Tugas Praktikum Natural Language Processing

Nama : Gavrilla Claudia

NIM : 21110004

Kelas : S1SD02A

```
pip install regex matplotlib sastrawi xlsxwriter nltk
```

```
Requirement already satisfied: regex in /usr/local/lib/python3.10/dist-packages (2023
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: sastrawi in /usr/local/lib/python3.10/dist-packages (1
Requirement already satisfied: xlsxwriter in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (fro
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
```

```
#import modul
import pandas as pd
import numpy as np
import re as reg
import matplotlib.pyplot as plt
%matplotlib inline


data = pd.read_csv('Jungkook.csv')
data
```

	Unnamed: 0	created_at	id_str	full_text	quote_count	repl
0	0	Wed Oct 04 03:23:42 +0000 2023	1709408926565826963	#JungKook_GOLDEN 👇👇	0	
1	1	Wed Oct 04 03:23:42 +0000 2023	1709408926435889254	This announcement doesn’t say anything about a...	0	
2	2	Wed Oct 04 03:23:40 +0000 2023	1709408921524281824	Inspired by the golden moments of #JungKook, h...	0	
3	3	Wed Oct 04 03:23:40 +0000 2023	1709408921104908629	Inspired by the golden moments of #JungKook, h...	0	
4	4	Wed Oct 04 03:23:40 +0000 2023	1709408920123683144	방탄소년단 방탄 버터 가디건 포카 양도 bts butter cardigan mini...	0	
...	...	...	...	...	...	
712	712	Wed Oct 04 03:15:40 +0000 2023	1709406906966577280	Noturnas CHARTMYS no SPF x3 e na AM rodando, ...	0	
713	713	Wed Oct 04 03:15:40 +0000 2023	1709406905230373215	Jungkook GOLEDN album unopen sealed sell 정국 골...	0	
714	714	Wed Oct 04 03:15:39 +0000 2023	1709406903237743058	LISTOO TAEHYUNG LE HACE EL WATER CHALLENGE A J...	0	
715	715	Wed Oct 04 03:15:39	1709406903141642430	jungkook bobo	0	

```
#Preprocessing
import re as reg
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

slangs={'yg':'yang', 'tdk':'tidak', 'pd':'pada', 'mlh':'malah', 'jgn':'jangan', 'jg':'jug
        'dlm':'dalam','dgn':'dengan', 'poto':'foto', 'g':'tidak', 'n':'dan', 'ad':'ada',
        "ahaha": "haha", "aj": "saja", "ajep-ajep": "dunia gemerlap", "ak": "saya", "akik
        "ancur": "hancur", "anjrit": "anjing", "anter": "antar", "ap2": "apa-apa", "apasi
        "aseekk": "asyik", "asekk": "asyik", "asem": "asam", "aspal": "asli tetapi palsu"
        "ayank": "sayang", "b4": "sebelum", "bakalan": "akan", "bandes": "bantuan desa",
        "bcanda": "bercanda", "bdg": "bandung", "begajulan": "nakal", "beliin": "belikan"
        "bosan", "beud": "banget", "bg": "abang", "bgmn": "bagaimana", "bgt": "banget", "
        "blegug": "bodoh", "blh": "boleh", "bln": "bulan", "blum": "belum", "bnci": "benc
        "bohong", "boljug": "boleh juga", "bonek": "bocah nekat", "boyeh": "boleh", "br":
        "bt": "buat", "btw": "ngomong-ngomong", "buaya": "tidak setia", "bubbu": "tidur",
        "cabal": "sabar", "cadas": "keren", "calo": "makelar", "can": "belum", "capcus":
        "cengengesan": "tertawa", "cepet": "cepat", "cew": "cewek", "chuyunk": "sayang",
        "ckp": "cakep", "cmiiw": "correct me if i'm wrong", "cmpur": "campur", "cong": "b
        "cucok": "cocok", "cuex": "cuek", "cumi": "Cuma miscall", "cups": "culun", "curan
        "d": "di", "dah": "deh", "dapet": "dapat", "de": "adik", "dek": "adik", "demen":
        "dimintak": "diminta", "disono": "di sana", "dket": "dekat", "dkk": "dan kawan-ka
