MODUL 10: TEXT CLASSIFICATION

10.1 Deskripsi Singkat

Klasifikasi teks (text classification) merupakan salah satu tugas penting dalam pembelajaran mesin (machine learning). Pada text classification, kita melakukan proses penetapan tag/kategori ke suatu dokumen agar secara otomatis dan cepat dapat menganalisis teks untuk digunakan pada aplikasi lanjutan seperti analisis sentimen, deteksi spam, pelabelan topik, deteksi hoax, dll. Data tidak terstruktur berupa teks pada email, media sosial, aplikasi chat, respon survei, dll tersedia secara melimpah saat ini. Teks dapat diolah menjadi sumber informasi yang kaya, namun karena sifatnya yang tidak terstruktur, ada tantangan tersendiri untuk melakukan ekstraksi pengetahuan dari teks tersebut.

10.2 Tujuan Praktikum

Setelah praktikum pada modul 10 ini diharapkan mahasiswa mempunyai kompetensi sebagai berikut:

1) Dapat mempraktekkan preprocessing teks dan analisis data eksploratif untuk persiapan text classification.
2) Dapat mempraktekkan TF-IDF sebagai feature dari suatu teks.
3) Dapat mempraktekkan text classification dengan Naive Bayes.

10.3 Material Praktikum

Tidak ada

10.4 Kegiatan Praktikum

Text classification digunakan untuk mengkategorikan sekumpulan label/kategori yang telah ditentukan sebelumnya ke suatu kumpulan teks. Text classification dapat digunakan untuk mengatur, menyusun, dan mengkategorikan hampir semua jenis teks. Secara umum, ada beberapa tahapan dalam text classification, yaitu:

a. Penyiapan library dan dataset

b. Analisis data eksploratif

c. Text pre-processing

d. Ekstraksi feature dari teks

e. Pembangunan model klasifikasi teks

f. Evaluasi model klasifikasi

Pada modul 10 ini kita akan mempraktekkan penggunaan vektor TF-IDF sebagai feature dari data teks serta melakukan klasifikasi teks dengan model Naive Bayes.

A. Penyiapan Library dan Dataset

Untuk tahap persiapan, kita akan meng-import beberapa package/library pada Python yang diperlukan baik nantinya untuk pre-processing teks, pembangunan model, tokenisasi, dsb.

```
### 1. Tahap Persiapan
Import package/library yang diperlukan
"""
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt

# untuk pre-processing teks
import re, string

# bag of words
from sklearn.feature_extraction.text import TfidfVectorizer

#untuk pembangunan model
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, f1_score, accuracy_score,
confusion_matrix
```

Berikutnya, kita bisa menyiapkan data training yang diperlukan. Kali ini kita akan menggunakan dataset analisis sentimen pada tweet (Koto and Rahmaningtyas, 2017) yang diambil dari:

https://github.com/indolem/indolem/tree/main/sentiment

Tugas kita adalah memprediksi apakah suatu tweet memiliki sentimen positif (kode 1) atau negatif (kode 0). Data training dan testing dapat diambil dari:

https://github.com/indolem/indolem/tree/main/sentiment/data

Dalam modul ini, data yang digunakan yaitu `train0.csv` dan `test0.csv`. Setelah itu, sesuaikan dengan alamat file pada komputer ataupun Google Drive Anda.

```
# Uncomment baris-baris berikut jika file data training disimpan di komputer
# import os
# os.chdir('/Users/xxx/Documents/')
# df_train=pd.read_csv('train0.csv')
from google.colab import drive
drive.mount("/content/drive", force_remount=True)
df_train=pd.read_csv('/content/drive/MyDrive/kuliah/Information   Retrieval
22-23/Bahan Modul/bahan_latihan10/train0.csv')
print(df_train.shape)
df_train.head()
# Baris-baris berikut digunakan jika file data training disimpan di Google
Drive
from google.colab import drive
drive.mount("/content/drive", force_remount=True)
```

```
df_train=pd.read_csv('/content/drive/MyDrive/kuliah/Information Retrieval 22-23/Bahan Modul/bahan_latihan10/train0.csv')
print(df_train.shape)
df_train.head()
```

	sentence	sentiment
0	Kangen NaBil @RealSyahnazS @bangbily RaGa @Raf...	1
1	Doa utk orang yg mberi makan: Ya Allah! Berila...	1
2	Setiap kali HP aku bunyi, aku selalu berharap ...	1
3	Belum pernah sedekat ini wawancara dgn Afgan S...	1
4	Dulu masa first pergi award show amatlah malas...	1

