Praktikum M4 Data Mining II: Crawling Wikipedia

Nama: Amalika Ari Anindya

NIM: 164221029

Kelas: Data Mining II SD-A2

1. Lakukan seluruh percobaan pada modul ini dan berikan analisis yang kalian temukan
   Percobaan di modul dilampirkan pada pdf atau ipynb satu.

2. Jelaskan perbedaan hasil dari Preprocessing menggunakan NLTK, TextBlob, dan Sastrawi dan berikan contohnya!
   Diberikan dua teks dalam Bahasa Inggris dan Bahasa Indonesia untuk percobaan.

```
teks_inggris = "Joanna Paresi was the last one left – the last living person in a family who had been market traders for hund
teks_inggris
```

[3]:
```
"Joanna Paresi was the last one left – the last living person in a family who had been market traders for hundreds of years.
She was born in a village at the bottom of high mountains, and she had lived there all her life. At the top of the mountain
s, the stone fruit grew. In autumn, the fruit fell down the mountains. Most of the stone fruit got lost and no one could fin
d them again. But some fruit fell into a small valley. Joanna's family were the only people who knew about the valley. When
the stone fruit fell from the trees, they were black and hard. It took four long months for them to become ripe. They turned
from black to grey and, finally, to silver. Then people could enjoy the sweet, sun-coloured fruit inside."
```

```
teks_indonesia = "Di sebuah perkampungan di kaki Gunung Tinjau, ada sepuluh orang bersaudara yang biasa disebut Bujang Sembil
teks_indonesia
```

[4]:
```
'Di sebuah perkampungan di kaki Gunung Tinjau, ada sepuluh orang bersaudara yang biasa disebut Bujang Sembilan. Si sulung be
rnama Kukuban dan si bungsu bernama Sani. Mereka mempunyai seorang paman bernama Datuk Limbatang. Datuk Limbatang pun mempun
yai seorang putra bernama Giran. Suatu hari, Datuk Limbatang berkunjung ke rumah Bujang Sambilan dan di saat itulah Sani dan
Girang menyadari bahwa mereka saling menaruh hati. Seiring berjalannya waktu, ketika musim panen tiba kampung tersebut menga
dakan adu silat.'
```

Percobaan NTLK menggunakan teks berbahasa Inggris.

Inggris

In [7]:
```
# Tokenize

from nltk.tokenize import word_tokenize
hasil_token = word_tokenize(teks_inggris)
hasil_token
```

Out[7]:
```
['Joanna',
 'Paresi',
 'was',
 'the',
 'last',
 'one',
 'left',
 '–',
 'the',
 'last',
 'living',
 'person',
 'in',
 'a',
 'family',
 'who',
 'had',
 'been',
 'market',
```

```
# Stopwords

from nltk.corpus import stopwords

a = set(stopwords.words('english'))
text1 = word_tokenize(teks_inggris.lower())
hasil_sw = [x for x in text1 if x not in a]
print(hasil_sw)
```

```
['joanna', 'paresi', 'last', 'one', 'left', '–', 'last', 'living', 'person', 'family', 'market', 'traders', 'hundreds', 'yea
rs', '.', 'born', 'village', 'bottom', 'high', 'mountains', ',', 'lived', 'life', '.', 'top', 'mountains', ',', 'stone', 'fr
uit', 'grew', '.', 'autumn', ',', 'fruit', 'fell', 'mountains', '.', 'stone', 'fruit', 'got', 'lost', 'one', 'could', 'fin
d', '.', 'fruit', 'fell', 'small', 'valley', '.', 'joanna', "'s", 'family', 'people', 'knew', 'valley', '.', 'stone', 'frui
t', 'fell', 'trees', ',', 'black', 'hard', '.', 'took', 'four', 'long', 'months', 'become', 'ripe', '.', 'turned', 'black',
'grey', ',', 'finally', ',', 'silver', '.', 'people', 'could', 'enjoy', 'sweet', ',', 'sun-coloured', 'fruit', 'inside',
'.']
```

```
# Stemming

from nltk.stem.lancaster import LancasterStemmer
from nltk.stem.porter import PorterStemmer
from nltk.stem.snowball import SnowballStemmer

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 hasil_sw), '\n')
```

```
Lancaster :  joann pares last on left – last liv person famy market trad hundr year . born vil bottom high mountain , liv li
f . top mountain , ston fruit grew . autumn , fruit fel mountain . ston fruit got lost on could find . fruit fel smal valley
. joann 's famy peopl knew valley . ston fruit fel tre , black hard . took four long month becom rip . turn black grey , fin
, silv . peopl could enjoy sweet , sun-coloured fruit insid .

Porter :  joanna paresi last one left – last live person famili market trader hundr year . born villag bottom high mountain
, live life . top mountain , stone fruit grew . autumn , fruit fell mountain . stone fruit got lost one could find . fruit f
ell small valley . joanna 's famili peopl knew valley . stone fruit fell tree , black hard . took four long month becom ripe
. turn black grey , final , silver . peopl could enjoy sweet , sun-colour fruit insid .

SnowBall :  joanna paresi last one left – last live person famili market trader hundr year . born villag bottom high mountai
n , live life . top mountain , stone fruit grew . autumn , fruit fell mountain . stone fruit got lost one could find . fruit
fell small valley . joanna 's famili peopl knew valley . stone fruit fell tree , black hard . took four long month becom rip
e . turn black grey , final , silver . peopl could enjoy sweet , sun-colour fruit insid .
```

```
# Lemmatization

from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

lemmatized_words = [lemmatizer.lemmatize(word) for word in hasil_sw]
print(lemmatized_words)
```

```
['joanna', 'paresi', 'last', 'one', 'left', '-', 'last', 'living', 'person', 'family', 'market', 'trader', 'hundred', 'yea
r', '.', 'born', 'village', 'bottom', 'high', 'mountain', ',', 'lived', 'life', '.', 'top', 'mountain', ',', 'stone', 'frui
t', 'grew', '.', 'autumn', ',', 'fruit', 'fell', 'mountain', '.', 'stone', 'fruit', 'got', 'lost', 'one', 'could', 'find',
'.', 'fruit', 'fell', 'small', 'valley', '.', 'joanna', "'s", 'family', 'people', 'knew', 'valley', '.', 'stone', 'fruit',
'fell', 'tree', ',', 'black', 'hard', '.', 'took', 'four', 'long', 'month', 'become', 'ripe', '.', 'turned', 'black', 'gre
y', ',', 'finally', ',', 'silver', '.', 'people', 'could', 'enjoy', 'sweet', ',', 'sun-coloured', 'fruit', 'inside', '.']
```

