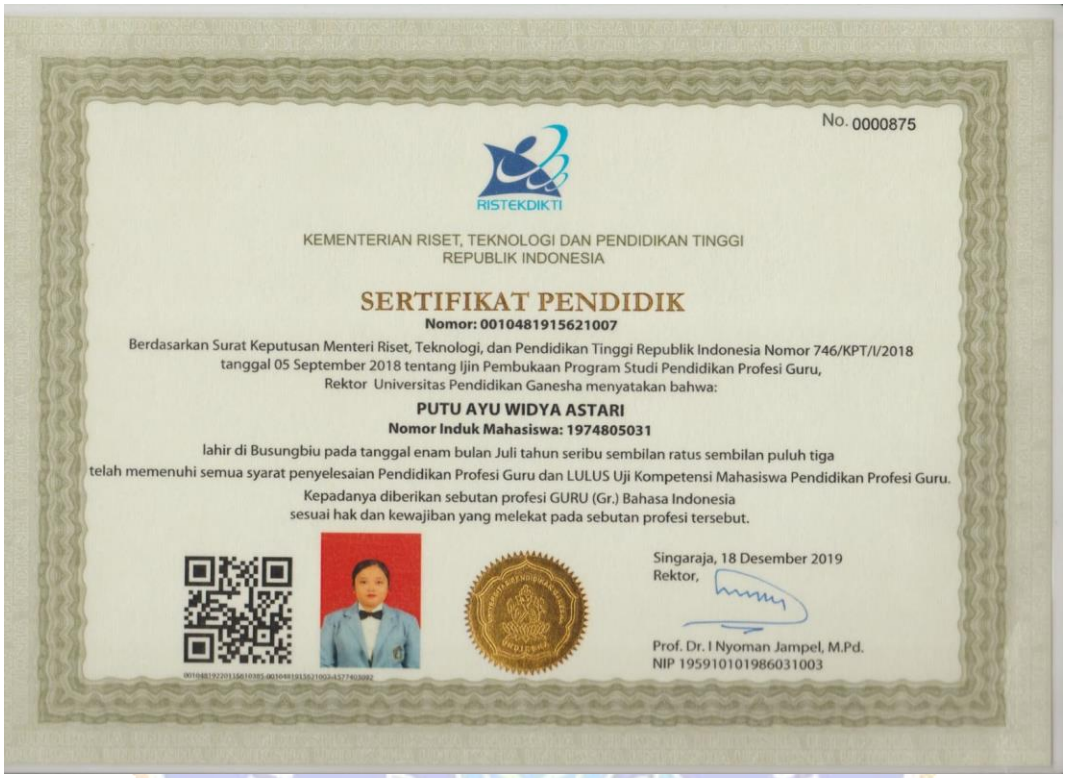
LAMPIRAN

Lampiran 1. Riwayat Hidup

RIWAYAT HIDUP

Daniel Sihite lahir di Medan pada tanggal 27 November 2001. Penulis berstatus Warga Negara Indonesia (WNI) dan berkeyakinan Agama Kristen Protestan. Alamat tinggal penulis saat ini berada di Jalan Laksamana, Kelurahan Baktiseraga, Kecamatan Buleleng, Kabupaten Buleleng, Bali. Penulis memulai pendidikan di SDN 064018 Medan dan lulus pada tahun 2013, kemudian melanjutkan pendidikan di SMP Swasta Brigjend Katamso I dan lulus tahun 2016. Setelah lulus dari jenjang SMP, kemudian penulis melanjutkan ke jenjang Sekolah Menengah Kejuruan (SMK) di Perguruan Swasta Brigjend Katamso I dengan mengambil jurusan Teknik Komputer jaringan (TKJ) dan lulus pada tahun 2019. Setelah penulis lulus dari jenjang SMK, penulis melanjutkan pendidikan ke Perguruan tinggi Universitas Pendidikan Ganesha dengan mengambil Program Studi (S1) Sistem Informasi, Jurusan Teknik Informatika, Fakultas Teknik dan Kejuruan.

