BAB 4 – TEXT MINING

TUJUAN BELAJAR

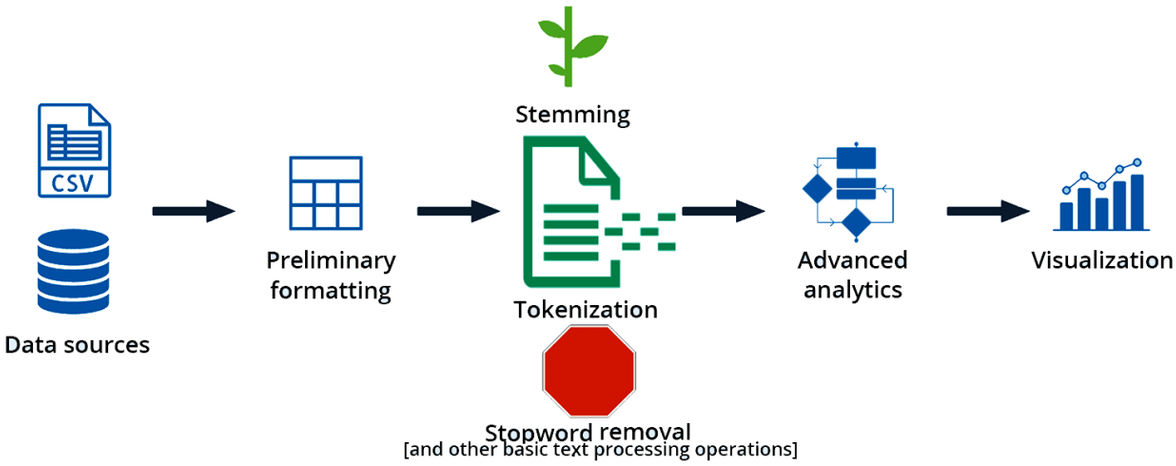
1. Memahami dan menerapkan *Text Mining* menggunakan Python
2. Mampu untuk mendapatkan pengetahuan dan insight dari data Text

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 adalah NLTK, TextBlob, dan Sastrawi untuk text yang menggunakan Bahasa Indonesia.
- Untuk menginstall NLTK, cukup lakukan perintah pip install nltk dan lakukan import. TextBlob menggunakan perintah pip install textblob dan Sastrawi menggunakan perintah pip install Sastrawi.

1. NLTK

- 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')
```

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

```
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']
```

- 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 dan TextBlob dapat digunakan.
- Beberapa algoritma stemming tersedia untuk Bahasa inggris di NLTK, meliputi Porter Stemmer, Lancaster Stemmer, WordNet Lemmatizer, dan SnowBall

```
Slide Type -
# Contoh Stemming di NLTK
from nltk.stem.lancaster import LancasterStemmer
from nltk.stem.porter import PorterStemmer
from nltk.stem.snowball import SnowballStemmer

S = 'presumably I would like to MultiPly my provision, saying tHat without crYing'
print('Sentence: ',S)

stemmer_list = [LancasterStemmer, PorterStemmer, SnowballStemmer]
names = ['Lancaster', 'Porter', 'SnowBall']
for stemmer_name,stem in zip(names,stemmer_list):
    if stemmer_name == 'SnowBall':
        st = stem('english')
    else:
        st = stem()
    print(stemmer_name,': ',' '.join(st.stem(s) for s in S.split()))
# perhatikan, kita tidak melakukan case normalization (lowercase)
# Hasil stemming bisa tidak bermakna
```

```
Sentence:  presumably I would like to MultiPly my provision, saying tHat without crYing
Lancaster :  presum i would lik to multiply my provision, say that without cry
Porter :  presum i would like to multipli my provision, say that without cri
SnowBall :  presum i would like to multipli my provision, say that without cri
```

- 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”))
```