Percobaan NTLK menggunakan teks berbahasa Indonesia.

```
# Tokenize

from nltk.tokenize import word_tokenize
hasil_token = word_tokenize(teks_indonesia)
hasil_token
```

```
['Di',
 'sebuah',
 'perkampungan',
 'di',
 'kaki',
 'Gunung',
 'Tinjau',
 ',',
 'ada',
 'sepuluh',
 'orang',
 'bersaudara',
 'yang',
 'biasa',
 'disebut',
 'Bujang',
 'Sembilan',
 '.',
 'Si',
 'sulung',
 'bernama',
```

NLTK harus menambahkan dataset baru untuk stopwords Bahasa Indonesia.

```
from nltk.corpus import stopwords
from nltk.data import find

try:
    # Check if the stopwords dataset is available
    find('corpora/stopwords.zip')
except:
    print("Stopwords dataset not found. Downloading...")
    from nltk import download
    download('stopwords')

# Load the stopwords corpus
stop_words = stopwords.words('indonesian')

# Function to remove stopwords from a list of words
def remove_stopwords(words, stopwords):
    return [word for word in words if word not in stopwords]
```

```
# Stopwords

from nltk.corpus import stopwords

text1 = word_tokenize(teks_indonesia.lower())
filtered_words = remove_stopwords(text1, stop_words)
print(filtered_words)
```

```
['perkampungan', 'kaki', 'gunung', 'tinjau', ',', 'sepuluh', 'orang', 'bersaudara', 'bujang', 'sembilan', '.', 'si', 'sulun
g', 'bernama', 'kukuban', 'si', 'bungsu', 'bernama', 'sani', '.', 'paman', 'bernama', 'datuk', 'limbatang', '.', 'datuk', 'l
imbatang', 'putra', 'bernama', 'giran', '.', ',', 'datuk', 'limbatang', 'berkunjung', 'rumah', 'bujang', 'sambilan', 'sani',
'girang', 'menyadari', 'menaruh', 'hati', '.', 'seiring', 'berjalannya', ',', 'musim', 'panen', 'kampung', 'mengadakan', 'ad
u', 'silat', '.']
```

NLTK tidak dapat menerapkan Stemming dan Lemmatization dalam Bahasa Indonesia dan harus menggunakan Sastrawi.

