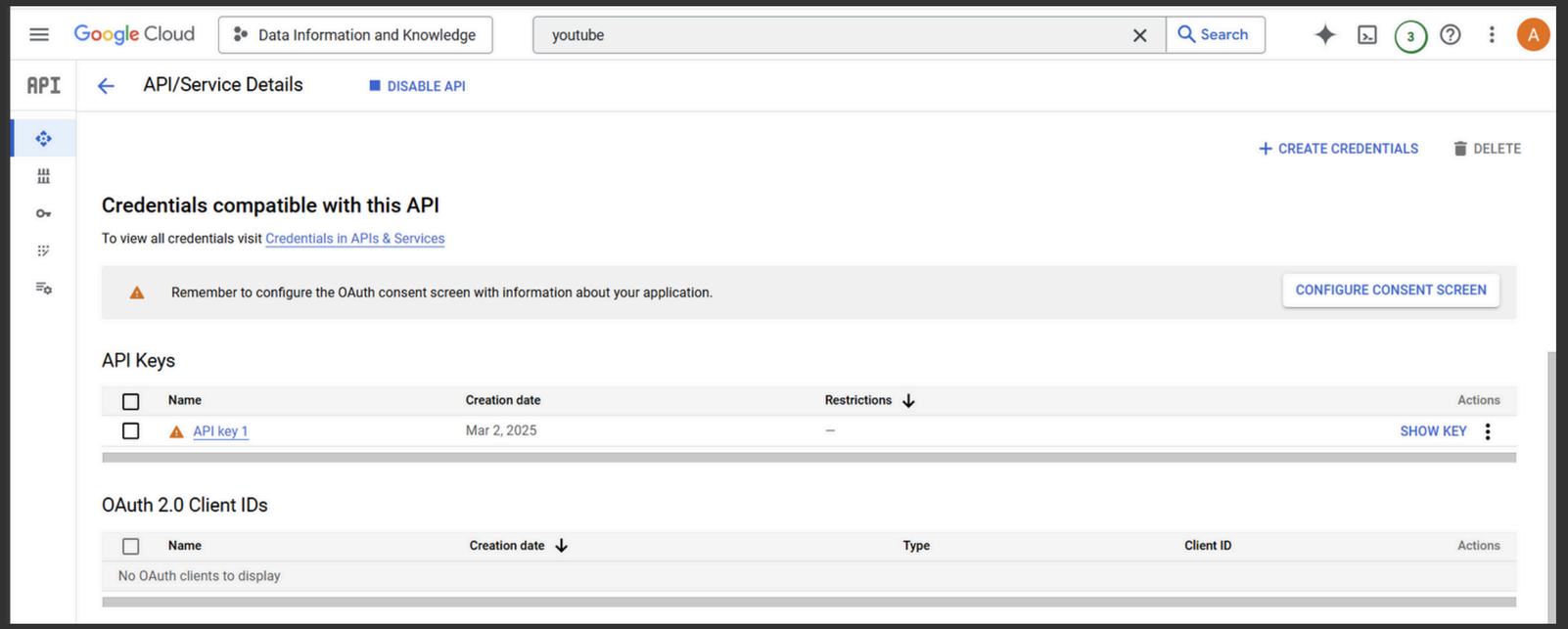
- Dalam proyek ini, saya melakukan analisis sentimen terhadap komentar-komentar yang ditinggalkan oleh warga Indonesia pada video YouTube Raymond Chin yang membahas Danantara. Data yang digunakan dikumpulkan melalui YouTube API v3, yang memungkinkan ekstraksi komentar secara langsung dari platform YouTube.

TOOLS

- VS Code
- Python
- Mongo DB
- Youtube API V3
- Youtube Platform

EXTRACT

LETS GO EXECUTE (EXTRACT)



- Untuk dapatkan API Key nya, pergi ke Google Cloud dan cari Youtube Data API V3
- Click Create Credentials untuk dapatkan API Key mu

LETS GO EXECUTE (EXTRACT)

```
dev = "Your API Key"
```

- Copy Your API Key, dan Youtube ID pada Program , untuk mengambil comments dari Youtube

```
import googleapiclient.discovery
import pandas as pd

api_service_name = "youtube"
api_version = "v3"
DEVELOPER_KEY = dev

youtube = googleapiclient.discovery.build(
    api_service_name, api_version, developerKey=DEVELOPER_KEY)

request = youtube.commentThreads().list(
    part="snippet",
    videoId="Tl6fDFA4xNI",
    maxResults=100
)

comments = []

# Execute the request.
response = request.execute()

# Get the comments from the response.
for item in response['items']:
    comment = item['snippet']['topLevelComment']['snippet']
    public = item['snippet']['isPublic']
    comments.append([
        comment['authorDisplayName'],
        comment['publishedAt'],
        comment['likeCount'],
        comment['textOriginal'],
        public
    ])
```

```
while (1 == 1):
  try:
   nextPageToken = response['nextPageToken']
  except KeyError:
   break
  nextPageToken = response['nextPageToken']
  # Create a new request object with the next page token.
  nextRequest = youtube.commentThreads().list(part="snippet", videoId="Tl6fDFA4xNI", maxResults=100, pageToken=nextPageToken)
  # Execute the next request.
  response = nextRequest.execute()
  # Get the comments from the next response.
  for item in response['items']:
    comment = item['snippet']['topLevelComment']['snippet']
    public = item['snippet']['isPublic']
    comments.append([
        comment['authorDisplayName'],
        comment['publishedAt'],
        comment['likeCount'],
        comment['textOriginal'],
        public
    ])

df = pd.DataFrame(comments, columns=['author', 'updated_at', 'like_count', 'text','public'])
df.info()
```

LETS GO EXECUTE (EXTRACT)

```
df.info()
```

- Cek Data nya apakah sudah dimuat, dan berapa entri, data type, dll

```
df.head(10)
```

- Jika ingin lebih yakin, cek data nya dalam bentuk tabel dengan df.head()