        "dongs": "dong", "dpt": "dapat", "dri": "dari", "drmn": "darimana", "drtd": "dari
        "egp": "emang gue pikirin", "eke": "aku", "elu": "kamu", "emangnya": "memangnya",
        "fifo": "first in first out", "folbek": "follow back", "fyi": "sebagai informasi"
        "gan": "juragan", "gaptek": "gagap teknologi", "gatek": "gagap teknologi", "gawe"
        "gepeng": "gelandangan dan pengemis", "ghiy": "lagi", "gile": "gila", "gimana": "
        "gn": "begini", "goblok": "bodoh", "golput": "golongan putih", "gowes": "mengayuh
        "gua": "saya", "guoblok": "goblok", "gw": "saya", "ha": "tertawa", "haha": "terta
        "hlm": "halaman", "hny": "hanya", "hoax": "isu bohong", "hr": "hari", "hrus": "ha
        "ilfil": "tidak suka", "imho": "in my humble opinion", "imoetz": "imut", "item":
        "jayus": "tidak lucu", "jdi": "jadi", "jem": "jam", "jga": "juga", "jgnkan": "jan
        "jutek": "galak", "k": "ke", "kab": "kabupaten", "kabor": "kabur", "kacrut": "kac
        "kamtibmas": "keamanan dan ketertiban masyarakat", "kamuwh": "kamu", "kanwil": "k
        "kayanya": "kayaknya", "kbr": "kabar", "kdu": "harus", "kec": "kecamatan", "kejur
        "kepengen": "mau", "kepingin": "mau", "kepsek": "kepala sekolah", "kesbang": "kes
        "kibul": "bohong", "kimpoi": "kawin", "kl": "kalau", "klianz": "kalian", "kloter"
        "knp": "kenapa", "kodya": "kota madya", "komdis": "komisi disiplin", "komsov": "k
        "kpn": "kapan", "krenz": "keren", "krm": "kirim", "kt": "kita", "ktmu": "ketemu",
        "lam": "salam", "lamp": "lampiran", "lanud": "landasan udara", "latgab": "latihan
        "lgsg": "langsung", "liat": "lihat", "litbang": "penelitian dan pengembangan", "l
        "lp": "lupa", "luber": "langsung, umum, bebas, dan rahasia", "luchuw": "lucu", "l
        "kptsan":"keputusan", "krik": "garing", "krn": "karena", "ktauan": "ketahuan", "k
        "lambreta": "lambat", "lansia": "lanjut usia", "lapas": "lembaga pemasyarakatan",
        "linmas": "perlindungan masyarakat", "lmyan": "lumayan", "lngkp": "lengkap", "loc
        "luchu": "lucu", "luff": "cinta", "luph": "cinta", "lw": "kamu", "lwt": "lewat",
        "maen": "main", "mahatma": "maju sehat bersama", "mak": "ibu", "makasih": "terima
        "markus": "makelar kasus", "mba": "mbak", "mending": "lebih baik", "mgkn": "mungk
        "mnt": "minta", "moge": "motor gede", "mokat": "mati", "mosok": "masa", "msh": "m
        "mumet": "pusing", "muna": "munafik", "munaslub": "musyawarah nasional luar biasa
        "naon": "apa", "napol": "narapidana politik", "naq": "anak", "narsis": "bangga pa
        "nelfon": "menelepon", "ngabis2in": "menghabiskan", "ngakak": "tertawa", "ngambek
        "ngaruh": "berpengaruh", "ngawur": "berbicara sembarangan", "ngeceng": "kumpul ba
        "ngemeng": "bicara terus-terusan", "ngerti": "mengerti", "nggak": "tidak", "ngiku
        "ngomongin": "membicarakan", "ngumpul": "berkumpul", "ni": "ini", "nyasar": "ters
```

```
        "ok": "ok", "priksa": "periksa", "pro": "profesional", "psn": "pesan", "psti": "p
        "red": "redaksi", "reg": "register", "rejeki": "rezeki", "renstra": "rencana stra
        "sosbud": "sosial-budaya", "sospol": "sosial-politik", "sowry": "maaf", "spd": "s