Lampiran 2. Sertifikat Pendidik



No. 0000875

RISTEKDIKTI

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
REPUBLIK INDONESIA

SERTIFIKAT PENDIDIK

Nomor: 0010481915621007

Berdasarkan Surat Keputusan Menteri Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia Nomor 746/KPT/I/2018 tanggal 05 September 2018 tentang Ijin Pembukaan Program Studi Pendidikan Profesi Guru, Rektor Universitas Pendidikan Ganesha menyatakan bahwa:

PUTU AYU WIDYA ASTARI

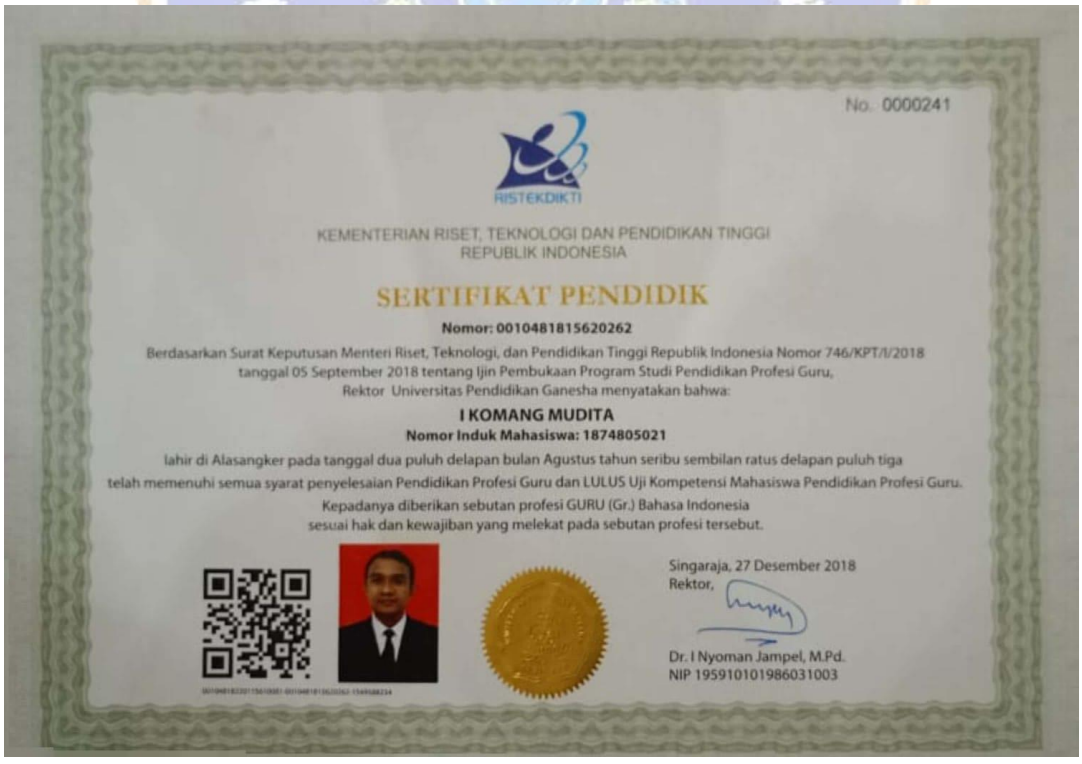
Nomor Induk Mahasiswa: 1974805031

lahir di Busungbiu pada tanggal enam bulan Juli tahun seribu sembilan ratus sembilan puluh tiga telah memenuhi semua syarat penyelesaian Pendidikan Profesi Guru dan LULUS Uji Kompetensi Mahasiswa Pendidikan Profesi Guru.

Kepadanya diberikan sebutan profesi GURU (Gr.) Bahasa Indonesia sesuai hak dan kewajiban yang melekat pada sebutan profesi tersebut.

