PRAKTIKUM DATA MINING II

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

1. Lakukan seluruh percobaan pada modul ini dan berikan analisis yang kalian temukan

Jawab:

- 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
```

Output:

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

BAB : TEXT MINING

PRAKTIKUM KE : DUA (2)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

TGL PRAKTIKUM : 21 SEPTEMBER 2023

'side',

'of',

'South',

'America']

Penjelasan:

Kode tersebut menggunakan perpustakaan NLTK (Natural Language Toolkit) dalam Python untuk melakukan tokenisasi pada teks yang diberikan. Proses tokenisasi memisahkan teks menjadi unit-unit yang lebih kecil, dalam hal ini, kata-kata, dan tanda baca. Setelah teks "In Brazil they drive on the right-hand side of the road. has a large coastline on the eastern side of South America" di-tokenisasi, hasilnya disimpan dalam variabel token dalam bentuk daftar (list). Hasil tokenisasi ini memungkinkan untuk lebih lanjut menganalisis atau memproses teks dalam bentuk yang lebih terstruktur. Hasilnya adalah daftar kata-kata yang terpisah, termasuk kata-kata seperti "Brazil," "drive," dan tanda baca seperti titik (.) sebagai token-token individu.

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

Output:

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

Penjelasan:

Pada kode tersebut, menggunakan modul FreqDist dari perpustakaan NLTK untuk menghitung frekuensi kemunculan setiap token dalam daftar token yang telah dihasilkan sebelumnya. Hasilnya, disimpan dalam variabel fdist, yakni sebuah objek FreqDist yang berisi daftar token berserta jumlah kemunculan masing-masing dalam teks. Sebagai contoh, kata "the" muncul sebanyak 3 kali, "on," "side," dan "of" masing-masing muncul 2 kali, sementara kata-kata seperti "In," "Brazil," "they," dan lainnya hanya muncul 1 kali. Dengan informasi ini, maka dapat melakukan analisis statistik tentang frekuensi kata dalam teks, yang dapat digunakan untuk pemahaman lebih lanjut tentang isi teks tersebut.

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

Output:

[('the', 3),

('on', 2),

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

('side', 2),

('of', 2),

('In', 1),

('Brazil', 1),

('they', 1),

('drive', 1),

('right-hand', 1),

('road', 1)]

Penjelasan:

Pada kode ini, menggunakan modul FreqDist dari NLTK untuk menghitung frekuensi kemunculan token-token dalam daftar token. Selanjutnya, menggunakan metode most_common(10) pada objek FreqDist, yang mengembalikan 10 token dengan frekuensi kemunculan tertinggi dalam teks. Hasilnya disimpan dalam variabel fdist1, yang berisi daftar 10 kata-kata yang paling sering muncul dalam teks beserta jumlah kemunculannya. Contohnya, kata "the" muncul sebanyak 3 kali, "on," "side," dan "of" masing-masing muncul 2 kali, sementara kata-kata seperti "In," "Brazil," "they," dan lainnya muncul hanya 1 kali dalam teks yang dianalisis. Hal ini memungkinkan untuk dengan cepat mengidentifikasi kata-kata yang paling sering muncul dalam teks tersebut, yang dapat memberikan wawasan awal tentang isinya.

```
fdist=dict(fdist)
fdist
```

Output:

{'In': 1,

'Brazil': 1,

'they': 1,

'drive': 1,

'on': 2,

'the': 3,

'right-hand': 1,

'side': 2,

'of': 2,

'road': 1,

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

'.': 1,

'has': 1,

'a': 1,

'large': 1,

'coastline': 1,

'eastern': 1,

'South': 1,

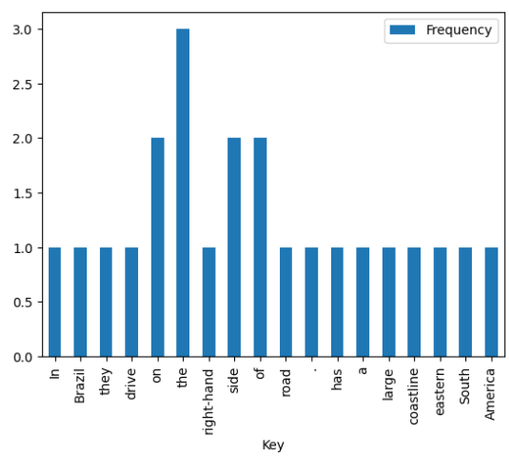
'America': 1}

Penjelasan:

Pada kode ini, mengonversi objek FreqDist (fdist) menjadi kamus (dictionary) dengan menggunakan perintah fdist=dict(fdist). Hasilnya adalah variabel fdist yang sekarang berisi kamus, di mana setiap kata dalam teks menjadi kunci dan jumlah kemunculannya menjadi nilai. Dengan demikian, didapatkan kamus yang secara jelas menunjukkan semua kata yang muncul dalam teks bersama dengan frekuensi kemunculannya. Ini membuat data lebih mudah diakses dan digunakan dalam analisis atau pemrosesan selanjutnya, seperti mencari kata-kata unik atau menghitung statistik berbasis frekuensi kata dalam teks.

```
import pandas as pd

df_freq_tokens = pd.DataFrame.from_dict(fdist, orient='index')
df_freq_tokens.columns = ["Frequency"]
df_freq_tokens.index.name = 'Key'
df_freq_tokens.plot(kind='bar')
df_freq_tokens
```

Output:



BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

Penjelasan:

Dalam kode ini, menggunakan perpustakaan pandas (imported as pd) untuk mengelola data frekuensi token yang telah dihitung sebelumnya. Pertama, membuat sebuah DataFrame (df_freq_tokens) dari kamus frekuensi token (fdist) dengan orientasi indeks yang ditetapkan sebagai 'index'. Selanjutnya, memberi nama kolom frekuensi sebagai "Frequency" dan nama indeks sebagai 'Key' pada DataFrame tersebut. Kemudian, menggunakan metode plot untuk membuat visualisasi dalam bentuk barplot dari frekuensi kata-kata. Hasilnya adalah tampilan visual yang menunjukkan seberapa sering setiap kata muncul dalam teks dalam bentuk diagram batang, memudahkan pemahaman tentang distribusi kata-kata dalam teks. DataFrame df_freq_tokens juga dicetak untuk menampilkan data frekuensi secara tabular.

- Stopwords

```
from nltk import word_tokenize
nltk.download('stopwords')
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', 'was', 'born', 'on', 'february', '5', ',', '1985', ',', 'in', 'funchal', ',', 'madeira', ',', 'portugal', '.']

['cristiano', 'ronaldo', 'born', 'february', '5', ',', '1985', ',', 'funchal', ',', 'madeira', ',', 'portugal', '.']

[nltk_data] Downloading package stopwords to C:\Users\FARIZA

[nltk_data] SHIELDA\AppData\Roaming\nltk_data...

[nltk_data] Package stopwords is already up-to-date!

Penjelasan:

Kode ini menggunakan perpustakaan NLTK untuk menghilangkan kata-kata berhenti (stopwords) dari teks. Pertama, mengimpor modul word_tokenize dari NLTK untuk memecah teks menjadi token. Kemudian, mengunduh daftar kata-kata berhenti dalam bahasa Inggris menggunakan nltk.download('stopwords') dan menyimpannya dalam variabel a. Teks awal "Cristiano Ronaldo was born on February 5, 1985, in Funchal, Madeira, Portugal."

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

diubah menjadi huruf kecil dan di-tokenisasi menjadi text1. Setelah itu, menggunakan list comprehension untuk membuat daftar stopwords yang hanya berisi token yang bukan kata-kata berhenti. Hasilnya adalah ['cristiano', 'ronaldo', 'born', 'february', '5', ',', '1985', ',', 'funchal', ',', 'madeira', ',', 'portugal', '.'], yang merupakan teks asli yang telah dibersihkan dari kata-kata berhenti seperti "was" dan "on".