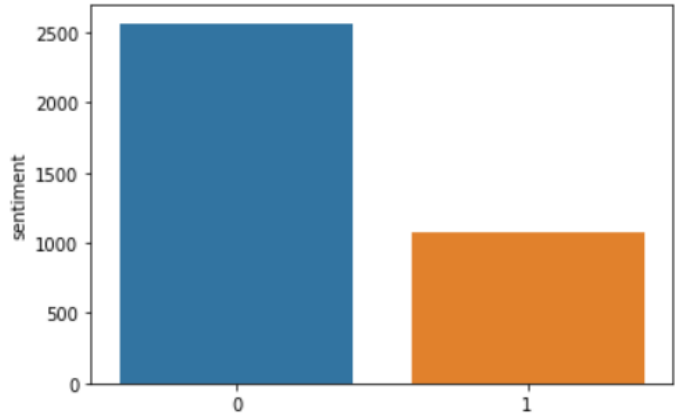
```
df_test=pd.read_csv('/content/drive/MyDrive/kuliah/Information Retrieval 22-23/Bahan Modul/bahan_latihan10/test0.csv')
```

B. Analisis Data Eksploratif

Pada tahap eksplorasi data (*Exploratory Data Analysis/EDA*) ini, kita akan mencoba untuk melihat distribusi kelas, pengecekan missing data, penghitungan jumlah kata, karakter, dan visualisasi data.

```
# CLASS DISTRIBUTION
# mengecek apakah dataset yang digunakan balance atau tidak
x=df_train['sentiment'].value_counts()
print(x)
sns.barplot(x.index,x)
```

```
0    2567
1    1071
Name: sentiment, dtype: int64
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fbd001e3f90>
```



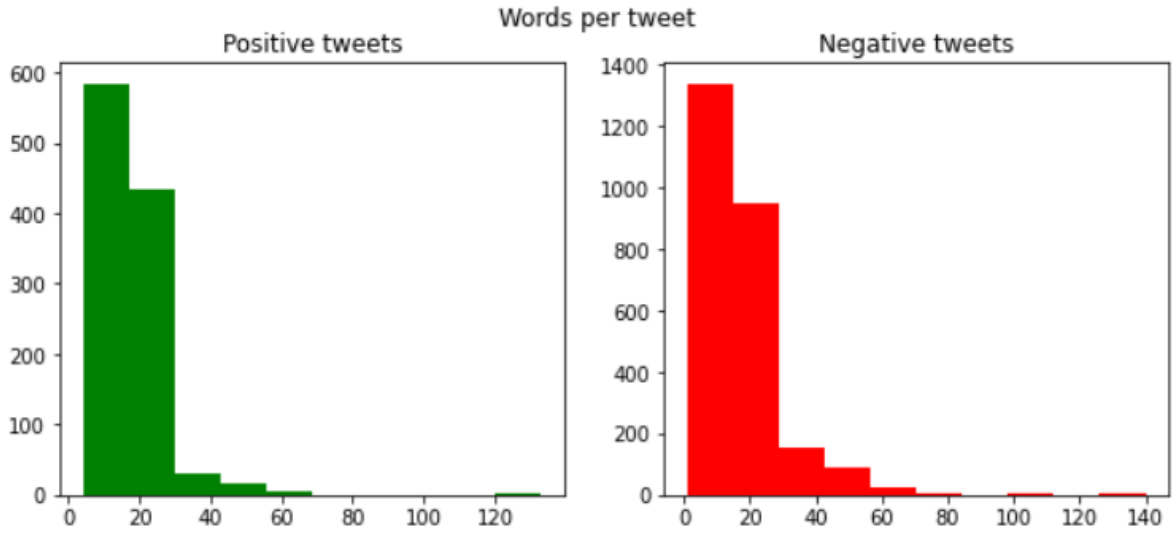
```
# Memeriksa missing values
df_train.isna().sum()
```

```
sentence     0
sentiment    0
dtype: int64
```

```
#1. WORD-COUNT
df_train['word_count']      =      df_train['sentence'].apply(lambda      x:
len(str(x).split()))
print(df_train[df_train['sentiment']==1]['word_count'].mean()) #Positive
print(df_train[df_train['sentiment']==0]['word_count'].mean()) #Negative

#2. CHARACTER-COUNT
df_train['char_count'] = df_train['sentence'].apply(lambda x: len(str(x)))
print(df_train[df_train['sentiment']==1]['char_count'].mean()) #Positive
print(df_train[df_train['sentiment']==0]['char_count'].mean()) #Negative

#3. UNIQUE WORD-COUNT
df_train['unique_word_count']    =    df_train['sentence'].apply(lambda    x:
len(set(str(x).split())))
print(df_train[df_train['sentiment']==1]['unique_word_count'].mean())
#Positive
print(df_train[df_train['sentiment']==0]['unique_word_count'].mean())
#Negative
```

```
16.985060690943044
16.684456564082588
121.1484593837535
111.01051811453058
16.166199813258636
15.502532138683287
```

```
# Plotting word-count per tweet
fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,4))
train_words=df_train[df_train['sentiment']==1]['word_count']
ax1.hist(train_words,color='green')
ax1.set_title('Positive tweets')
train_words=df_train[df_train['sentiment']==0]['word_count']
ax2.hist(train_words,color='red')
ax2.set_title('Negative tweets')
fig.suptitle('Words per tweet')
plt.show()

df_train=df_train.drop(columns=['word_count','char_count','unique_word_coun
t'])
```



