222112058_Feza Raffa Arnanda_Penugasan 6

October 7, 2023

[ ]:
```
path = "C:/Users/FEZA/My Drive/00. Drive PC/1.STIS/5. Semester 5/Information␣
↪Retrieval [IR] P/berita"
```

Casefolding

[ ]:
```
def case_folding(text):
    text = text.lower()
    return text
```

Tokenisasi

[ ]:
```
import nltk
# nltk.download('punkt') # Download data yang diperlukan untuk tokenisasi
from nltk.tokenize import word_tokenize
def tokenisasi(text):
    tokens = word_tokenize(text)
    return tokens
```

Eliminasi Stopword

[ ]:
```
from nltk.corpus import stopwords
stop_words = set(stopwords.words('Indonesian'))
```

[ ]:
```
def eliminasi_stopword(token):
    return [kata for kata in token if kata not in stop_words]
```

Stemming

[ ]:
```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
def stemming_sastrawi(tokens):
    # Membuat stemmer
    factory = StemmerFactory()
    stemmer = factory.create_stemmer()
    return [stemmer.stem(token) for token in tokens]
```

[ ]:
```
import re
import os
inverted_index = {}
doc_dict = {}
i = 1
```

```
for filename in os.listdir(path):
    if (filename.endswith('.txt')):
        file_path = os.path.join(path, filename)
        # Ekstrak angka dari nama file menggunakan regular expressions
        match = re.search(r'\d+', filename)
        if match:
            doc_id = match.group() # Mengambil angka dari nama file sebagai␣
 ↪dokumen ID
            with open (file_path, mode='r', encoding='utf-8') as file:
                text = file.read()
                hasil_case_folding = case_folding(text)
                token = tokenisasi(hasil_case_folding)
                token_bersih = eliminasi_stopword(token)
                stemm_token = stemming_sastrawi(token_bersih)
                stemm_token_final = [item for item in stemm_token if item !=␣
 ↪''] # membersihkan term kosong pada hasil stemming sebelumnya
                # Menggabungkan hasil stemming menjadi sebuah teks/paragraf
                doc_dict[doc_id] = ' '.join(stemm_token_final)
                for term in set(stemm_token_final): # penggunaan set untuk␣
 ↪mengantisipasi duplikasi term pada sebuah dokumen
                    if term in inverted_index:
                        inverted_index[term].append(doc_id)
                    else:
                        inverted_index[term] = [doc_id]
```