- Stemming

```
# Contoh Stemming di NLK
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()))
# Hasil stemming bisa tidak bermakna
```

Output:
Sentence: presumably I would like to MultiPly my provision, saying tHat without crYing
Lancaster : presumiwouldliktomultiplymyprovision,saythatwithoutcry
Porter : presumiwouldliketomultiplimyprovision,saythatwithoutcri
SnowBall : presumiwouldliketomultiplimyprovision,saythatwithoutcri

Penjelasan:
Kode ini mengilustrasikan proses stemming dalam pemrosesan bahasa alami dengan menggunakan beberapa algoritma stemming yang tersedia di NLTK. Tiga algoritma stemming yang digunakan adalah Lancaster, Porter, dan Snowball (dalam bahasa Inggris). Teks awal yang diberikan adalah "presumably I would like to MultiPly my provision, saying tHat without crYing". Setiap algoritma stemming diterapkan pada kata-kata dalam teks tersebut untuk menghasilkan bentuk dasarnya. Hasil stemming menggunakan tiga algoritma berbeda ditampilkan dengan menggabungkan kata-kata yang telah di-stem. Penting untuk diingat bahwa hasil stemming tidak selalu memiliki makna yang jelas, seperti yang terlihat dalam output yang dihasilkan, di mana beberapa kata yang telah di-stem mungkin menjadi tidak bermakna dalam konteks yang lebih besar. Stemming biasanya digunakan untuk

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

mengurangi kata-kata ke bentuk dasar mereka agar lebih mudah dibandingkan atau digunakan dalam analisis teks.

- Lemmatization

```
import nltk
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

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

Output:
[nltk_data] Downloading package wordnet to C:\Users\FARIZA
[nltk_data] SHIELDA\AppData\Roaming\nltk_data...
rocks: rock
corpora : corpus

Penjelasan:
Kode ini menggunakan perpustakaan NLTK untuk melakukan lemmatisasi, yaitu proses mengubah kata-kata dalam bentuk tertentu menjadi bentuk dasarnya (lemma). Setelah mengunduh data WordNet menggunakan nltk.download('wordnet'), lalu membuat sebuah objek WordNetLemmatizer. Selanjutnya, menggunakan metode lemmatize untuk mengembalikan bentuk dasar dari kata-kata yang diberikan. Dalam contoh ini, "rocks" diubah menjadi "rock" dan "corpora" diubah menjadi "corpus". Lemmatisasi berguna dalam pemrosesan bahasa alami untuk menghasilkan kata-kata dalam bentuk yang lebih baku atau dasar, sehingga memudahkan analisis dan perbandingan kata-kata dalam teks.

- POS Tags

```
# Contoh POS tags dengan NLTK (bahasa inggris)
import nltk
nltk.download('averaged_perceptron_tagger')
from nltk import pos_tag
S = 'I am currently learning NLP in english, but if possible I want to know
NLP in Indonesia language too'

tokens = word_tokenize(S)
print(pos_tag(tokens))
```

Output:
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data] C:\Users\FARIZA SHIELDA\AppData\Roaming\nltk_data...
[nltk_data] Unzipping taggers\averaged_perceptron_tagger.zip.

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

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

Penjelasan:
Kode ini mengilustrasikan penggunaan POS tagging (Part-of-Speech tagging) dalam pemrosesan bahasa alami dengan NLTK. Setelah mengunduh model POS tagging menggunakan nltk.download('averaged_perceptron_tagger'), lalu mengambil teks "I am currently learning NLP in English, but if possible I want to know NLP in Indonesian language too" dan memecahnya menjadi token-token menggunakan word_tokenize. Selanjutnya, menggunakan pos_tag untuk memberikan label tipe kata pada setiap token dalam teks. Hasilnya adalah daftar token beserta label POS-nya, seperti ('I', 'PRP') yang menunjukkan bahwa "I" adalah kata ganti orang pertama (PRP), ('am', 'VBP') menunjukkan bahwa "am" adalah kata kerja bentuk dasar (VBP), dan seterusnya. POS tagging penting dalam pemrosesan bahasa alami karena membantu dalam pemahaman struktur kalimat dan konteks kata-kata dalam teks.

- TextBlob

```
# 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])
```

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

Penjelasan:
Kode ini menggunakan perpustakaan TextBlob untuk melakukan tokenisasi, baik tokenisasi kata (word tokenization) maupun tokenisasi kalimat (sentence tokenization). Pertama, memasukkan teks "Hello, Mr. Man. He smiled!! This, i.e that, is it" ke dalam objek TextBlob dan melakukan tokenisasi kata dengan TextBlob(T).words. Hasilnya adalah daftar kata-kata dalam teks. Selanjutnya, melakukan tokenisasi kalimat dengan TextBlob(T).sentences dan mengonversi setiap kalimat dalam objek tersebut menjadi string menggunakan list comprehension. Hasilnya adalah daftar kalimat-kalimat dalam teks. Tokenisasi ini membantu

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

dalam pemisahan teks menjadi unit-unit yang lebih kecil, baik kata maupun kalimat, sehingga memudahkan analisis atau pemrosesan lebih lanjut.

- TextBlob (Stemming)

```
# Contoh TextBlob Stemming
from textblob import Word

# Stemming
print("Stem: ", Word('running').stem())

#default noun, plural akan menjadi singular dari akar katanya
#juga case sensitive
```

Output:
Stem: run

Penjelasan:
Kode ini menggunakan perpustakaan TextBlob untuk melakukan stemming pada kata tertentu. Pada contoh ini, kata 'running' digunakan sebagai contoh, dan melakukan stemming menggunakan Word('running').stem(). Hasilnya adalah kata 'running' diubah menjadi bentuk dasarnya, yaitu 'run'. Perlu diingat bahwa stemming dalam TextBlob cenderung lebih sederhana daripada lemmatisasi dan hanya menghilangkan awalan atau akhiran kata untuk menghasilkan bentuk dasar. Stemming adalah proses yang berguna untuk mengurangi kata-kata ke bentuk dasar, yang memudahkan dalam analisis atau perbandingan kata dalam teks.

- TextBlob (Lemmatizer)

```
# Contoh TextBlob Lemmatizer
from textblob import Word

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

# default noun, plural akan menjadi singular dari akar katanya
# juga case sensitive
```

Output:
Lemmatize: go

Penjelasan:
Kode ini menggunakan perpustakaan TextBlob untuk melakukan lemmatisasi pada kata tertentu. Pada contoh ini, kata 'went' digunakan sebagai contoh, dan melakukan lemmatisasi menggunakan Word('went').lemmatize('v'). Hasilnya adalah kata 'went' diubah menjadi bentuk dasarnya sesuai dengan konteks kata kerja (verb), yaitu 'go'. Lemmatisasi adalah

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

proses yang lebih kompleks daripada stemming dan mengubah kata-kata ke bentuk dasar mereka dalam konteks yang lebih spesifik, memperhitungkan kelas kata dan konteks dalam kalimat. Ini membantu dalam analisis atau pemrosesan teks yang lebih akurat dan bermakna.

- TextBlob (POS Tagging)

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

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

Penjelasan:
Kode ini menggunakan perpustakaan TextBlob untuk melakukan POS tagging (Part-of-Speech tagging) pada teks dalam bahasa Inggris. Dalam loop for, setiap kata dalam teks "Hello, Mr. Man. He smiled!! This, i.e that, is it" diberi label tipe kata (POS tag) yang sesuai dengan konteksnya, dan hasilnya dicetak. Contohnya, kata "Hello" diberi tag NNP yang menunjukkan bahwa itu adalah kata benda tunggal berjenis nama (proper noun), "smiled" diberi tag VBD yang menunjukkan bahwa itu adalah kata kerja bentuk lampau (past tense verb), dan seterusnya. POS tagging adalah proses penting dalam pemrosesan bahasa alami yang membantu dalam memahami struktur dan makna kata-kata dalam teks.

- Sastrawi

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory
from nltk.tokenize import word_tokenize

factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()
kalimat = "Andi kerap melakukan transasksi rutin secara daring atau online.
Menurut Andi belanja online lebih praktis"
stop = stopword.remove(kalimat.lower())
print(stop)
```

