HACKTIV8

Pengelompokan / Klasifikasi Berita / Artikel Menjadi beberapa topik (Berita dunia, olahraga, bisnis, teknologi)

Group 3 - SI6-02

HACKTIV8

Nama Anggota Kelompok :

- ALFIYYAH AJENG NURARDITA (UNIVERSITAS TANJUNGPURA – TEKNIK INFORMATIKA)
- ALIF FATURRAHMAN (UNIVERSITAS JAMBI – MANAJEMEN)
- ALVYN STANE (UNIVERSITAS MIKROSKIL – TEKNIK INFORMATIKA)
- ALI TEGUH APRILIYANTO (UNIVERSITAS TANJUNGPURA – REKAYASA SISTEM KOMPUTER)
- ANDI NIRINA NURSIANA ZASQIA (UNIVERSITAS TADULAKO– TEKNIK INFORMATIKA)
- ANGELA STERA MENTARI (UNIVERSITAS TELKOM – TEKNIK TELEKOMUNIKASI)

Pengelompokan / Klasifikasi Berita / Artikel

Background & Problem Statement

Pengelompokan atau klasifikasi teks adalah proses otomatis untuk mengkategorikan dokumen teks ke dalam satu atau lebih kategori yang telah ditentukan sebelumnya. Teknik-teknik ini telah banyak digunakan dalam berbagai aplikasi, seperti sistem rekomendasi, manajemen konten, dan mesin pencari. Dengan menggunakan model machine learning dan natural language processing (NLP), kita dapat membangun sistem yang mampu memahami dan mengklasifikasikan teks dengan akurasi tinggi.

Dalam era digital saat ini, volume berita dan artikel yang diproduksi setiap hari sangat besar. Banyaknya informasi ini membuat sulit bagi pengguna untuk menemukan dan mengkategorikan konten yang relevan sesuai dengan minat mereka. Oleh karena itu, dibutuhkan suatu sistem yang mampu mengelompokkan atau mengklasifikasikan berita dan artikel ke dalam beberapa topik, seperti olahraga, teknologi, bisnis, dan sebagainya, secara otomatis. Sistem ini diharapkan dapat membantu pengguna untuk lebih mudah dalam menavigasi dan menemukan konten yang relevan dengan minat mereka.

HACKTIV8

IMPORT LIBRARY

Import library sangat bermanfaat untuk meningkatkan efisiensi, kualitas, dan kinerja dalam pengembangan perangkat lunak. Dengan mengimpor library, kita dapat menggunakan fungsi, kelas, dan modul yang telah dibuat, sehingga menghemat waktu dan usaha serta meningkatkan produktivitas.

```
!pip install -U -q ibm-watson-machine-learning
!pip install -q python-dotenv
!pip install -U scikit-learn==1.1
```

```
#Import Libraries
import re
import os
import json
import nltk
import joblib
import sklearn
import tarfile

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

from nltk.tokenize import word_tokenize
from nltk.stem.porter import PorterStemmer
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB

from transformers import AutoTokenizer, Trainer, TrainingArguments, AutoModelForSequenceClassification

from dotenv import load_dotenv

from ibm_watson_machine_learning import APIClient

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

DATA LOADING

Data loading berfungsi untuk mengambil data dari berbagai sumber seperti file lokal (misalnya CSV, Excel), database, atau API, dan memuatnya ke dalam program Python untuk diproses lebih lanjut. Proses ini melibatkan membaca data dari sumber eksternal, mengonversinya ke dalam struktur data yang sesuai dalam Python

```
[ ]  !mkdir ~/.kaggle
     !cp /content/kaggle.json ~/.kaggle

     cp: cannot stat '/content/kaggle.json': No such file or directory

[ ]  !ls ~/.kaggle

     ls: cannot access '/root/.kaggle': No such file or directory

[ ]  !kaggle datasets download -d amananandrai/ag-news-classification-dataset

     Dataset URL: https://www.kaggle.com/datasets/amananandrai/ag-news-classification-dataset
     License(s): unknown
     Downloading ag-news-classification-dataset.zip to /content
      35% 4.00M/11.4M [00:00<00:00, 41.2MB/s]
     100% 11.4M/11.4M [00:00<00:00, 68.7MB/s]

[ ]  !unzip -q /content/ag-news-classification-dataset.zip
```

DATA LOADING

Hasil menunjukkan bahwa dataset df_train dan df_test sudah cukup bersih dari duplikasi dan missing value pada kolom-kolom yang telah diperiksa. Hal ini penting untuk memastikan kualitas data sebelum melanjutkan ke tahap analisis atau pemrosesan data lebih lanjut.