[ ]:
```
inverted_index
```

[ ]:
```
{'signifikan': ['1', '4'],
 'protokol': ['1'],
 'hijau': ['1'],
 '24': ['1'],
 'cek': ['1'],
 'jakarta': ['1', '2', '3', '4', '5'],
 'siti': ['1'],
 'wilayah': ['1', '5'],
 'nadia': ['1'],
 'laku': ['1'],
 'hitung': ['1'],
 'rencana': ['1', '2'],
 'terap': ['1'],
 'masyarakat': ['1'],
 'direktur': ['1'],
 'cegah': ['1'],
 'dr': ['1', '5'],
 'tingkat': ['1', '4'],
 'januari': ['1', '2'],
 'tular': ['1'],
```

```
 'langsung': ['1'],
 'zona': ['1'],
 '-': ['1', '2', '3', '4', '5'],
 '3': ['1'],
 'picu': ['1'],
 'longgar': ['1'],
 'kaji': ['1'],
 '34': ['1'],
 'kendali': ['1'],
 'baru': ['1', '3', '4'],
 'giat': ['1'],
 'bebas': ['1'],
 'kemenkes': ['1', '4'],
 'desember': ['1'],
 'perintah': ['1'],
 'sehat': ['1'],
 'batas': ['1'],
 'kab kota': ['1'],
 'mobilitas': ['1'],
 'p2pml': ['1'],
 'menteri': ['1'],
 'sakit': ['1'],
 'ppkm': ['1'],
 'bijak': ['1'],
 'https': ['1', '2', '3', '4', '5'],
 '2': ['1', '3'],
 'tahap': ['1'],
 'health detik com berita-detikhealth d-5816690 wilayah-kamu-sudah-bebas-
covid-19-cek-34-kabkota-zona-hijau-terbaru': ['1'],
 'ri': ['1', '3', '4'],
 'level': ['1'],
 '2021': ['1'],
 'covid-19': ['1', '2', '3', '4', '5'],
 'tarmizi': ['1'],
 'prof': ['2', '3'],
 'lantas': ['2'],
 'djoerban': ['2', '3'],
 'vaksinasi': ['2'],
 'influenza': ['2'],
 'zubairi': ['2', '3'],
 'ada': ['2'],
 'ketua': ['2', '3'],
 'rutin': ['2'],
 'gantung': ['2'],
 'beri': ['2'],
 'tiga': ['2', '3'],
 'satgas': ['2', '3'],
```

```
 'dokter': ['2', '3'],
 'idi': ['2', '3'],
 'booster': ['2', '3'],
 'ikat': ['2', '3'],
 'jelas': ['2'],
 'indonesia': ['2', '3', '4'],
 'dosis': ['2', '3'],
 'kait': ['2', '3'],
 '2022': ['2', '3'],
 'health detik com berita-detikhealth d-5816582 vaksin-covid-19-bakal-rutin-
setiap-tahun-tergantung-ini-penjelasannya': ['2'],
 'vaksin': ['2', '3'],
 'pasti': ['2'],
 'turut': ['2'],
 'corona': ['3', '5'],
 '1-2': ['3'],
 'cs': ['3'],
 'bukti': ['3'],
 'strain': ['3'],
 'varian': ['3', '4', '5'],
 'efektivitas': ['3'],
 'singgung': ['3'],
 '1': ['3'],
 'aku': ['3'],
 'turun': ['3', '5'],
 'dasar': ['3'],
 'sebut': ['3'],
 'lawan': ['3'],
 'virus': ['3'],
 'ikut': ['3'],
 'alami': ['3', '4'],
 'jenis': ['3'],
 'pfizer': ['3'],
 'health detik com berita-detikhealth d-5816534 ri-mulai-suntikkan-booster-
di-2022-masihkah-ampuh-lawan-varian-delta-cs': ['3'],
 'suntik': ['3'],
 'riset': ['3'],
 'delta': ['3', '4', '5'],
 'moderna': ['3'],
 'ampuh': ['3'],
 'pakar': ['3'],
 '90': ['4'],
 'alpha': ['4'],
 'alert': ['4'],
 'sulawesi': ['4'],
 'november': ['4'],
 'balitbangkes': ['4'],
```

```
 'tambah': ['4'],
 'beta': ['4'],
 'jawa': ['4'],
 'health detik com berita-detikhealth d-5812940 alert-kasus-varian-delta-
covid-19-di-dki-meningkat': ['4'],
 'dki': ['4'],
 'data': ['4'],
 '13': ['4'],
 'utara': ['4'],
 'barat': ['4', '5'],
 '165': ['4'],
 'asal': ['4'],
 '86': ['4'],
 'total': ['4'],
 '1 327': ['4'],
 'medis': ['5'],
 'health detik com berita-detikhealth d-5813949 corona-di-as-mendadak-naik-lagi-
usai-serangan-delta-sempat-mereda': ['5'],
 'catat': ['5'],
 'serang': ['5'],
 'puncak': ['5'],
 'nasihat': ['5'],
 'serikat': ['5'],
 'stabil': ['5'],
 'area': ['5'],
 'timur': ['5'],
 'minggu': ['5'],
 'gedung': ['5'],
 'laut': ['5'],
 '57': ['5'],
 'anthony': ['5'],
 'panas': ['5'],
 'persen': ['5'],
 'reda': ['5'],
 'senin': ['5'],
 'amerika': ['5'],
 'kepala': ['5'],
 'musim': ['5'],
 'fauci': ['5'],
 '15 11 2021': ['5'],
 'dadak': ['5'],
 'pasien': ['5'],
 'as': ['5'],
 'gelombang': ['5'],
 'nasional': ['5'],
 'pasca': ['5'],
 'putih': ['5']}
```