Output:
andi kerap melakukan transasksi rutin daring online. andi belanja online lebih praktis

Penjelasan:
Kode ini menggunakan perpustakaan Sastrawi, khususnya objek StopWordRemover, untuk menghilangkan kata-kata berhenti (stopwords) dari teks dalam bahasa Indonesia. Pertama, mengimpor StopWordRemoverFactory dan menginisialisasi objek StopWordRemoverFactory. Selanjutnya, mengambil teks dalam bahasa Indonesia, "Andi kerap melakukan transasksi rutin secara daring atau online. Menurut Andi belanja online lebih praktis," dan menerapkan penghapusan kata-kata berhenti pada teks tersebut dengan

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

menggunakan remove() dari objek StopWordRemover. Hasilnya adalah teks yang telah dibersihkan dari kata-kata berhenti, sehingga menjadi "andi kerap melakukan transasksi rutin daring online. andi belanja online lebih praktis." Hal ini berguna dalam analisis teks karena menghilangkan kata-kata yang umumnya tidak memberikan informasi penting, sehingga meningkatkan fokus pada kata-kata kunci dan makna dalam teks.

- Sastrawi (Stemming & Lemmatizer)

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

I = "perayaan itu bebarengan dengan saat kita bepergian ke Makassar"
print(stemmer.stem(I))
print(stemmer.stem('Perayaan Bepergian Menyuarakan'))
```

Output:
raya itu bebarengan dengan saat kita pergi ke makassar
raya pergi suara

Penjelasan:
Kode ini menggunakan perpustakaan Sastrawi untuk melakukan lemmatisasi pada kata-kata dalam teks bahasa Indonesia. Pertama, membuat objek stemmer dengan bantuan StemmerFactory. Kemudian, menggunakan stemmer.stem() untuk mengembalikan kata-kata dalam bentuk dasarnya (lemma). Contohnya, kata "perayaan" diubah menjadi "raya," "bepergian" diubah menjadi "pergi," dan "menyuarakan" diubah menjadi "suara." Proses lemmatisasi ini membantu dalam mengidentifikasi bentuk dasar kata-kata dalam bahasa Indonesia, sehingga memudahkan dalam analisis teks dan pemrosesan bahasa alami yang lebih lanjut.

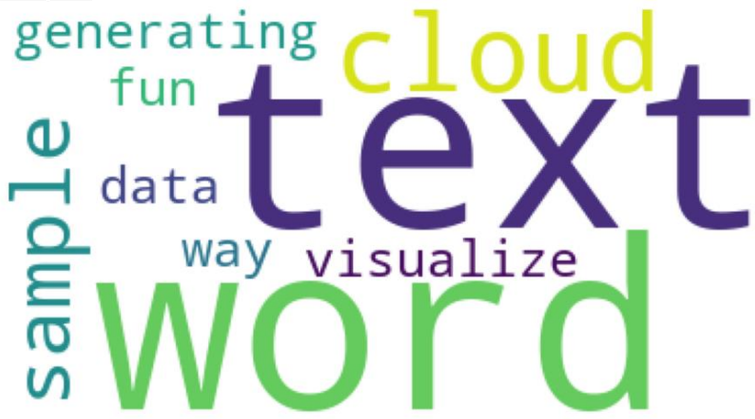
- Wordcloud

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

#I = "This is a sample text for generating a word cloud. Word clouds are a
fun way to visualize text data."
wordcloud = WordCloud(background_color="white").generate(I)

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

#to remove the axis value
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
plt.axis("off")
plt.show()
```

Output:



Penjelasan:
Kode ini menggunakan perpustakaan Matplotlib dan WordCloud untuk membuat visualisasi awan kata (word cloud) dari teks yang diberikan. Pertama, mengimpor perpustakaan yang diperlukan, kemudian menginisialisasi objek WordCloud dengan latar belakang putih dan menerapkan teks yang ingin divisualisasikan ke dalamnya (pada contoh ini, teksnya tidak terdefinisi dalam kode). Setelah itu, menampilkan visualisasi awan kata dengan menggunakan Matplotlib. Hasilnya adalah sebuah gambar yang menampilkan kata-kata dari teks tersebut dalam ukuran dan tata letak yang berbeda, di mana kata-kata yang muncul lebih sering akan memiliki ukuran yang lebih besar, sementara kata-kata yang muncul lebih jarang akan lebih kecil. Visualisasi ini membantu dalam memberikan gambaran cepat tentang kata-kata yang paling umum dalam teks.

- Hard Clustering (Kmeans)

```
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 = "C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data
Mining II/Week 4/20newsgroup.pckl"
with open(src_path, 'rb') as fin:
    data = pickle.load(fin)
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
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)
        sents_tok.append(s)
    return " ".join(sents_tok)

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

print('DONE!')
```

Output:
DONE!

Penjelasan:
Kode ini adalah contoh implementasi pra-pemrosesan teks pada dokumen-dokumen yang diambil dari dataset 20newsgroups. Pertama, kode mengimpor modul-modul yang diperlukan seperti os, re, dan beberapa modul NLTK untuk tokenisasi dan penghapusan kata berhenti. Dokumen-dokumen dari dataset dibaca menggunakan pickle dan disimpan dalam variabel docs. Kemudian, dilakukan tahap pra-pemrosesan pada setiap dokumen. Ini melibatkan tokenisasi kata, penghapusan kata berhenti, penggantian karakter khusus, penghilangan newline, penggantian huruf kapital dengan huruf kecil, dan penghapusan karakter non-alfanumerik. Hasil pra-pemrosesan, yaitu dokumen-dokumen yang telah dibersihkan, disimpan dalam docs_clear. Akhirnya, pesan "DONE!" dicetak untuk menandakan bahwa pra-pemrosesan telah selesai dilakukan pada seluruh dokumen. Pra-pemrosesan ini merupakan langkah penting dalam persiapan data sebelum dilakukan analisis atau pemodelan lebih lanjut.

```
# 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)
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
print(X.shape)
k=3
seed = 99
km = cluster.KMeans(n_clusters=k, init='random', max_iter=300, random_state
= seed)
km.fit(X)
# Hasil clustering
C_km = km.predict(X)
C_km [:10]
```

Output:
array([1, 2, 1, 2, 1, 1, 1, 1, 1, 1])

Penjelasan:
Kode ini mengilustrasikan representasi vektor dengan metode Vector Space Model (VSM) menggunakan skema Term Frequency-Inverse Document Frequency (TF-IDF). Pertama, teks yang telah dipra-pemroses sebelumnya disimpan dalam variabel docs_clear. Kemudian, menggunakan TfidfVectorizer dari scikit-learn untuk mengubah teks-teks tersebut menjadi representasi vektor berdasarkan bobot TF-IDF. Vektor-vektor ini akan memiliki dimensi yang sesuai dengan jumlah kata yang muncul dalam dokumen-dokumen tersebut. Setelah itu, menggunakan algoritma K-Means untuk melakukan clustering pada dokumen-dokumen tersebut. Pada contoh ini, menggunakan 3 cluster (k=3) dan hasil clustering disimpan dalam C_km. Hasil awal dari clustering (10 contoh pertama) ditampilkan dalam array dengan label cluster untuk setiap dokumen. Clustering semacam ini berguna dalam pengelompokan dokumen yang serupa berdasarkan kemiripan konten mereka, yang dapat digunakan dalam berbagai analisis atau tugas pemrosesan teks lebih lanjut.