```
# Menghitung jumlah duplikasi
print("Jumlah duplikasi:", df_train.duplicated().sum())
print("Jumlah duplikasi:", df_test.duplicated().sum())
```

```
Jumlah duplikasi: 0
Jumlah duplikasi: 0
```

```
# Mengecek missing value dari kedua dataset
print(df_train.isnull().sum())
print()
print(df_test.isnull().sum())
```

```
Class Index    0
Title          0
Description    0
dtype: int64

Class Index    0
Title          0
Description    0
dtype: int64
```

Data Processing & EDA

Data processing dan EDA sering kali berjalan beriringan dalam siklus analisis data. Data processing memastikan bahwa data berada dalam kondisi yang baik dan siap untuk dianalisis, sedangkan EDA membantu memahami dan mengekstrak wawasan dari data tersebut.

Hasil dari EDA dapat mengarahkan langkah-langkah lebih lanjut dalam data processing, seperti transformasi tambahan atau pembersihan lebih lanjut, sebelum data tersebut digunakan dalam model pembelajaran mesin atau analisis statistik yang lebih mendalam.

HACKTIV8

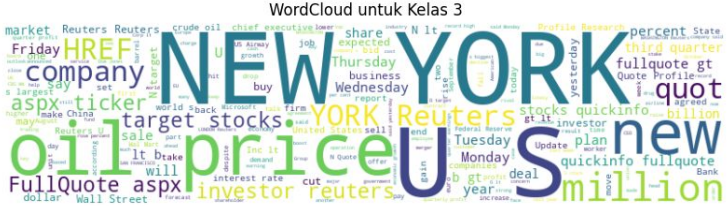
EDA

Visualisasikan kata-kata yang paling sering muncul dalam teks yang terkait dengan setiap kelas dalam dataset.

WordCloud untuk Kelas 1

WordCloud untuk Kelas 2

WordCloud untuk Kelas 3



WordCloud untuk Kelas 4

Feature Engineering

Mengubah teks mentah menjadi fitur-fitur yang lebih representatif dan berguna untuk model machine learning. Langkah-langkah yang dilakukan adalah:

1. Mendefinisikan kata umum yang tidak memiliki banyak arti atau juga disebut stopwords
2. Membuat fungsi preprocessing text, fungsi ini melakukan pemrosesan text yaitu:
    1. Mengubah menjadi lowercase
    2. Menghapus source berita
    3. Menghapus tanda baca
    4. Menghapus mention dan hashtags
    5. Menghapus baris baru
    6. Menghapus spasi berlebih
    7. Menghapus link
    8. Menghapus kata yang kurang dari sama dengan 2 huruf
    9. Tokenization
    10. Menghapus stopwords dan stemming. Menggabungkan Kembali kata menjadi 1 string

    menggabungkan Kembali token menjadi satu string yang telah dibersihkan
3. Text yang sudah diproses ditampung dalam kolom baru bernama ‘text_processed’

```python
# Membuat fungsi untuk preprocessing data
def text_preprocessing(text):
  # Mengubah menjadi lowercase
  text = text.lower()

  # Menghapus source berita
  text = re.sub(r'\([^)]*\)', '', text)

  # Menghapus tanda baca
  text = re.sub(r'[^\w\s]', '', text)

  # Menghapus mention dan hashtags
  text = re.sub("@[A-Za-z0-9_]+", " ", text)
  text = re.sub("#[A-Za-z0-9_]+", " ", text)

  # Menghapus baris baru
  text = re.sub(r"\\n", " ",text)

  # Menghapus spasi berlebih
  text = re.sub(r'\s+', ' ', text).strip()

  # Menghapus link
  text = re.sub(r"http\S+", " ", text)
  text = re.sub(r"www.\S+", " ", text)

  # Menghapus kata yang kurang dari sama dengan 2 huruf
  text = ' '.join([word for word in text.split() if len(word) > 2])

  # Tokenization
  tokens = word_tokenize(text)

  # Menghapus Stopword dan Stemming
  stemmer = PorterStemmer()
  tokens = [stemmer.stem(word) for word in tokens if word not in stpwds_id]