[ ]:
```
doc_dict
```

[ ]:
```
{'1': 'wilayah bebas covid-19 cek 34 kab kota zona hijau baru jakarta - perintah
rencana terap laku batas giat masyarakat ppkm level 3 hitung 24 desember 2021 2
januari 2021 menteri sehat ri bijak ppkm level 3 tahap kaji direktur cegah
kendali sakit tular langsung p2pml kemenkes ri dr siti nadia tarmizi ppkm level
3 terap covid-19 signifikan picu tingkat mobilitas longgar protokol sehat https
health detik com berita-detikhealth d-5816690 wilayah-kamu-sudah-bebas-
covid-19-cek-34-kabkota-zona-hijau-terbaru',
 '2': 'vaksin covid-19 rutin gantung jelas jakarta - beri booster dosis tiga
vaksin covid-19 indonesia rencana januari 2022 lantas ada vaksinasi covid-19
vaksinasi influenza ketua satgas covid-19 ikat dokter indonesia idi prof zubairi
djoerban pasti kait turut vaksin covid-19 booster vaksinasi covid-19 https
health detik com berita-detikhealth d-5816582 vaksin-covid-19-bakal-rutin-
setiap-tahun-tergantung-ini-penjelasannya',
 '3': 'ri suntik booster 2022 ampuh lawan varian delta cs jakarta - pakar aku
vaksin dosis 1-2 alami turun efektivitas varian corona varian delta booster
dosis tiga vaksin covid-19 indonesia 2022 jenis vaksin ikut strain virus baru
ketua satgas covid-19 ikat dokter indonesia idi prof zubairi djoerban singgung
riset kait efektivitas vaksin covid-19 dosis 1 2 sebut dasar riset efektivitas
vaksin covid-19 pfizer moderna bukti turun lawan varian delta https health detik
com berita-detikhealth d-5816534 ri-mulai-suntikkan-booster-di-2022-masihkah-
ampuh-lawan-varian-delta-cs',
 '4': 'alert varian delta covid-19 dki tingkat jakarta - data baru balitbangkes
kemenkes ri 13 november tambah varian delta tambah jawa barat 165 dki jakarta 90
sulawesi utara 86 balitbangkes dki jakarta alami tingkat varian delta signifikan
varian varian alpha varian delta beta indonesia asal dki jakarta total 1 327
https health detik com berita-detikhealth d-5812940 alert-kasus-varian-delta-
covid-19-di-dki-meningkat',
 '5': 'corona as dadak serang delta reda jakarta - covid-19 wilayah amerika
serikat as covid-19 catat stabil pasca serang varian delta musim panas kepala
nasihat medis gedung putih dr anthony fauci senin 15 11 2021 nasional turun 57
persen minggu puncak gelombang varian delta musim panas pasien covid-19 area
barat timur laut dadak https health detik com berita-detikhealth d-5813949
corona-di-as-mendadak-naik-lagi-usai-serangan-delta-sempat-mereda'}
```

Vocabulary List dari Inverted Index

[ ]:
```
vocab = list(inverted_index.keys())
print(f'Vocabulary List: {vocab}')
```

```
Vocabulary List: ['signifikan', 'protokol', 'hijau', '24', 'cek', 'jakarta',
'siti', 'wilayah', 'nadia', 'laku', 'hitung', 'rencana', 'terap', 'masyarakat',
'direktur', 'cegah', 'dr', 'tingkat', 'januari', 'tular', 'langsung', 'zona',
'-', '3', 'picu', 'longgar', 'kaji', '34', 'kendali', 'baru', 'giat', 'bebas',
'kemenkes', 'desember', 'perintah', 'sehat', 'batas', 'kab kota', 'mobilitas',
'p2pml', 'menteri', 'sakit', 'ppkm', 'bijak', 'https', '2', 'tahap', 'health
detik com berita-detikhealth d-5816690 wilayah-kamu-sudah-bebas-
```

```
covid-19-cek-34-kabkota-zona-hijau-terbaru', 'ri', 'level', '2021', 'covid-19',
'tarmizi', 'prof', 'lantas', 'djoerban', 'vaksinasi', 'influenza', 'zubairi',
'ada', 'ketua', 'rutin', 'gantung', 'beri', 'tiga', 'satgas', 'dokter', 'idi',
'booster', 'ikat', 'jelas', 'indonesia', 'dosis', 'kait', '2022', 'health detik
com berita-detikhealth d-5816582 vaksin-covid-19-bakal-rutin-setiap-tahun-
tergantung-ini-penjelasannya', 'vaksin', 'pasti', 'turut', 'corona', '1-2',
'cs', 'bukti', 'strain', 'varian', 'efektivitas', 'singgung', '1', 'aku',
'turun', 'dasar', 'sebut', 'lawan', 'virus', 'ikut', 'alami', 'jenis', 'pfizer',
'health detik com berita-detikhealth d-5816534 ri-mulai-suntikkan-booster-
di-2022-masihkah-ampuh-lawan-varian-delta-cs', 'suntik', 'riset', 'delta',
'moderna', 'ampuh', 'pakar', '90', 'alpha', 'alert', 'sulawesi', 'november',
'balitbangkes', 'tambah', 'beta', 'jawa', 'health detik com berita-detikhealth
d-5812940 alert-kasus-varian-delta-covid-19-di-dki-meningkat', 'dki', 'data',
'13', 'utara', 'barat', '165', 'asal', '86', 'total', '1 327', 'medis', 'health
detik com berita-detikhealth d-5813949 corona-di-as-mendadak-naik-lagi-usai-
serangan-delta-sempat-mereda', 'catat', 'serang', 'puncak', 'nasihat',
'serikat', 'stabil', 'area', 'timur', 'minggu', 'gedung', 'laut', '57',
'anthony', 'panas', 'persen', 'reda', 'senin', 'amerika', 'kepala', 'musim',
'fauci', '15 11 2021', 'dadak', 'pasien', 'as', 'gelombang', 'nasional',
'pasca', 'putih']
```