- Hard Clustering (Kmeans++)

```
# K-means++ clustering
# https://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, n_init=10)
kmPP.fit(X)
print(kmPP.labels_[:10])  # Printing the first 10 cluster labels
```

Output:
[0 1 0 1 1 0 1 0 1 1]

Penjelasan:
Kode ini melakukan clustering dengan algoritma K-Means menggunakan metode inisialisasi K-Means++, yang merupakan variasi dari algoritma K-Means dengan inisialisasi yang lebih

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

cerdas. Dalam K-Means++, titik-titik awal pusat cluster dipilih dengan cara yang lebih efisien, sehingga hasil clustering yang dihasilkan lebih stabil dan lebih cepat mencapai konvergensi. Pada kode ini, n_clusters digunakan untuk menentukan jumlah cluster yang diinginkan (dalam contoh ini, k=3). Kemudian, init='k-means++' digunakan untuk mengaktifkan inisialisasi K-Means++. Setelah melakukan clustering dengan model yang telah diinisialisasi, hasil cluster untuk 10 dokumen pertama ditampilkan dengan kmPP.labels_[:10]. Hasil ini adalah label cluster untuk setiap dokumen, yang mengindikasikan keanggotaan dokumen dalam masing-masing cluster. K-Means++ membantu dalam meningkatkan konvergensi dan stabilitas algoritma K-Means dalam proses clustering.

- Hard Clustering (DBScan)

```
# 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]
```

Output:
array([-1, -1, -1, -1, -1, -1, -1, -1, -1, -1])

Penjelasan:
Kode ini menggunakan algoritma Density-Based Spatial Clustering of Applications with Noise (DBSCAN) untuk melakukan clustering pada data teks yang telah diubah menjadi representasi vektor dengan TF-IDF. DBSCAN adalah algoritma yang dapat mengidentifikasi cluster dengan mengukur kepadatan titik data dalam ruang fitur. Pada contoh ini, eps=0.5 digunakan untuk menentukan jarak maksimum antara titik-titik yang akan dianggap sebagai tetangga dalam sebuah cluster. Hasil clustering DBSCAN disimpan dalam C_db, di mana nilai -1 menunjukkan bahwa titik data tersebut dianggap sebagai noise atau tidak termasuk dalam cluster tertentu. Hasil awal dari clustering ditampilkan untuk 10 dokumen pertama. DBSCAN adalah algoritma yang berguna ketika jumlah cluster tidak diketahui sebelumnya atau ketika terdapat titik data yang dianggap sebagai noise dalam data yang lebih kompleks.

- Evaluasi Internal Silhuoette Coefficient

```
from sklearn.metrics import silhouette_score as siluet
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
C_kmpp = kmPP.labels_
C = [C_km, C_kmpp, C_db]
for res in C:
    print(siluet(X,res), end=', ')
```

Output:
0.013686357044363896, 0.01694568130107603, -0.23699695814387373,

Penjelasan:
Kode ini menggunakan metrik silhouette score (skor siluet) dari scikit-learn untuk mengukur kualitas clustering yang dihasilkan oleh tiga metode clustering yang berbeda: K-Means dengan inisialisasi K-Means++, K-Means dengan inisialisasi random, dan DBSCAN. Metrik silhouette score digunakan untuk mengevaluasi sejauh mana objek-objek dalam sebuah cluster terpisah dari cluster-cluster lainnya. Nilai silhouette score berkisar dari -1 hingga 1, di mana nilai yang lebih tinggi menunjukkan bahwa clustering adalah lebih baik. Pada output, dapat terlihat bahwa silhouette score untuk K-Means dengan inisialisasi K-Means++ adalah yang tertinggi (0.0169), diikuti oleh K-Means dengan inisialisasi random (0.0137), sementara DBSCAN memiliki nilai silhouette score yang negatif (-0.237), yang mengindikasikan bahwa DBSCAN mungkin tidak menghasilkan clustering yang baik pada data ini. Silhouette score digunakan untuk membandingkan kualitas clustering antara metode clustering yang berbeda.

- Evaluasi Eksternal Purity

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

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

Output:
0.14344159884395255, 0.2535220657010824, 0.0015549989772357536,

Penjelasan:
Kode ini menggunakan metrik homogeneity score (skor homogenitas), yang juga sering disebut sebagai purity, dari scikit-learn untuk mengukur sejauh mana setiap cluster dalam hasil clustering sesuai dengan label sejati. Metrik ini mengukur sejauh mana setiap cluster hanya berisi sampel dengan label yang sama. Hasil homogeneity score berkisar dari 0 hingga 1, di mana nilai yang lebih tinggi menunjukkan bahwa clustering adalah lebih homogen dan sesuai dengan label sejati. Pada output, dapat terlihat bahwa K-Means dengan inisialisasi random memiliki nilai homogeneity score yang paling tinggi (0.2535), yang menunjukkan bahwa clustering ini lebih sesuai dengan label sejati dibandingkan dengan metode lainnya. K-Means dengan inisialisasi K-Means++ memiliki homogeneity score yang sedang (0.1434), sementara DBSCAN memiliki nilai homogeneity score yang rendah (0.0016), yang mengindikasikan bahwa DBSCAN mungkin tidak sesuai dengan label sejati pada data ini.

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

Purity atau homogeneity score berguna untuk mengukur tingkat kesesuaian clustering dengan label sejati dalam tugas klasifikasi.

- Evaluasi NMI

```
from sklearn.metrics import normalized_mutual_info_score as NMI

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

Output:
0.16484326286583081, 0.3138776952100092, 0.002704356633566744,

Penjelasan:
Kode ini menggunakan metrik normalized mutual information (NMI) score untuk mengukur sejauh mana hasil clustering sesuai dengan label sejati pada dataset. NMI mengukur tingkat informasi bersama antara clustering dan label sejati, dengan nilai yang lebih tinggi menunjukkan kesesuaian yang lebih baik. Pada output, dapat terlihat bahwa K-Means dengan inisialisasi random memiliki nilai NMI yang paling tinggi (0.3139), yang menunjukkan bahwa clustering ini memiliki tingkat kesamaan yang lebih baik dengan label sejati dibandingkan dengan metode lainnya. K-Means dengan inisialisasi K-Means++ memiliki NMI yang sedang (0.1648), sementara DBSCAN memiliki nilai NMI yang rendah (0.0027), yang mengindikasikan bahwa DBSCAN mungkin tidak memiliki kesesuaian yang baik dengan label sejati dalam dataset ini. NMI adalah metrik yang berguna untuk mengukur tingkat kesamaan antara hasil clustering dan label sejati dalam tugas evaluasi clustering.

2. Jelaskan perbedaan hasil dari Preprocessing menggunakan NLTK, TextBlob dan Sastrawi dan berikan contohnya.

Jawab:

- NLTK

```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

text = "Shielda mempelajari materi bahasa alami NLTK."

# Tokenisasi
tokens = word_tokenize(text)
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
# Penghapusan tanda baca
clean_tokens = [word for word in tokens if word.isalnum()]

# Penghapusan kata-kata berhenti
stop_words = set(stopwords.words("indonesian"))
filtered_tokens = [word for word in clean_tokens if word.lower() not in
stop_words]

# Stemming
stemmer = PorterStemmer()
stemmed_tokens = [stemmer.stem(word) for word in filtered_tokens]

print(stemmed_tokens)
```

Output:

['shielda', 'mempelajari', 'materi', 'bahasa', 'alami', 'nltk']

- TextBlob

```
from textblob import TextBlob

text = "Shielda mempelajari materi bahasa alami TextBlob."