Percobaan TextBlob menggunakan teks berbahasa Inggris.

```
from textblob import TextBlob

# Tokenisasi kata
tb_token = TextBlob(teks_inggris).words
print(tb_token)
```

```
['Joanna', 'Paresi', 'was', 'the', 'last', 'one', 'left', '-', 'the', 'last', 'living', 'person', 'in', 'a', 'family', 'wh
o', 'had', 'been', 'market', 'traders', 'for', 'hundreds', 'of', 'years', 'She', 'was', 'born', 'in', 'a', 'village', 'at',
'the', 'bottom', 'of', 'high', 'mountains', 'and', 'she', 'had', 'lived', 'there', 'all', 'her', 'life', 'At', 'the', 'top',
'of', 'the', 'mountains', 'the', 'stone', 'fruit', 'grew', 'In', 'autumn', 'the', 'fruit', 'fell', 'down', 'the', 'mountain
s', 'Most', 'of', 'the', 'stone', 'fruit', 'got', 'lost', 'and', 'no', 'one', 'could', 'find', 'them', 'again', 'But', 'som
e', 'fruit', 'fell', 'into', 'a', 'small', 'valley', 'Joanna', "'s", 'family', 'were', 'the', 'only', 'people', 'who', 'kne
w', 'about', 'the', 'valley', 'When', 'the', 'stone', 'fruit', 'fell', 'from', 'the', 'trees', 'they', 'were', 'black', 'an
d', 'hard', 'It', 'took', 'four', 'long', 'months', 'for', 'them', 'to', 'become', 'ripe', 'They', 'turned', 'from', 'blac
k', 'to', 'grey', 'and', 'finally', 'to', 'silver', 'Then', 'people', 'could', 'enjoy', 'the', 'sweet', 'sun-coloured', 'fru
it', 'inside']
```

```
from textblob import Word

# Stemming
tb_token.stem()
```

```
WordList(['joanna', 'paresi', 'wa', 'the', 'last', 'one', 'left', '-', 'the', 'last', 'live', 'person', 'in', 'a', 'famili',
'who', 'had', 'been', 'market', 'trader', 'for', 'hundr', 'of', 'year', 'she', 'wa', 'born', 'in', 'a', 'villag', 'at', 'th
e', 'bottom', 'of', 'high', 'mountain', 'and', 'she', 'had', 'live', 'there', 'all', 'her', 'life', 'at', 'the', 'top', 'o
f', 'the', 'mountain', 'the', 'stone', 'fruit', 'grew', 'in', 'autumn', 'the', 'fruit', 'fell', 'down', 'the', 'mountain',
'most', 'of', 'the', 'stone', 'fruit', 'got', 'lost', 'and', 'no', 'one', 'could', 'find', 'them', 'again', 'but', 'some',
'fruit', 'fell', 'into', 'a', 'small', 'valley', 'joanna', "'s", 'famili', 'were', 'the', 'onli', 'peopl', 'who', 'knew', 'a
bout', 'the', 'valley', 'when', 'the', 'stone', 'fruit', 'fell', 'from', 'the', 'tree', 'they', 'were', 'black', 'and', 'har
d', 'it', 'took', 'four', 'long', 'month', 'for', 'them', 'to', 'becom', 'ripe', 'they', 'turn', 'from', 'black', 'to', 'gre
y', 'and', 'final', 'to', 'silver', 'then', 'peopl', 'could', 'enjoy', 'the', 'sweet', 'sun-colour', 'fruit', 'insid'])
```

```
# Lemmatizer
tb_token.lemmatize()
```

```
WordList(['Joanna', 'Paresi', 'wa', 'the', 'last', 'one', 'left', '-', 'the', 'last', 'living', 'person', 'in', 'a', 'famil
y', 'who', 'had', 'been', 'market', 'trader', 'for', 'hundred', 'of', 'year', 'She', 'wa', 'born', 'in', 'a', 'village', 'a
t', 'the', 'bottom', 'of', 'high', 'mountain', 'and', 'she', 'had', 'lived', 'there', 'all', 'her', 'life', 'At', 'the', 'to
p', 'of', 'the', 'mountain', 'the', 'stone', 'fruit', 'grew', 'In', 'autumn', 'the', 'fruit', 'fell', 'down', 'the', 'mounta
in', 'Most', 'of', 'the', 'stone', 'fruit', 'got', 'lost', 'and', 'no', 'one', 'could', 'find', 'them', 'again', 'But', 'som
e', 'fruit', 'fell', 'into', 'a', 'small', 'valley', 'Joanna', "'s", 'family', 'were', 'the', 'only', 'people', 'who', 'kne
w', 'about', 'the', 'valley', 'When', 'the', 'stone', 'fruit', 'fell', 'from', 'the', 'tree', 'they', 'were', 'black', 'an
d', 'hard', 'It', 'took', 'four', 'long', 'month', 'for', 'them', 'to', 'become', 'ripe', 'They', 'turned', 'from', 'black',
'to', 'grey', 'and', 'finally', 'to', 'silver', 'Then', 'people', 'could', 'enjoy', 'the', 'sweet', 'sun-coloured', 'fruit',
'inside'])
```