Top 3 Document Retrieval

[ ]:
```
query = 'vaksin corona jakarta'
```

Membuat term frequency

[ ]:
```
def termFrequency(vocab, query): # term frequency berdasarkan query
    tf_query = {}
    for word in vocab:
        tf_query[word] = query.count(word)
    return tf_query

tf_query = termFrequency(vocab, query)
tf_query
```

[ ]:
```
{'signifikan': 0,
 'protokol': 0,
 'hijau': 0,
 '24': 0,
 'cek': 0,
 'jakarta': 1,
 'siti': 0,
 'wilayah': 0,
 'nadia': 0,
 'laku': 0,
 'hitung': 0,
 'rencana': 0,
```

```
 'terap': 0,
 'masyarakat': 0,
 'direktur': 0,
 'cegah': 0,
 'dr': 0,
 'tingkat': 0,
 'januari': 0,
 'tular': 0,
 'langsung': 0,
 'zona': 0,
 '-': 0,
 '3': 0,
 'picu': 0,
 'longgar': 0,
 'kaji': 0,
 '34': 0,
 'kendali': 0,
 'baru': 0,
 'giat': 0,
 'bebas': 0,
 'kemenkes': 0,
 'desember': 0,
 'perintah': 0,
 'sehat': 0,
 'batas': 0,
 'kab kota': 0,
 'mobilitas': 0,
 'p2pml': 0,
 'menteri': 0,
 'sakit': 0,
 'ppkm': 0,
 'bijak': 0,
 'https': 0,
 '2': 0,
 'tahap': 0,
 'health detik com berita-detikhealth d-5816690 wilayah-kamu-sudah-bebas-
covid-19-cek-34-kabkota-zona-hijau-terbaru': 0,
 'ri': 0,
 'level': 0,
 '2021': 0,
 'covid-19': 0,
 'tarmizi': 0,
 'prof': 0,
 'lantas': 0,
 'djoerban': 0,
 'vaksinasi': 0,
 'influenza': 0,
```

```
 'zubairi': 0,
 'ada': 0,
 'ketua': 0,
 'rutin': 0,
 'gantung': 0,
 'beri': 0,
 'tiga': 0,
 'satgas': 0,
 'dokter': 0,
 'idi': 0,
 'booster': 0,
 'ikat': 0,
 'jelas': 0,
 'indonesia': 0,
 'dosis': 0,
 'kait': 0,
 '2022': 0,
 'health detik com berita-detikhealth d-5816582 vaksin-covid-19-bakal-rutin-
setiap-tahun-tergantung-ini-penjelasannya': 0,
 'vaksin': 1,
 'pasti': 0,
 'turut': 0,
 'corona': 1,
 '1-2': 0,
 'cs': 0,
 'bukti': 0,
 'strain': 0,
 'varian': 0,
 'efektivitas': 0,
 'singgung': 0,
 '1': 0,
 'aku': 0,
 'turun': 0,
 'dasar': 0,
 'sebut': 0,
 'lawan': 0,
 'virus': 0,
 'ikut': 0,
 'alami': 0,
 'jenis': 0,
 'pfizer': 0,
 'health detik com berita-detikhealth d-5816534 ri-mulai-suntikkan-booster-
di-2022-masihkah-ampuh-lawan-varian-delta-cs': 0,
 'suntik': 0,
 'riset': 0,
 'delta': 0,
 'moderna': 0,
```

```
 'ampuh': 0,
 'pakar': 0,
 '90': 0,
 'alpha': 0,
 'alert': 0,
 'sulawesi': 0,
 'november': 0,
 'balitbangkes': 0,
 'tambah': 0,
 'beta': 0,
 'jawa': 0,
 'health detik com berita-detikhealth d-5812940 alert-kasus-varian-delta-
covid-19-di-dki-meningkat': 0,
 'dki': 0,
 'data': 0,
 '13': 0,
 'utara': 0,
 'barat': 0,
 '165': 0,
 'asal': 0,
 '86': 0,
 'total': 0,
 '1 327': 0,
 'medis': 0,
 'health detik com berita-detikhealth d-5813949 corona-di-as-mendadak-naik-lagi-
usai-serangan-delta-sempat-mereda': 0,
 'catat': 0,
 'serang': 0,
 'puncak': 0,
 'nasihat': 0,
 'serikat': 0,
 'stabil': 0,
 'area': 0,
 'timur': 0,
 'minggu': 0,
 'gedung': 0,
 'laut': 0,
 '57': 0,
 'anthony': 0,
 'panas': 0,
 'persen': 0,
 'reda': 0,
 'senin': 0,
 'amerika': 0,
 'kepala': 0,
 'musim': 0,
 'fauci': 0,
```

```
'15 11 2021': 0,
'dadak': 0,
'pasien': 0,
'as': 0,
'gelombang': 0,
'nasional': 0,
'pasca': 0,
'putih': 0}
```