        "sumbangin": "sumbangkan", "sy": "saya", "syp": "siapa", "tabanas": "tabungan pem
        "tekor": "rugi", "telkom": "telekomunikasi", "telp": "telepon", "temen2": "teman-
        "thd": "terhadap", "thx": "terima kasih", "tipi": "TV", "tkg": "tukang", "tll": "
        "tnda": "tanda", "tnh": "tanah", "togel": "toto gelap", "tp": "tapi", "tq": "teri
        "reklamuk": "reklamasi", "sma": "sama", "tren": "trend", "ngehe": "kesal", "mz":
        "zonk": "bodoh", "rights": "benar", "simiskin": "miskin", "ngumpet": "sembunyi",
        "masy": "masyarakat", "still": "masih", "tauk": "tahu", "mbual": "bual", "tioghoa
        "rubahnn": "rubah", "trlalu": "terlalu", "nyela": "cela", "heters": "pembenci", "
        "setting": "atur", "seting": "akting", "next": "lanjut", "waspadalah": "waspada",
        "jentelmen": "berani", "buangbuang": "buang", "tsangka": "tersangka", "kurng": "k
        "tahi": "kotoran", "tirani": "tiran", "tilep": "tilap", "happy": "bahagia", "tak"
        "taik": "tahi", "wkwkkw": "tertawa", "ahokncc": "ahok", "istaa": "nista", "benarj
        'ny':'nya', 'htm':'harga tiket masuk', 'cm':'cuma', 'slalu':'selalu', 'tingi':'ti
processed_comments = []

for sentence in data['full_text']:
    # Remove all the special characters
    processed_comment = reg.sub(r'\W', ' ', str(sentence))

    # Converting to Lowercase
    processed_comment = processed_comment.lower()

    #Remove number
    processed_comment = reg.sub(r'\d+', ' ', processed_comment)

    # remove all single characters
    processed_comment = reg.sub(r'\s+[a-zA-Z]\s+', ' ', processed_comment)

    #remove duplicate character
    pattern=reg.compile(r"(.)\1{1,}",reg.DOTALL)
    processed_comment=pattern.sub(r"\1",processed_comment)


    #Corrected Slang words
    words = processed_comment.split()
    rfrm=[slangs[word] if word in slangs else word for word in words]
    processed_comment= " ".join(rfrm)

    #remove stopword
    factory = StopWordRemoverFactory()
    more_stopword = ['tak', 'jd', 'per', 'nya', 'terjemah', 'diterjemahkan', 'oleh', 'gog
                     'ada','bersih', 'salur', 'baru', 'purwokerto', 'batas', 'hotel',
                     'com', 'kamu', 'http', 'https', 'htps', 'htp', 'gak', 'jadi', 'le
                     'apa', 'paling', 'semua', 'lah', 'ihwm', 'od', 'pakai', 'hayo', '
    stopwords = factory.get_stop_words() + more_stopword
    temp = [t for t in reg.findall(r'\b[a-z]+-?[a-z]+\b',processed_comment) if t not in s
    processed_comment = ' '.join(temp)

    #stemming
    stemmer = StemmerFactory().create_stemmer()
    processed_comment = stemmer.stem(processed_comment)
```

```
    # Substituting multiple spaces with single space
    processed_comment = reg.sub(r'\s+', ' ', processed_comment, flags=reg.I)

    processed_comments.append(processed_comment)

#output data Preprocessing
processed_comments
```

```
['',
 'this anouncement doesn say anything about special guest and it doesn have photo
that says it with jungkok like the last one think these are truly just listening
parties to bost the song',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'bts buter cardigan mini photocard phoca pc rm jin hope jimin jk jungkok md co
fiom zpms',
 'drank litle to much wine and now im tipsy but just remembered have date with