Singaraja, 18 Desember 2019
Rektor,

Prof. Dr. I Nyoman Jampel, M.Pd.
NIP 195910101986031003



No. 0000241

RISTEKDIKTI

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
REPUBLIK INDONESIA

SERTIFIKAT PENDIDIK

Nomor: 0010481815620262

Berdasarkan Surat Keputusan Menteri Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia Nomor 746/KPT/I/2018 tanggal 05 September 2018 tentang Ijin Pembukaan Program Studi Pendidikan Profesi Guru, Rektor Universitas Pendidikan Ganesha menyatakan bahwa:

I KOMANG MUDITA

Nomor Induk Mahasiswa: 1874805021

lahir di Alasangker pada tanggal dua puluh delapan bulan Agustus tahun seribu sembilan ratus delapan puluh tiga telah memenuhi semua syarat penyelesaian Pendidikan Profesi Guru dan LULUS Uji Kompetensi Mahasiswa Pendidikan Profesi Guru.

Kepadanya diberikan sebutan profesi GURU (Gr.) Bahasa Indonesia sesuai hak dan kewajiban yang melekat pada sebutan profesi tersebut.

Singaraja, 27 Desember 2018
Rektor,

Dr. I Nyoman Jampel, M.Pd.
NIP 195910101986031003

Lampiran 3. Surat Keterangan Validasi

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
JURUSAN TEKNIK INFORMATIKA
Alamat Jalan Udayana No 11, Singaraja 81116
Telepon (0362) 25571 Fax. (0362) 25571
Laman http://ftk.undiksha.ac.id

SURAT KETERANGAN VALIDASI
LABELING DATASET

Yang bertanda tangan di bawah ini.

Nama : I Komang Mudita, S.Pd
NIP : 198308282022211003

Menerangkan bahwa Mahasiswa Universitas Pendidikan Ganesha di bawah ini:

Nama : Daniel Sihite
NIM : 1915091016
Prodi/Jurusan : Sistem Informasi/Fakultas Teknik Dan Kejuruan

Memang benar bahwa dataset yang sudah dilabeli telah divalidasi pada tanggal 26 Agustus 2024. Demikian surat keterangan ini dibuat dengan sebenarnya untuk dapat digunakan sebagaimana mestinya.

Singaraja, 26 Agustus 2024
Ahli Pakar I,

I Komang Mudita, S.Pd
NIP. 198308282022211003

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
JURUSAN TEKNIK INFORMATIKA
Alamat Jalan Udayana No 11, Singaraja 81116
Telepon (0362) 25571 Fax. (0362) 25571
Laman http://ftk.undiksha.ac.id

SURAT KETERANGAN VALIDASI
LABELING DATASET

Yang bertanda tangan di bawah ini.

Nama : Putu Ayu Widya Astari, S.Pd.
NIP : 199307062019032015

Menerangkan bahwa Mahasiswa Universitas Pendidikan Ganesha di bawah ini:

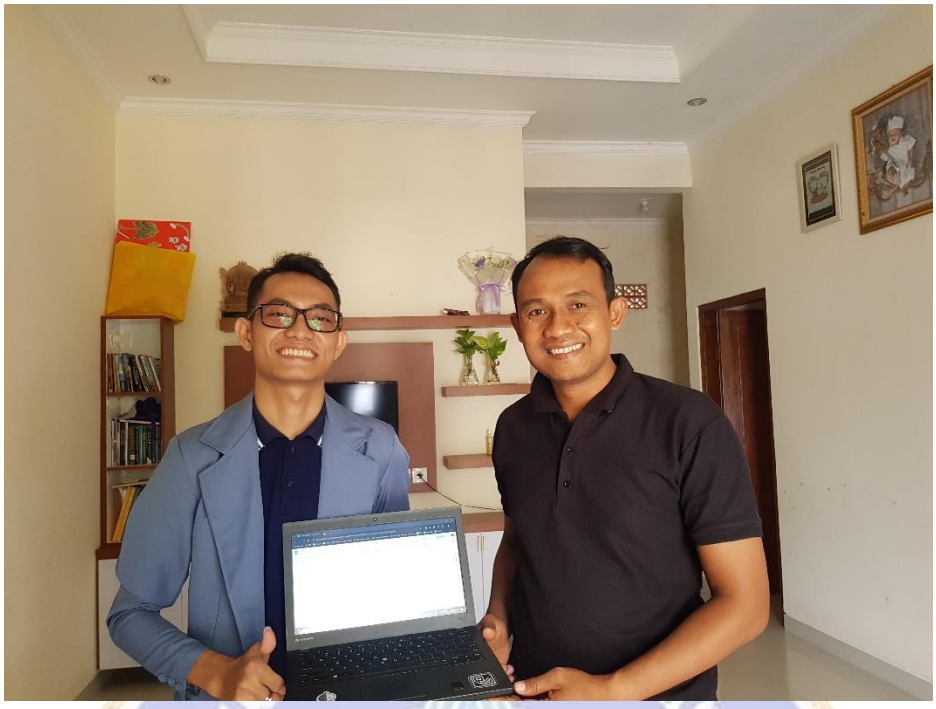
Nama : Daniel Sihite
NIM : 1915091016
Prodi/Jurusan : Sistem Informasi/Fakultas Teknik Dan Kejuruan