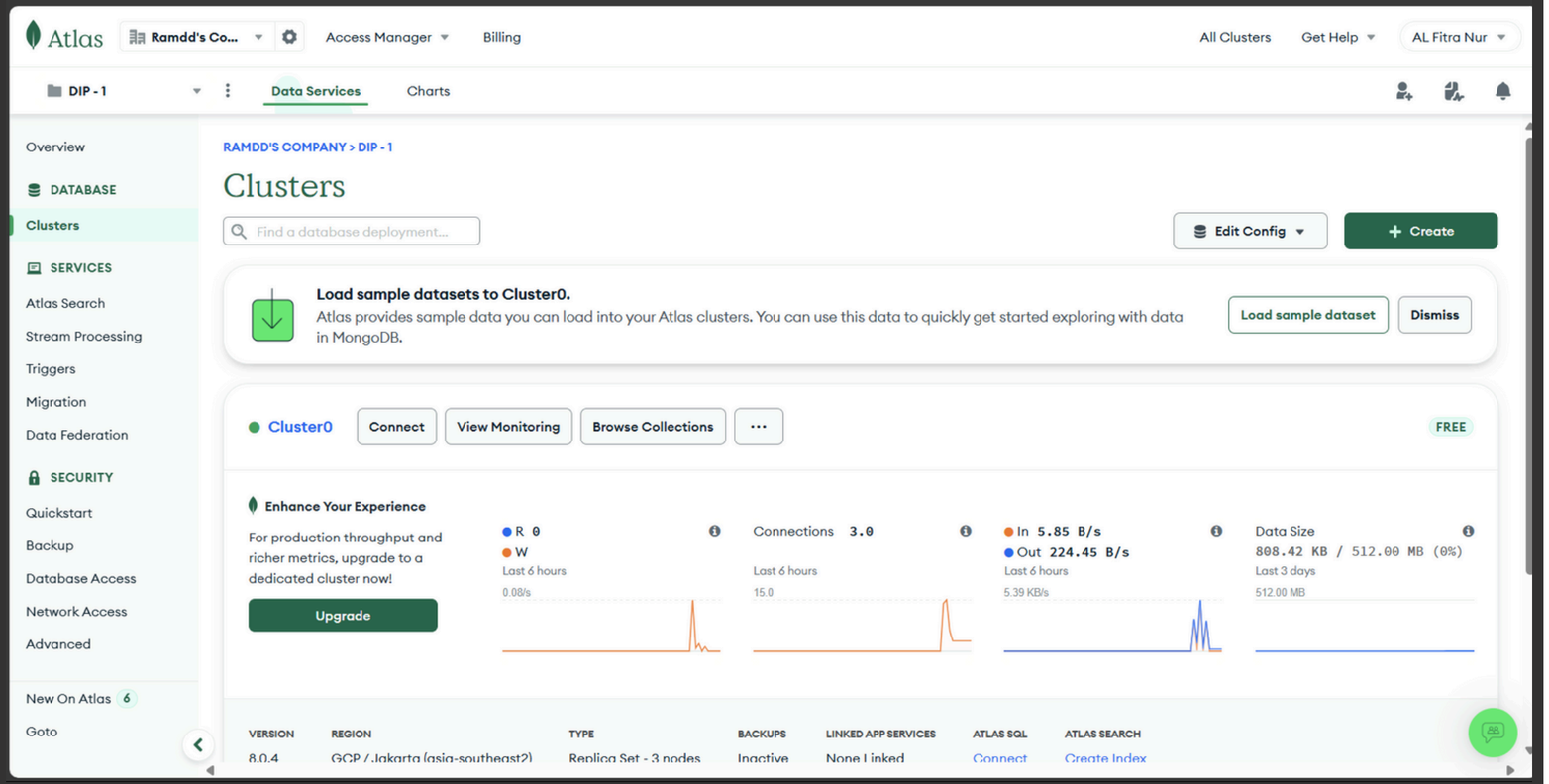
	author	updated_at	# like_count	text	public
0	@ahmadsyaifuddin3553	2025-03-02T02:50:13Z	0	Ayo taruh an ga bakal an di korupsi l	True
1	@lenyselvianti4031	2025-03-02T02:49:21Z	0	Semoga ada transparansi😳	True
2	@andrihermansyah7	2025-03-02T02:47:54Z	0	Itulah yang ditakutkan. Eksekusi amb	True

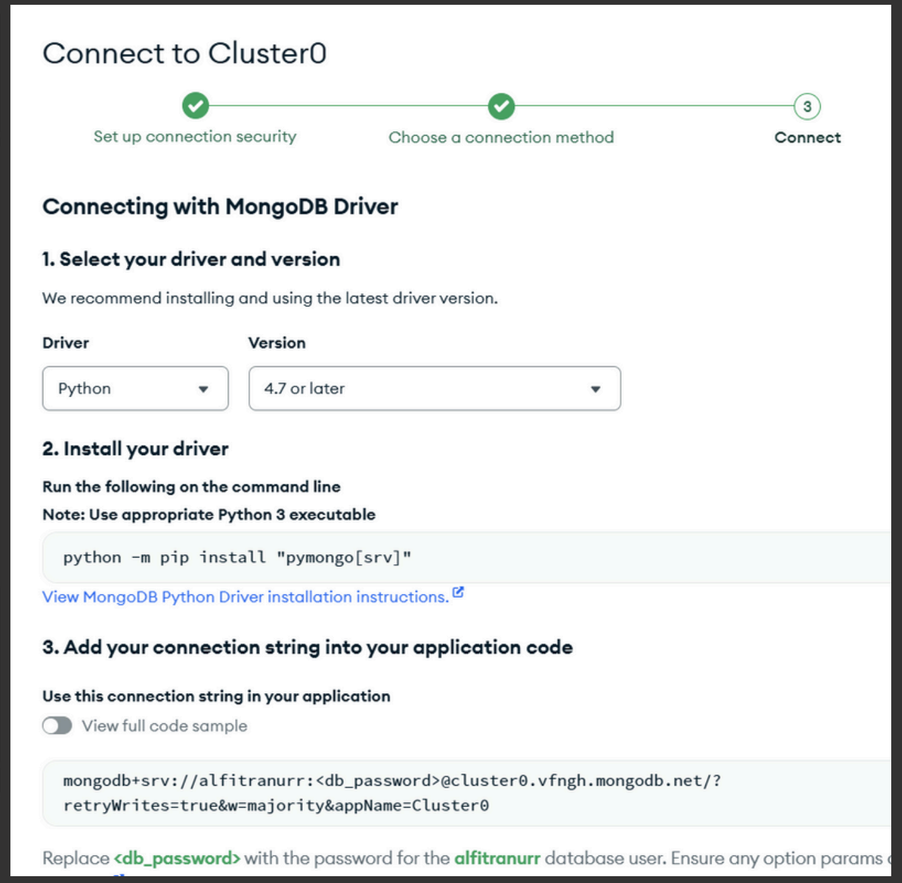
```
df.to_csv('Data_Scrapped_Youtube_RaymondChin.csv', index=False)

```

- Langkah terakhir, download datanya dalam bentuk CSV

LOAD

LETS GO EXECUTE (LOAD)





- Pergi ke MongoDB untuk membuat Database, kemudian Click Connect dan pilih Driver, kemudian sesuaikan dengan seperti pada gambar, dan copy connection string nya

LETS GO EXECUTE (LOAD)

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
from pymongo import MongoClient
import emoji
import csv
import json
```

- Masuk ke dalam Program, dan berikut adalah beberapa library yang harus di impor

```python
# 1. Koneksi ke MongoDB Atlas
# Ganti <db_password> dengan kata sandi MongoDB Anda
mongo_uri = "mongodb+srv://alfitranurr:YourCode@cluster0.vfngh.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0"
client = MongoClient(mongo_uri)

# Pilih database dan collection
db = client['danantara_db']  # Nama database, bisa diganti sesuai keinginan
collection = db['youtube_comments']  # Nama collection, bisa diganti sesuai keinginan
```

- Pastekan URL yang sudah di copy dari MongoDb dan ubah sesuai PW mu, dan sesuaikan untuk nama database dan tabel nya untuk ditampilkan di Mongo DB

LETS GO EXECUTE (LOAD)

```python
# 2. Fungsi untuk memuat data CSV ke MongoDB
def load_csv_to_mongodb(csv_file_path):
    # Baca file CSV menggunakan pandas
    df = pd.read_csv(csv_file_path)

    # Konversi DataFrame ke list of dictionaries
    data = df.to_dict(orient='records')

    # Hapus data lama di collection (opsional, uncomment jika ingin reset)
    # collection.delete_many({})

    # Insert data ke MongoDB
    collection.insert_many(data)
    print(f"Berhasil memuat {len(data)} dokumen ke MongoDB!")
```

- Berikut adalah program untuk memuat data CSV ke dalam mongo DB, dengan mengubah awalnya berbentuk dataframe menjadi list of dictionary

```python
# 3. Fungsi untuk membaca data dari MongoDB ke DataFrame
def read_from_mongodb():
    # Ambil semua data dari collection
    data = list(collection.find())

    # Konversi ke DataFrame
    df = pd.DataFrame(data)

    # Hapus kolom '_id' yang otomatis dibuat oleh MongoDB (opsional)
    if '_id' in df.columns:
        df = df.drop('_id', axis=1)

    return df
```

- Kemudian read file yang ada di mongoDB

LETS GO EXECUTE (LOAD)

```python
# 4. Main execution
if __name__ == "__main__":
    # Ganti dengan path file CSV Anda
    csv_file_path = "Data_Scrapped_Youtube_RaymondChin.csv"