Percobaan TextBlob menggunakan teks berbahasa Indonesia.

```
from textblob import TextBlob

# Tokenisasi kata
tb_token = TextBlob(teks_indonesia).words
print(tb_token)
```

```
['Di', 'sebuah', 'perkampungan', 'di', 'kaki', 'Gunung', 'Tinjau', 'ada', 'sepuluh', 'orang', 'bersaudara', 'yang', 'biasa',
'disebut', 'Bujang', 'Sembilan', 'Si', 'sulung', 'bernama', 'Kukuban', 'dan', 'si', 'bungsu', 'bernama', 'Sani', 'Mereka',
'mempunyai', 'seorang', 'paman', 'bernama', 'Datuk', 'Limbatang', 'Datuk', 'Limbatang', 'pun', 'mempunyai', 'seorang', 'putr
a', 'bernama', 'Giran', 'Suatu', 'hari', 'Datuk', 'Limbatang', 'berkunjung', 'ke', 'rumah', 'Bujang', 'Sambilan', 'dan', 'd
i', 'saat', 'itulah', 'Sani', 'dan', 'Girang', 'menyadari', 'bahwa', 'mereka', 'saling', 'menaruh', 'hati', 'Seiring', 'berj
alannya', 'waktu', 'ketika', 'musim', 'panen', 'tiba', 'kampung', 'tersebut', 'mengadakan', 'adu', 'silat']
```

Stemming dan Lemmatization tidak dapat digunakan dalam Bahasa Indonesia pada TextBlob. Jika memang ingin menggunakan TextBlob maka harus menambahkan bantuan library Sastrawi.

```
from textblob import TextBlob
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

# Initialize Sastrawi stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# Stem the text
tbstring = ' '.join(tb_token)
stemmed_text = stemmer.stem(tbstring)
print(stemmed_text)
```

```
di buah kampung di kaki gunung tinjau ada puluh orang saudara yang biasa sebut bujang sembilan si sulung nama ban dan si bun
gsu nama sani mereka punya orang paman nama datuk limbatang datuk limbatang pun punya orang putra nama gir suatu hari datuk
limbatang kunjung ke rumah bujang sambil dan di saat itu sani dan girang sadar bahwa mereka saling taruh hati iring jalan wa
ktu ketika musim panen tiba kampung sebut ada adu silat
```

Percobaan Sastrawi menggunakan teks berbahasa Inggris.

Sastrawi tidak dapat digunakan pada teks Bahasa Inggris.

Percobaan Sastrawi menggunakan teks berbahasa Indonesia.

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

factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()

stop = stopword.remove(teks_indonesia.lower())
print(stop)
```

```
sebuah perkampungan kaki gunung tinjau, sepuluh orang bersaudara biasa disebut bujang sembilan. si sulung bernama kukuban si
bungsu bernama sani. mempunyai seorang paman bernama datuk limbatang. datuk limbatang mempunyai seorang putra bernama giran.
suatu hari, datuk limbatang berkunjung rumah bujang sambilan di itulah sani girang menyadari mereka saling menaruh hati. sei
ring berjalannya waktu, musim panen tiba kampung tersebut mengadakan adu silat.
```

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

print(stemmer.stem(stop))
```

```
buah kampung kaki gunung tinjau puluh orang saudara biasa sebut bujang sembilan si sulung nama ban si bungsu nama sani punya
orang paman nama datuk limbatang datuk limbatang punya orang putra nama gir suatu hari datuk limbatang kunjung rumah bujang
sambil di itu sani girang sadar mereka saling taruh hati iring jalan waktu musim panen tiba kampung sebut ada adu silat
```

3. Crawling dataset dengan total 10 pada berbagai judul artikel Wikipedia berdasarkan daftar topik sesuai dengan akhiran nim dan wajib dalam topik yang sama

   NIM 29 mendapat topik 09 yaitu TV Indonesia https://id.wikipedia.org/wiki/Daftar_stasiun_televisi_di_Indonesia

```python
import wikipedia
import csv

wikipedia.set_lang("id")

judul = ["TVRI (saluran televisi)", "GTV (Indonesia)", "Indosiar", "Antv", "Trans7", "NET.",
         "Kompas TV", "CNN Indonesia", "MNCTV", "Nusantara TV"]

with open('wikipedia_pages.csv', mode='w', newline='', encoding='utf-8') as file:
    writer = csv.writer(file)
    writer.writerow(["Judul", "Konten"])

    for i in judul:
        try:
            halaman = wikipedia.page(i)
            konten = halaman.content
            writer.writerow([i, konten])
            print(f"Berhasil menyimpan: {i}")
        except wikipedia.exceptions.DisambiguationError as e:
            print(f"Disambiguasi untuk: {i} - Dilewati")
        except wikipedia.exceptions.PageError as e:
            print(f"Halaman tidak ditemukan untuk: {i} - Dilewati")
```