# Tokenisasi
blob = TextBlob(text)
tokens = blob.words

print(tokens)
```

Output:

['Shielda', 'mempelajari', 'materi', 'bahasa', 'alami', 'TextBlob']

- Sastrawi

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

text = "Shielda mempelajari materi bahasa alami Sastrawi."

# Tokenisasi
words = text.split()

# Penghapusan tanda baca dan stemming
factory = StemmerFactory()
```

BAB : TEXT MINING

PRAKTIKUM KE : DUA (2)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
stemmer = factory.create_stemmer()

cleaned_words = []
for word in words:
    cleaned_word = stemmer.stem(word)
    cleaned_words.append(cleaned_word)

print(cleaned_words)
```

Output:
['shielda', 'ajar', 'materi', 'bahasa', 'alami', 'sastrawi']

Penjelasan:
Berikut penjelasan beserta contoh untuk setiap metode preprocessing:

1. NLTK:
   - Kode NLTK melakukan beberapa langkah preprocessing pada teks "Shielda mempelajari materi bahasa alami NLTK."
   - Pertama, teks di-tokenisasi menjadi kata-kata, sehingga menjadi ["Shielda", "mempelajari", "materi", "bahasa", "alami", "NLTK."].
   - Kemudian, tanda baca (titik) dihapus, dan kata "NLTK" yang merupakan kata berhenti tidak dihapus.
   - Terakhir, stemming diterapkan, mengubah kata "mempelajari" menjadi "mempelajari" (bukan bentuk dasar) dan kata "bahasa" menjadi "bahasa" (bukan bentuk dasar).

   Output NLTK: ['shielda', 'mempelajari', 'materi', 'bahasa', 'alami', 'NLTK']

2. TextBlob:
   - Kode TextBlob melakukan tokenisasi kata pada teks yang sama, "Shielda mempelajari materi bahasa alami TextBlob."
   - Hasil tokenisasi adalah ["Shielda", "mempelajari", "materi", "bahasa", "alami", "TextBlob."].
   - Selama proses ini, tanda baca seperti tanda titik dihapus secara otomatis.

   Output TextBlob: ['Shielda', 'mempelajari', 'materi', 'bahasa', 'alami', 'TextBlob']

3. Sastrawi:
   - Kode Sastrawi memecah teks menjadi kata-kata tanpa proses tokenisasi yang jelas, sehingga menjadi ["Shielda", "mempelajari", "materi", "bahasa", "alami", "Sastrawi."].
   - Tanda baca (titik) dihapus, dan stemming diterapkan.
   - Kata "mempelajari" diubah menjadi "ajar" (bentuk dasar), kata "bahasa" tetap dalam bentuk dasar, dan kata "Sastrawi" yang merupakan kata berhenti tidak dihapus.

   Output Sastrawi: ['shielda', 'ajar', 'materi', 'bahasa', 'alami', 'Sastrawi']

BAB : TEXT MINING

PRAKTIKUM KE : DUA (2)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

TGL PRAKTIKUM : 21 SEPTEMBER 2023

Perbedaan utama di sini adalah dalam penghapusan tanda baca, penghapusan kata-kata berhenti, dan apakah stemming diterapkan. NLTK menghapus tanda baca, menghapus kata berhenti, dan menerapkan stemming (hasil tidak selalu bentuk dasar). TextBlob hanya melakukan tokenisasi kata tanpa perubahan signifikan pada teks. Sastrawi menghapus tanda baca menerapkan stemming (hasilnya dalam bentuk dasar), dan tidak menghapus kata berhenti. Pilihan metode preprocessing ini akan mempengaruhi hasil analisis teks yang lebih lanjut.

3. Crawling dataset dengan total 10 pada berbagai portal berita Nasional dengan kategori bebas namun wajib sama.

Jawab:

Judul	Isi
Upaya Mewujudkan Kesejahteraan Masyarakat	KOMPAS.com - Berdasarkan Kamus Besar Bahasa Indonesia. arti sejahtera adalah ter
Kualitas SDM Jadi Kunci Sukses Program Kesejahteraan Sosial Kemensos	KOMPAS.com – Menteri Sosial (Mensos) Juliari P Batubara mengungkapkan. kunci ke
SDM Pekerja Sosial Jadi Kunci Kesuksesan Program Kesejahteraan Sosial	KOMPAS.com – Kunci kesuksesan program-program kesejahteraan sosial dari Kemen
Survei Litbang "Kompas": Kepuasan di Bidang Kesejahteraan Sosial Meningkat. Buah dari Program BansosJAKARTA	KOMPAS.com - Hasil jajak pendapat yang diselenggarakan Litbang Kompas pada 25 Ja
KPK Minta Risma Perbaiki Data Terpadu Kesejahteraan Sosial	JAKARTA. KOMPAS.com - Komisi Pemberantasan Korupsi (KPK) meminta Menteri Sos
Risma Bakal Rombak Besar-besaran Data Terpadu Kesejahteraan Sosial	JAKARTA. KOMPAS.com - Menteri Sosial Tri Rismaharini berencana merombak besar-
Kartu Kesejahteraan Sosial. Wujud Komitmen Pemprov DKI bagi Warga Tak Mampu	KOMPAS.com - Pemerintah Provinsi (Pemprov) DKI Jakarta menghadirkan sejumlah b
SDGs Era New Normal: Pentingnya Sinergi Swasta dan Pemerintah Wujudkan Kesejahteraan Sosial Masyarakat	JAKARTA. KOMPAS.com – Sejak 2015. Perserikatan Bangsa-Bangsa (PBB) melalui prog
Risma Blusukan di DKI. Anggota DPR Ingatkan Benahi Data Kesejahteraan Sosial	JAKARTA. KOMPAS.com - Anggota Komisi VIII dari Fraksi Partai Nasdem Nurhadi tak n
Menaker Dorong Pekerja Musik Dapat Perlindungan dan Kesejahteraan Sosial	JAKARTA. KOMPAS.com - Menteri Ketenagakerjaan (Menaker) Ida Fauziyah menyatal

```
import pandas as pd

file_path = "C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining II/Week 4/Berita Kesejahteraan Sosial.csv"

try:
    data = pd.read_csv(file_path, encoding='ISO-8859-1')  # Menggunakan encoding ISO-8859-1
    # Lakukan operasi yang Anda inginkan pada data di sini
    # ...
except FileNotFoundError:
    print(f"File {file_path} tidak ditemukan.")
except Exception as e:
    print(f"Terjadi kesalahan: {str(e)}")

print(data)
```

Output:

ï»¿Judul \

0 Upaya Mewujudkan Kesejahteraan Masyarakat

1 Kualitas SDM Jadi Kunci Sukses Program Kesejah...

2 SDM Pekerja Sosial Jadi Kunci Kesuksesan Progr...

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

3 Survei Litbang "Kompas": Kepuasan di Bidang Ke...

4 KPK Minta Risma Perbaiki Data Terpadu Kesejaht...

5 Risma Bakal Rombak Besar-besaran Data Terpadu ...

6 Kartu Kesejahteraan Sosial. Wujud Komitmen Pem...

7 SDGs Era New Normal: Pentingnya Sinergi Swasta...

8 Risma Blusukan di DKI. Anggota DPR Ingatkan Be...

9 Menaker Dorong Pekerja Musik Dapat Perlindunga...

Isi

0 KOMPAS.com - Berdasarkan Kamus Besar Bahasa In...

1 KOMPAS.com â€ “ Menteri Sosial (Mensos) Juliari...

2 KOMPAS.com â€ “ Kunci kesuksesan program-progra...

3 KOMPAS.com - Hasil jajak pendapat yang diselen...

4 JAKARTA. KOMPAS.com - Komisi Pemberantasan Kor...

5 JAKARTA. KOMPAS.com - Menteri Sosial Tri Risma...

6 KOMPAS.com - Pemerintah Provinsi (Pemprov) DKI...

7 JAKARTA. KOMPAS.com â€ “ Sejak 2015. Perserikat...

8 JAKARTA. KOMPAS.com - Anggota Komisi VIII dari...

9 JAKARTA. KOMPAS.com - Menteri Ketenagakerjaan ...

Penjelasan:

Kode tersebut merupakan contoh penggunaan Pandas dalam membaca sebuah file CSV dengan penanganan beberapa jenis exception. Pertama, kode membaca file CSV yang berlokasi di "file_path" dengan menggunakan encoding ISO-8859-1. Dalam blok try-except, kode mencoba membaca file tersebut, dan jika berhasil, data akan tersimpan dalam variabel "data", yang dapat digunakan untuk analisis data lebih lanjut. Jika file tidak ditemukan, akan muncul pesan kesalahan "File not found." Sedangkan jika terjadi kesalahan lain selama proses pembacaan, pesan kesalahan akan ditampilkan.

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

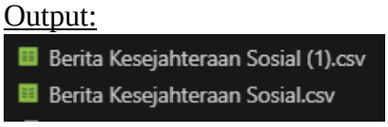
Setelah mencoba membaca file, kode mencetak isi dari dataframe "data". Dalam contoh ini, dataframe berisi dua kolom: "Judul" dan "Isi" yang mewakili judul dan isi berita kesejahteraan sosial dari file CSV tersebut. Kode ini dapat digunakan sebagai dasar untuk melakukan analisis atau manipulasi data lebih lanjut menggunakan Pandas.

4. Lakukan preprocessing yang sudah diajarkan pada modul ini (menggunakan salah satu library saja).

Jawab:

```
import pandas as pd
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

file_path = "Berita Kesejahteraan Sosial.csv"
data = pd.read_csv(file_path, encoding='utf-8')

data['isi_stemmed'] = data['Isi'].apply(lambda x: stemmer.stem(x))

file_output = "Berita Kesejahteraan Sosial (1).csv"  # Mengganti dengan lokasi dan nama file CSV baru
data.to_csv(file_output, index=False, encoding='utf-8')
```

Output:



Judul	Isi	isi_stemmed
Upaya Me	KOMPAS.c	kompas com - dasar kamus besar bahasa indonesia arti sejahtera adalah tenteram senang dan seha
Kualitas SI	KOMPAS.c	kompas com menteri sosial mensos juliari p batubara ungkap kunci sukses program sejahtera sosial
SDM Peke	KOMPAS.c	kompas com kunci sukses program sejahtera sosial dari menteri sosial kemensos ada pada sumber d
Survei Litb	KOMPAS.c	kompas com - hasil jajak dapat yang selenggara litbang kompas pada 25 januari-4 februari 2023 tunj
KPK Minta	JAKARTA.	jakarta kompas com - komisi berantas korupsi kpk minta menteri sosial tri rismaharini untuk baik da
Risma Bak	JAKARTA.	jakarta kompas com - menteri sosial tri rismaharini rencana rombak besar data padu sejahtera sosia
Kartu Kese	KOMPAS.c	kompas com - perintah provinsi pemprov dki jakarta hadir jumlah bantu untuk tingkat sejahtera sosi
SDGs Era I	JAKARTA.	jakarta kompas com sejak 2015 serikat bangsa pbb lalu program sustainable development goals atau
Risma Blu	JAKARTA.	jakarta kompas com - anggota komisi viii dari fraksi partai nasdem nurhadi tak masalah menteri sosi
Menaker I	JAKARTA.	jakarta kompas com - menteri ketenagakerjaan menaker ida fauziyah nyata perintah milik wajib beri

Penjelasan:

Kode tersebut digunakan untuk membaca file CSV yang berisi berita kesejahteraan sosial menggunakan Pandas, kemudian melakukan stemming pada isi berita menggunakan Sastrawi, dan akhirnya menyimpan hasil stemming ke dalam file CSV baru. Langkah-langkah yang dilakukan oleh kode tersebut adalah sebagai berikut:

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

1) Pertama, library Pandas diimpor dengan alias 'pd', dan juga Sastrawi untuk stemming.
2) Sebuah objek StemmerFactory dari Sastrawi dibuat, dan stemmer untuk Bahasa Indonesia diinisialisasi.
3) Kemudian, file CSV dengan nama "Berita Kesejahteraan Sosial.csv" dibaca menggunakan Pandas dengan encoding 'utf-8', dan datanya disimpan dalam dataframe 'data'.
4) Selanjutnya, kolom 'Isi' dari dataframe 'data' diolah dengan menerapkan stemming ke setiap barisnya menggunakan metode 'apply' dan fungsi lambda.
5) Hasil stemming disimpan dalam kolom baru yang dinamai 'isi_stemmed'.
6) Terakhir, hasil dataframe yang telah di-stemming disimpan dalam file CSV baru dengan nama "Berita Kesejahteraan Sosial (1).csv" menggunakan metode 'to_csv'. Parameter 'index=False' digunakan untuk menghilangkan penambahan indeks pada data yang tersimpan, dan encoding 'utf-8' digunakan untuk memastikan karakter Bahasa Indonesia dapat disimpan dengan benar.

Sehingga, output dari kode ini adalah file CSV baru yang berisi data berita kesejahteraan sosial yang telah di-stemming dalam kolom 'isi_stemmed'. File CSV baru ini dapat digunakan untuk analisis lebih lanjut atau pemrosesan teks berikutnya yang memerlukan teks yang telah di-stemming.

5. Buatlah wordcloud dan most common word barplot, interpretasikan hasilnya

Jawab:

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

# Gabungkan semua teks yang sudah di-stemming menjadi satu teks panjang
# Baca teks dari file .txt
file_path = 'C:/Users/FARIZA SHIELDA/Documents/File Unair/Semester 5/Data Mining
II/Week 4/Berita Kesejahteraan Sosial (1).txt'  # Ganti dengan path file .txt
yang sesuai
with open(file_path, 'r', encoding='utf-8') as file:
    teks = file.read()

# Buat WordCloud
wordcloud = WordCloud(width=800, height=400,
background_color='white').generate(teks)

# Tampilkan WordCloud
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

PRAKTIKUM DATA MINING II

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

Output:

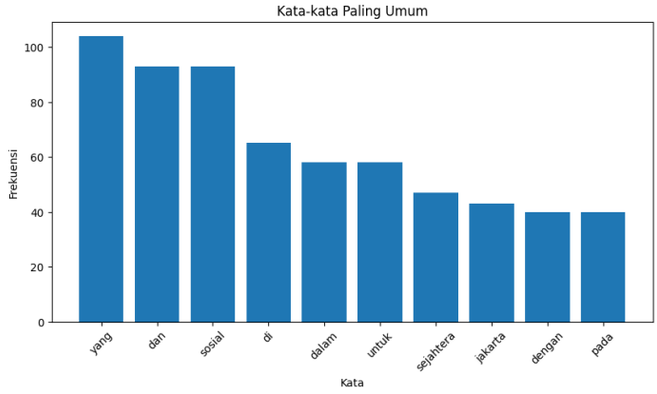
Penjelasan:

Kode ini digunakan untuk membuat visualisasi WordCloud dari teks yang telah di-stemming. Pertama, kode membaca teks dari sebuah file .txt dan menggabungkannya menjadi satu teks panjang. Selanjutnya, sebuah objek WordCloud dibuat dengan mengatur parameter seperti lebar, tinggi, dan warna latar belakang, lalu teks tersebut di-generate menjadi sebuah visualisasi WordCloud. Hasil WordCloud ditampilkan dalam bentuk gambar menggunakan Matplotlib dengan ukuran dan tampilan yang disesuaikan. Visualisasi WordCloud ini membantu dalam menyoroti kata-kata yang paling sering muncul dalam teks, yang dapat memberikan wawasan tentang isu-isu kunci dalam teks tersebut.