    # Load data CSV ke MongoDB
    load_csv_to_mongodb(csv_file_path)

    # Baca data dari MongoDB ke DataFrame
    df = read_from_mongodb()

    # Tampilkan informasi dasar DataFrame
    print(df.info())
    print(df.head())
```

```
Berhasil memuat 1758 dokumen ke MongoDB!
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3515 entries, 0 to 3514
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   author      3514 non-null   object
 1   updated_at  3515 non-null   object
 2   like_count  3515 non-null   int64
 3   text        3515 non-null   object
 4   public      3515 non-null   bool
dtypes: bool(1), int64(1), object(3)
memory usage: 113.4+ KB
None
                author            updated_at  like_count  \
0  @ahmadsyaifuddin3553  2025-03-02T02:50:13Z           0
1    @lenyselvianti4031  2025-03-02T02:49:21Z           0
2     @andrihermansyah7  2025-03-02T02:47:54Z           0
3     @agussupriadi6083  2025-03-02T02:43:54Z           0
4           @Bonsai-q4f  2025-03-02T02:41:49Z           0

                                                text  public
0  Ayo taruh an ga bakal an di korupsi kaya jiwas...    True
1                          Semoga ada transparansi😳     True
2  Itulah yang ditakutkan  Eksekusi ambunadul  an        True
```

- Execute Program didalam main, jika sudah akan menampilkan seperti pada gambar

TRANSFORM

LETS GO EXECUTE (TRANSFORM)

```python
# Tentukan kolom yang berisi tipe list
list_columns = ['text']  # Ganti dengan nama kolom yang sesuai

# Mengonversi kolom dengan tipe list menjadi string
for col in list_columns:
    df[col] = df[col].apply(lambda x: str(x) if isinstance(x, list) else x)

# Cek duplikat setelah konversi
duplikat = df.duplicated()
print(f"Jumlah duplikat: {duplikat.sum()}")
print("Baris yang duplikat:")
print(df[duplikat])

```

- Cek duplikat data nya

```
df.isnull().sum()
✓ 0.0s
author         1
updated_at     0
like_count     0
text           0
public         0
dtype: int64
```

```python
# Menghapus baris dengan nilai null di kolom 'text'
df = df.dropna(subset=['author'])
```

```python
# Periksa kembali jumlah nilai null
print(df.isnull().sum())
```

- Cek Missing Value
- Ternyata ada 1 di kolom “Author”, kemudian bersihkan dan tampilkan lagi untuk memastikan apakah benar-benar sudah dibersihkan

LETS GO EXECUTE (TRANSFORM)

```
[2]  df.info()
     ✓ 0.0s

<class 'pandas.core.frame.DataFrame'>
Index: 1757 entries, 0 to 1756
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   author      1757 non-null   object
 1   updated_at  1757 non-null   object
 2   like_count  1757 non-null   int64 
 3   text        1757 non-null   object
 4   public      1757 non-null   bool  
dtypes: bool(1), int64(1), object(3)
memory usage: 70.3+ KB
```

- Cek lagi datanya apakah sudah benar
- Dari yang semula 1758, kini menjadi 1757 setelah PreProcessing

```
# Menyimpan df_cleaned sebagai file CSV
df.to_csv('data_cleaned_Youtube_RaymondChin.csv', index=False)
```

- Simpan datanya sebagai csv

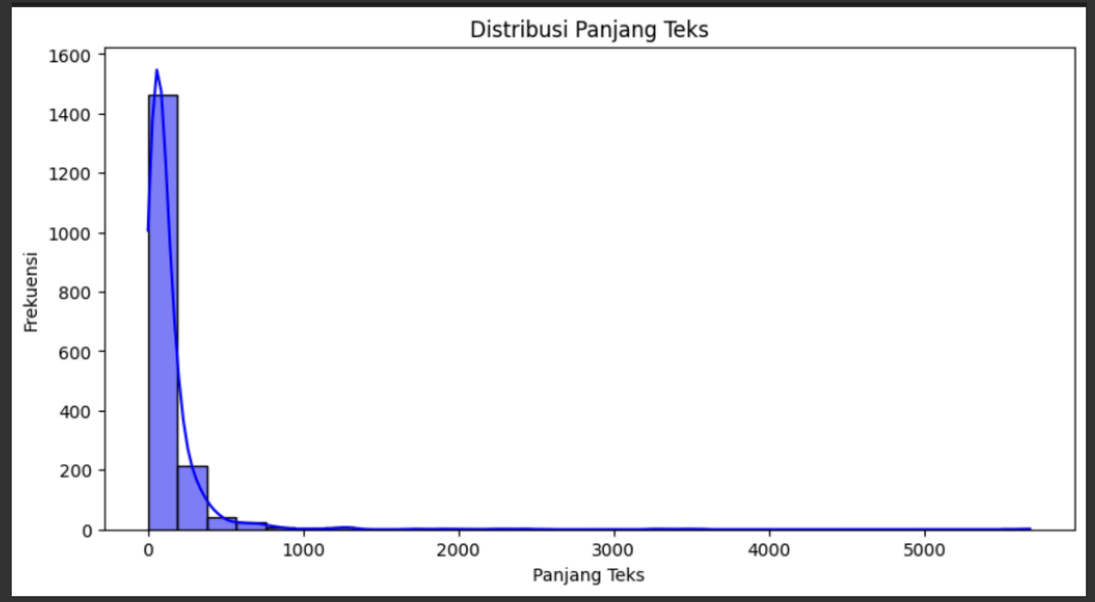
```
# Menghapus semua data lama
collection.delete_many({})

# Mengonversi DataFrame menjadi list of dictionaries
data_baru = df.to_dict(orient='records')

# Memuat data baru ke MongoDB
collection.insert_many(data_baru)
print(f"Data berhasil dimuat kembali dengan {len(data_baru)} dokumen.")
```

- Load lagi datanya ke mongo DB untuk perubahan data terbaru

ANALYZE & VISUALIZATION

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
# Analisis panjang teks
df['text_length'] = df['text'].apply(lambda x: len(str(x)))


plt.figure(figsize=(10,5))
sns.histplot(df['text_length'], bins=30, kde=True, color='blue')
plt.xlabel("Panjang Teks")
plt.ylabel("Frekuensi")
plt.title("Distribusi Panjang Teks")
plt.show()
```



- Grafik menunjukkan bahwa sebagian besar teks dalam dataset kita memiliki panjang yang pendek, sekitar 0-500 karakter, dengan jumlah terbanyak mencapai lebih dari 1.400, sementara teks panjang di atas 1.000 karakter sangat jarang ditemukan. Hal ini menunjukkan bahwa konten kita didominasi oleh teks pendek

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
# WordCloud untuk melihat kata yang sering muncul
text_corpus = " ".join(df['text'].dropna().astype(str))
# Tema warna (misalnya: biru ke ungu)


stopwords = set(STOPWORDS)
stopwords.update(["yg", "dg", "rt", "dgn", "ny", "d", 'klo',
                  'kalo', 'amp', 'biar', 'bikin', 'bilang',
                  'gak', 'ga', 'krn', 'nya', 'nih', 'sih',
                  'si', 'tau', 'tdk', 'tuh', 'utk', 'ya',
                  'jd', 'jgn', 'sdh', 'aja', 'n', 't',
                  'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                  'wkwk', 'wkwkwk', 'nggak', 'gakk', 'gk', 'tp', 'lg',
                  'bgt', 'dr', 'gw', 'gue', 'gpp', 'aja', 'deh', 'kok',
                  'doi', 'banget', 'bangettt', 'cuma', 'kali', 'yaa',
                  'udah', 'k', 'sy', 'm', 'bang', 'kak', 'kakak', 'min',
                  'mas', 'mba', 'mbak', 'anjay', 'anjir', 'anj', 'lah',
                  'lho', 'loh', 'lu', 'loe', 'lo', 'kamu', 'aku', 'gue',
                  'gw', 'gua', 'gue', 'kita', 'mereka', 'kalian', 'aku', 'saya',
                  'dia', 'mereka', 'kamu', 'kalian', 'kau', 'engkau', 'anda',
                  'beliau', 'ia', 'mereka', 'kita', 'ini', 'itu', 'sana', 'sini',
                  'mana', 'kapan', 'kenapa', 'bagaimana', 'berapa', 'siapa', 'dimana',
                  'apa', 'siapa', 'mana', 'kapan', 'mengapa', 'bagaimana', 'berapa',
                  'kabur', 'aja', 'dulu', 'indonesia', 'luar', 'negeri'])