- Dalam NLTK, Pos Tagging dapat digunakan

```
# Contoh POS tags dengan NLTK (bahasa Inggris)
from nltk import pos_tag
S = 'I am currently learning NLP in English, but if possible I want to know NLP in Indonesian language too'

tokens = word_tokenize(S)
print(pos_tag(tokens))
# Tidak lagi hanya 9 macam tags seperti yang dibahas ahli bahasa (linguist)
```

```
[('I', 'PRP'), ('am', 'VBP'), ('currently', 'RB'), ('learning', 'VBG'), ('NLP', 'NNP'), ('in', 'IN'), ('English', '
NNP'), (',', ','), ('but', 'CC'), ('if', 'IN'), ('possible', 'JJ'), ('I', 'PRP'), ('want', 'VBP'), ('to', 'TO'), ('
know', 'VB'), ('NLP', 'NNP'), ('in', 'IN'), ('Indonesian', 'JJ'), ('language', 'NN'), ('too', 'RB')]
```

2. TextBlob (Text Binary large Object)

- Tokenisasi menggunakan Textblob juga dapat dilakukan dengan menggunakan sintax

```
# Contoh tokenisasi dengan TextBlob
from textblob import TextBlob

T = "Hello, Mr. Man. He smiled!! This, i.e. that, is it."
sentence_tokens = TextBlob(T).sentences

# Tokenisasi kata
print(TextBlob(T).words)

# Tokenisasi kalimat
print([str(sent) for sent in sentence_tokens])
```

```
['Hello', 'Mr', 'Man', 'He', 'smiled', 'This', 'i.e', 'that', 'is', 'it']
['Hello, Mr. Man.', 'He smiled!!', 'This, i.e.', 'that, is it.']
```

- TextBlob juga dapat digunakan untuk Stemming dan Lemmatization

```
# Contoh TextBlob Stemming & Lemmatizer
from textblob import Word
# Stemming
print("Stem: ", Word('running').stem())

# Lemmatizer
print("Lemmatize: ", Word('went').lemmatize('v'))

# default Noun, plural akan menjadi singular dari akar katanya
# Juga case sensitive
```

```
Stem:  run
Lemmatize:  go
```

- TextBlob juga menyediakan fitur untuk Pos tagging

```
# Contoh POS tag dengan TextBlob pada bahasa Inggris
for word, pos in TextBlob(T).tags:
    print(word, pos, end=', ')
```

```
Hello NNP, Mr. NNP, Man NNP, He PRP, smiled VBD, This DT, i.e NN, that DT, is VBZ, it PRP,
```

3. Sastrawi

- Untuk Bahasa Indonesia, terdapat library dengan nama PySastrawi yang dapat digunakan

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from nltk.tokenize import word_tokenize
factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()
kalimat = "Andi kerap melakukan transaksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis
stop = stopword.remove(kalimat.lower())
print(stop)
```

```
andi kerap melakukan transaksi rutin daring online. andi belanja online lebih praktis & murah.
```

- Stemming dan lemma pada Sastrawi

Slide Type -

```
# Lemmatizer dengan Sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
stemmer = StemmerFactory().create_stemmer()

I = "perayaan itu berbarengan dengan saat kita bepergian ke Makassar"
print(stemmer.stem(I))
print(stemmer.stem("Perayaan Bepergian Menyuarakan"))
# Ada beberapa hal yang berbeda antara Sastrawi dan modul-modul diatas.
# Apa sajakah?
```