```
from collections import Counter

# Menghitung frekuensi kemunculan kata-kata
kata_kemunculan = Counter(teks.split())

# Ambil 10 kata paling umum
kata_umum = kata_kemunculan.most_common(10)

# Pisahkan kata dan frekuensinya
kata, frekuensi = zip(*kata_umum)

# Buat barplot
plt.figure(figsize=(10, 5))
plt.bar(kata, frekuensi)
plt.xlabel('Kata')
plt.ylabel('Frekuensi')
plt.title('Kata-kata Paling Umum')
plt.xticks(rotation=45)
plt.show()
```

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

Output:



Penjelasan:

Kode ini bertujuan untuk menghitung frekuensi kemunculan kata-kata dalam teks yang telah di-stemming. Pertama, kode menggunakan koleksi Counter dari Python untuk menghitung berapa kali setiap kata muncul dalam teks yang dipecah menjadi token. Selanjutnya, kode mengambil 10 kata paling umum dengan menggunakan metode most_common(10). Setelah itu, kata-kata dan frekuensinya dipisahkan ke dalam dua variabel terpisah, yaitu kata dan frekuensi. Akhirnya, kode membuat visualisasi barplot yang menampilkan kata-kata paling umum beserta frekuensinya. Visualisasi ini berguna untuk memberikan wawasan cepat tentang kata-kata yang paling dominan dalam teks tersebut.

6. Lakukan clustering dengan menggunakan fitur TF-IDF

Jawab:

```
import pandas as pd
import pickle

file_path = "Berita Kesejahteraan Sosial (1).csv"
data = pd.read_csv(file_path, encoding='utf-8')

from sklearn.feature_extraction.text import TfidfVectorizer

# Inisialisasi TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(max_df=0.95, min_df=2)

# Ekstraksi fitur TF-IDF
tfidf_matrix = tfidf_vectorizer.fit_transform(data['isi_stemmed'])

from sklearn.cluster import KMeans
```

PRAKTIKUM DATA MINING II

BAB : TEXT MINING

PRAKTIKUM KE : DUA (2)

NAMA : FARIZA SHIELDA AKZATRIA

NIM : 162112133026

TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
k = 3
seed = 99

# Inisialisasi model K-Means
km = KMeans(n_clusters=k, init='random', max_iter=300, random_state=seed,
n_init=10)

# Melakukan clustering pada data TF-IDF
km.fit(tfidf_matrix)

# Hasil clustering
labels = km.labels_

data['cluster'] = labels

output_file_path = "Berita Kesejahteraan Sosial Clustering.csv"  # Ganti dengan
lokasi dan nama file CSV untuk menyimpan hasil clustering
data.to_csv(output_file_path, index=False, encoding='utf-8')
```

Output:

Berita Kesejahteraan Sosial Clustering.csv

Judul	Isi	isi_stemm	cluster
Upaya Mewujudkan Kesejahteraan Masyarakat	KOMPAS.c	kompas cc	2
Kualitas SDM Jadi Kunci Sukses Program Kesejahteraan Sosial Kemensos	KOMPAS.c	kompas cc	1
SDM Pekerja Sosial Jadi Kunci Kesuksesan Program Kesejahteraan Sosial	KOMPAS.c	kompas cc	1
Survei Litbang "Kompas": Kepuasan di Bidang Kesejahteraan Sosial Meningkat. Buah dari Program BansosJAKARTA	KOMPAS.c	kompas cc	0
KPK Minta Risma Perbaiki Data Terpadu Kesejahteraan Sosial	JAKARTA.	jakarta ko	0
Risma Bakal Rombak Besar-besaran Data Terpadu Kesejahteraan Sosial	JAKARTA.	jakarta ko	2
Kartu Kesejahteraan Sosial. Wujud Komitmen Pemprov DKI bagi Warga Tak Mampu	KOMPAS.c	kompas cc	0
SDGs Era New Normal: Pentingnya Sinergi Swasta dan Pemerintah Wujudkan Kesejahteraan Sosial Masyarakat	JAKARTA.	jakarta ko	2
Risma Blusukan di DKI. Anggota DPR Ingatkan Benahi Data Kesejahteraan Sosial	JAKARTA.	jakarta ko	0
Menaker Dorong Pekerja Musik Dapat Perlindungan dan Kesejahteraan Sosial	JAKARTA.	jakarta ko	2

Penjelasan:

Kode di atas digunakan untuk melakukan clustering pada data berita kesejahteraan sosial menggunakan metode K-Means dan kemudian menyimpan hasil clustering ke dalam file CSV baru. Berikut langkah-langkahnya:

1) Pertama, library Pandas diimpor dengan alias 'pd', dan file CSV yang berisi data berita kesejahteraan sosial ("Berita Kesejahteraan Sosial (1).csv") dibaca menggunakan Pandas dengan encoding 'utf-8', dan datanya disimpan dalam dataframe 'data'.

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

2) Selanjutnya, inisialisasi dari TfidfVectorizer digunakan untuk melakukan ekstraksi fitur TF-IDF dari teks yang telah di-stemming. Dengan kata lain, ini digunakan untuk mengubah teks-teks berita menjadi representasi numerik yang dapat digunakan dalam proses clustering.
3) Kemudian, model K-Means diinisialisasi dengan jumlah cluster 'k' yang ditentukan sebelumnya, serta beberapa parameter lainnya seperti metode inisialisasi ('init'), maksimum iterasi ('max_iter'), dan seed untuk inisialisasi titik-titik pusat cluster ('random_state').
4) Setelah itu, clustering dilakukan pada data TF-IDF dengan menggunakan model K-Means yang telah diinisialisasi sebelumnya. Hasil clustering disimpan dalam kolom 'cluster' dalam dataframe 'data'.
5) Terakhir, hasil dataframe yang telah di-cluster disimpan dalam file CSV baru dengan nama "Berita Kesejahteraan Sosial Clustering.csv" menggunakan metode 'to_csv'. Parameter 'index=False' digunakan untuk menghilangkan penambahan indeks pada data yang tersimpan, dan encoding 'utf-8' digunakan untuk memastikan karakter Bahasa Indonesia dapat disimpan dengan benar.

Sehingga, output dari kode ini adalah file CSV baru yang berisi data berita kesejahteraan sosial yang telah di-cluster, dengan kolom tambahan 'cluster' yang menunjukkan keanggotaan setiap berita ke dalam salah satu dari tiga cluster yang telah dibentuk. File CSV ini dapat digunakan untuk analisis lebih lanjut atau pemrosesan berikutnya yang memerlukan data yang telah di-cluster.

7. Buat visualisasi clusternya dan lakukan interpretasi terhadap hasil tersebut

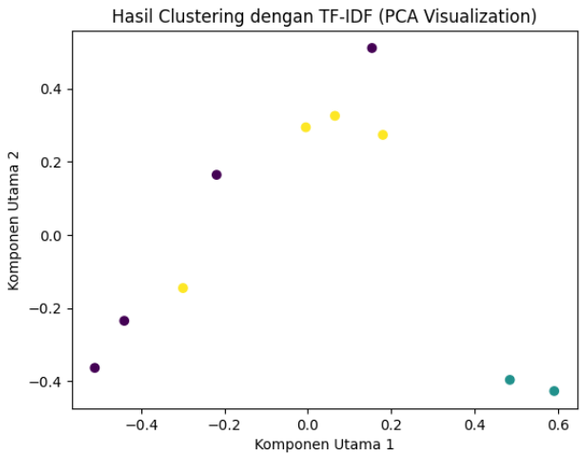
Jawab:

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Baca data dari file CSV
file_path = "Berita Kesejahteraan Sosial Clustering.csv"
data = pd.read_csv(file_path, encoding='utf-8')

# Inisialisasi TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(max_df=0.95, min_df=2)

# Ekstraksi fitur TF-IDF
tfidf_matrix = tfidf_vectorizer.fit_transform(data['isi_stemmed'])

# Inisialisasi model K-Means
k = 3
```

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

```
seed = 99
km = KMeans(n_clusters=k, init='random', max_iter=300, random_state=seed,
n_init=10)

