2211110013_Muhammad_Naufal_Farabbi_NLP_PRAK_7

November 12, 2023

[2]:
```
pip install fasttext
```

```
Collecting fasttext
  Downloading fasttext-0.9.2.tar.gz (68 kB)
                           68.8/68.8 kB
481.5 kB/s eta 0:00:00
  Preparing metadata (setup.py) … done
Collecting pybind11>=2.2 (from fasttext)
  Using cached pybind11-2.11.1-py3-none-any.whl (227 kB)
Requirement already satisfied: setuptools>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from fasttext) (67.7.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages
(from fasttext) (1.23.5)
Building wheels for collected packages: fasttext
  Building wheel for fasttext (setup.py) … done
  Created wheel for fasttext:
filename=fasttext-0.9.2-cp310-cp310-linux_x86_64.whl size=4199772
sha256=ba7e2b52651b91c386d6f0d2f5ca797679598b601aa6adcd0566ca8c4aba2a1d
  Stored in directory: /root/.cache/pip/wheels/a5/13/75/f811c84a8ab36eedbaef977a
6a58a98990e8e0f1967f98f394
Successfully built fasttext
Installing collected packages: pybind11, fasttext
Successfully installed fasttext-0.9.2 pybind11-2.11.1
```

[3]:
```
pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages
(3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages
(from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk) (2023.6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from nltk) (4.66.1)
```

[5]:
```
nltk.download('punkt')
```

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

[5]: True

[4]:
```python
import re
import string
import nltk
import gensim
import fasttext
import itertools
import numpy as np
import pandas as pd
import seaborn as sns
import tensorflow as tf
from nltk.corpus import stopwords
from nltk import bigrams
from tensorflow import keras
import matplotlib.pyplot as plt
from nltk.tokenize import word_tokenize
from tensorflow.keras.models import Sequential
from gensim.models import Word2Vec, KeyedVectors
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.layers import Embedding, Bidirectional, LSTM, Dense
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer,␣
 ↪HashingVectorizer
```

[6]:
```
pip install PyPDF2
```

```
Collecting PyPDF2
  Downloading pypdf2-3.0.1-py3-none-any.whl (232 kB)
                         232.6/232.6

kB 2.2 MB/s eta 0:00:00
Installing collected packages: PyPDF2
Successfully installed PyPDF2-3.0.1
```

0.1 Ekstrak teks PDF

Ekstraksi dilakukan agar segala tulisan didalam file dapat diambil untuk selanjutnya diproses. Ekstraksi dilakukan dengan mencari per kalimat yang selanjutnya ditampung ke dalam kolom 'Kalimat' dalam dataframe (df)

[9]:
```
import PyPDF2
import pandas as pd
def extract_sentences (pdf_path):
    data = {'Kalimat': []}

    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[ 'Kalimat'].extend(sentences)
    return pd.DataFrame (data)
# Example usage
pdf_path='/content/MALIN_KUNDANG.pdf'
df = extract_sentences(pdf_path)
# Display the DataFrame print (df)
df
```

[9]:
```
                                                Kalimat
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      Seminggu 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]
```

0.2 Pre-processing

Pre-processing dilakukan untuk mempermudah pengolahan teks selanjutnya. Dilakukan dengan beberapa tahap seperti, menghapus tanda baca, angka, konversi huruf kecil, tokenisasi, menghapus stopwords. Sehingga, nantinya teks lebih efektif dan efisien untuk dianalisis.

