LAPORAN PRAKTIKUM NATURAL LANGUAGE PROCESSING
VECTORIZATIONN

Oleh

Muhammad Ihsan Prawira Hutomo

2211110022

PROGRAM STUDI S1 SAINS DATA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2022

Tugas

Buatlah representasi data dari teks cerita malin kundang dalam bentuk PDF berikut.

Metode Representasi yang harus Anda tampilkan adalah:

1. One hot encoding

2. Hash

3. Co-occurance matrix

4.Word2Vec

5.Fast text

Hasil tugas dikumpulkan dalam bentuk PDF yang berisi kode program

File PDF

https://lms.ittelkom-pwt.ac.id/pluginfile.php/224129/mod_assign/introattachment/0/MALIN_KUNDANG.pdf?forcedownload=1

Github link : https://github.com/pwawiwa/nlp-praktikum/blob/main/Vektorisasi-part-2/NLP_Tugas_Vektorisasi.ipynb

Extract PDF

```
pip install PyPDF2
```

```
Collecting PyPDF2
  Downloading pypdf2-3.0.1-py3-none-any.whl (232 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 232.6/232.6 kB 4.2 MB/s eta 0:00:00
Installing collected packages: PyPDF2
Successfully installed PyPDF2-3.0.1
```

```
[3] import PyPDF2
    import pandas as pd

    def extract_sentences(pdf_path):
        data = {'Sentence': []}

        with open(pdf_path, 'rb') as file:
            pdf_reader = PyPDF2.PdfReader(file)

            for page_num in range(len(pdf_reader.pages)):
                page = pdf_reader.pages[page_num]
                text = page.extract_text()

                # Split text into sentences
                sentences = text.split('.')

                # Add sentences to the data dictionary
                data['Sentence'].extend(sentences)

        return pd.DataFrame(data)

    # Example usage
    pdf_path = '/content/MALIN_KUNDANG.pdf'
    df = extract_sentences(pdf_path)

    # Display the DataFrame
    print(df)
```

```
                                               Sentence
0     MALIN KUNDANG \nPada suatu waktu, hiduplah seb...
1      Keluarga tersebut terdiri \ndari ayah, ibu da...
2      Karena kondisi keuangan keluarga yang \nmempr...
3     \nMaka tinggallah si Malin dan ibunya di gubug...
4      Semingg u, dua minggu, sebulan, dua \nbulan b...
...                                                 ...
3734   Tubuh Prabu Dewata Cengkar dilempar Aji Saka ...
3735   \nAji Saka kemudian dinobatkan menjadi raja M...
3736                  I a memboyong ayahnya ke \nistana
3737   Berkat pemerintahan yang adil dan bijaksana, ...
3738

[3739 rows x 1 columns]
```

Pembersihan dengan lowercase, digit, tanda baca dan stopwords

```
[5]  df['Sentence'] = df['Sentence'].apply(lambda x: x.str.lower())
```

```
[7]  import re
```

```
[8]  # Membersihkan /n
     def clean_text1(text):
         return re.sub(r'\n([a-z])', r' \1', text)

     # Apply the function to the 'Text' column
     df['Sentence'] = df['Sentence'].apply(clean_text1)
```

```
[10] import pandas as pd
     import re
     import string


     # digit and punctuation removal function
     def remove_digits_and_punctuation(text):
         cleaned_text = re.sub(r'[\d' + re.escape(string.punctuation) + ']', '', text)
         return cleaned_text

     # Apply the function to the 'Sentence' column
     df['Sentence'] = df['Sentence'].apply(remove_digits_and_punctuation)

     # Display the result
     print(df['Sentence'])
```

```
     # Stopword Removal
     import nltk
     from nltk.corpus import stopwords

     nltk.download('stopwords')

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

     def remove_stopwords(text):
         words = text.split()
         filtered_words = [word for word in words if word.lower() not in stop_words]
         return ' '.join(filtered_words)

     df['Sentence'] = df['Sentence'].apply(remove_stopwords)
```

```
WordListCorpusReader: stopwords
nltk.corpus.reader.wordlist.WordListCorpusReader instance
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Hasil

```
[12] df['Sentence']