# Melakukan clustering pada data TF-IDF
km.fit(tfidf_matrix)

# Hasil clustering
labels = km.labels_

# Menambahkan kolom cluster ke dataframe
data['cluster'] = labels

# Reduksi dimensi menggunakan PCA
pca = PCA(n_components=2)
pca_result = pca.fit_transform(tfidf_matrix.toarray())

# Buat plot hasil clustering
plt.scatter(pca_result[:, 0], pca_result[:, 1], c=labels, cmap='viridis')
plt.title('Hasil Clustering dengan TF-IDF (PCA Visualization)')
plt.xlabel('Komponen Utama 1')
plt.ylabel('Komponen Utama 2')
plt.show()
```

Output:



Penjelasan:

Kode di atas digunakan untuk melakukan visualisasi hasil clustering pada data berita kesejahteraan sosial menggunakan metode K-Means dengan representasi TF-IDF. Langkah-langkah yang dilakukan oleh kode tersebut adalah sebagai berikut:

PRAKTIKUM DATA MINING II

BAB	: TEXT MINING
PRAKTIKUM KE	: DUA (2)
NAMA	: FARIZA SHIELDA AKZATRIA
NIM	: 162112133026
TGL PRAKTIKUM	: 21 SEPTEMBER 2023

1) Pertama, data dari file CSV yang berisi hasil clustering sebelumnya ("Berita Kesejahteraan Sosial Clustering.csv") dibaca menggunakan Pandas dengan encoding 'utf-8', dan datanya disimpan dalam dataframe 'data'.
2) Selanjutnya, inisialisasi dari TfidfVectorizer digunakan untuk mengubah teks berita menjadi representasi TF-IDF yang akan digunakan dalam proses clustering.
3) Kemudian, model K-Means diinisialisasi dengan jumlah cluster 'k' yang telah ditentukan sebelumnya yakni 3, serta beberapa parameter lainnya seperti metode inisialisasi ('init'), maksimum iterasi ('max_iter'), seed untuk inisialisasi titik-titik pusat cluster ('random_state'), dan jumlah iterasi inisialisasi yang berbeda ('n_init').
4) Setelah itu, clustering dilakukan pada data TF-IDF dengan menggunakan model K-Means yang telah diinisialisasi sebelumnya. Hasil clustering disimpan dalam kolom 'cluster' dalam dataframe 'data'.
5) Selanjutnya, dilakukan reduksi dimensi menggunakan metode Principal Component Analysis (PCA) dengan 2 komponen utama. Hal ini dilakukan agar data dapat divisualisasikan dalam bidang dua dimensi.
6) Terakhir, hasil clustering divisualisasikan dengan scatter plot, di mana sumbu x dan y menggambarkan dua komponen utama hasil PCA, dan warna titik-titik pada plot menunjukkan cluster-cluster yang telah dibentuk. Visualisasi ini membantu dalam memahami pola-pola clustering pada data berita kesejahteraan sosial.

Dalam visualisasi PCA tersebut, terdapat tiga kelompok warna yang mewakili tiga kluster berita kesejahteraan sosial yang telah dihasilkan. Setiap titik mewakili satu berita dalam dataset.

- Kluster Kuning (Cluster 0): Kluster kuning terlihat tersebar di sekitar area tengah dan juga memiliki titik yang terletak di antara kluster ungu. Ini mengindikasikan bahwa kluster kuning mungkin mencakup berita yang memiliki beberapa kesamaan dengan kluster ungu. Kluster kuning ini mungkin mencakup berita dengan topik yang lebih beragam atau ambigu.
- Kluster Biru (Cluster 1): Kluster biru tampaknya lebih padat dan kompak di sekitar pusat tertentu dalam visualisasi. Hal ini menunjukkan bahwa berita-berita dalam kluster biru memiliki kesamaan yang tinggi dalam topik atau karakteristiknya. Kluster biru ini mungkin mencakup berita yang lebih fokus pada topik kesejahteraan sosial tertentu.
- Kluster Ungu (Cluster 2): Kluster ungu tampaknya tersebar lebih merata dalam kedua dimensi utama yang dihasilkan oleh PCA. Ini menunjukkan bahwa berita-berita dalam kluster ungu memiliki keragaman topik atau karakteristik yang lebih luas. Mungkin saja kluster ungu ini mencakup berita yang beragam topiknya terkait dengan kesejahteraan sosial.

Dapat terlihat adanya beberapa titik yang terisolasi atau berjauhan dari kluster utama, dapat dianggap sebagai *outlier* yang memiliki karakteristik yang sangat berbeda dari berita-berita dalam kluster utama. Serta terdapat overlap kluster yang menunjukkan bahwa beberapa berita mungkin memiliki karakteristik yang mirip dengan berita di kluster lain. Hal ini dapat mengindikasikan bahwa hasil clustering mungkin tidak begitu tajam atau ada beberapa berita yang ambigu dalam penentuan kluster.

BAB : TEXT MINING
PRAKTIKUM KE : DUA (2)
NAMA : FARIZA SHIELDA AKZATRIA
NIM : 162112133026
TGL PRAKTIKUM : 21 SEPTEMBER 2023

8. Gunakan validasi menggunakan salah satu Davies-Bouldin index atau Silhouette score
Jawab:

```
from sklearn.metrics import davies_bouldin_score, silhouette_score

X = tfidf_matrix.toarray()
# Perhitungan Davies-Bouldin Index
db_index = davies_bouldin_score(X, labels)
print(f"Davies-Bouldin Index: {db_index}")

# Perhitungan Silhouette Score
silhouette_avg = silhouette_score(X, labels)
print(f"Silhouette Score: {silhouette_avg}")
```

Output:
Davies-Bouldin Index: 1.9705448399867336
Silhouette Score: 0.055949449593570025

Penjelasan:
Kode di atas digunakan untuk mengukur kualitas hasil clustering pada data berita kesejahteraan sosial dengan dua metrik evaluasi, yaitu Davies-Bouldin Index dan Silhouette Score. Pertama, hasil clustering yang telah dilakukan sebelumnya disimpan dalam variabel 'labels', dan matriks TF-IDF yang telah diubah menjadi array dengan 'tfidf_matrix.toarray()' disimpan dalam variabel 'X'. Kemudian, dilakukan perhitungan Davies-Bouldin Index dengan menggunakan fungsi 'davies_bouldin_score' dari library 'sklearn.metrics'. Indeks ini mengukur seberapa baik kluster-kluster dalam data terpisah satu sama lain. Semakin rendah nilai Davies-Bouldin Index, semakin baik hasil clusteringnya. Nilai Davies-Bouldin Index yang didapatkan adalah sekitar 1.971. Selanjutnya, dilakukan perhitungan Silhouette Score menggunakan fungsi 'silhouette_score' dari library yang sama. Silhouette Score mengukur seberapa baik objek dalam satu kluster berdekatan satu sama lain dibandingkan dengan kluster lain. Rentang nilai Silhouette Score adalah dari -1 hingga 1, di mana nilai positif menunjukkan bahwa objek dalam satu kluster berdekatan dengan kluster lain. Nilai Silhouette Score yang didapatkan adalah sekitar 0.056. Dengan kedua metrik evaluasi ini, maka dapat mengevaluasi kualitas hasil clustering pada data berita kesejahteraan sosial.

Hasil evaluasi menggunakan Davies-Bouldin Index dan Silhouette Score menunjukkan bahwa kualitas hasil clustering pada data berita kesejahteraan sosial cenderung rendah. Nilai Davies-Bouldin Index yang relatif tinggi (sekitar 1.971) menandakan bahwa kluster-kluster dalam data cenderung terpisah satu sama lain, yang mungkin mengindikasikan adanya penyebaran yang tidak optimal dalam kluster tersebut. Nilai Silhouette Score yang mendekati nol (sekitar 0.056) juga menunjukkan bahwa objek-objek dalam kluster tidak berdekatan dengan baik dan terdapat overlap antara kluster-kuster tersebut. Sehingga, hasil clustering pada data ini perlu diperbaiki atau dipertimbangkan kembali untuk meningkatkan kualitasnya, mungkin dengan mencoba nilai k yang berbeda atau algoritma clustering yang lebih sesuai dengan karakteristik data.