C. Teks Pre-processing

Pada tahap ini, Anda akan melakukan beberapa teknik pre-processing teks sebelum melakukan pemodelan. Anda dapat menambahkan atau mengurangi tahapan pre-processing yang dirasa perlu berdasarkan sumber data teks yang digunakan.

```
# untuk pre-processing teks

#1. Common text preprocessing
text = "@user    Teks ini mau dibersihkan. Ada beberapa karakter seperti: <br>, ?, :, ''  spasi berlebih dan tab     .  "

# mengubah ke huruf kecil (lowercase) dan menghapus tanda baca, karakter aneh dan strip
def preprocess(text):
    text = text.lower() #lowercase text
    text=text.strip()  #Menghapus leading/trailing whitespace
    text = re.sub('@[^\s]+','atUser',text) #mengubah @user menjadi atUser
    text = re.sub(r'#([^\s]+)', r'\1', text) #menghapus hashtag di depan suatu kata
    text= re.compile('<.*?>').sub('', text) #Menghapus HTML tags/markups
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub(' ', text) #Replace punctuation with space. Careful since punctuation can sometime be useful
    text = re.sub('\s+', ' ', text)  #Menghapus extra space dan tabs
    text = re.sub(r'\[[0-9]*\]',' ',text) #[0-9] matches any digit (0 to 10000...)
    text= re.sub(r'[^\w\s]', '', str(text).strip())
    text = re.sub(r'\d',' ',text) #matches any digit from 0 to 100000..., \D matches non-digits
    text = re.sub(r'\s+',' ',text) #\s matches any whitespace, \s+ matches multiple whitespace, \S matches non-whitespace

    return text
preprocess(text)
```

```
def tokenisasi(text):
    tokens = text.split(" ")
    return tokens

#STOPWORD ELIMINATION DAN STEMMING
def stemming(text, stemmer):
    # stemming process
    output   = stemmer.stem(text)
    return output

def stemming_stopword_elim(text, stopwords, stemmer):
    output = ""
    for token in tokenisasi(text):
      if not token in stopwords:
        output   = output + stemming(token, stemmer) + " "
    return output[:-1]

#FINAL PREPROCESSING
from spacy.lang.id import Indonesian
import spacy
nlp = Indonesian()  # use directly
nlp = spacy.blank('id')  # blank instance'
stopwords = nlp.Defaults.stop_words

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

def finalpreprocess(string, stopwords, stemmer):
    return stemming_stopword_elim(preprocess(string), stopwords, stemmer)
df_train['clean_text'] = df_train['sentence'].apply(lambda x:
finalpreprocess(x, stopwords, stemmer))

df_train.head()

df_train['clean_text'] = df_train['text'].apply(lambda x:
finalpreprocess(x))
df_train=df_train.drop(columns=['word_count','char_count','unique_word_coun
t'])
df_train.head()
```

Preprocess tanpa eliminasi stopword dan stemming:

	sentence	sentiment	clean_text
0	Kangen NaBil @RealSyahnazS @bangbily RaGa @Raf...	1	kangen nabil atUser atUser raga atUser atUser ...
1	Doa utk orang yg mberi makan: Ya Allah! Berila...	1	doa utk orang yg mberi makan ya allah berilah ...
2	Setiap kali HP aku bunyi, aku selalu berharap ...	1	setiap kali hp aku bunyi aku selalu berharap i...
3	Belum pernah sedekat ini wawancara dgn Afgan S...	1	belum pernah sedekat ini wawancara dgn afgan s...
4	Dulu masa first pergi award show amatlah malas...	1	dulu masa first pergi award show amatlah malas...