Di atas merupakan suatu fungsi untuk mengumpulkan 10 judul wikipedia untuk dicrawling. Variabel judul merupakan 10 judul halaman yang akan dicrawling. Baris 'with open' akan membuat file csv baru yang berisi hasil dari crawling. Fungsi perulangan for digunakan untuk menjalankan crawling dan memberikan peringatan berhasil atau gagalnya crawling. Berikut adalah output yang diberikan oleh Python.

```
Berhasil menyimpan: TVRI (saluran televisi)
Berhasil menyimpan: GTV (Indonesia)
Berhasil menyimpan: Indosiar
Berhasil menyimpan: Antv
Berhasil menyimpan: Trans7
Berhasil menyimpan: NET.
Berhasil menyimpan: Kompas TV
Berhasil menyimpan: CNN Indonesia
Berhasil menyimpan: MNCTV
Berhasil menyimpan: Nusantara TV
```

Berikut adalah hasil crawling berupa csv.

```
import pandas as pd
data = pd.read_csv("wikipedia_pages.csv")
data
```

	Judul	Konten
0	TVRI (saluran televisi)	TVRI (terkadang disebut sebagai TVRI Nasional)...
1	GTV (Indonesia)	GTV (pelafalan dalam bahasa Indonesia: [dʒitif...
2	Indosiar	Indosiar (secara resmi bernama Indosiar Visual...
3	Antv	ANTV (pelafalan dalam bahasa Indonesia: [antɛf...
4	Trans7	Trans7 (sebelumnya bernama TV7) adalah sebuah ...
5	NET.	NET. (disebut juga sebagai NET TV; singkatan d...
6	Kompas TV	Kompas TV adalah salah satu jaringan televisi ...
7	CNN Indonesia	CNN Indonesia adalah sebuah jaringan televisi ...
8	MNCTV	MNCTV (pelafalan dalam bahasa Indonesia: [emen...
9	Nusantara TV	Nusantara TV (digayakan dengan huruf kecil sem...

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

   Preprocessing dilakukan menggunakan tokenize, penghilangan stopwords, dan stemming. Langkah preprocessing di sini digunakan menggunakan library NLTK dan Sastrawi. Berikut adalah kode untuk tokenizing.

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

def preprocess(doc):
    sents = word_tokenize(doc)
    sents_tok = list() #tokenisasi kalimat
    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 data["Konten"]:
    docs_clear.append(preprocess(d))
```

Khusus untuk menghilangkan stopwords digunakan library Sastrawi untuk Bahasa Indonesia.

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from nltk.tokenize import word_tokenize
factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()

kalimat = []
for i in docs_clear:
    stop = stopword.remove(i)
    kalimat.append(stop)
print(kalimat)
print(type(kalimat))
print(len(kalimat))
```

Stemming juga dilakukan menggunakan Sastrawi.

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

hasil_stem = []
for i in kalimat:
    stemmers = stemmer.stem(i)
    hasil_stem.append(stemmers)
print(hasil_stem)
```

Berikut adalah preview hasil dari preprocessing.