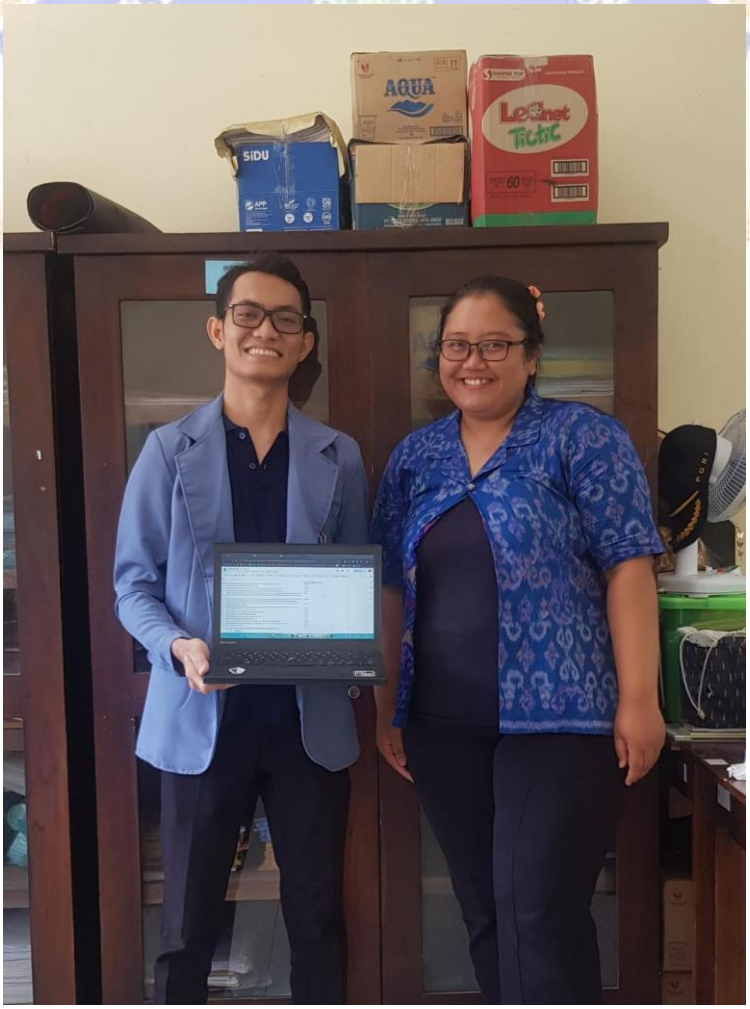
Memang benar bahwa dataset yang sudah dilabeli telah divalidasi pada tanggal 26 Agustus 2024. Demikian surat keterangan ini dibuat dengan sebenarnya untuk dapat digunakan sebagaimana mestinya.