wordcloud = WordCloud(width=800,height=400,background_color="black",colormap="cool",stopwords=STOPWORDS,
                      max_words=200,contour_width=3,contour_color="white").generate(text_corpus)


plt.figure(figsize=(10,5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Word Cloud Post Tiktok")
plt.show()
```

- berikut adalah wordcloud sebelum adanya preprocessing

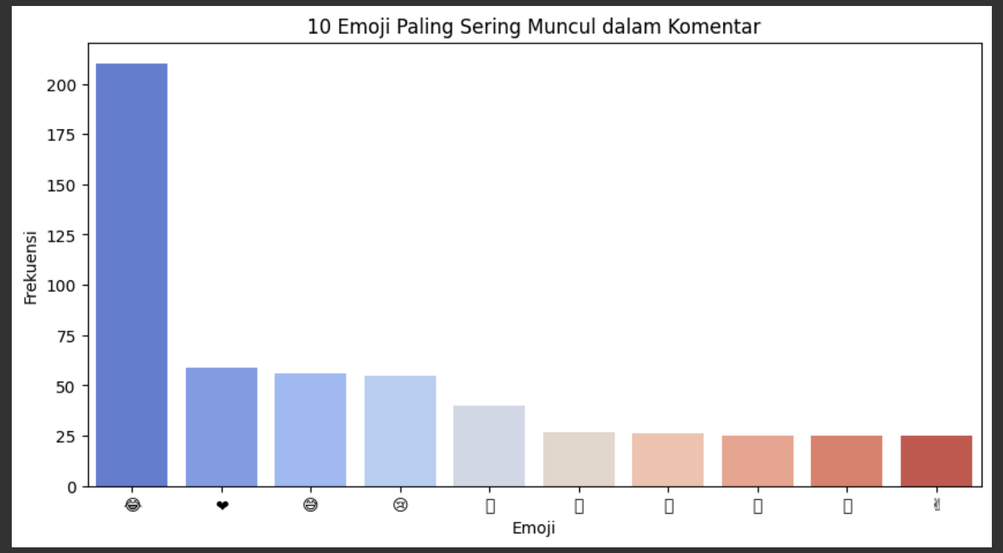
LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
# Check the result of demojize

df['text'].apply(lambda x: emoji.demojize(x) if isinstance(x, str) else x).head()
```

```
0    Ayo taruh an ga bakal an di korupsi kaya jiwas...
1                  Semoga ada transparansi:crying_face:
2    Itulah yang ditakutkan. Eksekusi amburadul, ap...
3    DANANTARA ITU LEMBAGA NEGARA BUKAN KOH....TERU...
4    Jk berkaca dari SDM penguasa negeri ini sih sg...
Name: text, dtype: object
```

- Mengubah emoji menjadi sebuah kata, dan memvisualisasikan dalam top 10 Emoji paling sering muncul

```python
def extract_emojis(text):
    return [char for char in text if emoji.is_emoji(char)]

all_emojis = [emj for text in df['text'].dropna().astype(str) for emj in extract_emojis(text)]
emoji_freq = Counter(all_emojis)
top_emojis = emoji_freq.most_common(10)
emoji_chars, emoji_counts = zip(*top_emojis)

plt.figure(figsize=(10, 5))
sns.barplot(x=list(emoji_chars), y=list(emoji_counts), palette="coolwarm")
plt.xlabel("Emoji")
plt.ylabel("Frekuensi")
plt.title("10 Emoji Paling Sering Muncul dalam Komentar")
plt.show()
```



LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
import pandas as pd
import re
import string
import nltk
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from nltk.corpus import stopwords
from nltk.tokenize import regexp_tokenize
```

```python
# Ensure the NLTK stopwords list is downloaded
nltk.download('stopwords')
```

```python
# Membuat stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()
```

```python
# Fungsi untuk menghapus URL dan HTML
def remove_urls_and_html(text):
    # Menghapus URL
    text = re.sub(r'http\S+', '', text)
    # Menghapus tag HTML
    text = re.sub(r'<.*?>', '', text)
    return text
```

```python
# Fungsi untuk menghapus emoji
def remove_emoji(text):
    return emoji.replace_emoji(text, replace='')
```

```python
# Fungsi cleansing teks
def cleansing(df):
    df_clean = df.astype(str)  # Mengubah semua entri menjadi string
    df_clean = df_clean.str.lower()  # Mengubah menjadi huruf kecil
    df_clean = [re.sub(r"\d+", "", i) for i in df_clean]  # Menghapus angka
    df_clean = [re.sub(r'[^\w]', ' ', i) for i in df_clean]  # Menghapus karakter non-alfanumerik
    df_clean = [re.sub(r'\s+', ' ', i) for i in df_clean]  # Mengganti spasi ganda menjadi satu
    df_clean = [i.strip() for i in df_clean]  # Menghapus spasi di awal dan akhir
    return df_clean
```

```python
# Daftar stopwords Bahasa Indonesia dari NLTK
stopwords_list = stopwords.words('indonesian')
stopwords_list.extend([
    "yg", "dg", "rt", "dgn", "ny", "d", "klo", "kalo", "amp", "biar", "bikin", "bilang",
    "gak", "ga", "krn", "nya", "nih", "sih", "si", "tau", "tdk", "tuh", "utk", "ya", "jd",
    "jgn", "sdh", "aja", "n", "t", "nyg", "hehe", "pen", "u", "nan", "loh", "rt", "wkwk",
    "wkwkwk", "nggak", "gakk", "gk", "tp", "lg", "bgt", "dr", "gw", "gue", "gpp", "aja", "deh",
    "kok", "doi", "banget", "bangettt", "cuma", "kali", "yaa", "udah", "k", "sy", "m", "bang",
    "kak", "kakak", "min", "mas", "mba", "mbak", "anjay", "anjir", "anj", "lah", "lho", "loh",
    "lu", "loe", "lo", "kamu", "aku", "gue", "gw", "gua", "gue", "kita", "mereka", "kalian",
    "aku", "saya", "dia", "mereka", "kamu", "kalian", "kau", "engkau", "anda", "beliau", "ia",
    "mereka", "kita", "ini", "itu", "sana", "sini", "mana", "kapan", "kenapa", "bagaimana",
    "berapa", "siapa", "dimana", "apa", "siapa", "mana", "kapan", "mengapa", "bagaimana",
    "berapa", "kabur", "aja", "dulu", "indonesia", "luar", "negeri", "dan"
])
```

```python
# Fungsi untuk menghapus stopwords
def remove_stopwords(tokens):
    return [word for word in tokens if word not in stopwords_list]
```

- Preprocessing lebih lanjut untuk sentiment analysis nya
- Preprocessing mencakup Stopword. Stemming dengan Sastrawi, Tokenize, Lower case, menghapus angka, URL maupun HTML, menghapus emoji, dll.

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
# Fungsi utama untuk memproses DataFrame
def process_dataframe(df):
    df_clean = cleansing(df['text'])

    # Menambahkan kolom 'clean_comment' untuk teks yang sudah dibersihkan
    df['clean_comment'] = df_clean

    # Tokenisasi
    df['tokens'] = df_clean  # Simpan hasil cleansing langsung ke kolom tokens

    # Tokenisasi teks
    df['tokens'] = df['tokens'].apply(tokenize_text)

    # Menghapus stopwords
    df['tokens_nostopwords'] = df['tokens'].apply(remove_stopwords)

    return df
```

```python
# Proses DataFrame
df_processed = process_dataframe(df)
```

```python
# Check the result
df['nonstop_comment'] = df['tokens'].apply(lambda x: [word for word in x if word not in stopwords_list])
df.head()
```