Membuat Word Document Frequency

[ ]:
```python
def wordDocFre(vocab, doc_dict):
    df = {}
    for word in vocab:
        frq = 0
        for doc in doc_dict.values():
            if word in tokenisasi(doc):
                frq = frq + 1
        df[word] = frq
    return df
```

Membuat IDF

[ ]:
```python
import numpy as np
def inverseDocFre(vocab, doc_fre, length): # fungsi untuk menghasilkan idf
    idf = {}
    for word in vocab:
        idf[word] = 1 + np.log((length + 1) / (doc_fre[word]+1))
    return idf
```

[ ]:
```python
def termFrequencyInDoc(vocab, doc_dict):
    tf_docs = {}
    for doc_id in doc_dict.keys():
        tf_docs[doc_id] = {}
    for word in vocab:
        for doc_id, doc in doc_dict.items():
            tf_docs[doc_id][word] = doc.count(word)
    return tf_docs
```

TF-IDF

[ ]:
```python
def tfidf(vocab, tf, idf_scr, doc_dict):
    tf_idf_scr = {}
    for doc_id in doc_dict.keys():
        tf_idf_scr[doc_id] = {}
    for word in vocab:
        for doc_id, doc in doc_dict.items():
            tf_idf_scr[doc_id][word] = tf[doc_id][word] * idf_scr[word]
    return tf_idf_scr
```

[ ]:
```
import math
def cosine_sim(vec1, vec2):
    vec1 = list(vec1)
    vec2 = list(vec2)
    dot_prod = 0
    for i, v in enumerate(vec1):
        dot_prod += v * vec2[i]
    mag_1 = math.sqrt(sum([x**2 for x in vec1]))
    mag_2 = math.sqrt(sum([x**2 for x in vec2]))
    # Menggunakan numpy.asscalar() untuk mengubah array menjadi scalar
    mag_1 = np.squeeze(mag_1)
    mag_2 = np.squeeze(mag_2)
    return dot_prod / (mag_1 * mag_2)
```

[ ]:
```
from collections import OrderedDict
def topk(doc_dict, TD, q, k):
    relevance_scores = {}
    i = 0
    for doc_id in doc_dict.keys():
        relevance_scores[doc_id] = cosine_sim(q, TD[:, i])
        i = i + 1

    sorted_value = OrderedDict(sorted(relevance_scores.items(), key=lambda x:␣
 ↪x[1], reverse = True))
    top_k = {j: sorted_value[j] for j in list(sorted_value)[:k]}
    # penghitungan time complexity (disusun oleh banyak dokumen + proses␣
 ↪pengurutan + seleksi top k)
    time_complexity_k = len(doc_dict) + (len(doc_dict) * (len(doc_dict).
 ↪bit_length() - 1)) + k
    return top_k, time_complexity_k
```

[ ]:
```
def retriev(vocab, query, doc_dict, k):
    tf_query = termFrequency(vocab, query)
    idf = inverseDocFre(vocab, wordDocFre(vocab, doc_dict), len(doc_dict))


    TQ = np.zeros((len(vocab), 1))

    for word in vocab: # iterasi untuk pemobobotan tf-idf term-query matriks
        ind1 = vocab.index(word) # memberikan index pada tiap kata pada vocab
        TQ[ind1][0] = tf_query[word]*idf[word]

    # implementasi fungsi pembobotan tf-idf antara tiap term dalam vocab dan␣
 ↪tiap dokumen di dalam corpus untuk digunakan dalam konstruksi term-document␣
 ↪matriks
    tf_idf = tfidf(vocab, termFrequencyInDoc(vocab, doc_dict), idf, doc_dict)
```

```
    # inisialisasi term-query matriks dengan matriks 0 dengan banyak baris␣
↪sebanyak len(vocab) dan banyak kolom sebanyak len(doc_dict)
    TD = np.zeros((len(vocab), len(doc_dict)))
    for word in vocab: # iterasi untuk konstruksi term-document matriks
        for doc_id, doc in tf_idf.items():
            ind1 = vocab.index(word)
            ind2 = list(tf_idf.keys()).index(doc_id)
            TD[ind1][ind2] = tf_idf[doc_id][word]

    # implementasi fungsi pemilihan top k dokumen beserta penghitungan time␣
↪complexitinya
    top_k_results, complexity_k = topk(doc_dict, TD, TQ, k)

    # penghitungan time complexity (disusun oleh penghitungan tf_query +␣
↪pembuatan TQ + penghitungan tf-idf + pembuatan TD + penghitungan top k)
    time_complexity_main = len(vocab) + len(vocab) + (len(vocab) *␣
↪len(doc_dict)) + (len(vocab) * len(doc_dict)) + len(doc_dict) + complexity_k

    return top_k_results, TQ, TD, time_complexity_main
```