```
['tvri terkadang sebut tvri nasional salur televisi publik nasional utama milik lpp televisi republik indonesia mulai uda
ra sejak 24 agustus 1962 jadi salur televisi pertama indonesia siar tvri tayang aneka ragam program mulai berita informas
i hibur hingga olahraga waktu siar 24 jam hari salur belum jaring stasiun tvri daerah siar analog jakarta sekitar hingga
tahun 2022 kini siar penuh mandiri siar digital sejarah awal siar tvri diri perintah indonesia rangka sambut pesta olahra
ga asia 1962 siar coba mulai tayang acara hari merdeka republik indonesia ke-17 tanggal 17 agustus 1962 acara sebut siar
halaman istana merdeka jalan medan merdeka jakarta pusat format hitam-putih dukung pancar cadang kuat 100 watt tiga kamer
a siar pancar kanal 5 vhf beri nama salur lima lama 3 jam pukul 07 30-11 02 wib tanggal 17-23 agustus 1962 tvri laku siar
coba pukul 19 00-20 00 wib lama minggu 24 agustus 1962 mulai siar pola teknik pukul 14 30 wib jam 16 00-19 00 wib tvri re
smi udara pertama kali acara siar langsung upacara buka pesta olahraga asia 1962 stadion utama gelora bung karno tvri tay
ang siar putar pesta olahraga asia 1962 pancar salur 9 9 vhf daya siar 10 kw salur 5 vhf manfaat salur cadang salur 9 ala
mi rusa tvri tayang siar langsung mulai pukul 10 00 wib hingga 14 00 wib siar tunda mulai pukul 20 45 wib hingga 23 00 wi
b program asi games iv siar tvri rupa buat produksi film negara pfn asi games ke-iv akhir siar tvri sempat nonaktif lama
beberapa waktu sejak 12 september 1962 hingga udara kala 19 september 1962 henti siar jadi pihak tvri maupun perintah mil
ik ide acara apa siar lanjut harap henti acara siap konsolidasi kemudian tvri mulai siar kala 11 oktober 1962 sama resmi
studio pandu siar wanita pertama nama sus salamun 14 november 1962 tvri laku siar pertama studio kantor pusat senayan tvr
i siar malam hari lama 1 jam 5 hari minggu kemudian sejak 1963 panjang jadi 2 jam kurun waktu 1964-1967 waktu siar panjan
g jadi 3 5 jam tvri mulai tayang iklan 1 maret 1963 iklan makan 10 jam siar tvri 90 -nya tuju program hibur 30 acara didi
k terang budaya 60 iklan tahun sama tvri mulai rekam buat acara luar studio guna video tape recorder kemudian hari iklan
tvri satu jadi satu program khusus nama mana suka siar niaga kembang lanjut awal siar tvri nikmat jakarta jawa barat irin
```

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

   Pembuatan wordcloud menggunakan library matplotplib dan wordcloud. Berikut adalah hasil wordcloud dari keseluruhan teks.

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

hasil_string = ' '.join(hasil_stem)
wordcloud = WordCloud(background_color="white").generate(hasil_string)

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

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

Terlihat dari wordcloud bahwa kata yang paling besar antara lain: siar, milik, program, jadi, acara, indonesia, tpi, dan sebut. Kata siar kemungkinan berasal dari kata ‘siaran’ atau ‘bersiar’ yang pastinya banyak digunakan mengingat topik yang diambil adalah ‘Siaran Televisi’. Kata Indonesia juga memiliki frekuensi banyak mengingat yang diambil adalah Acara Televisi di Indonesia.

Untuk barplot akan diambil kata teratas sejumlah 20. Jika diambil semua maka barplot tidak terbaca. Sebelum membuat barplot dibutuhkan penghitungan frekuensi kata terbanyak dalam teks.