0       malin kundang hiduplah keluarga nelayan pesisi...
1          keluarga ayah anak lakilaki nama malin kundang
2       kondisi keuangan keluarga memprihatinkan sang ...
3                         tinggallah si malin ibunya gubug
4       semingg u minggu sebulan ayah malin kampung ha...
                              ...
3734    tubuh prabu dewata cengkar dilempar aji saka j...
3735              aji saka dinobatkan raja medang kamulan
3736                       i a memboyong ayahnya istana
3737    berkat pemerintahan adil bijaksana aji saka me...
3738
Name: Sentence, Length: 3739, dtype: object
```

One Hot Encoding

OHE

```
[42] OHE = pd.get_dummies(df['Sentence'].str.split(expand=True).stack(), drop_first=True).groupby(level=0).max()

[54] OHE['Sentence']= df['Sentence']

[62] OHE
```

	aaa	aaaa	aan	aat	abad	abadi	abangnya	abdi	abdinya	abu	...	yelamatkan	yet	yik	yikan	yir	yosaku	yuk	yut	zam	Sentence
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin kundang hiduplah keluarga nelayan pesisi...
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	keluarga ayah anak lakilaki nama malin kundang
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	kondisi keuangan keluarga memprihatinkan sang ...
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	tinggallah si malin ibunya gubug
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	semingg u minggu sebulan ayah malin kampung ha...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3733	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	prabu dewata cengkar marah serban aji s aka me...
3734	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	tubuh prabu dewata cengkar dilempar aji saka j...
3735	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	aji saka dinobatkan raja medang kamulan
3736	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	i a memboyong ayahnya istana
3737	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	berkat pemerintahan adil bijaksana aji saka me...

3535 rows × 4884 columns

Ada modifikasi pada pd.get_dummies yang biasanya bisa langsung digunakan untuk one hot encoding karena data kata-kata berada di dalam kalimat. Modifikasinya adalah fungsi split untuk memisahkan kalimat dalam kata dan expand untuk menambah kolom baru seiring bertambahnya kata baru yang ingin dilakukan OHE. Fungsi seterusnya yaitu mengelompokkan data berdasarkan indeks utama dari MultiIndex (indeks DataFrame asli) dan menerapkan fungsi lambda untuk membuat Seri berisi nilai 1 dengan indeks yang sama seperti kata-kata. Akhirnya, unstack(fill_value=0) mengubah bentuk data, mengisi nilai NaN dengan 0, menghasilkan DataFrame yang telah di-OHE kan.

Pencarian kalimat yang terdapat kata ‘malin’ atau kolom malin bernilai == 1.

```
OHE[OHE['malin']==1]
```

	aaa	aaaa	aan	aat	abad	abadi	abangnya	abdi	abdinya	abu	...	yelamatkan	yet	yik	yikan	yir	yosaku	yuk	yut	zam	Sentence
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin kundang hiduplah keluarga nelayan pesisi...
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	keluarga ayah anak lakilaki nama malin kundang
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	tinggallah si malin ibunya gubug
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	semingg u minggu sebulan ayah malin kampung ha...
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ibunya menggantikan posisi ayah malin mencari ...
6	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin anak cerdas nakal
8	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin me ngejar ayam tersandung batu lengan ka...
10	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	beranjak dewasa malin kundang kasihan ibunya b...
12	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin tertarik ajakan nakhoda kapal dagang dul...
13	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin kundang mengutarakan maksudnya ibunya
14	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ibunya s emula setuju maksud malin kundang mal...
15	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	bekal perlengkapan malin dermaga diantar ibunya
16	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	anakku engkau berhasil orang berkecukupan kau ...
17	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	kapal dinaiki malin diiringi lambaian tangan m...
18	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	kapal malin kundang bany ak belajar ilmu pelay...
19	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	perjalanan kapal dinaiki malin kundang serang ...
22	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin kundang beruntung dibunuh ba jak laut pe...
23	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	malin kundang terkatungkatung ditengah laut ak...
24	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	sisa tenaga ad a malin kundang berjalan desa t...
25	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	sesampainya desa te rsebut malin kundang ditolong
27	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	desa malin terdampar desa ya ng subur
28	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	keuletan kegigihannya malin kelamaa n berhasil...
30	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	kaya raya malin kundang m empersunting gadis i...
31	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	berita malin kundang kaya raya te menikah mali

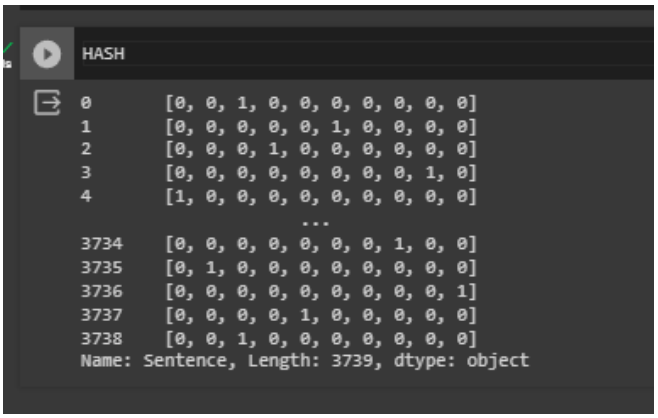
Hash Vectorization

HASH