Preprocess dengan eliminasi stopword dan stemming:

	sentence	sentiment	clean_text
0	Kangen NaBil @RealSyahnazS @bangbily RaGa @Raf...	1	kangen nabil atuser atuser raga atuser atuser ...
1	Doa utk orang yg mberi makan: Ya Allah! Berila...	1	doa utk orang yg mberi makan ya allah ilah ber...
2	Setiap kali HP aku bunyi, aku selalu berharap ...	1	kali hp bunyi harap
3	Belum pernah sedekat ini wawancara dgn Afgan S...	1	dekat wawancara dgn afgan syahreza updateblog ...
4	Dulu masa first pergi award show amatlah malas...	1	first pergi award show malas nak pikir baju sk...

D. Ekstraksi Feature dari Data Teks

Salah satu feature dari data teks yang dapat digunakan yaitu dalam bentuk vektor dari bag of word dengan menggunakan bobot TF.IDF. Gunakan kode di bawah ini untuk mendapatkan vektor TF.IDF dari sekumpulan input teks.

```
X_train = df_train['clean_text']
y_train = df_train['sentiment']
X_test = df_test['clean_text']
y_test = df_test['sentiment']

# TF-IDF
# Konversi x_train ke vector karena model hanya dapat memproses angka, bukan kata/karakter
tfidf_vectorizer = TfidfVectorizer(use_idf=True)
X_train_vectors_tfidf = tfidf_vectorizer.fit_transform(X_train)
# tfidf digunakan pada kalimat yang belum ditokenisasi, berbeda dengan word2vec
# Hanya men-transform x_test (bukan fit dan transform)
X_test_vectors_tfidf = tfidf_vectorizer.transform(X_test)
# Jangan melakukan fungsi fit() TfidfVectorizer ke data testing karena hal itu akan
# mengubah indeks kata & bobot sehingga sesuai dengan data testing. Sebaliknya, lakukan
# fungsi fit pada data training, lalu gunakan hasil model pada data training tadi pada
# data testing untuk menunjukkan fakta bahwa Anda menganalisis data testing hanya
# berdasarkan apa yang dipelajari tanpa melihat data testing itu sendiri sebelumnya.
```

E. Pembangunan Model Klasifikasi Teks dengan Naive Bayes

Naive Bayes merupakan model klasifikasi probabilistik berdasarkan Teorema Bayes, yang menggunakan probabilitas untuk membuat prediksi berdasarkan pengetahuan sebelumnya tentang kondisi yang mungkin terkait. Algoritma ini paling cocok untuk kumpulan data besar karena mempertimbangkan setiap fitur secara independen, menghitung probabilitas setiap kategori, dan kemudian memprediksi kategori dengan probabilitas tertinggi.

```
"""#### NB (tf-idf)"""
nb_tfidf = MultinomialNB()
nb_tfidf.fit(X_train_vectors_tfidf, y_train)  #model

#Melakukan prediksi nilai y pada dataset testing
y_predict = nb_tfidf.predict(X_test_vectors_tfidf)
y_prob = nb_tfidf.predict_proba(X_test_vectors_tfidf)[:,1]
```

F. Evaluasi Model Klasifikasi

Evaluasi dari model klasifikasi dapat dilakukan dengan kode berikut.

```
print(classification_report(y_test,y_predict))
print('Confusion Matrix:',confusion_matrix(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.79      0.96      0.87       713
           1       0.80      0.39      0.52       298

    accuracy                           0.79      1011
   macro avg       0.80      0.67      0.69      1011
weighted avg       0.79      0.79      0.76      1011

Confusion Matrix: [[685  28]
 [183 115]]
```

Pastikan Anda dapat melakukan interpretasi dari hasil evaluasi tersebut.

10.5 Penugasan

1. Lakukan duplikasi dari latihan di modul ini dengan Google Colab kalian masing-masing.
2. Berlatihlah dengan data lain.