jungkok at am tmrw co nd xyvcxat',
 'golden is coming jungkok solo album golden by jungkok is coming goldenbyjk',
 'golden is coming jungkok solo album golden by jungkok is coming goldenbyjk',
 'dreamjeons signed up on weverse for jungkok golden autographed poster rafle
lucky winers maybe be one of them beginers luck',
 'pre order jungkok solo album golden weverse ver random set set dm co
vtsurwelek',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'iamstef dbyjungkok co ainruauwo',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
november rd pm kst golden by jungkok is coming',
 'al this things don work out you know you should take an imbecile you know
personaly to the pastor and bom watch god and the pastor not in action is god so
stupid to create her that way as you al say bluejays napolirealmadrid',
 'cari po album jungkok non profit po start dp',
 'jungkok knows how to make it royal co aedrh cr',
 '',
 'mozi jungkok dbyjungkok co fwtd tm',
 'to preparada golden is coming jungkok solo album golden by jungkok is coming
goldenbyjk co dmc mzj ml',
 'euphoriasluna it okay everyone makes mistakes',
 'favojekey goldenjk bismilah listening to song solo single by jungkok of feat
jackharlow co do dp xv',
 'omg jungkok album',
 'inspired by the golden moments of jungkok his solo album golden is coming pre
save and pre order now the new album by the golden giant pop star coming out
```

```
     november rd pm kst golden by jungkok is coming',
      'eu bia tendo surtos do album do jungkok',
      'inspired by the golden moments of jungkok his solo album golden is coming pre
     save and pre order now the new album by the golden giant pop star coming out
     november rd pm kst golden by jungkok is coming',
      'se viene album adik jungkok se viene arte',
      'jungkok xylitol bts co wnm smile smile',
```

```
import nltk
nltk.download('punkt')
```

```
     [nltk_data] Downloading package punkt to /root/nltk_data...
     [nltk_data]   Unzipping tokenizers/punkt.zip.
     True
```

```
processed_comments = ''.join(processed_comments)
```

```
type(processed_comments)
```

```
     str
```

```
from nltk.tokenize import word_tokenize
Docs = processed_comments
Sentences = nltk.sent_tokenize(Docs)
print ("Kalimat Awal: ", Sentences)
hasil_tokenizing = nltk.word_tokenize(Docs)
print("Setelah Tokenizing: ", hasil_tokenizing)
```

```
     Kalimat Awal:  ['this anouncement doesn say anything about special guest and it doesn
     Setelah Tokenizing:  ['this', 'anouncement', 'doesn', 'say', 'anything', 'about', 'sp
```

```
len(hasil_tokenizing)
```

```
     12559
```

```
#Ekstraksi fitur / representasi dokumen menggunakan TF
from sklearn.feature_extraction.text import CountVectorizer
tf_vectorizer = CountVectorizer(max_df=1.0, min_df=1)

tf = tf_vectorizer.fit_transform(hasil_tokenizing)

#hasil representasi
tf_terms = tf_vectorizer.get_feature_names_out()
print(tf_vectorizer.get_feature_names_out())
matrix = tf.toarray()
print(matrix)
```

```
     ['abangshould' 'ability' 'able' ... 'zrvpwhkgmhay' 'zscjtpkeq' 'zudlqx']
     [[0 0 0 ... 0 0 0]
      [0 0 0 ... 0 0 0]
      [0 0 0 ... 0 0 0]
      ...
```

```
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

Menggunakan LDA