```python
import pandas as pd
import hashlib


def hash_vectoring(text, vector_size):
    # Inisialisasi vektor dengan nilai 0
    vector = [0] * vector_size

    # Konversi teks menjadi hash
    hashed_text = hashlib.sha256(text.encode()).hexdigest()

    # Ambil sebagian dari hash (sesuai dengan panjang vektor)
    hash_subset = hashed_text[:vector_size]

    # Konversi hash menjadi bilangan bulat (integer)
    hash_integer = int(hash_subset, 16)

    # Modulus hash dengan ukuran vektor untuk mendapatkan indeks
    index = hash_integer % vector_size

    # Set nilai indeks vektor menjadi 1
    vector[index] = 1

    return vector

# Define the vector size (you can adjust this based on your needs)
vector_size = 10

# Apply hash_vectoring to the 'Sentence' column
HASH = df['Sentence'].apply(lambda x: hash_vectoring(x, vector_size))
```

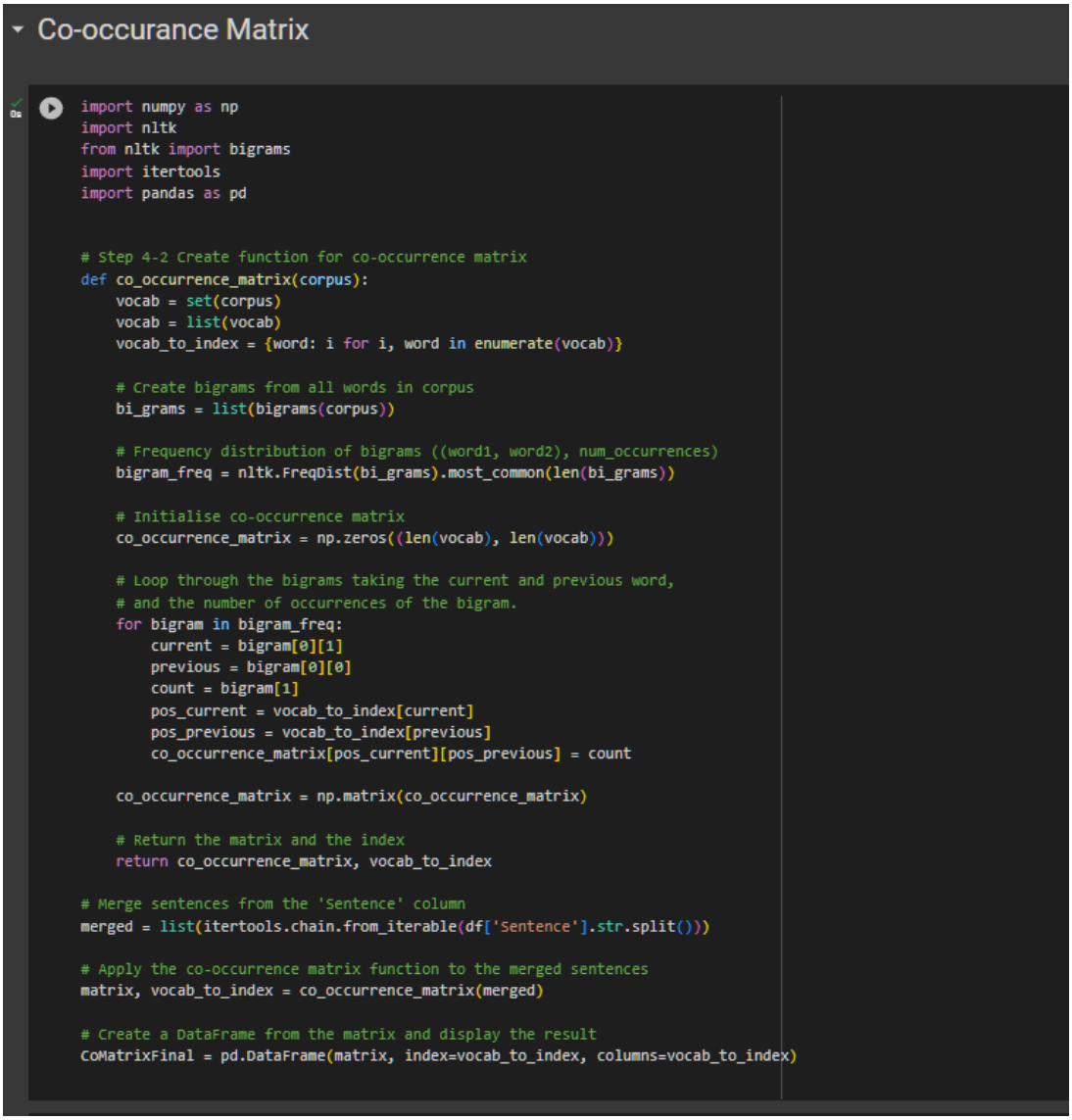
Model Hash vectorization adalah mengubah data teks menjadi representasi numerik yang dalam hal ini adalah biner. Output dari Hash adalah sebuah row kalimat tertentu akan lebih dekat ke salah satu fitur tertentu.

Pada vector size atau jumlah fitur yang digunakan adalah 10 buah sehingga didapat hasil sebagai berikut.