  # Menggabungkan kembali kata menjadi 1 string
  cleaned_text = ' '.join(tokens)

  return cleaned_text
```

PEMODELAN DATA

Model LinearSVC dinilai lebih baik dari pada ke 2 permodelan lainnya karena berdasarkan akurasi dan **F1-scorenya** yang lebih tinggi. Model ini juga menunjukkan kinerja terbaik dalam akurasi keseluruhan dan performa yang seimbang di semua kelas.

LinearSVC

	precision	recall	f1-score	support
0	0.93	0.90	0.91	1900
1	0.96	0.98	0.97	1900
2	0.89	0.88	0.88	1900
3	0.89	0.91	0.90	1900
accuracy			0.92	7600
macro avg	0.92	0.92	0.92	7600
weighted avg	0.92	0.92	0.92	7600

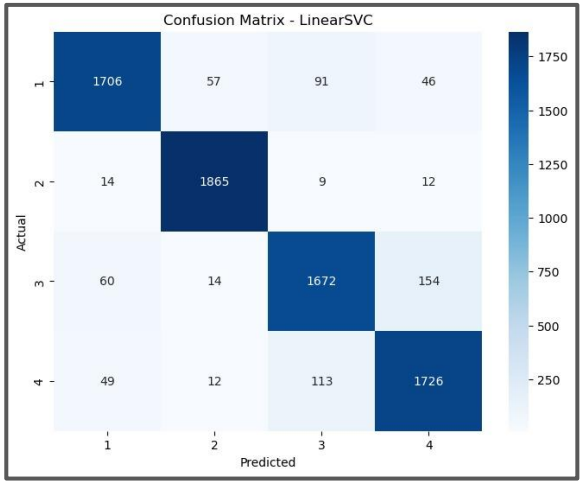
Logistic Regression

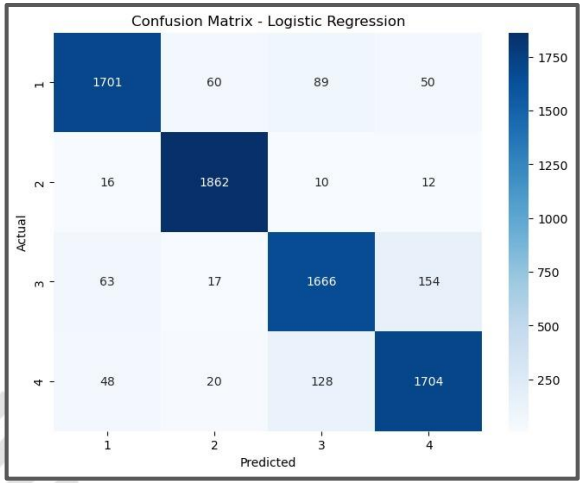
	precision	recall	f1-score	support
0	0.93	0.90	0.91	1900
1	0.95	0.98	0.97	1900
2	0.88	0.88	0.88	1900
3	0.89	0.90	0.89	1900
accuracy			0.91	7600
macro avg	0.91	0.91	0.91	7600
weighted avg	0.91	0.91	0.91	7600

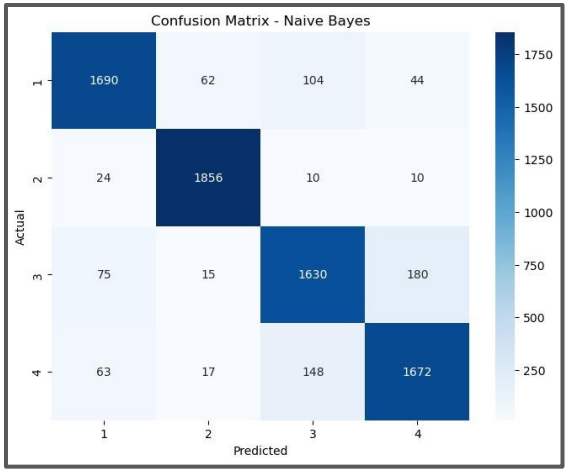
Naives Bayes

	precision	recall	f1-score	support
0	0.91	0.89	0.90	1900
1	0.95	0.98	0.96	1900
2	0.86	0.86	0.86	1900
3	0.88	0.88	0.88	1900
accuracy			0.90	7600
macro avg	0.90	0.90	0.90	7600
weighted avg	0.90	0.90	0.90	7600

CONFUSION MATRIX







DEPLOYMENT

Inisialisasi:

Kode dimulai dengan memuat variabel lingkungan dari **file .env** menggunakan **load_dotenv().**

Kemudian, dikonfigurasi dictionary **wml_credentials** dengan API key dan lokasi (dalam kasus ini, **us-south**) untuk instance **WML.**

```
#Memuat variabel lingkungan dari file .env
load_dotenv()

#Mengambil nilai dari variabel lingkungan
api_key = '8FRki8Wu66A_RA9mxls9-Nzk1zWl5rkxTQK8z9GE256-'
location = 'us-south'

#Menggunakan nilai untuk membuat wml_credentials
wml_credentials = {
    "apikey": api_key,
    "url": f'https://{location}.ml.cloud.ibm.com'
}
#Memulai WML client
client = APIClient(wml_credentials)

client.spaces.list(limit=10)
```

DEPLOYMENT

Persiapan Model:

Kode mendefinisikan metadata untuk model, termasuk nama, tipe, dan UID spesifikasi perangkat lunak.

Model kemudian disimpan di repository WML menggunakan metode store_model, yang menerima nama file model (tarfile_name), metadata, dan data pelatihan sebagai argumen.

Deployment Model:

Metode create digunakan untuk mengembangkan model. Metode ini menerima UID artefak model yang disimpan (published_model_id) dan properti deployment sebagai argumen.

Properti deployment termasuk nama deployment ("**project capstone NLP**") dan konfigurasi online kosong.