```
from sklearn.decomposition import LatentDirichletAllocation as LDA

n_topics = 10 #untuk mendapatkan jumlah topik terbaik perlu trial
lda = LDA(n_components=n_topics, learning_method='batch', random_state=0)
lda.fit(tf)
lda
```

```
LatentDirichletAllocation(random_state=0)
```

```
vsm_topics = lda.transform(tf)
#tampilkan hasil
print(vsm_topics)
```

```
[[0.05       0.05       0.05       ... 0.05       0.05       0.55      ]
 [0.05       0.55       0.05       ... 0.05       0.05       0.05      ]
 [0.05       0.05       0.05       ... 0.05       0.05       0.05      ]
 ...
 [0.05       0.05       0.05       ... 0.54999999 0.05       0.05      ]
 [0.05       0.05       0.05       ... 0.05       0.05       0.05      ]
 [0.05       0.05       0.05       ... 0.05       0.05       0.05      ]]
```

```
len(vsm_topics)
```

```
12559
```

```
#Tampilkan nilai-nilai setiap fitur
print(lda.components_)
```

```
[[1.09999998 0.1        1.09999998 ... 0.1        0.1        0.1       ]
 [0.1        1.09999997 0.1        ... 0.1        0.1        0.1       ]
 [0.1        0.1        0.1        ... 0.1        0.1        0.1       ]
 ...
 [0.1        0.1        0.1        ... 0.1        0.1        0.1       ]
 [0.1        0.1        0.1        ... 0.1        1.09999997 0.1       ]
 [0.1        0.1        0.1        ... 0.1        0.1        0.1       ]]
```

```
#hasil label topic model untuk setiap dokumen
import numpy as np
topics = np.argmax(vsm_topics, axis=1)
topics
```

```
array([9, 1, 6, ..., 7, 3, 3])
```

```
#mencetak word fitur dengan nilai tertinggi pada setiap topik
n_top_words = 10 # jumlah fitur tertinggi yang kita tentukan
topic_words = {}
for topic, comp in enumerate(lda.components_):
 word_idx = np.argsort(comp)[::-1][:n_top_words] # susun indeks secara terbalik, sehingga
 # store the words most relevant to the topic
 topic_words[topic] = [tf_vectorizer.get_feature_names_out()[i]+' '+str(comp[i]) for i in
```

```
type(topic_words)
```

```
dict
```

```
topic_words
```

```
{0: ['is 349.09999998602297',
  'his 144.09999998599756',
  'rd 132.09999998599432',
  'kst 130.09999998599335',
  'el 47.09999998591712',
  'cominginspired 32.09999998586052',
  'amp 24.099999985801077',
  'that 18.099999985721254',
  'have 17.09999998570238',
  'ems 16.099999985681098'],
 1: ['golden 674.0999999796329',
  'the 471.0999999796179',
  'now 139.0999999795778',
  'out 134.09999997957627',
  'star 132.09999997957493',
  'bts 66.09999997951284',
  'me 42.09999997944153',
  'est 42.09999997944153',
  'stu 11.099999978876241',
  'del 11.099999978876241'],
 2: ['coming 323.0999999880826',
  'order 158.0999999880663',
  'for 51.09999998800221',
  'ver 47.0999999879941',
  'jk 20.099999987854453',
  'be 20.09999998785445',
  'live 15.099999987772335',
  'las 13.099999987721397',
  'or 11.09999998765145',
  'stationhead 11.099999987651449'],
 3: ['jungkok 602.0999999842757',
  'new 132.09999998424328',
  'pm 131.09999998424337',
  'moments 129.09999998424271',
  'rasa 42.099999984140574',
  'so 36.099999984115314',
  'xylitol 28.099999984064496',
  'po 25.099999984036952',
  'with 22.09999998400179',
  'con 21.09999998398781'],
 4: ['by 443.099999989426',
  'set 67.09999998937177',
  'al 21.099999989231073',
```

```
 'pob 16.099999989166115',
 'get 12.099999989074226',
 'just 12.099999989074226',
 'eu 11.09999998904054',
 'jung 10.099999988999917',
 'thank 10.099999988999917',
 'mi 9.099999988949978'],
5: ['save 134.09999999101524',
 'dp 76.09999999099479',
 'smile 28.099999990912707',
 'do 27.099999990907893',
 'he 20.09999999086022',
 'jimin 12.09999999073601',
 'at 11.099999990707444',
```