```
HASH

0       [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
1       [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
2       [0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
3       [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
4       [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
                    ...
3734    [0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
3735    [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
3736    [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
3737    [0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
3738    [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
Name: Sentence, Length: 3739, dtype: object
```

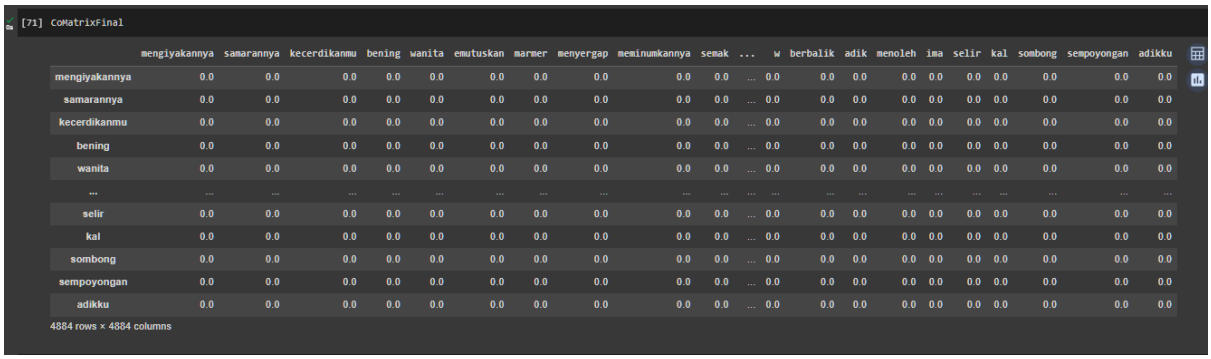
Co-occurrence Matrix



```
Co-occurance Matrix

import numpy as np
import nltk
from nltk import bigrams
import itertools
import pandas as pd


# Step 4-2 Create function for co-occurrence matrix
def co_occurrence_matrix(corpus):
    vocab = set(corpus)
    vocab = list(vocab)
    vocab_to_index = {word: i for i, word in enumerate(vocab)}

    # Create bigrams from all words in corpus
    bi_grams = list(bigrams(corpus))

    # Frequency distribution of bigrams ((word1, word2), num_occurrences)
    bigram_freq = nltk.FreqDist(bi_grams).most_common(len(bi_grams))

    # Initialise co-occurrence matrix
    co_occurrence_matrix = np.zeros((len(vocab), len(vocab)))

    # Loop through the bigrams taking the current and previous word,
    # and the number of occurrences of the bigram.
    for bigram in bigram_freq:
        current = bigram[0][1]
        previous = bigram[0][0]
        count = bigram[1]
        pos_current = vocab_to_index[current]
        pos_previous = vocab_to_index[previous]
        co_occurrence_matrix[pos_current][pos_previous] = count

    co_occurrence_matrix = np.matrix(co_occurrence_matrix)