[10]:
```
nltk.download('stopwords')
def clean_text(input_text):
    # Menghapus tanda baca
```

```
    translator = str.maketrans("", "", string.punctuation)
    text_without_punct = input_text.translate(translator)

    # Menghapus angka
    text_without_numbers = re.sub(r'\d', '', text_without_punct)

    # Mengonversi huruf kecil
    cleaned_text = text_without_numbers.lower()

    # Tokenisasi teks
    tokens = nltk.word_tokenize(cleaned_text)

    # Menghapus stopwords
    stop_words = set(stopwords.words('indonesian'))
    filtered_tokens = [word for word in tokens if word.lower() not in␣
 ↪stop_words]

    # Menggabungkan kembali teks dari token yang telah difilter
    cleaned_text = ' '.join(filtered_tokens)

    return cleaned_text


df['Kalimat'] = df['Kalimat'].apply(clean_text)
print(df)
```

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

                                                Kalimat
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     seminggg 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

[3739 rows x 1 columns]
```

0.3 One Hot Encoding

Mengubah data kategorikal menjadi sebuah vektor biner dengan nilai 1 pada kategori yang sesuai dan 0 untuk kategori lainnya.

[11]:
```
ONE_HOT=pd.get_dummies(df['Kalimat'].str.split(expand=True).stack(),␣
 ↪drop_first=True).groupby(level=0).max()
ONE_HOT['Kalimat']=df['Kalimat']
ONE_HOT
```

[11]:
```
      aaa  aaaa  aan  aat  abad  abadi  abangnya  abdi  abdinya  abu  …  \
0       0     0    0    0     0      0         0     0        0    0  …
1       0     0    0    0     0      0         0     0        0    0  …
2       0     0    0    0     0      0         0     0        0    0  …
3       0     0    0    0     0      0         0     0        0    0  …
4       0     0    0    0     0      0         0     0        0    0  …
…     …    …   …   …   …     …     …        …    …       …   …  …
3733    0     0    0    0     0      0         0     0        0    0  …
3734    0     0    0    0     0      0         0     0        0    0  …
3735    0     0    0    0     0      0         0     0        0    0  …
3736    0     0    0    0     0      0         0     0        0    0  …
3737    0     0    0    0     0      0         0     0        0    0  …

      yelamatkan  yet  yik  yikan  yir  yosaku  yuk  yut  zam  \
0              0    0    0      0    0       0    0    0    0
1              0    0    0      0    0       0    0    0    0
2              0    0    0      0    0       0    0    0    0
3              0    0    0      0    0       0    0    0    0
4              0    0    0      0    0       0    0    0    0
…            …   …   …     …   …      …   …   …   …
3733           0    0    0      0    0       0    0    0    0
3734           0    0    0      0    0       0    0    0    0
3735           0    0    0      0    0       0    0    0    0
3736           0    0    0      0    0       0    0    0    0
3737           0    0    0      0    0       0    0    0    0

                                                Kalimat
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…
…                                                   …
3733  prabu dewata cengkar marah serban aji s aka me…
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…

[3535 rows x 4884 columns]
```

0.4 Hash Vectoring

Mengubah data atau teks non-numerik menjadi representasi numerik (biasanya dalam bentuk bilangan bulat atau vektor biner) dengan cara yang terdistribusi secara acak. Fungsi hash memetakan data masukan (misalnya, kata atau objek) ke nilai hash yang sesuai, yang kemudian dapat digunakan sebagai representasi numerik dari data tersebut.

[12]:
```python
import hashlib
def hash_vectoring(text):
    # 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

vector_size = 15
df_HASH = df["Kalimat"].apply(hash_vectoring)
print(df_HASH)
```

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

0.5 Co-Occurence Matriks

[116]:
```
# Membuat dataset menjadi list dalam lists (per kalimat dijadikan list)
df_lists = df['Kalimat'].apply(lambda x: x.split()).tolist()
df_lists[1:5]
```

[116]:
```
[['keluarga', 'ayah', 'anak', 'lakilaki', 'nama', 'malin', 'kundang'],
 ['kondisi',
  'keuangan',
  'keluarga',
  'memprihatinkan',
  'sang',
  'ayah',
  'memutuskan',
  'mencari',
  'nafkah',
  'negeri',
  'seberang',
  'mengarungi',
  'lautan',
  'luas'],
 ['tinggallah', 'si', 'malin', 'ibunya', 'gubug'],
 ['seminggg',
  'u',
  'minggu',
  'sebulan',
  'ayah',
  'malin',
  'kampung',
  'halamannya']]
```

[14]:
```
import numpy as np
import nltk
from nltk import bigrams
import itertools
import pandas as pd