[ ]:
```
k = 3
top_3_result, TQ, TD, time = retriev(vocab, query, doc_dict, k)
print(f'Term-query matriks:\n{TQ}')
print(f'\nTerm-document matriks:\n{TD}')
print(f'\nHasil perankingan top {k} dokumen:')

i = 1
for no_doc, cosine_similarity in top_3_result.items():
    print(f'{i}. Dokumen {no_doc} dengan nilai cosine similarity =␣
↪{cosine_similarity}')
    i += 1
```

```
Term-query matriks:
[[0.        ]
 [0.        ]
 [0.        ]
 [0.        ]
 [0.        ]
 [1.        ]
 [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.         ]
[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.        ]
[1.69314718]
[0.        ]
[0.        ]
[1.69314718]
[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.         ]
 [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.         ]
 [0.         ]
 [0.         ]
 [0.         ]
 [0.         ]]

Term-document matriks:
```

```
[[ 1.69314718  0.          0.          1.69314718  0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [ 1.          1.          1.          4.          1.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 3.38629436  0.          0.          0.          1.69314718]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 1.69314718  1.69314718  0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 1.69314718  0.          0.          0.          1.69314718]
 [ 1.69314718  0.          0.          3.38629436  0.        ]
 [ 1.69314718  1.69314718  0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [16.         18.         19.         12.         16.        ]
 [10.49306144  0.          2.09861229  4.19722458  2.09861229]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          2.09861229  0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.81093022  0.          1.40546511  1.40546511  0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [ 1.69314718  0.          0.          1.69314718  0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 4.19722458  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.79175947  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 6.29583687  0.          0.          0.          0.        ]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 1.          1.          1.          1.          1.        ]
 [11.85203026  6.77258872 18.62461899  3.38629436  3.38629436]
 [ 2.09861229  0.          0.          0.          0.        ]
 [ 2.79175947  0.          0.          0.          0.        ]
```

```
[ 8.43279065  5.62186043 15.46011619 12.64918597  7.02732554]
[ 6.29583687  0.          0.          0.          0.        ]
[ 3.38629436  0.          0.          0.          1.69314718]
[ 3.          7.          4.          2.          3.        ]
[ 2.09861229  0.          0.          0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          2.09861229  0.          0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          6.29583687  0.          0.          0.        ]
[ 0.          2.09861229  0.          0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          2.09861229  0.          0.          6.29583687]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          4.19722458  0.          0.          0.        ]
[ 0.          4.19722458  0.          0.          0.        ]
[ 2.09861229  4.19722458  2.09861229  2.09861229  2.09861229]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          3.38629436  5.07944154  0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          1.69314718]
[ 0.          4.19722458  0.          0.          0.        ]
[ 0.          2.81093022  2.81093022  1.40546511  0.        ]
[ 0.          1.69314718  5.07944154  0.          0.        ]
[ 0.          1.69314718  1.69314718  0.          0.        ]
[ 0.          1.69314718  5.07944154  0.          0.        ]
[ 0.          2.79175947  0.          0.          0.        ]
[ 0.         11.85203026  8.4657359   0.          0.        ]
[ 0.          2.09861229  0.          0.          0.        ]
[ 0.          2.09861229  0.          0.          0.        ]
[ 0.          0.          1.69314718  0.          3.38629436]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          4.19722458  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          7.02732554  9.83825576  2.81093022]
[ 0.          0.          6.29583687  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[10.15888308 13.54517744 11.85203026 10.15888308 13.54517744]
[ 2.09861229  0.          2.09861229  0.          0.        ]
[ 0.          0.          3.38629436  0.          1.69314718]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          6.29583687  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          1.69314718  1.69314718  0.        ]
```

```
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          2.79175947  0.          0.        ]
[ 0.          0.          4.19722458  0.          0.        ]
[ 0.          0.          4.19722458  0.          0.        ]
[ 0.          0.          5.62186043  7.02732554  5.62186043]
[ 0.          0.          2.09861229  0.          0.        ]
[ 0.          0.          4.19722458  0.          0.        ]
[ 0.          0.          2.09861229  0.          0.        ]
[ 2.09861229  0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          4.19722458  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          4.19722458  0.        ]
[ 0.          0.          0.          4.19722458  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.79175947  0.        ]
[ 0.          0.          0.         10.49306144  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  2.09861229]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          1.69314718  1.69314718]
[ 0.          2.09861229  2.09861229  2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.09861229  0.        ]
[ 0.          0.          0.          2.79175947  0.        ]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.79175947]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          6.29583687]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          4.19722458]
[ 0.          0.          0.          0.          2.09861229]
[ 0.          0.          0.          0.          4.19722458]
[ 0.          0.          0.          0.          2.09861229]
```

```
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          4.19722458]
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          2.79175947]
 [ 0.          0.          0.          0.          6.29583687]
 [ 0.          0.          0.          0.          2.09861229]
 [10.49306144 16.78889831 12.59167373  4.19722458 18.8875106 ]
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          2.09861229]
 [ 0.          0.          0.          0.          2.09861229]]

Hasil perankingan top 3 dokumen:
1. Dokumen 2 dengan nilai cosine similarity = [0.22852767]
2. Dokumen 3 dengan nilai cosine similarity = [0.1661475]
3. Dokumen 5 dengan nilai cosine similarity = [0.07374686]

Time complexity: 1895
```

