BAB 4 – TEXT MINING

TUJUAN BELAJAR

1. Memahami dan menerapkan *Text Mining* menggunakan Python
2. Mampu untuk mendapatkan pengetahuan dan insight dari data Text

DASAR TEORI

- Text mining merupakan istilah untuk mencari informasi dan insight dari data yang berupa teks
- Data teks dapat memiliki struktur semi maupun teks bebas
- Contoh data yang memiliki struktur semi yaitu html, json, iklan koran, daftar Pustaka, dsb
- contoh data teks bebas yaitu esai, berita, buku, dsb.
- Dalam python, toolkit yang dapat digunakan untuk melakukan text mining salah satunya adalah NLTK
- Untuk menginstall NLTK, cukup lakukan perintah pip install nltk dan lakukan import
- Tokenisasi merupakan langkah awal dalam pemrosesan teks, langkah ini mengubah sebuah teks menjadi list yang berisi elemen berupa setiap kata dari teks tersebut.
- Untuk melakukan tokenisasi, perintah string.split() dapat dilakukan, atau menggunakan NLTK

```
import nltk


text = "In Brazil they drive on the right-hand side of the
road. has a large coastline on the eastern side of South
America"

from nltk.tokenize import word_tokenize
token = word_tokenize(text)
token
```

```
['In','Brazil','they','drive', 'on','the', 'right hand',
'side', 'of', 'the', 'road', '.', 'Brazil', 'has', 'a',
'large', 'coastline', 'on', 'the', 'eastern', 'side', 'of',
'South', 'America']
```

- Setelah tokenisasi, kita dapat mengetahui berapa kali sebuah kata muncul dalam teks tersebut dengan mencari frekuensinya
- Frekuensi kata yang muncul dalam sebuah teks dapat menjadi tanda awal tingkat kepentingan kata tersebut dalam korpus
- Untuk mencari frekuensi kata, dapat menggunakan fungsi FreqDist dari NLTK

```
from nltk.probability import FreqDist
fdist = FreqDist(token)
fdist

output:
"FreqDist({'the': 3, 'Brazil': 2, 'on': 2, 'side': 2, 'of':
2, 'In': 1, 'they': 1, 'drive': 1, 'right-hand': 1, 'road':
1, ...})"
```

- Atau, dengan menggunakan method most_common untuk mencari beberapa kata dengan frekuensi tertinggi dalam korpus

```
from nltk.probability import FreqDist
fdist = FreqDist(token)
fdist1 = fdist.most_common(10)
fdist1

output:
[('the', 3),
 ('Brazil', 2),
 ('on', 2),
 ('side', 2),
 ('of', 2),
 ('In', 1),
 ('they', 1),
 ('drive', 1),
 ('right-hand', 1),
 ('road', 1)]
```

- Frekuensi kata ini juga dapat divisualisasikan menggunakan barplot

```
import pandas as pd

df_freq_tokens = pd.DataFrame.from_dict(freqdist1,
orient='index')
df_freq_tokens.columns = ['Frequency']
df_freq_tokens.index.name = 'Key'

df_freq_tokens.plot(kind='bar')
```

- Stemming merupakan salah satu langkah pra pemrosesan teks. Yaitu mengubah sebuah kata menjadi bentuk dasarnya, misalkan: “Waiting” menjadi “Wait”, dan “Mengubah” menjadi “Ubah”
- Proses ini merupakan proses yang *language-dependent* sehingga tidak semua library dengan stemmer method memiliki output yang sama.
- Untuk stemming dalam Bahasa Inggris, NLTK dapat digunakan
- Beberapa algoritma stemming tersedia untuk Bahasa inggris di NLTK, meliputi Porter Stemmer, Lancaster Stemmer, WordNet Lemmatizer, dan SnowBall
- ```
  from nltk.stem import PorterStemmer
  pst = PorterStemmer()
  pst.stem(“waiting”)

  output:
  'wait'
  ```
- ```
  from nltk.stem import LancasterStemmer
  lst = LancasterStemmer()
  stm = [“giving”, “given”, “given”, “gave”]
  for word in stm :
      print(word+ “:” +lst.stem(word))

  output:
  giving:giv
  given:giv
  given:giv
  gave:gav
  ```
- Bentuk lanjutan dari Stemming adalah Lemmatization
- Proses lemmatization merupakan proses yang lebih kompleks dari stemming, karena meliputi perubahan bentuk kata ke bentuk dasarnya. Contoh: “going”, “went”, “gone” memiliki output yang sama ketika dilakukan lemmatisasi yakni “go”
- Dalam python, lemmatisasi Bahasa Inggris dapat dilakukan menggunakan beberapa algoritma, yakni WordNet Lemmatizer, SpaCy Lemmatizer, TextBlob, dan Stanford CoreNLP

-
```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

print("rocks :", lemmatizer.lemmatize("rocks"))
print("corpora :", lemmatizer.lemmatize("corpora"))
```

- Untuk Bahasa Indonesia, terdapat library stemming dan lemma dengan nama PySastrawi

```
In [138]: # import Sastrawi package
          from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

          # token without stopword
          list_tokens = tokens_without_stopword

          # stem
          output   = [(token + " : " + stemmer.stem(token)) for token in list_tokens]

          output

Out[138]: ['positif : positif',
           'virus : virus',
           'corona : corona',
           'april : april',
           'orang : orang',
           'pasien : pasien',
           'sembuh : sembuh',
           'ri : ri',
           'meninggal : tinggal']
```

- Stopwords merupakan kata yang paling banyak muncul dalam sebuah teks. Biasanya tidak memiliki makna tertentu, contoh “the” “an” “a” “on” “is”.

```
from nltk.corpus import stopwords

# tokenize text
freq_tokens

# get Indonesian stopword
list_stopwords = set(stopwords.words('indonesian'))


#remove stopword pada list token
tokens_without_stopword = [word for word in
freq_tokens if not word in list_stopwords]


print(tokens_without_stopword)
```

-
```
from nltk import word_tokenize
from nltk.corpus import stopwords
a = set(stopwords.words('english'))
text = "Cristiano Ronaldo was born on February 5,
1985, in Funchal, Madeira, Portugal."

text1 = word_tokenize(text.lower())

print(text1)stopwords = [x for x in text1 if x not in
a]
print(stopwords)

output:
['cristiano', 'ronaldo', 'born', 'february', '5',
'1985',  'funchal' , 'madeira', 'portugal']
```
- Wordcloud merupakan salah satu cara visualisasi deskriptif pada data teks, sifatnya mirip dengan barplot frekuensi namun semakin besar frekuensi kata tersebut, semakin besar ukuran kata tersebut dalam wordcloud
-
```
from matplotlib import pyplot as plt
from wordcloud import WordCloud

wordcloud =
WordCloud(background_color="white").generate(string)


#plot the wordcloud
plt.figure(figsize = (12, 12))
plt.imshow(wordcloud)

#to remove the axis value
plt.axis("off")
plt.show()
```

LATIHAN DAN TUGAS PRAKTIKUM

1. Download data teks dari halaman berikut: https://raw.githubusercontent.com/ruzcmc/medmon/main/jawapos19012021.csv
2. Buatlah wordcloud dan most common word barplot, interpretasikan hasilnya
3. Lakukan clustering dengan menggunakan fitur TF-IDF
4. Buat visualisasi clusternya dan lakukan interpretasi terhadap hasil tersebut
5. Dokumentasikan setiap langkah diatas dan tulis laporannya
6. Kumpulkan dalam format pdf