```
for topic, words in topic_words.items():
 print('Topic: %d' % topic)
 print(' %s' % ', '.join(words))
```

```
Topic: 0
 is 349.09999998602297, his 144.09999998599756, rd 132.09999998599432, kst 130.099999
Topic: 1
 golden 674.0999999796329, the 471.0999999796179, now 139.0999999795778, out 134.0999
Topic: 2
 coming 323.0999999880826, order 158.0999999880663, for 51.09999998800221, ver 47.099
Topic: 3
 jungkok 602.0999999842757, new 132.09999998424328, pm 131.09999998424337, moments 12
Topic: 4
 by 443.099999989426, set 67.09999998937177, al 21.099999989231073, pob 16.0999999891
Topic: 5
 save 134.09999999101524, dp 76.09999999099479, smile 28.099999990912707, do 27.09999
Topic: 6
 adik 105.0999999911746, que 62.099999991151236, in 33.09999999110073, it 28.09999999
Topic: 7
 co 255.09999998873346, pop 131.09999998871714, weverse 71.0999999886882, en 39.09999
Topic: 8
 album 430.099999983539, pre 274.0999999835272, solo 201.09999998352058, november 136
Topic: 9
 and 177.09999998891155, of 168.09999998891058, giant 129.0999999889025, on 45.099999
```

Interpretasi hasil :

Misal topic 0, didalamnya terdapat kata is, his, rd, kst, el, dsb dengan probabilitas masing - masing. Di mana kita dapat melihat bahwa kata is memiliki nilai 349.09. Artinya bahwa kata "is" dalam topik tersebut memiliki distrusi probalitas paling tinggi.

Menggunakan LSA

```
from sklearn.feature_extraction.text import TfidfVectorizer
vektor = TfidfVectorizer(max_features=400)
```

```
df = pd.read_csv('Jungkook.csv')
```

```
df.head()
```

	Unnamed: 0	created_at	id_str	full_text	quote_count	reply_
0	0	Wed Oct 04 03:23:42 +0000 2023	1709408926565826963	#JungKook_GOLDEN 👇👇	0	
1	1	Wed Oct 04 03:23:42 +0000 2023	1709408926435889254	This announcement doesn’t say anything about a...	0	
2	2	Wed Oct 04 03:23:40 +0000 2023	1709408921524281824	Inspired by the golden moments of #JungKook, h...	0	
3	3	Wed Oct 04 03:23:40 +0000 2023	1709408921104908629	Inspired by the golden moments of #JungKook, h...	0	
4	4	Wed Oct 04 03:23:40 +0000 2023	1709408920123683144	방탄소년단 방탄 버터 가디건 포카 양도 bts butter cardigan mini...	0	

```
df = df['full_text']
```

```
df = pd.DataFrame(df)
```

```
df.head()
```

	full_text
0	#JungKook_GOLDEN 👇👇
1	This announcement doesn’t say anything about a...
2	Inspired by the golden moments of #JungKook, h...
3	Inspired by the golden moments of #JungKook, h...
4	방탄소년단 방탄 버터 가디건 포카 양도 bts butter cardigan mini...

```
#menghitung tf-idf dengan TfidfTransformer
vektor_dt=vektor.fit_transform(df['full_text'].values.astype('U'))
print (vektor_dt)
print (vektor_dt.shape)
```

```
  (0, 171)      1.0
  (1, 290)      0.24880749992040554
  (1, 319)      0.1599302330861236
  (1, 187)      0.25485142252629095
```