    # Return the matrix and the index
    return co_occurrence_matrix, vocab_to_index

# Merge sentences from the 'Sentence' column
merged = list(itertools.chain.from_iterable(df['Sentence'].str.split()))

# Apply the co-occurrence matrix function to the merged sentences
matrix, vocab_to_index = co_occurrence_matrix(merged)

# Create a DataFrame from the matrix and display the result
CoMatrixFinal = pd.DataFrame(matrix, index=vocab_to_index, columns=vocab_to_index)
```

Menggunakan fungsi yang ada di praktikum dengan modifikasi pada variable merged yang ditambahkan fungsi split untuk memecah kata di kalimat.

Hasilnya seperti berikut:



```
[71] CoMatrixFinal
```

	mengiyakannya	samarannya	kecerdikanmu	bening	wanita	emutuskan	marmer	menyergap	meminumkannya	semak	...	w	berbalik	adik	menoleh	ima	selir	kal	sombong	sempoyongan	adikku
mengiyakannya	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
samarannya	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
kecerdikanmu	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
bening	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
wanita	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
selir	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
kal	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
sombong	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
sempoyongan	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
adikku	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

4884 rows × 4884 columns

Co-occurrence berguna untuk mencari kata yang sering muncul bersama pada suatu konteks. Sebagai contoh kata ‘malin’ sering muncul berdekatan (kata sebelum dan sesudah) dengan beberapa kata berikut:

```
[73] CoMatrixFinal['malin'][CoMatrixFinal['malin']==1]

diiringi        1.0
me              1.0
mendesak        1.0
istrinya        1.0
mencari         1.0
kun             1.0
anak            1.0
kelamaa         1.0
tertarik        1.0
dermaga         1.0
ibunya          1.0
terdampar       1.0
menengadahkan   1.0
kampung         1.0
bersembunyi     1.0
Name: malin, dtype: float64
```

Word2Vec

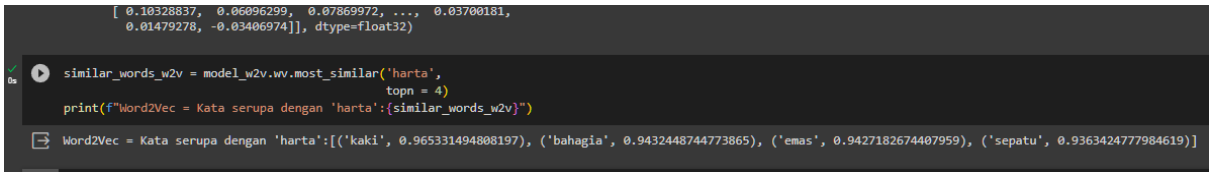
```
Word2Vec

from gensim.models import Word2Vec

+ Code + Text

[17] # Split sentences into words
     df['Words'] = df['Sentence'].apply(lambda x: x.split())

     # Display the result
     print(df['Words'].tolist())

     [['malin', 'kundang', 'hiduplah', 'keluarga', 'nelayan', 'pesisir', 'pantai', 'wilayah', 'sumatra'], ['keluarga', 'ayah', 'anak

[103] model_w2v = Word2Vec(sentences = df['Words'], vector_size = 10, window = 10, min_count=1, workers = 4)

[104] import numpy as np

[105] words = list(model_w2v.wv.index_to_key)
      vector_W2V = [model_w2v.wv[word] for word in words]
      vector_W2V = np.array(vector_W2V)
```

Pembuatan model Word2Vec dimulai dengan pemecahan kata-kata dalam kalimat dan membuatnya dalam array, kemudian model Word2Vec dibuat dengan beberapa parameter seperti vector_size = 10 yang berarti hasil vector akan berdimensi 10, window = 10 yang berarti model akan mempertimbangkan 10 kata sebelum dan sesudah kata target, min_count = 1 berarti sebuah kata akan tercatat jika muncul setidaknya 1 kali dalam kalimat, dan worker =4 adalah penggunaan cpu dalam training.

Hasil Word2Vec