```
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist

# Tokenize the text into words
tokens = word_tokenize(hasil_string)

# Filter out words of length 1 or less
filtered_words = [word for word in tokens if len(word) > 1]

# Create a frequency distribution
fdist = FreqDist(filtered_words)

# Display the most common words
print(fdist.most_common(10))
```

```
[('siar', 536), ('tv', 317), ('televisi', 266), ('acara', 239), ('indonesia', 236), ('milik', 184), ('tahun', 174), ('jadi', 173), ('program', 164), ('tvri', 161)]
```

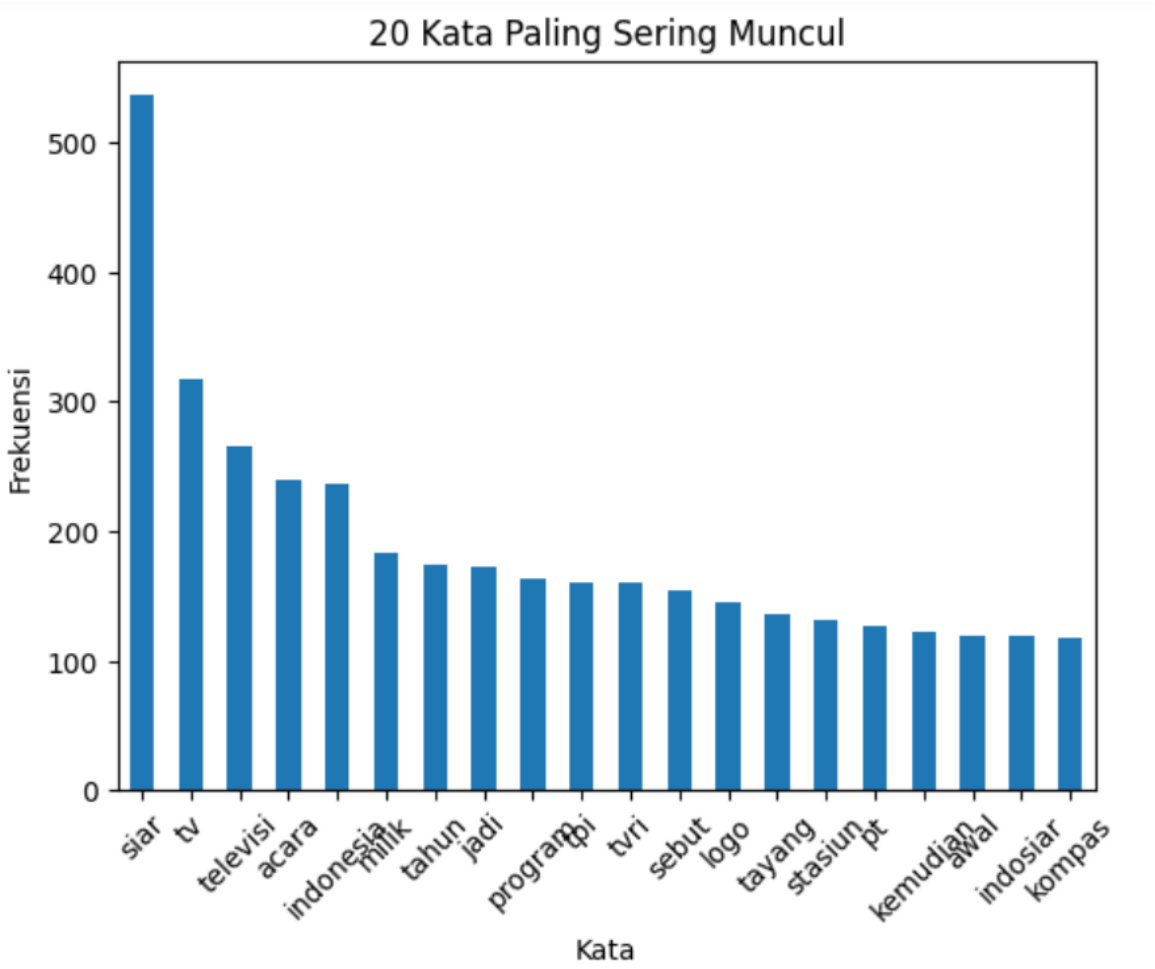
Terlihat sama seperti wordcloud bahwa kata ‘siar’, ‘televisi’, ‘acara’, dan ‘indonesia’ menduduki peringkat teratas. Berikut adalah kode untuk membuat bar plot 20 kata dengan frekuensi teratas.

```
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_top_20 = df_freq_tokens.sort_values(by='Frequency', ascending=False).head(20)

df_top_20.plot(kind='bar', legend=False)
plt.title('20 Kata Paling Sering Muncul')
plt.ylabel('Frekuensi')
plt.xlabel('Kata')
plt.xticks(rotation=45)
plt.show()
```



Dari bar plot terlihat jelas kata yang paling banyak disebut dan frekuensinya. Kata ‘siar’ disebutkan lebih dari 500 kali, kata ‘tv’ lebih dari 300 kali, dan kata ‘televisi’ antara 250 hingga 300 kali.

6. Lakukan clustering dengan menggunakan fitur TF-IDF. NIM 29 mendapatkan metode Hierarchical Clustering.
   Clustering dimulai dengan menghitung TF-IDF Vectorization.

TF-IDF Vectorization

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(hasil_stem)
```