```
  (1, 173)      0.23045526315699844
  (1, 43)       0.22347806655769126
  (1, 231)      0.23870753520321114
  (1, 312)      0.21087158639027384
  (1, 186)      0.22347806655769126
  (1, 169)      0.05891812793835312
  (1, 360)      0.20100824833281278
  (1, 311)      0.21210302639359133
  (1, 142)      0.2146901410992338
  (1, 159)      0.3842094211781831
  (1, 41)       0.10994072015986994
  (1, 292)      0.2618286191255981
  (1, 26)       0.22347806655769126
  (1, 42)       0.25485142252629095
  (1, 271)      0.27008089117181083
  (1, 317)      0.20512582979428412
  (2, 179)      0.13217245317085524
  (2, 11)       0.13217245317085524
  (2, 20)       0.13217245317085524
  (2, 224)      0.13033966569908154
  (2, 237)      0.13070077216944584
  :     :
  (713, 273)    0.6176929086246765
  (713, 275)    0.6371612537582619
  (713, 232)    0.2636091069930398
  (713, 75)     0.11123344150475804
  (713, 149)    0.111453607322724
  (713, 399)    0.25992556825663293
  (713, 34)     0.21402135498711194
  (713, 169)    0.0694983419660179
  (714, 303)    0.5550985211319187
  (714, 183)    0.49061671560486164
  (714, 102)    0.3593560105313779
  (714, 175)    0.4817502401271333
  (714, 75)     0.1938144182983579
  (714, 149)    0.19419803773295416
  (714, 169)    0.12109470442185268
  (715, 169)    1.0
  (716, 315)    0.458225479011412
  (716, 141)    0.47077055026187564
  (716, 340)    0.458225479011412
  (716, 45)     0.43777303979682997
  (716, 75)     0.16955165822224283
  (716, 149)    0.16988725405578883
  (716, 157)    0.19000409743682486
  (716, 169)    0.21187080008723413
  (716, 171)    0.16789412894375624
(717, 400)
```

```
# topic modeling using LSA
from sklearn.decomposition import TruncatedSVD
lsa_model = TruncatedSVD(n_components=5, n_iter=10, random_state=42)

lsa_top=lsa_model.fit_transform(vektor_dt)
print(lsa_top)
```

```
[[ 0.14465356  0.26529895 -0.60592495 -0.46799927 -0.49998212]
 [ 0.10226278  0.05280024  0.0016908   0.08483551  0.05467641]
```

```
       [ 0.990713   -0.08935521  0.03982456 -0.04034218  0.0492801 ]
       ...
       [ 0.03863553  0.27404647 -0.0500248   0.1279869   0.10786427]
       [ 0.19149988  0.31454389  0.03473359  0.63961924 -0.43348463]
       [ 0.12807024  0.29936246 -0.14213118  0.05559805 -0.05364207]]
```

```
len(lsa_top)
```

```
     717
```

```
lsa_model.get_params()
```

```
     {'algorithm': 'randomized',
      'n_components': 5,
      'n_iter': 10,
      'n_oversamples': 10,
      'power_iteration_normalizer': 'auto',
      'random_state': 42,
      'tol': 0.0}
```

```
# Memunculkan nilai lsa setiap topik
l=lsa_top[0]
print("Topik- Topik:")
for i,topic in enumerate(l):
  print("Topic ",i," : ",topic*100)
```

```
     Topik- Topik:
     Topic  0  :  14.465355726370488
     Topic  1  :  26.52989479563579
     Topic  2  :  -60.59249511173618
     Topic  3  :  -46.79992699805632
     Topic  4  :  -49.99821159914404
```

```
# Memunculkan jumlah kata-kata dalam setiap topik
print(lsa_model.components_.shape)
print(lsa_model.components_)
```

```
     (5, 400)
     [[ 0.00113542  0.00116164  0.00175317 ...  0.00180275  0.00180275
        0.00677111]
      [ 0.00670117  0.00652341  0.01849782 ...  0.12024669  0.12024669
        0.05651562]
      [-0.00367943 -0.00178049 -0.00357358 ...  0.22034709  0.22034709
       -0.04746391]
      [ 0.00066557  0.00580704  0.01207578 ... -0.10881123 -0.10881123
        0.00913752]
      [ 0.00517642  0.0088676   0.01950038 ... -0.09243465 -0.09243465
       -0.02233595]]
```

```
# Word/ kata paling penting dalam setiap topik
vocab = vektor.get_feature_names_out()
for i, comp in enumerate(lsa_model.components_):
```