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))
```

```python
    # 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

merged = list(itertools.chain.from_iterable(df_lists))
matrix, vocab_to_index = co_occurrence_matrix(merged)

CoMatrixFinal = pd.DataFrame(matrix, index=vocab_to_index,
 columns=vocab_to_index)
print(CoMatrixFinal)
```

```
             mengadakan  tingkat  anakanak  pemuda  kesal  langgang  perompak  \
mengadakan          0.0      0.0       0.0     0.0    0.0       0.0       0.0
tingkat             0.0      0.0       0.0     0.0    0.0       0.0       0.0
anakanak            0.0      0.0       0.0     0.0    0.0       0.0       0.0
pemuda              0.0      0.0       0.0     0.0    0.0       0.0       0.0
kesal               0.0      0.0       0.0     0.0    0.0       0.0       0.0
…                     …        …         …       …      …         …         …
lambat              0.0      0.0       0.0     0.0    0.0       0.0       0.0
ungsu               0.0      0.0       0.0     0.0    0.0       0.0       0.0
tom                 0.0      0.0       0.0     0.0    0.0       0.0       0.0
berkelakuan         0.0      0.0       0.0     0.0    0.0       0.0       0.0
rampokannya         0.0      0.0       0.0     0.0    0.0       0.0       0.0

             prab  bag  punggungnya  …  sejuk  kerja   ua  berdiri  gigih  \
mengadakan    0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
tingkat       0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
anakanak      0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
pemuda        0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
kesal         0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
…               …    …            …  …      …      …    …
lambat        0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
ungsu         0.0  0.0          0.0  …    0.0    0.0  0.0      0.0    0.0
```

```
tom            0.0  0.0         0.0  …   0.0    0.0  0.0      0.0    0.0
berkelakuan    0.0  0.0         0.0  …   0.0    0.0  0.0      0.0    0.0
rampokannya    0.0  0.0         0.0  …   0.0    0.0  0.0      0.0    0.0

             lambat  ungsu  tom  berkelakuan  rampokannya
mengadakan      0.0    0.0  0.0          0.0          0.0
tingkat         0.0    0.0  0.0          0.0          0.0
anakanak        0.0    0.0  0.0          0.0          0.0
pemuda          0.0    0.0  0.0          0.0          0.0
kesal           0.0    0.0  0.0          0.0          0.0
…             …    …  …        …          …
lambat          0.0    0.0  0.0          0.0          0.0
ungsu           0.0    0.0  0.0          0.0          0.0
tom             0.0    0.0  1.0          0.0          0.0
berkelakuan     0.0    0.0  0.0          0.0          0.0
rampokannya     0.0    0.0  0.0          0.0          0.0

[4884 rows x 4884 columns]
```

Baris dan kolom dalam matriks ini mewakili kata-kata dalam korpus yang digunakan. Setiap sel dalam matriks berisi angka yang menunjukkan seberapa sering dua kata muncul bersama-sama dalam konteks yang telah ditentukan. Nilai diagonal utama (misalnya, "lambat" di baris dan kolom sekian) umumnya diatur ke nol, karena ini menunjukkan seberapa sering sebuah kata muncul bersama dengan dirinya sendiri dalam konteks yang sama, yang biasanya tidak relevan. Nilai sel yang lebih besar menunjukkan bahwa kata-kata tersebut cenderung muncul bersama lebih sering dalam konteks yang sama. Contohnya, di baris "lapar" dan kolom "haus", nilai adalah 5 (jika dilihat dari file excelnya), menunjukkan bahwa kata "lapar" dan "haus" muncul bersama dalam lima konteks.

0.6 WORD2VEC

[75]:
```
# Membangun model Word2Vec
corpus=df['Kalimat']
tokenized_corpus = [sentence.split() for sentence in corpus]
model_w2v = Word2Vec(tokenized_corpus, vector_size=150, window=25, min_count=1,␣
  ↪sg=1)
```

[76]:
```
words=list(model_w2v.wv.index_to_key)
vector_W2V = [model_w2v.wv[word] for word in words]
vector_W2V = np.array(vector_W2V)
```

[115]:
```
words[1:10]
```

[115]:
```
['raja', 'si', 'sang', 'kancil', 'anak', 'puteri', 'istana', 'a', 'orang']
```

[78]:
```
vec_word2vec=pd.DataFrame(vector_W2V)
print(vec_word2vec)
```