1 Penugasan Praktikum 6

Evaluasi untuk Unranked Retrieval Set

Mencari skor Precision dan Recall

[ ]:
```python
def main_unranked(rel_docs):
    retrieved_rel_doc3 = [value for value in list(top_3_result.keys()) if value␣
 ↪in rel_docs]
    precission = len(retrieved_rel_doc3)/len(top_3_result)*100
    recall = len(retrieved_rel_doc3)/len(rel_docs)*100
    f1Score = 2 * precission * recall / (precission + recall)
    return precission, recall, f1Score
```

Dokumen sebenarnya yang sesuai query berdasarkan relevance judgement yaitu berita2 dan berita3.

[ ]:
```python
rel_judgement = {
'1': 0,
'2': 1,
'3': 1,
'4': 0,
'5': 0,
}

rel_docs = [] # inisilisasi list kosong untuk menyimpan dokumen yang relevan
for doc_id, rel in rel_judgement.items():
    if rel==1:
        rel_docs.append(doc_id)
```

```
precission, recall, f1score = main_unranked(rel_docs)
print('Metrik evaluasi untuk unranked retrieval set adalah sebagai berikut:')
print(f'Nilai precission: {precission}\nNilai recall: {recall}\nNilai F1-score:␣
↪{f1score}')
```

```
Metrik evaluasi untuk unranked retrieval set adalah sebagai berikut:
Nilai precission: 66.66666666666666
Nilai recall: 100.0
Nilai F1-score: 80.0
```

Pada kode di atas, variabel dictionary rel_judgement digunakan untuk memberi kode pada berita2 dan berita 3 karena berita tersebut merupakan berita relevan berdasarkan relevance judgment.

Evaluasi untuk Ranked Retrieval Set

[ ]:
```
import numpy as np
def compute_prf_metrics(I, score, I_Q):
    """Compute precision, recall, F-measures and other
    evaluation metrics for document-level retrieval

    Notebook: C7/C7S3_Evaluation.ipynb

    Args:
        I (np.ndarray): Array of items
        score (np.ndarray): Array containing the score values of the times
        I_Q (np.ndarray): Array of relevant (positive) items

    Returns:
        P_Q (float): Precision
        R_Q (float): Recall
        F_Q (float): F-measures sorted by rank
        BEP (float): Break-even point
        F_max (float): Maximal F-measure
        P_average (float): Mean average
        X_Q (np.ndarray): Relevance function
        rank (np.ndarray): Array of rank values
        I_sorted (np.ndarray): Array of items sorted by rank
        rank_sorted (np.ndarray): Array of rank values sorted by rank
    """
    # Compute rank and sort documents according to rank
    K = len(I)
    index_sorted = np.flip(np.argsort(score))
    I_sorted = I[index_sorted]
    rank = np.argsort(index_sorted) + 1
    rank_sorted = np.arange(1, K+1)

    # Compute relevance function X_Q (indexing starts with zero)
```

```
    # X_Q = np.zeros(K, dtype=bool)
    # for i in range(K):
    #     if I_sorted[i] in I_Q:
    #         X_Q[i] = True
    X_Q = np.isin(I_sorted, I_Q)
    # P_Q = np.cumsum(X_Q) / np.arange(1, K+1)

    # Compute precision and recall values (indexing starts with zero)
    M = len(I_Q)
    # P_Q = np.zeros(K)
    # R_Q = np.zeros(K)
    # for i in range(K):
    #     r = rank_sorted[i]
    #     P_Q[i] = np.sum(X_Q[:r]) / r
    #     R_Q[i] = np.sum(X_Q[:r]) / M
    P_Q = np.cumsum(X_Q) / np.arange(1, K+1)
    R_Q = np.cumsum(X_Q) / M

    # Break-even point
    BEP = P_Q[M-1]
    # Maximal F-measure
    sum_PR = P_Q + R_Q
    sum_PR[sum_PR == 0] = 1  # Avoid division by zero
    F_Q = 2 * (P_Q * R_Q) / sum_PR
    F_max = F_Q.max()
    # Average precision
    P_average = np.sum(P_Q * X_Q) / len(I_Q)

    return P_Q, R_Q, F_Q, BEP, F_max, P_average, X_Q, rank, I_sorted,␣
↪rank_sorted
```