Singaraja, 26 Agustus 2024
Ahli Pakar II,

Putu Ayu Widya Astari, S.Pd.
NIP. 199307062019032015

Lampiran 4. Foto Bersama Pakar (Guru Bahasa Indonesia)





Lampiran 5. Proses *Crawling Data*

```
# Import required Python package
!pip install pandas

# Install Node.js (because tweet-harvest built using Node.js)
!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
!curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-
repo.gpg.key | sudo gpg --dearmor -o
/etc/apt/keyrings/nodesource.gpg
!NODE_MAJOR=20 && echo "deb [signed-
by=/etc/apt/keyrings/nodesource.gpg]
https://deb.nodesource.com/node_$NODE_MAJOR.x nodistro main" |
sudo tee /etc/apt/sources.list.d/nodesource.list
!sudo apt-get update
!sudo apt-get install nodejs -y
!node -v

# Crawl Data
filename = Dataset.csv'
search_keyword = 'pinjol #pinjol akulaku kredivo easycash
lang:id since:2023-09-01 until:2024-01-31'
limit = 500

# Specify the path to your CSV file
file_path = f"tweets-data/{filename}"

# Read the CSV file into a pandas DataFrame
df = pd.read_csv(file_path, delimiter=",")

# Display the DataFrame
display(df)

# Cek jumlah data yang didapatkan
num_tweets = len(df)
print(f"Jumlah tweet dalam dataframe adalah {num_tweets}.")
```

Lampiran 6. Proses *Cleaning Data*

```
# Cleaning data
def clean_tweet(tweet):
    # Menghapus karakter non-alphanumeric dan username
    tweet = re.sub('[^0-9a-zA-Z@]+', ' ', tweet)
    tweet = re.sub(r'@\w+', '', tweet)
    tweet = re.sub(r'@', '', tweet)
    # Hapus URL
    tweet = re.sub(r'http\S+|www\S+|https\S+', '', tweet)
    # Menghapus angka dalam kata
    tweet = re.sub(r'\d', '', tweet)
    # Menghapus spasi di awal dan akhir teks
    tweet = tweet.strip()
    # Menghapus duplikasi karakter berlebihan
    tweet = re.sub(r'(.)\1+', r'\1', tweet)
    return tweet
#mengaplikasikan cleaning data pada setiap tweet
df['Tweet'] = df['Tweet'].apply(clean_tweet)
# Menampilkan data setelah dilakukan preprocessing
display(df.head(10))
```

Lampiran 7. Proses *Case Folding*

```
# Fungsi untuk melakukan case folding
def case_folding(tweet):
    return tweet.lower()

#mengaplikasikan Case Folding pada setiap tweet
df['Tweet'] = df['Tweet'].apply(case_folding)

# Menampilkan data setelah dilakukan preprocessing
display(df.head(10))
```

Lampiran 8. Proses *Stopword Removal*

```
# Fungsi untuk melakukan stopword removal
def remove_stopwords(tweet, stopwords):
    words = nltk.word_tokenize(tweet)
    filtered_words = [word for word in words if word not in
stopwords]
    return ' '.join(filtered_words)

# Baca file eksternal yang berisi stop words tambahan
def read_additional_stopwords(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        additional_stopwords = file.read().splitlines()
    return additional_stopwords

# Baca file eksternal yang berisi stop words tambahan
additional_stopwords =
read_additional_stopwords('stopword.txt')

# Stop words tambahan manual
manual_stopwords = ['sih', 'trs','nya', 'yg', kl','dah','kalo',
'aju', 'aka', 'ye', 'yuk', 'ke', 'yalah', 'dehhh', 'ngga','w',
'aj','ga', 'mulu', 'gitu','askrl', 'tau','gatau', 'mah', 'ih',
'si','tuh','n','tt','mana','tuh','pak','ac','di','elu','hehe','
baik','soal','lah','ah','mingat','yujiem','adakm','able','nder'
, 'u', 'ni', 'loh', 'adan', 'aka','aju','ya','tu','ada','apan',
'masi', 'kali', 'nah', 'dg', 'gasi','hi', 'nih', 'an', 'anjir',
'mau', 'lu', 'gimana', 'gaje','yah','kan', 'pas','utk','disat',
'yang','loh','nga','deh','malah','jah', 'lo','ukt','itb','kok',
'gais' ,'gk' , 'ye', 'woi', 'ga','gw', 'gue','bwt','ku','plis',
'hah','g','gak','apa','ap','pa','aja','sdh','dh','udah','bgt','
wkwkwkwk','persen','ala']

# Download the 'stopwords' resource from NLTK
import nltk
nltk.download('stopwords')

# Menggabungkan stop words bawaan dengan stop words tambahan
from nltk.corpus import stopwords
stop_words = set(stopwords.words('indonesian') +
additional_stopwords + manual_stopwords)

# Lakukan stopword removal dengan stop words tambahan
df['Tweet'] = df['Tweet'].apply(lambda x: remove_stopwords(x,
stop_words))

# Menampilkan data setelah dilakukan preprocessing
display(df.head(10))
```