```
           0         1         2         3         4         5         6    \
0    -0.114086 -0.079794 -0.030212 -0.066008  0.164640 -0.218131 -0.046164
1    -0.099754 -0.075718 -0.038167 -0.065102  0.143348 -0.191161 -0.047286
2    -0.094924 -0.042961 -0.038607 -0.089244  0.163655 -0.226439 -0.056719
3    -0.094524 -0.059426 -0.033376 -0.088737  0.153762 -0.220889 -0.070693
4    -0.069247 -0.056096 -0.046556 -0.083823  0.152241 -0.203306 -0.060706
…          …         …         …         …         …         …         …
4879 -0.014075 -0.008424 -0.014014 -0.018467  0.033933 -0.029483 -0.006715
4880 -0.005460 -0.006171 -0.008373 -0.008417  0.010040 -0.021307 -0.009938
4881 -0.009584 -0.004450 -0.004733 -0.007926  0.010122 -0.024938 -0.007555
4882 -0.005578 -0.007537 -0.004540 -0.012707  0.009462 -0.015897 -0.010643
4883 -0.018353 -0.012831 -0.013830 -0.012267  0.024212 -0.033598 -0.014815

           7         8         9    …       140       141       142  \
0     0.214324 -0.009060 -0.018524  … -0.015923 -0.072631  0.106559
1     0.212442 -0.028277 -0.004110  … -0.026873 -0.055112  0.140067
2     0.205409 -0.018235 -0.010207  … -0.034894 -0.077152  0.090281
3     0.179295 -0.031419 -0.017594  … -0.010365 -0.063172  0.102875
4     0.190748 -0.025773 -0.008465  … -0.028610 -0.076094  0.098876
…          …         …         … …       …         …         …
4879  0.036181 -0.007061 -0.002929  …  0.000019 -0.014142  0.018201
4880  0.011340 -0.006940 -0.002600  …  0.004332 -0.004765  0.003585
4881  0.022405 -0.004390 -0.006207  … -0.006489 -0.010773  0.011643
4882  0.008163 -0.002169 -0.001883  … -0.001899  0.000927  0.012902
4883  0.037956 -0.002687 -0.007440  … -0.000057 -0.018539  0.022362

           143       144       145       146       147       148       149
0    -0.026739  0.102262  0.156552 -0.260656 -0.091037  0.082283 -0.300681
1     0.030528  0.142974  0.155869 -0.254003 -0.085751  0.060944 -0.301083
2     0.055797  0.122427  0.141543 -0.273211 -0.064552  0.085041 -0.277434
3     0.015827  0.117834  0.144605 -0.264616 -0.094843  0.066059 -0.290649
4     0.047610  0.110633  0.123310 -0.249639 -0.063271  0.079041 -0.255950
…          …         …         …         …         …         …         …
4879  0.007545  0.012779  0.029744 -0.039043 -0.013837  0.008725 -0.048939
4880  0.008511  0.009776  0.013976 -0.016687  0.000624  0.005357 -0.029979
4881  0.008130  0.012472  0.015647 -0.019275 -0.003679  0.000071 -0.029158
4882 -0.000147  0.001432  0.015575 -0.014776  0.000100  0.010989 -0.021194
4883 -0.001116  0.019482  0.019075 -0.039853 -0.005985  0.017334 -0.053835

[4884 rows x 150 columns]
```

0.7 FastText

[79]:
```
pip install gensim
```

[80]:
```
from gensim.models import FastText
from nltk.tokenize import word_tokenize
```