```
raya itu bareng dengan saat kita pergi ke makassar
raya pergi suara
```

- 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

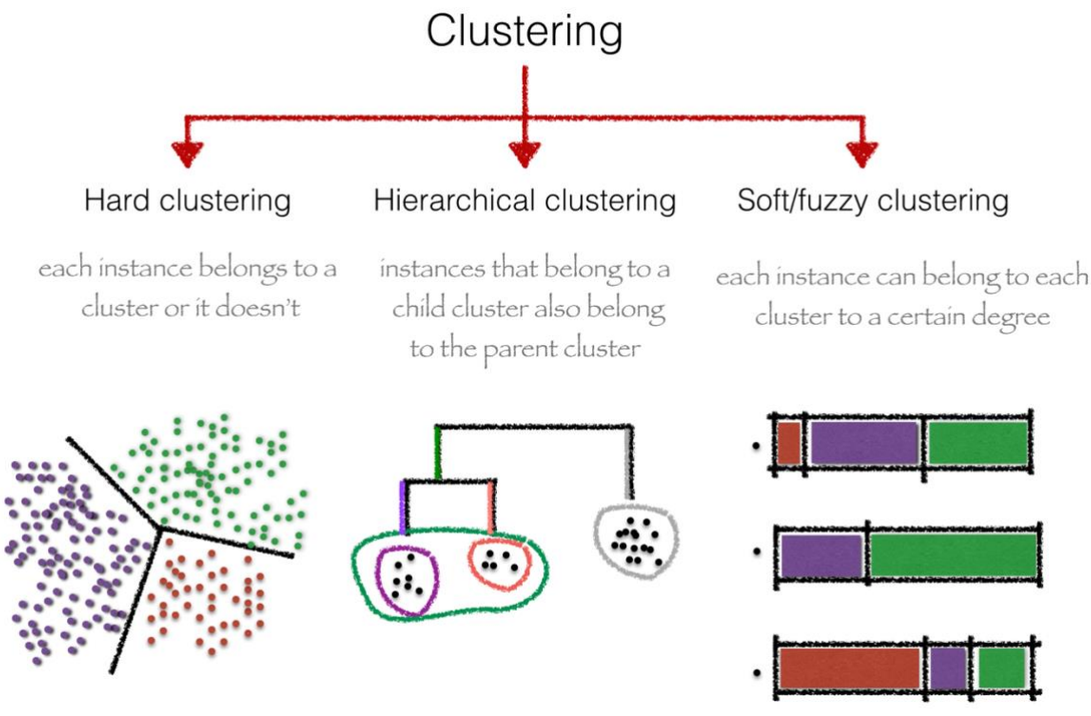
```
from matplotlib import pyplot as plt
from wordcloud import WordCloud

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()
```

4. Clustering



- Clustering merupakan metode pengelompokan data yang tidak memiliki label. Clustering dibagi menjadi tiga yaitu Hard Clustering, Hierarchical Clustering, Soft/Fuzzy Clustering.
- Metode Clustering Hard Clustering merupakan pengelompokan data dimana data termasuk pada kelompok tertentu atau tidak. Beberapa metode tersebut adalah Kmeans, Kmeans++, DB Scan.
- Kmeans

```
import pickle
import os
import re
from nltk import sent_tokenize
from nltk import word_tokenize
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))

src_name = "20newsgroup.pckl"
src_path = os.path.join("data", src_name)
with open(src_path, 'rb') as fin:
    data = pickle.load(fin)

docs = [doc for doc in data.data]
label = data.target

def preprocess(doc):
    sents = word_tokenize(doc)
    sents_tok = list() # tokenisasi kalimat
    sents = [t for t in sents if t not in stop_words]
    for s in sents:
        s = s.strip().lower() # case folding dan menghilangkan new line
        s = s.replace("\n", " ") # menggantikan \n dengan spasi
        s = re.sub(r'[^a-zA-Z0-9 ]', ' ', s) # menghapus simbol
        s = re.sub(' +', ' ', s) # menghapus repetitive space
        sents_tok.append(s)
    return " ".join(sents_tok)

docs_clear = list()
for d in docs:
    docs_clear.append(preprocess(d))

print('DONE!')
```

```
# representasi vektor dengan VSM-TFIDF
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import cluster


tfidf_vectorizer = TfidfVectorizer(max_df=0.95, min_df=2)
X = tfidf_vectorizer.fit_transform(docs_clear)
print(X.shape)
k = 3
seed = 99 # Sembarang nilai untuk Random generator, mengapa penting? agar ketika dijalankan ulang nilai randomnya te
km = cluster.KMeans(n_clusters=k, init='random', max_iter=300, random_state = seed)
km.fit(X)
# Hasil clusteringnya
C_km = km.predict(X)
C_km[:10]
```

- Kmeans++

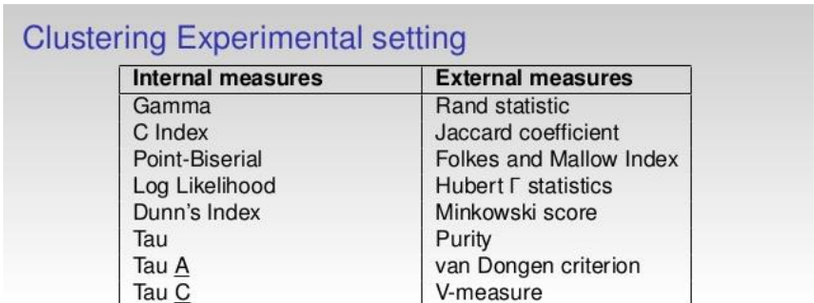
```
# k-means++ clustering
# http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
kmPP = cluster.KMeans(n_clusters=k, init='k-means++', max_iter=300, tol=0.0001,  random_state = seed)
kmPP.fit(X)
C_kmpp = kmPP.predict(X)
C_kmpp[:10]
```

- DB Scan