```
space_id = '10b5b9cb-5989-4308-911c-8a5cec675f7a' #id dari spacenya capstone-nlp-pro

client.set.default_space(space_id)

sofware_spec_uid = client.software_specifications.get_id_by_name("runtime-23.1-py3.10")
print(sofware_spec_uid)

#Define model name, autor name and email.
metadata = {
    client.repository.ModelMetaNames.NAME: "Project Capstone NLP",
    client.repository.ModelMetaNames.TYPE: "scikit-learn_1.1",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: '336b29df-e0e1-5e7d-b6a5-f6ab722625b2'
}

#Simpan model ke Watson Machine Learning repository
published_model = client.repository.store_model(
    model = tarfile_name,
    meta_props = metadata ,
    training_data = x_train_tfidf,
    training_target = y_train
)

print("Model successfully uploaded to Watson Machine Learning repository")
print(published_model)
```

DEPLOYMENT

Setelah pengembangan, kode mengambil rincian model yang dideploy menggunakan **metode get_details** dan mencetaknya dalam format **JSON.**

Model deployment ini mengikuti langkah-langkah yang biasa digunakan untuk mengembangkan model machine learning pada **WML**

```
published_model_id = client.repository.get_model_id(published_model)
model_details = client.repository.get_details(published_model_id)
print(json.dumps(model_details, indent=2))
```

DEPLOYMENT

Inisialisasi: Set up klien WML dan variabel lingkungan.

Persiapan Model: Persiapkan metadata model dan data pelatihan.

Penyimpanan Model: Simpan model di repository WML.

Deployment Model: Mengembangkan model menggunakan metode create.

Rincian Model: Ambil dan cetak rincian model yang dideploy.

```
deployment_props = {
    client.deployments.ConfigurationMetaNames.NAME: "project capstone NLP ",
    client.deployments.ConfigurationMetaNames.ONLINE: {}
    }

client.repository.ModelMetaNames.show()

deployment = client.deployments.create(
    artifact_uid=published_model_id,
    meta_props=deployment_props)
```

HASIL EVALUASI

Berdasarkan hasil evaluasi tiga model klasifikasi, yaitu **Logistic Regression, Naive Bayes**, dan **LinearSVC** pada dataset yang berisi **7600** sampel dengan 4 kelas, berikut adalah rincian performanya:

1. **Logistic Regression**
   - Akurasi: 0.91
   - Macro avg (Precision, Recall, F1-Score): 0.91
   - Weighted avg (Precision, Recall, F1-Score): 0.91
   - Masalah: Model mengalami masalah konvergensi, membutuhkan peningkatan iterasi atau scaling data.

2. **Naive Bayes**
   - Akurasi: 0.90
   - Macro avg (Precision, Recall, F1-Score): 0.90
   - Weighted avg (Precision, Recall, F1-Score): 0.90
   - Kelebihan: Mudah dan cepat dilatih.

3. **LinearSVC**
   - Akurasi: 0.92
   - Macro avg (Precision, Recall, F1-Score): 0.92
   - Weighted avg (Precision, Recall, F1-Score): 0.92
   - Kesimpulan: Performa terbaik secara keseluruhan.

KESIMPULAN

Untuk mengembangkan sistem klasifikasi otomatis yang mampu mengkategorikan berita dan artikel ke dalam beberapa topik utama, disarankan menggunakan **LinearSVC** karena secara keseluruhan Model ini dapat memberikan performa terbaik.

Dengan menggunakan **LinearSVC**, diharapkan sistem yang dikembangkan mampu mencapai target pencapaian dan metrik evaluasi seperti **akurasi, precision, recall, dan F1-score** yang diharapkan, sehingga membantu pengguna untuk lebih mudah menavigasi dan menemukan konten yang relevan dengan minat mereka.

Thank You

HACKTIV8

Alfiyyah Ajeng N.
UNIVERSITAS TANJUNGPURA
Teknik Informatika

Alif Faturrahman
UNIVERSITAS JAMBI
Manajemen

Alvyn Stane
UNIVERSITAS MIKROSKIL
Teknik Informatika

Ali Teguh A.
UNIVERSITAS TANJUNGPURA
Rekayasa Sistem Komputer

Andi Nirina N Z.
UNIVERSITAS TADULAKO
Teknik Informatika

Angela Stera M.
UNIVERSITAS TELKOM
Teknik Telekomunikasi

Hacktiv8id

Hacktiv8id

Hacktiv8

Hacktiv8 Indonesia