# word_count	clean_comment	tokens	tokens_nostopwords	nonstop_comment
19	ayo taruh an ga bakal an di korupsi k	['ayo', 'taruh', 'an', 'ga', 'bakal', 'an', 'c	['ayo', 'taruh', 'an', 'an', 'korupsi', 'kay	['ayo', 'taruh', 'an', 'an', 'korupsi', 'kay
3	semoga ada transparansi	['semoga', 'ada', 'transparansi']	['semoga', 'transparansi']	['semoga', 'transparansi']
55	itulah yang ditakutkan eksekusi amb	['itulah', 'yang', 'ditakutkan', 'eksekus	['ditakutkan', 'eksekusi', 'amburadul',	['ditakutkan', 'eksekusi', 'amburadul',
10	danantara itu lembaga negara bukar	['danantara', 'itu', 'lembaga', 'negara'	['danantara', 'lembaga', 'negara', 'koh	['danantara', 'lembaga', 'negara', 'koh
10	jk berkaca dari sdm penguasa negeri	['jk', 'berkaca', 'dari', 'sdm', 'penguasa	['jk', 'berkaca', 'sdm', 'penguasa', 'sgt	['jk', 'berkaca', 'sdm', 'penguasa', 'sgt

- Proses dataframe dengan membuat kolom baru bernama clean_comment, tokens, tokens_nostopwords, nonstop_comment

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
# Fungsi utama untuk memproses DataFrame
def process_dataframe(df):
    df_clean = cleansing(df['text'])

    # Menambahkan kolom 'clean_comment' untuk teks yang sudah dibersihkan
    df['clean_comment'] = df_clean

    # Tokenisasi
    df['tokens'] = df_clean  # Simpan hasil cleansing langsung ke kolom tokens

    # Tokenisasi teks
    df['tokens'] = df['tokens'].apply(tokenize_text)

    # Menghapus stopwords
    df['tokens_nostopwords'] = df['tokens'].apply(remove_stopwords)

    return df
```

```python
# Proses DataFrame
df_processed = process_dataframe(df)
```

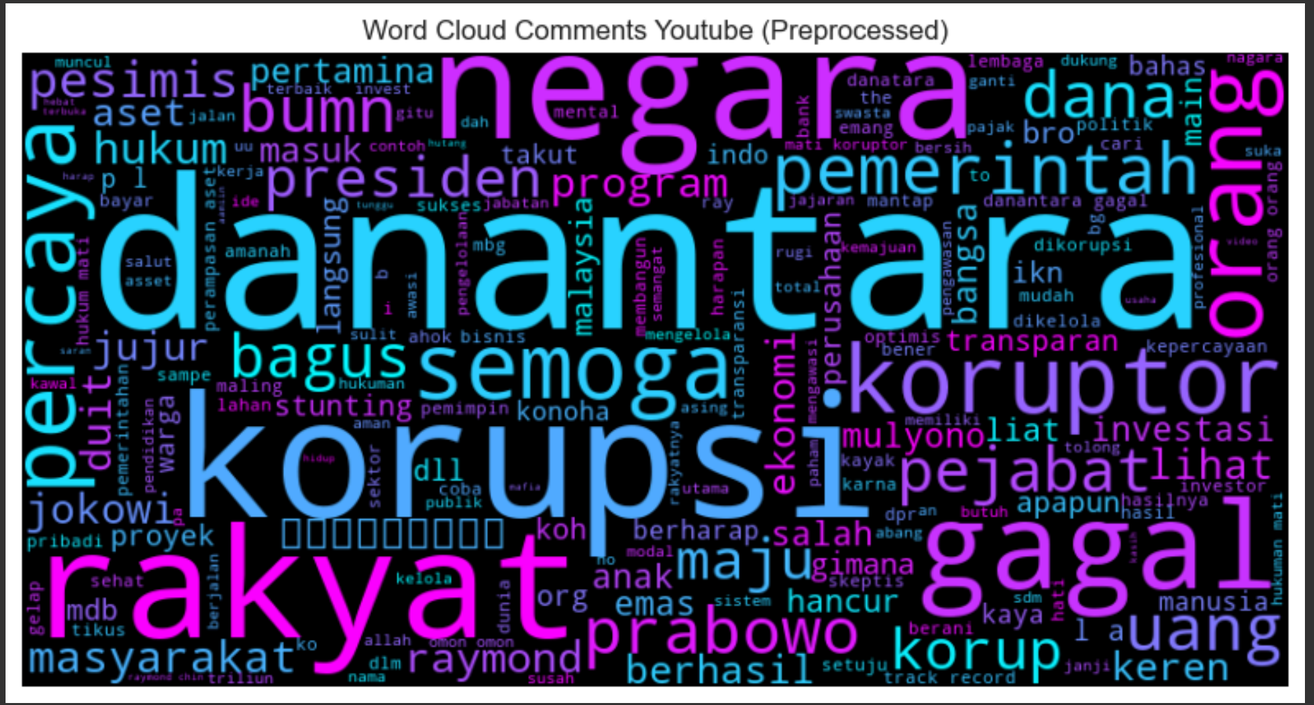
```python
# Check the result
df['nonstop_comment'] = df['tokens'].apply(lambda x: [word for word in x if word not in stopwords_list])
df.head()
```

# word_count	clean_comment	tokens	tokens_nostopwords	nonstop_comment
19	ayo taruh an ga bakal an di korupsi k	['ayo', 'taruh', 'an', 'ga', 'bakal', 'an', 'c	['ayo', 'taruh', 'an', 'an', 'korupsi', 'kay	['ayo', 'taruh', 'an', 'an', 'korupsi', 'kay
3	semoga ada transparansi	['semoga', 'ada', 'transparansi']	['semoga', 'transparansi']	['semoga', 'transparansi']
55	itulah yang ditakutkan eksekusi amb	['itulah', 'yang', 'ditakutkan', 'eksekus	['ditakutkan', 'eksekusi', 'amburadul',	['ditakutkan', 'eksekusi', 'amburadul',
10	danantara itu lembaga negara bukar	['danantara', 'itu', 'lembaga', 'negara'	['danantara', 'lembaga', 'negara', 'koh	['danantara', 'lembaga', 'negara', 'koh
10	jk berkaca dari sdm penguasa negeri	['jk', 'berkaca', 'dari', 'sdm', 'penguasa	['jk', 'berkaca', 'sdm', 'penguasa', 'sgt	['jk', 'berkaca', 'sdm', 'penguasa', 'sgt

- Proses dataframe dengan membuat kolom baru bernama clean_comment, tokens, tokens_nostopwords, nonstop_comment

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from nltk.corpus import stopwords

# Make sure stopwords are downloaded
nltk.download('stopwords')

# Get the stopwords from NLTK and convert it to a set
stopwords_list = set(stopwords.words('indonesian'))  # Convert to a set to avoid issues with iteration

# Combine all the tokens (without stopwords)
text_corpus_preprocessed = " ".join([" ".join(tokens) for tokens in df['tokens_nostopwords']])

# Create the word cloud
wordcloud_preprocessed = WordCloud(
    width=800, height=400, background_color="black", colormap="cool",
    stopwords=stopwords_list, max_words=200, contour_width=3, contour_color="white"
).generate(text_corpus_preprocessed)

# Plot the wordcloud
plt.figure(figsize=(10,5))
plt.imshow(wordcloud_preprocessed, interpolation='bilinear')
plt.axis("off")
plt.title("Word Cloud Comments Youtube (Preprocessed)")
plt.show()
```