Lampiran 9. Proses *Tokenizing*

```
# Fungsi untuk melakukan tokenisasi
def tokenize(tweet):
    tokens = nltk.word_tokenize(tweet)
    return tokens
#mengaplikasikan Tokenizing pada setiap tweet
df['Tweet'] = df['Tweet'].apply(tokenize)
# Menampilkan data setelah dilakukan preprocessing
display(df.head(10))
```

Lampiran 10. Proses *Stemming*

```
# Fungsi untuk melakukan stemming
def stem(tokens):
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    stemmed_tokens = [stemmer.stem(token) for token in tokens]
    return stemmed_tokens
# Membaca file eksternal
with open('kata-dasar.txt', 'r') as file:
    content = file.read()
    tokens = content.split()  # Mengambil token-token dari file
#mengaplikasikan cleaning data pada setiap tweet
df['Tweet'] = df['Tweet'].apply(lambda x: stem(x))
display(df.head(10))
```

Lampiran 11. Proses *Normalization*

```
# proses normalisasi
normalized_word = pd.read_csv(
    "normalisasi.csv")
normalized_word_dict = {}
for index, row in normalized_word.iterrows():
    if row[0] not in normalized_word_dict:
        normalized_word_dict[row[0]] = row[1]
def normalized_term(document):
    return [normalized_word_dict[term] if term in
normalized_word_dict else term for term in document]
df['Tweet'] = df['Tweet'].apply(normalized_term)
# Menampilkan data setelah dilakukan preprocessing
display(df.head(10))
```

Lampiran 12. Proses TF-IDF

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer

# Step 1: Membaca data CSV
data = pd.read_csv('dataset.csv')

# Step 2: Menghapus baris yang mengandung NaN
data.dropna(subset=['tweet'], inplace=True)

# Asumsi bahwa file CSV memiliki kolom 'text'
texts = data['tweet']  # Kolom teks

# Step 3: TF-IDF
vectorizer = TfidfVectorizer(stop_words='english', min_df=2)
X_tfidf = vectorizer.fit_transform(texts)

# Step 4: Mengevaluasi hasil TF-IDF

# Mendapatkan nama-nama fitur (kata-kata unik) yang dihasilkan
oleh TF-IDF
feature_names = vectorizer.get_feature_names_out()

# Mengubah hasil TF-IDF ke dalam bentuk DataFrame untuk
visualisasi
tfidf_df = pd.DataFrame(X_tfidf.toarray(),
columns=feature_names)
# Menampilkan beberapa baris pertama hasil TF-IDF
print("Matriks TF-IDF:")
print(tfidf_df.head())

# Menampilkan ukuran dari matriks TF-IDF
print(f"\nUkuran matriks TF-IDF: {X_tfidf.shape}")  # (jumlah
dokumen, jumlah kata unik)

# Menampilkan nilai rata-rata TF-IDF untuk tiap kata
print("\nNilai rata-rata TF-IDF untuk tiap kata:")
print(tfidf_df.mean().sort_values(ascending=False).head(10))  #
10 kata dengan TF-IDF rata-rata tertinggi
```

Lampiran 13. Proses Klasifikasi