```
vector_W2V

array([[ 0.24301793, -0.10094726,  0.40993246, ...,  0.2715208 ,
        -0.5350817 , -0.40523204],
       [ 0.3366652 , -0.11739132,  0.30948305, ...,  0.16600001,
        -0.58281446, -0.4651468 ],
       [ 0.39142478, -0.05559553,  0.457041  , ...,  0.22560218,
        -0.6260841 , -0.4434359 ],
       ...,
       [ 0.02885258,  0.01481642,  0.08708108, ..., -0.03916731,
         0.05117014, -0.0324481 ],
       [-0.02571236, -0.03237989, -0.0866741 , ...,  0.08821128,
        -0.03641912,  0.08899024],
       [ 0.10328837,  0.06096299,  0.07869972, ...,  0.03700181,
         0.01479278, -0.03406974]], dtype=float32)
```

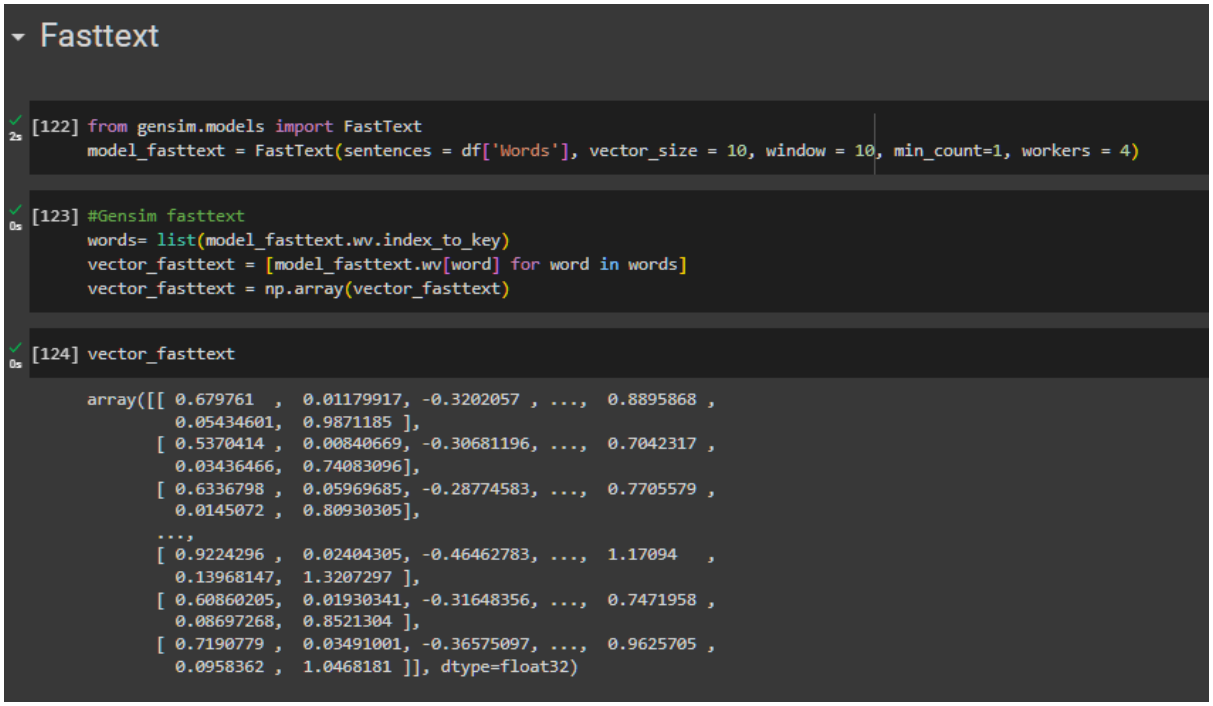
Pencarian kata yang mirip dengan menggunakan model yang dibuat. Kata yang dicari adalah ‘harta’.



Hasilnya kata ‘harta’ mirip atau dekat dengan kata ‘kaki’, ‘bahagia, ‘emas, ‘sepatu.

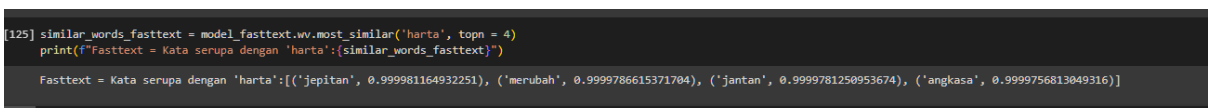
Beberapa hasil cukup memuaskan kecuali kata ‘kaki’ dan ‘sepatu’ yang kurang cocok dengan kata harta , mungkin terdapat kesalahan bias dalam model ataupun parameter kurang optimal.

Fasttext



Sebagian besar syntax mirip dengan Word2Vec hanya mengganti model yang digunakan yaitu Fasttext dan tetap menggunakan parameter yang sama.

Lalu, melakukan pencarian kata yang mirip, kata yang dicari adalah ‘harta’ seperti sebelumnya.



Hasil didapat sangat buruk 4 kata top yang didapat tidak cocok dengan kata harta, berbeda dengan model Word2Vec sebelumnya.