[ ]:
```
import pandas as pd
relevance_scores = {}
i = 0
for doc_id in doc_dict.keys():
    relevance_scores[doc_id] = cosine_sim(TQ, TD[:, i])
    i = i + 1

# mengubah value dari dictionary relevance_scores menjadi float
for key, value in relevance_scores.items():
    relevance_scores[key] = float(value[0])

I = np.array(list(relevance_scores.keys()))
score = np.array(list(relevance_scores.values()))
I_Q = np.array(['2', '3'])
output = compute_prf_metrics(I, score, I_Q)
P_Q, R_Q, F_Q, BEP, F_max, P_average, X_Q, rank, I_sorted, rank_sorted = output
```

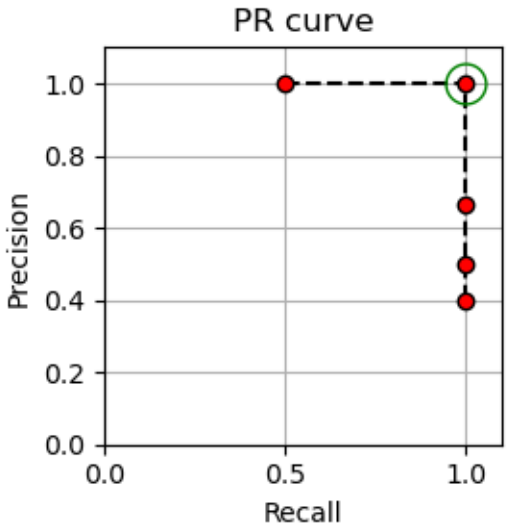
```
# Arrange output as tables
score_sorted = np.flip(np.sort(score))
df = pd.DataFrame({'Rank': rank_sorted, 'ID': I_sorted, 'Score': score_sorted,␣
 ↪'$\chi_\mathcal{Q}$': X_Q, 'P(r)': P_Q, 'R(r)': R_Q, 'F(r)': F_Q})
print(df)
print('Break-even point = %.2f' % BEP)
print('F_max = %.2f' % F_max)
print('Average precision =', np.round(P_average, 5))
```

```
   Rank ID     Score  $\chi_\mathcal{Q}$      P(r)  R(r)      F(r)
0     1  2  0.228528                True  1.000000   0.5  0.666667
1     2  3  0.166147                True  1.000000   1.0  1.000000
2     3  5  0.073747               False  0.666667   1.0  0.800000
3     4  4  0.051724               False  0.500000   1.0  0.666667
4     5  1  0.011144               False  0.400000   1.0  0.571429
Break-even point = 1.00
F_max = 1.00
Average precision = 1.0
```

Pada hasil di atas, diperoleh hasil relevance score untuk masing-masing dokumen

[ ]:
```
from matplotlib import pyplot as plt
def plot_PR_curve(P_Q, R_Q, figsize=(3, 3)):
    fig, ax = plt.subplots(1, 1, figsize=figsize)
    plt.plot(R_Q, P_Q, linestyle='--', marker='o', color='k', mfc='r')
    plt.xlim([0, 1.1])
    plt.ylim([0, 1.1])
    ax.set_aspect('equal', 'box')
    plt.title('PR curve')
    plt.xlabel('Recall')
    plt.ylabel('Precision')
    plt.grid()
    plt.tight_layout()
    ax.plot(BEP, BEP, color='green', marker='o', fillstyle='none',␣
 ↪markersize=15)
    ax.set_title('PR curve')
    plt.show()
    return fig, ax
```

[ ]:
```
plot_PR_curve(P_Q, R_Q, figsize=(3,3))
```



[ ]: (<Figure size 300x300 with 1 Axes>,
<Axes: title={'center': 'PR curve'}, xlabel='Recall', ylabel='Precision'>)