```
import pandas as pd
from sklearn.model_selection import StratifiedKFold,
cross_val_score
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import classification_report,
confusion_matrix
from sklearn.pipeline import Pipeline
from imblearn.pipeline import Pipeline as ImbPipeline  # Untuk
pipeline dengan SMOTE
from imblearn.over_sampling import SMOTE
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Fungsi untuk melakukan cross-validation, training, dan
evaluasi
def run_pipeline(X, y, use_smote=False):
    # Tentukan pipeline yang akan digunakan
    if use_smote:
        print("\n--- Model dengan SMOTE ---")
        pipeline = ImbPipeline([
            ('tfidf', TfidfVectorizer()),       # TF-IDF untuk
mengubah teks menjadi vektor numerik
            ('smote', SMOTE(random_state=42)),  # SMOTE untuk
oversampling
            ('nb', MultinomialNB())             # Model Naive
Bayes
        ])
    else:
        print("\n--- Model tanpa SMOTE ---")
        pipeline = Pipeline([
            ('tfidf', TfidfVectorizer()),  # TF-IDF untuk
mengubah teks menjadi vektor numerik
            ('nb', MultinomialNB())        # Model Naive Bayes
        ])

    # Gunakan StratifiedKFold untuk K-Fold Cross Validation
    kfold = StratifiedKFold(n_splits=10, shuffle=True,
random_state=42)
```

```
# Gunakan StratifiedKFold untuk K-Fold Cross Validation
    kfold = StratifiedKFold(n_splits=10, shuffle=True,
random_state=42)

    # Lakukan cross-validation untuk mendapatkan akurasi
    scores = cross_val_score(pipeline, X, y, cv=kfold,
scoring='accuracy')

    # Cetak akurasi setiap fold
    for i, score in enumerate(scores):
        print(f"Fold {i+1}: Akurasi = {score:.2f}")

    # Hitung dan cetak rata-rata akurasi dari 10-Fold Cross
Validation
    average_accuracy = np.mean(scores)
    print(f"\nRata-rata akurasi dari 10-Fold Cross Validation:
{average_accuracy:.2f}")

    # Variabel untuk menyimpan hasil akurasi setiap fold
    accuracies = []
    all_y_true = []  # Label asli dari semua fold
    all_y_pred = []  # Prediksi dari semua fold

    # Lakukan cross-validation secara manual untuk mendapatkan
report dan confusion matrix
    for train_index, test_index in kfold.split(X, y):
        X_train, X_test = X.iloc[train_index],
X.iloc[test_index]
        y_train, y_test = y.iloc[train_index],
y.iloc[test_index]

        # Latih model pada data pelatihan
        pipeline.fit(X_train, y_train)

        # Prediksi data uji
        y_pred = pipeline.predict(X_test)

        # Hitung akurasi
        accuracy = np.mean(y_pred == y_test)
        accuracies.append(accuracy)

        # Simpan hasil prediksi dan label asli
        all_y_true.extend(y_test)
        all_y_pred.extend(y_pred)
```

```
    # Temukan fold dengan akurasi tertinggi
    best_fold = np.argmax(accuracies) + 1  # Menambahkan 1
karena index dimulai dari 0
    print(f"\nFold dengan akurasi tertinggi adalah Fold
{best_fold} dengan akurasi {accuracies[best_fold-1]:.2f}")

    # Cetak classification report secara keseluruhan
    print("\nClassification Report:")
    print(classification_report(all_y_true, all_y_pred))

    # Buat confusion matrix secara keseluruhan
    conf_matrix = confusion_matrix(all_y_true, all_y_pred)

    # Plot confusion matrix dengan Seaborn
    plt.figure(figsize=(8, 6))
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=np.unique(y), yticklabels=np.unique(y))
    plt.xlabel('Predicted Label')
    plt.ylabel('True Label')
    plt.title('Confusion Matrix' + (' dengan SMOTE' if
use_smote else ' tanpa SMOTE'))
    plt.show()
```

```
# 1. Baca dataset dari CSV
df = pd.read_csv('set.csv')

# 2. Hapus baris dengan nilai kosong atau teks kosong
df.dropna(subset=['tweet', 'label'], inplace=True)
df = df[df['tweet'].str.strip() != '']

# 3. Pisahkan fitur dan label
X = df['tweet']  # Fitur (teks)
y = df['label']  # Label

# Jalankan pipeline tanpa SMOTE
run_pipeline(X, y, use_smote=False)

# Jalankan pipeline dengan SMOTE
run_pipeline(X, y, use_smote=True)
```