- Wordcloud setelah adanya preprocess lanjutan

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

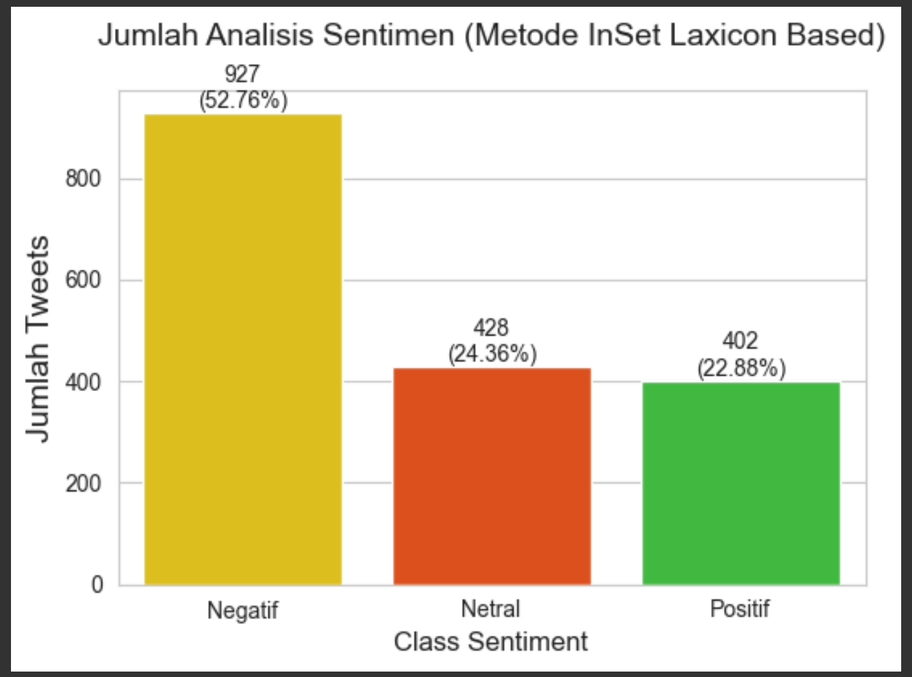
```python
# Kamus Leksikon postif dan negatif
positive_url = "https://raw.githubusercontent.com/fajri91/InSet/master/positive.tsv"
negative_url = "https://raw.githubusercontent.com/fajri91/InSet/master/negative.tsv"

pos_lex = set(pd.read_csv(positive_url, sep='\t', header=None)[0].str.lower())
neg_lex = set(pd.read_csv(negative_url, sep='\t', header=None)[0].str.lower())

# Fungsi untuk menentukan sentimen
def determine_sentiment(text):
    if isinstance(text, str):
        positive_count = sum(1 for word in text.split() if word in pos_lex)
        negative_count = sum(1 for word in text.split() if word in neg_lex)
        sentiment_score = positive_count - negative_count

        if sentiment_score > 0:
            sentiment = "Positif"
        elif sentiment_score < 0:
            sentiment = "Negatif"
        else:
            sentiment = "Netral"

        return sentiment_score, sentiment

    return 0, "Netral"

# Tentukan sentimen dan skor untuk setiap ulasan
df[['Score', 'Sentiment']] = df['clean_comment'].apply(lambda x: pd.Series(determine_sentiment(x)))

# Tampilkan hasilnya
print(df.head(5))

```

```
...
1                              [semoga, transparansi]      1   Positif
2  [ditakutkan, eksekusi, amburadul, pemerintah, ...    -16   Negatif
3  [danantara, lembaga, negara, koh, pengelolanya...     -3   Negatif
4         [jk, berkaca, sdm, penguasa, sgt, pesimis]      0    Netral
```

- Ambil kamus Inset Lexicon Based untuk Positif dan Negatif dari Github

LETS GO EXECUTE
(ANALYZE & VISUALIZATION)

```python
sentiment_count = df['Sentiment'].value_counts()
sns.set_style('whitegrid')


custom_colors = ["#FFD700", "#FF4500", "#32CD32"]
fig, ax = plt.subplots(figsize=(6, 4))
ax = sns.barplot(x=sentiment_count.index, y=sentiment_count.values, palette=custom_colors)


plt.title("Jumlah Analisis Sentimen (Metode InSet Laxicon Based)", fontsize=14, pad=20)
plt.xlabel("Class Sentiment", fontsize=12)
plt.ylabel("Jumlah Tweets", fontsize=14)


total = len(df['Sentiment'])
for i, count in enumerate(sentiment_count.values):
    percentage = f"{100 * count / total:.2f}%"
    ax.text(i, count + 0.10, f"{count}\n({percentage})", ha='center', va='bottom')

plt.show()
```



- Visualisasi Hasilnya, dan seperti yang tertera pada diagram bahwa Negatif berada pada posisi paling tinggi.
- Menandakan bahwa Danantara pada Video Youtube Raymond Chin memiliki konotasi sentimen negatif di kalangan masyarakat.

MACHINE LEARNING MODEL

LETS GO EXECUTE
(MACHINE LEARNING MODEL)

```python
from sklearn.model_selection import train_test_split
# Pembagian data
x_train, x_test, y_train, y_test = train_test_split(df['clean_comment'], df['Sentiment'], test_size=0.2, random_state=42)

```

```
x_train
```

```
y_train
```

```
x_test
```

```
y_test
```

- Membuat model machine learning dengan Train, Test Split terlebih dahulu

LETS GO EXECUTE (MACHINE LEARNING MODEL)

Teks Representation 1 : Bag of Words

```python
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
x = vectorizer.fit_transform(x_train)
train_bow = pd.DataFrame(x.toarray(), columns=vectorizer.get_feature_names_out())

# Menampilkan hasil
print("Representasi Bag of Words:")
print(train_bow)
```

✓ 0.0s Open 'train_bow' in Data Wrangler

```
Representasi Bag of Words:
      _jujur  aada  aamiiin  aamiin  aamin  aarrrrrggghhhhhhhhhhh  abadi  \
0          0     0        0       0      0                      0      0
1          0     0        0       0      0                      0      0
2          0     0        0       0      0                      0      0
3          0     0        0       0      0                      0      0
4          0     0        0       0      0                      0      0
...      ...   ...      ...     ...    ...                    ...    ...
1400       0     0        0       0      0                      0      0
1401       0     0        0       0      0                      0      0
1402       0     0        0       0      0                      0      0
1403       0     0        0       0      0                      0      0
1404       0     0        0       0      0                      0      0
```

```python
train_bow.columns
```

✓ 0.0s Python

```
Index(['_jujur', 'aada', 'aamiiin', 'aamiin', 'aamin',
       'aarrrrrrgggghhhhhhhhhhh', 'abadi', 'abah', 'abang', 'abangnya',
       ...
       'bedah', 'ideologi', 'kita', 'mas', 'tolong', 'AXLbikin', 'AXL', 'AXL',
       'DEWadOrA', 'DEWaDOяA'],
      dtype='object', length=5877)
```

```python
x_test_bow = vectorizer.transform(x_test)
test_bow = pd.DataFrame(x_test_bow.toarray(), columns=vectorizer.get_feature_names_out())
```

✓ 0.0s Python

- Text Reprentation Bag of Words dengan 3 Model Machine Learning (Random Forest, SVM, dan Naive Bayes)

LETS GO EXECUTE (MACHINE LEARNING MODEL)

Teks Representation 1 : Bag of Words

1. Random Forest

```
from sklearn.ensemble import RandomForestClassifier
rf_class = RandomForestClassifier(n_estimators=100, random_state=42)
rf_class.fit(train_bow, y_train)
```

[46] ✓ 2.0s

RandomForestClassifier

RandomForestClassifier(random_state=42)

```
test_rf_class = rf_class.predict(test_bow)
```

[47] ✓ 0.0s

```
from sklearn.metrics import classification_report
print('\nRandom Forest Classification Report\n')
print(classification_report(y_test, test_rf_class, target_names=['Negatif','Netral','Positif']))
```