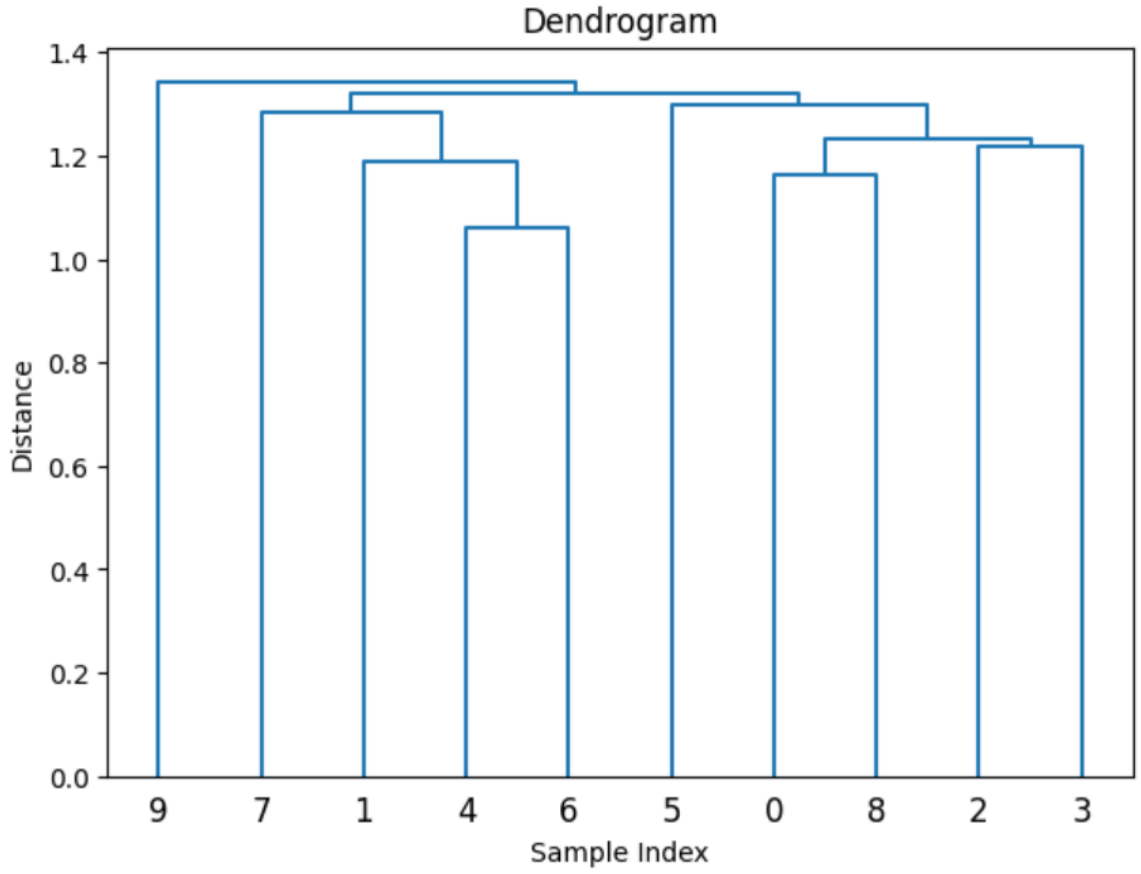
TF-IDF atau Term Frequency-Inverse Document Frequency merupakan penghitungan statistik yang digunakan dalam analisis teks yang menghitung frekuensi dan pentingnya suatu kata dalam dokumen atau teks.

7. Buat visualisasi clusternya dan lakukan interpretasi terhadap hasil tersebut
   Visualisasi cluster digunakan dengan membuat Dendogram.

```
from scipy.cluster.hierarchy import dendrogram, linkage
import matplotlib.pyplot as plt

# Create a linkage matrix
linked = linkage(tfidf_matrix.toarray(), method='ward')

# Plot the dendrogram
plt.figure(figsize=(7,5))
dendrogram(linked)
plt.title('Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()
```



Cluster dapat dibagi menjadi 9 cluster berdasarkan dendogram di atas.

Clustering kemudian dilanjutkan dengan label menggunakan kode berikut.

```
from sklearn.cluster import AgglomerativeClustering

# Define the number of clusters
num_clusters = 9

# Initialize AgglomerativeClustering with the number of clusters
hierarchical_cluster = AgglomerativeClustering(n_clusters=num_clusters, metric='euclidean', linkage='ward')

# Fit the model and predict cluster labels
cluster_labels = hierarchical_cluster.fit_predict(tfidf_matrix.toarray())

# Display the cluster labels
print("Cluster Labels:")
print(cluster_labels)
```

```
Cluster Labels:
[8 7 6 5 0 3 0 4 2 1]
```

8. Gunakan validasi menggunakan salah satu Davies-Bouldin index atau Silhouette score
   Validasi dilakukan menggunakan Davies-Bouldin Index atau DBI.

```
from sklearn.metrics import davies_bouldin_score

# Menghitung Davies-Bouldin index
db_index = davies_bouldin_score(tfidf_matrix.toarray(), cluster_labels)
print(f"Davies-Bouldin Index: {db_index}")
```

```
Davies-Bouldin Index: 0.48917432814279305
```

Nilai DBI ditunjukkan dengan semakin kecil dan mendekati nol, maka semakin baik nilai clustering. Di sini DBI menunjukkan nilai 0.489 yang lumayan mendekati nol. Bentuk cluster dinyatakan sudah baik meskipun dapat ditingkatkan lebih baik lagi.

WordCloud per Cluster

Kode yang digunakan

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

# Gabungkan semua judul artikel dalam cluster menjadi satu string
cluster_text = ' '.join(clusters[0])

# Buat WordCloud dari gabungan teks
wordcloud = WordCloud(background_color="white").generate(cluster_text)

# Plot the WordCloud
plt.figure(figsize=(12, 12))
plt.imshow(wordcloud, interpolation='bilinear')

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

Cluster Pertama

Cluster Kedua

Cluster Ketiga

Cluster Keempat

Cluster Kelima

Cluster Keenam

Cluster Ketujuh

Cluster Kedelapan

Cluster Kesembilan