```
# Tokenisasi setiap kalimat menjadi daftar kata-kata
tokenized_sentences = [word_tokenize(sentence) for sentence in df['Kalimat']]

# Buat dan latih model FastText
model_fasttext = FastText(sentences=tokenized_sentences, vector_size=150,␣
↪window=25, min_count=1, workers=4)
```

[81]:

```
words=list(model_fasttext.wv.index_to_key)
vector_FastText = [model_fasttext.wv[word] for word in words]
vector_FastText = np.array(vector_FastText)
```

[114]:

```
words[1:10]
```

[114]:

```
['raja', 'si', 'sang', 'kancil', 'anak', 'puteri', 'istana', 'a', 'orang']
```

[83]:

```
vec_fasttext=pd.DataFrame(vector_FastText)
print(vec_fasttext)
```

```
            0         1         2         3         4         5         6    \
0    -0.178200 -0.297179  0.416828  0.286388 -0.291728  0.215651 -0.890209
1    -0.148360 -0.247674  0.347771  0.241316 -0.241357  0.179396 -0.746582
2    -0.146200 -0.237473  0.333681  0.232010 -0.230617  0.174016 -0.713181
3    -0.429933 -0.717781  1.008868  0.694313 -0.703807  0.517780 -2.159927
4    -0.153019 -0.253026  0.355569  0.244433 -0.247884  0.182203 -0.760185
…          …         …         …         …         …         …         …
4879 -0.250210 -0.418123  0.586246  0.401740 -0.407185  0.301244 -1.255674
4880 -0.215448 -0.358584  0.503411  0.346557 -0.351662  0.259069 -1.077641
4881 -0.254315 -0.421427  0.594860  0.409058 -0.413853  0.304118 -1.272054
4882 -0.165540 -0.273895  0.381821  0.264429 -0.268952  0.197484 -0.817961
4883 -0.203141 -0.339625  0.476911  0.328276 -0.332267  0.245810 -1.020019

            7         8         9    …       140       141       142  \
0     0.602122  0.504625 -0.475022  …  0.551462 -0.047902 -0.245312
1     0.504554  0.420944 -0.396023  …  0.458589 -0.042299 -0.206651
2     0.480861  0.402895 -0.377548  …  0.440688 -0.041231 -0.197744
3     1.457097  1.222325 -1.146647  …  1.327358 -0.121381 -0.595702
4     0.516050  0.431325 -0.403533  …  0.469046 -0.042055 -0.208902
…          …         …         …  …       …         …         …
4879  0.848179  0.710061 -0.666694  …  0.771074 -0.069880 -0.347329
4880  0.729077  0.610061 -0.572947  …  0.663378 -0.059597 -0.297838
4881  0.860089  0.719064 -0.677843  …  0.781292 -0.072816 -0.352528
4882  0.554060  0.463071 -0.435398  …  0.504297 -0.045734 -0.226473
4883  0.687924  0.577472 -0.541799  …  0.627708 -0.055685 -0.280230

            143       144       145       146       147       148       149
0    -0.478233 -0.729434  0.496810  0.264620  0.516047  0.329265  0.113297
```

```
1    -0.402035 -0.608689  0.416425  0.222151  0.429514  0.273732  0.096912
2    -0.381615 -0.580359  0.396436  0.212583  0.414953  0.257949  0.093763
3    -1.160954 -1.763169  1.202816  0.639861  1.249053  0.790143  0.279051
4    -0.408960 -0.619569  0.421907  0.224369  0.442001  0.277554  0.098830
…         …         …         …         …         …         …         …
4879 -0.674388 -1.025524  0.701286  0.373695  0.726349  0.457734  0.164036
4880 -0.579651 -0.879524  0.601476  0.319963  0.626391  0.394456  0.139827
4881 -0.684878 -1.037363  0.709087  0.377105  0.738010  0.465405  0.163388
4882 -0.439866 -0.668839  0.456593  0.242839  0.476785  0.299077  0.105485
4883 -0.548321 -0.833312  0.568936  0.301334  0.590718  0.373504  0.132052

[4884 rows x 150 columns]
```

[84]:

```python
# representasi kata 'malin' pada w2v dan fasttext
word_w2v=model_w2v.wv['malin']
word_fasttext=model_fasttext.wv['malin']

print('Word2Vec:', word_w2v)
print('FastText:', word_fasttext)
```

```
Word2Vec: [-0.04396357 -0.03990632 -0.02886433 -0.105688    0.20746963
-0.22636747
 -0.06051996  0.22015017 -0.00626191 -0.01871684  0.24128166  0.02274396
 -0.18446083  0.40127742 -0.3042639   0.05701965  0.02058132  0.05361719
 -0.14848596  0.02885894 -0.21700425  0.02378188  0.1296603  -0.15756293
 -0.09710332  0.05980825 -0.2747287  -0.16497077 -0.05537954 -0.06704765
  0.00820155 -0.08461739 -0.19469045 -0.15234974 -0.08757959 -0.01760192
  0.38036376  0.09263624 -0.00243479 -0.31509435  0.14602499  0.17079453
 -0.19477265  0.06884856  0.22924139 -0.04581464  0.21880174 -0.17425922
  0.17704318 -0.00878582 -0.1227864   0.11435875 -0.12269876  0.19902974
 -0.14028524  0.16899943 -0.23505831  0.13680042 -0.00946294 -0.08574829
 -0.00688985 -0.286034    0.05701552 -0.00899219  0.1680523  -0.01705234
  0.15044071 -0.31996137 -0.31752053 -0.2937455  -0.03174764  0.16265854
 -0.0750914  -0.28581375 -0.15067077  0.29546362 -0.2196337  -0.18551153
 -0.22943309  0.20156302 -0.07453801 -0.03514314 -0.10596927  0.39787322
 -0.09598821  0.0451581   0.0277281   0.03214654  0.20295654  0.09352777
  0.06707703 -0.05300888  0.16502759  0.02147523  0.15065025  0.00915599
  0.23227382  0.01339632 -0.04557454  0.13596311 -0.13462412  0.10144224
 -0.06141223 -0.01026159 -0.04834579 -0.07432263 -0.00926503  0.03831899
 -0.35667896  0.0814826  -0.273333   -0.00552497  0.06449135 -0.20860922
  0.03526349  0.163071   -0.02754649  0.10647973  0.05236599 -0.07729051
 -0.21222548  0.22710837  0.11497246 -0.2670516  -0.0337228   0.14642315
  0.30883402 -0.14054032 -0.16260788 -0.08393211  0.25698152  0.03378035
  0.15557295  0.17847705  0.02999945  0.18676454 -0.43156093 -0.2173626
  0.02173061 -0.0772697  -0.03407215 -0.08313769  0.1633587   0.00551542
  0.08984667  0.15203469 -0.27894753 -0.05687897  0.04583459 -0.30736667]
FastText: [-1.34095877e-01 -2.21994683e-01  3.12383026e-01  2.13798150e-01
 -2.16432020e-01  1.61237940e-01 -6.64784372e-01  4.50500339e-01
```