[48] ✓ 0.0s

Random Forest Classification Report

	precision	recall	f1-score	support
Negatif	0.73	0.90	0.80	184
Netral	0.61	0.60	0.60	92
Positif	0.83	0.39	0.54	76
accuracy			0.71	352
macro avg	0.72	0.63	0.65	352
weighted avg	0.72	0.71	0.69	352

PROBLEMS 83 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS JUPYTER COMMENTS

2. SVM

```
from sklearn import svm
svm_class1 = svm.LinearSVC( random_state=42)
svm_class1.fit(train_bow, y_train)
```

[39] ✓ 0.0s

LinearSVC

LinearSVC(random_state=42)

```
test_svm_class1=svm_class1.predict(test_bow)
```

[40] ✓ 0.0s

Generate + Code + Markdo

```
print('\nClassification Report\n')
print(classification_report(y_test, test_svm_class1, target_names=['Negatif','Netral','Positif']))
```

[41] ✓ 0.0s

Classification Report

	precision	recall	f1-score	support
Negatif	0.83	0.81	0.82	184
Netral	0.53	0.60	0.56	92
Positif	0.71	0.63	0.67	76
accuracy			0.72	352
macro avg	0.69	0.68	0.68	352
weighted avg	0.72	0.72	0.72	352

3. Naive Bayes

```
from sklearn.naive_bayes import MultinomialNB

nb_class = MultinomialNB()
nb_class.fit(train_bow, y_train)
test_nb_class = nb_class.predict(test_bow)
```

[42] ✓ 0.1s

```
print('\nNaive Bayes Classification Report\n')
print(classification_report(y_test, test_nb_class, target_names=['Negatif','Netral','Positif']))
```

[43] ✓ 0.0s

Naive Bayes Classification Report

	precision	recall	f1-score	support
Negatif	0.64	0.96	0.77	184
Netral	0.64	0.15	0.25	92
Positif	0.66	0.46	0.54	76
accuracy			0.64	352
macro avg	0.65	0.52	0.52	352
weighted avg	0.64	0.64	0.58	352

LETS GO EXECUTE
(MACHINE LEARNING MODEL)

Teks Representation 2 : TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
X = tfidf_vectorizer.fit_transform(x_train)
train_tfidf=pd.DataFrame(X.toarray(),columns=vectorizer.get_feature_names_out())
train_tfidf.head()
```

✓ 0.4s 🐼 Open 'train_tfidf' in Data Wrangler

	# _jujur	# aada	# aamiiin	# aamiin
0	0.0	0.0	0.0	
1	0.0	0.0	0.0	
2	0.0	0.0	0.0	
3	0.0	0.0	0.0	
4	0.0	0.0	0.0	

5 rows x 5,877 cols 10 per page « ‹ Page 1 of 1 ›

```
test_tfidf = tfidf_vectorizer.transform(x_test)
TFIDF_test=pd.DataFrame(test_tfidf.toarray(),columns=tfidf_vectorizer.get_feature_names_out())
TFIDF_test.head()
```

✓ 0.5s 🐼 Open 'TFIDF_test' in Data Wrangler

	# _jujur	# aada	# aamiiin	# aamiin
0	0.0	0.0	0.0	
1	0.0	0.0	0.0	
2	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.32
4	0.0	0.0	0.0	

- Text Reprentation TF - IDF dengan 3 Model Machine Learning (Random Forest, SVM, dan Naive Bayes)

LETS GO EXECUTE (MACHINE LEARNING MODEL)

Teks Representation 2 : TF -IDF

1. Random Forest

```
from sklearn.ensemble import RandomForestClassifier
rf_class2 = RandomForestClassifier(n_estimators=100, random_state=42)
rf_class2.fit(train_tfidf, y_train)
```

✓ 1.9s

RandomForestClassifier

RandomForestClassifier(random_state=42)

```
test_rf_class2 = rf_class2.predict(test_tfidf)
```

✓ 0.0s

```
c:\Users\ASUS\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\utils\validation.py:27
  warnings.warn(
```

```
print('\nRandom Forest Classification TFIDF Report\n')
print(classification_report(y_test, test_rf_class2, target_names=['Negatif','Netral','Positif']))
```

✓ 0.0s

Random Forest Classification TFIDF Report

	precision	recall	f1-score	support
Negatif	0.70	0.91	0.79	184
Netral	0.59	0.52	0.55	92
Positif	0.81	0.34	0.48	76

2. SVM

```
svm_class2 = svm.LinearSVC( random_state=42)
svm_class2.fit(train_tfidf, y_train)
```

✓ 0.0s

LinearSVC

LinearSVC(random_state=42)

```
test_svm_class2 =svm_class2.predict(test_tfidf)
```

✓ 0.0s

```
c:\Users\ASUS\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\utils\validation.py:273
  warnings.warn(
```

```
print('\nClassification Report SVM TFIDF\n')
print(classification_report(y_test, test_svm_class2, target_names=['Negatif','Netral','Positif']))
```

✓ 0.0s

Classification Report SVM TFIDF

	precision	recall	f1-score	support
Negatif	0.77	0.88	0.82	184
Netral	0.59	0.46	0.52	92
Positif	0.75	0.70	0.72	76

PROBLEMS 83 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS JUPYTER COMMENTS

3. Naive Bayes

```
from sklearn.naive_bayes import MultinomialNB

nb_class2 = MultinomialNB()
nb_class2.fit(train_tfidf, y_train)
test_nb_class2 = nb_class2.predict(test_tfidf)
```

✓ 0.0s

```
c:\Users\ASUS\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\utils\validation.py:2
  warnings.warn(
```

```
print('\nNaive Bayes Classification Report\n')
print(classification_report(y_test, test_nb_class2, target_names=['Negatif','Netral','Positif']))
```

✓ 0.0s

Naive Bayes Classification Report

	precision	recall	f1-score	support
Negatif	0.55	1.00	0.71	184
Netral	0.75	0.03	0.06	92
Positif	1.00	0.18	0.31	76
accuracy			0.57	352
macro avg	0.77	0.41	0.36	352
weighted avg	0.70	0.57	0.45	352

LETS GO EXECUTE (MACHINE LEARNING MODEL)

Best Model Accuracy

```python
def find_best_model(y_test, models_predictions):
    best_model = None
    best_accuracy = -1

    for model_name, predictions in models_predictions.items():
        report = classification_report(y_test, predictions, target_names=['Negatif', 'Netral', 'Positif'], output_dict=True)
        accuracy = report['accuracy']

        if accuracy > best_accuracy:
            best_accuracy = accuracy
            best_model = model_name

    return best_model, best_accuracy


models_predictions = {
    'RF_BOW': test_rf_class,
    'SVM_BOW': test_svm_class1,
    'NB_BOW': test_nb_class,
    'RF_TFIDF': test_rf_class2,
    'SVM_TFIDF': test_svm_class2,
    'NB_TFIDF': test_nb_class2,
}