```
Slide Type -
# DBSCAN
# http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
import numpy as np

dbscan = cluster.DBSCAN(eps=0.5)
dbscan.fit(X)
C_db = dbscan.labels_.astype(int)
C_db[:10]
```

- Evaluasi Clustering Evaluasi clustering dapat dibagi menjadi 2 kelompok: Eksternal dan Internal. Eksternal: menghitung seberapa baik sebuah cluster dibandingkan dengan nground truth. Internal: menghitung seberapa baik cluster yang terbentuk berdasarkan intra-cluster similarity dan inter-cluster similarity.



Clustering Experimental setting

Internal measures	External measures
Gamma	Rand statistic
C Index	Jaccard coefficient
Point-Biserial	Folkes and Mallow Index
Log Likelihood	Hubert Γ statistics
Dunn's Index	Minkowski score
Tau	Purity
Tau A	van Dongen criterion
Tau C	V-measure
Somer's Gamma	Completeness
Ratio of Repetition	Homogeneity
Modified Ratio of Repetition	Variation of information
Adjusted Ratio of Clustering	Mutual information
Fagan's Index	Class-based entropy
Deviation Index	Cluster-based entropy
Z-Score Index	Precision
D Index	Recall
Silhouette coefficient	F-measure

Table: Internal and external clustering evaluation measures.

- Salah satu contoh Evaluasi Internal yaitu Silhouette coefficient yang menghitung kemiripan data point dengan cluster tempat data point tsb berada dibandingkan dengan cluster tetangga. Range nilai [-1, +1]. Semakin tinggi nilainya menunjukkan semakin baik cluster yang terbentuk.

```
from sklearn.metrics import silhouette_score as siluet
C = [C_km, C_kmpp, C_db]

for res in C:
    print(siluet(X,res), end=', ')

# NOTE: Silhouette coefficient hanya cocok untuk k-means
```

```
0.017194119526845738, 0.018754806794862373, -0.23717769755957757,
```

- Salah satu evaluasi eksternal yaitu Purity yang menghitung jumlah data point yang benar dalam suatu cluster. Untuk menggunakan evaluasi ini, membutuhkan label data dari ground truth.

```
from sklearn.metrics.cluster import homogeneity_score as purity

for res in C:
    print(purity(label,res), end=', ')
```

```
0.24079149127678492, 0.06758730134236998, 0.0015549989772357536,
```

- Evaluasi menggunakan NMI

```
# Evaluasi eksternal NMI
from sklearn.metrics import normalized_mutual_info_score as NMI

for res in C:
    print(NMI(label,res), end=', ')
```

```
0.30209398183003827, 0.10666569115117253, 0.002704356633566744,
```

LATIHAN DAN TUGAS PRAKTIKUM

1. Lakukan seluruh percobaan pada modul ini dan berikan analisis yang kalian temukan
2. Jelaskan perbedaan hasil dari Preprocessing menggunakan NLTK, TextBlob dan Sastrawi dan berikan contohnya.
3. Crawling dataset dengan total 10 pada berbagai portal berita Nasional dengan kategori bebas namun wajib sama.
4. Lakukan preprocessing yang sudah diajarkan pada modul ini (menggunakan salah satu library saja).
5. Buatlah wordcloud dan most common word barplot, interpretasikan hasilnya
6. Lakukan clustering dengan menggunakan fitur TF-IDF
7. Buat visualisasi clusternya dan lakukan interpretasi terhadap hasil tersebut
8. Gunakan validasi menggunakan salah satu Davies-Bouldin index atau Silhouette score
9. Dokumentasikan setiap langkah diatas dan tulis laporannya
10. Kumpulkan dalam format pdf