best_model, best_accuracy = find_best_model(y_test, models_predictions)
print(f"The best performing model is: {best_model} with accuracy: {best_accuracy:.2%}")
```

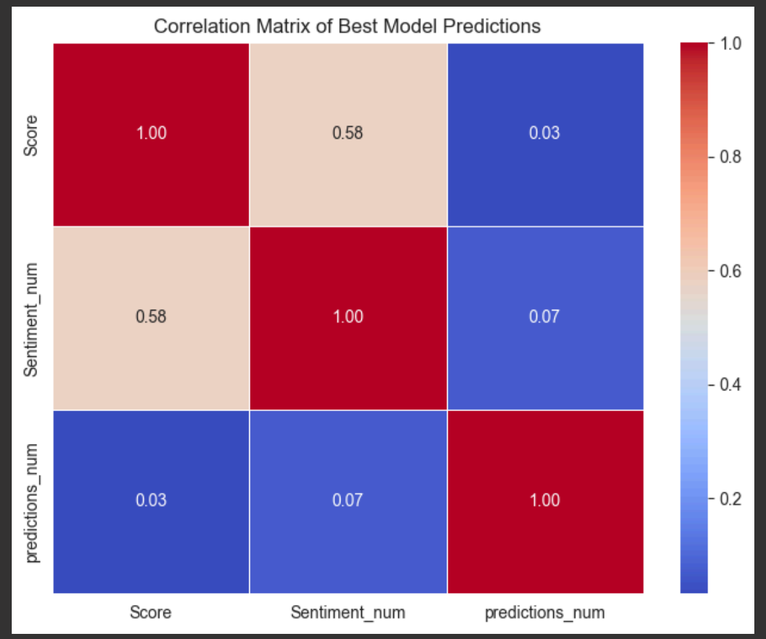
```
The best performing model is: SVM_TFIDF with accuracy: 73.01%
```

- Best model Accuracy ada di model TF - IDF (SVM) dengan score : 73,01%
- Dengan kata lain, perlu adanya peningkatan dan revisi pada program

INSIGHT

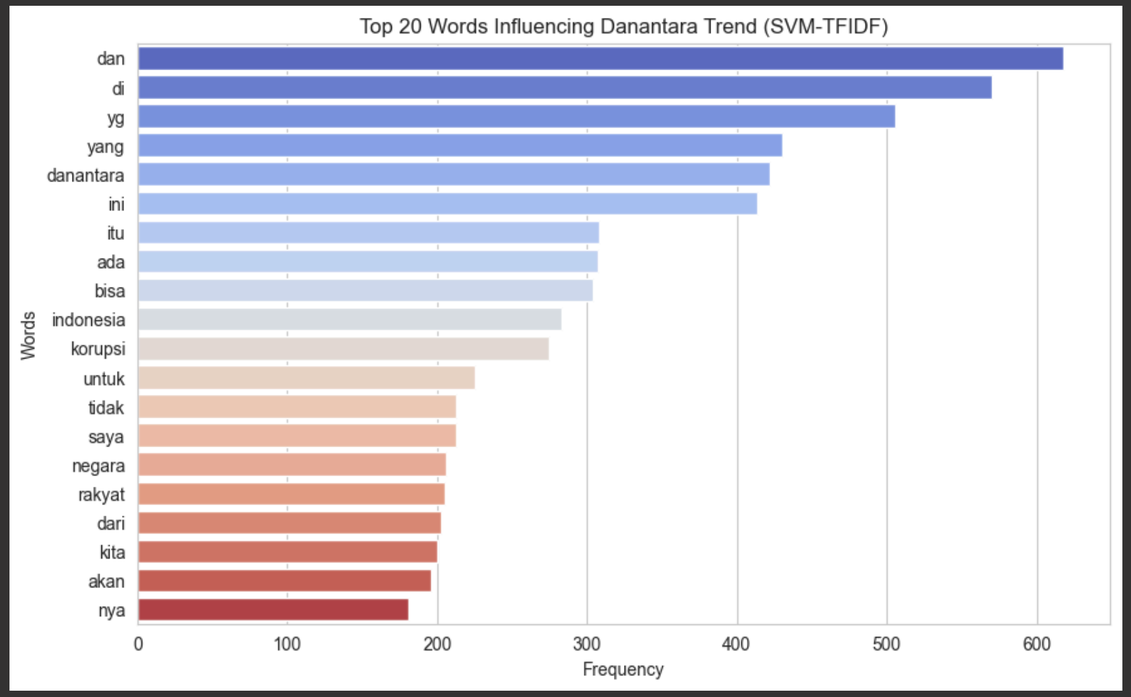
INSIGHT

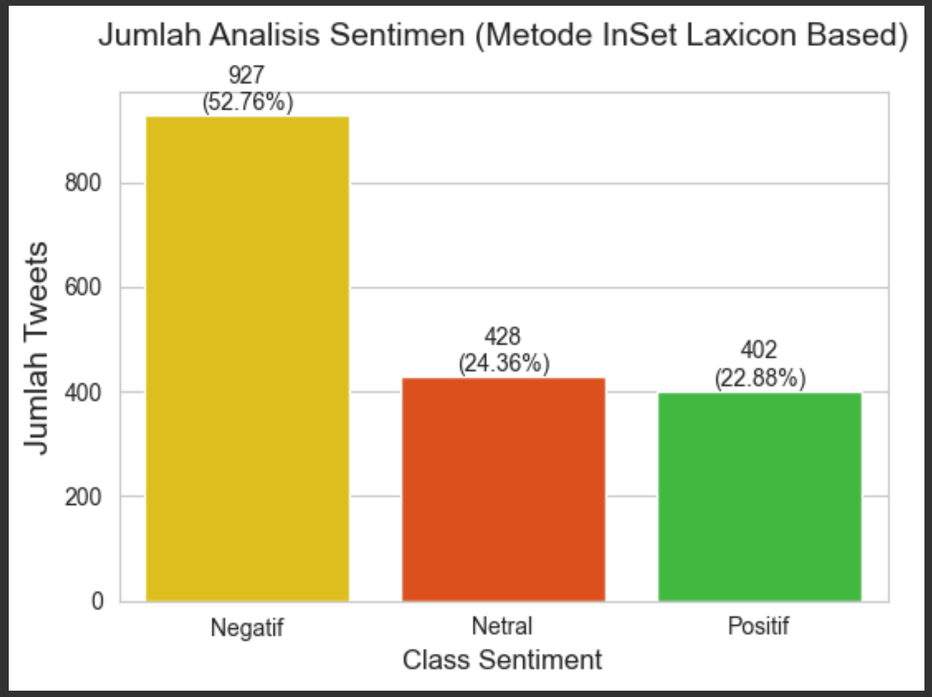
```python
best_model_predictions = pd.DataFrame({'predictions': test_svm_class2})
combined_df = pd.concat([df, best_model_predictions], axis=1)

sentiment_mapping = {'Negatif': 0, 'Netral': 1, 'Positif': 2}
combined_df['Sentiment_num'] = combined_df['Sentiment'].map(sentiment_mapping)
combined_df['predictions_num'] = combined_df['predictions'].map(sentiment_mapping)

correlation_matrix = combined_df[['Score', 'Sentiment_num', 'predictions_num']].corr()

plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5)
plt.title('Correlation Matrix of Best Model Predictions')
plt.show()
```



- Berdasarkan Correlation Matrix tersebut, terdapat korelasi yang cukup kuat antara Score dan Sentiment_num sebesar 0.58, menunjukkan bahwa nilai skor memiliki hubungan positif dengan sentimen asli. Namun, korelasi antara predictions_num dan Sentiment_num hanya 0.07, menandakan bahwa **model masih memiliki akurasi rendah dalam memprediksi sentimen sesuai data asli**.
- Insight: **Model perlu ditingkatkan**, terutama dalam mendeteksi sentimen positif dan netral, dengan optimasi fitur atau pemilihan model yang lebih kompleks.

INSIGHT





Analisis top 20 words tentang Danantara Indonesia menunjukkan bahwa masyarakat memandang Danantara sebagai agen perubahan dalam mendorong investasi strategis dan pemulihan ekonomi nasional. Namun, hasil analisis sentimen dengan metode InSet Lexicon Based menunjukkan mayoritas sentimen yang muncul adalah negatif (52.76%), menandakan adanya kekhawatiran masyarakat terhadap efektivitas dan transparansi pengelolaan investasi. Kata-kata seperti korupsi, tidak, dan masalah memperkuat adanya ketidakpercayaan terhadap tata kelola yang bersih. Meskipun demikian, kehadiran kata bisa, manfaat, dan kita menunjukkan harapan bahwa Danantara mampu membawa dampak positif bagi kesejahteraan masyarakat jika dikelola dengan baik dan transparan. Insight ini mencerminkan perlunya peningkatan komunikasi dan transparansi dalam menjawab kekhawatiran publik serta memperkuat peran Danantara sebagai agen perubahan yang inklusif dan berkelanjutan.

THANK YOU