```
 3.78056943e-01 -3.52634758e-01  4.30979848e-01 -8.10823366e-02
 4.71930534e-01  2.97945857e-01 -3.56768668e-01 -2.16335699e-01
 2.40474284e-01  9.38376598e-03 -9.11095813e-02 -5.69938794e-02
-3.89027953e-01 -2.85665631e-01  9.67324227e-02  1.24272473e-01
-5.55740535e-01  3.02484393e-01 -5.63721776e-01  3.96856815e-01
 7.96611607e-02 -5.31737030e-01  2.63936788e-01 -2.38479957e-01
 1.53560624e-01 -3.87853622e-01 -4.24298346e-01 -2.59824932e-01
-2.77469277e-01  1.26818614e-03 -1.94101438e-01 -9.57328603e-02
-2.46095017e-01 -1.91898420e-01 -1.85705632e-01  3.99361163e-01
 5.19177258e-01  2.80114233e-01 -1.09597176e-01 -1.66285224e-02
 8.07098448e-02  5.01330435e-01 -3.66217524e-01 -2.81573832e-01
 3.35933790e-02 -4.14694905e-01  4.88706499e-01  5.76640546e-01
-6.73340484e-02 -2.37309709e-01  1.57276601e-01 -7.51707777e-02
-1.09208096e-02 -1.82809517e-01  8.04862753e-02  1.96561486e-01
-9.88412276e-03  8.76134168e-03 -5.79998255e-01 -5.38905933e-02
 3.29787195e-01  3.17839235e-02 -2.03207005e-02  1.10819772e-01
 4.62651759e-01  5.16397471e-04 -3.18357438e-01  6.03758037e-01
 6.03425913e-02  2.83869326e-01 -5.31369336e-02  2.72854835e-01
-5.31443357e-02  3.99411500e-01  3.87507319e-01  5.95565438e-02
 1.33609146e-01  2.18366727e-01 -1.26710474e-01 -1.58616304e-01
 9.19022858e-02 -9.72179249e-02  2.40686402e-01 -3.54764372e-01
 1.03570022e-01  1.38315156e-01  7.14682400e-01 -5.64116418e-01
 2.39079729e-01  3.08074594e-01  2.53283262e-01  5.48247933e-01
-2.91211128e-01  1.07731268e-01  4.85109746e-01 -1.23702928e-01
 3.49814802e-01 -1.23520643e-01 -3.08917612e-01  4.31339443e-01
 3.46975267e-01  2.28485689e-01  1.19652830e-01  2.61803508e-01
-5.76213636e-02 -5.85290529e-02  6.42392114e-02  3.97735447e-01
 2.70326704e-01  8.79972428e-02 -6.37173653e-01 -1.02746695e-01
-3.52462083e-01  2.90895730e-01 -2.62654692e-01 -1.19382888e-01
 4.56914216e-01  4.77018088e-01  4.23555106e-01 -2.24181488e-01
-7.58978724e-01  1.69635490e-01  7.71470740e-02 -4.69627917e-01
 1.88917473e-01  7.58859366e-02  8.12632516e-02  4.96749490e-01
-7.89374486e-02 -1.53566882e-01  3.36284429e-01  3.09922360e-02
 4.09346282e-01 -3.64326425e-02 -1.82975501e-01 -3.59290242e-01
-5.43712795e-01  3.70073706e-01  1.95685178e-01  3.87136906e-01
 2.42917806e-01  8.29681158e-02]
```

Pada setiap model, setiap kata akan direpresentasikan dengan dimensi vektor sebanyak 150. Sesuai dengan ukuran vektor yang telah ditetapkan di awal saat model dibangun.

[89]:
```python
similar_words_w2v= model_w2v.wv.most_similar('bondowoso', topn=4)
similar_words_fasttext = model_fasttext.wv.most_similar('bondowoso', topn=4)

print(f'Word2Vec - kata serupa dengan "bondowoso":{similar_words_w2v}')
print(f'FastText - kata serupa dengan "bondowoso":{similar_words_fasttext}')
```

```
Word2Vec - kata serupa dengan "bondowoso":[('loro', 0.9985447525978088),
('bandung', 0.9984880685806274), ('biji', 0.9984799027442932), ('jin',
```

```
0.9984443783760071)]
FastText - kata serupa dengan "bondowoso":[('berbondongbondong',
0.9999478459358215), ('diperintah', 0.9999463558197021), ('jonggrang',
0.9999458193778992), ('memainkan', 0.9999455809593201)]
```

Pada kasus ini, baik model Word2Vec atau model FastText menunjukan hasil yang sesuai (kata yang mirip memang konteksnya mendekati satu sama lain) walaupun perlu beberapa peningkatan agar hasilnya lebih akurat.

[96]:
```
# untuk model word2vec pre-trained
# inspeksi model
print(f'ukuran vektor: {model_w2v.vector_size}')
print(f'ukuran window: {model_fasttext.window}')
print(f'jumlah kata vektor: {model_w2v.wv}')
print(f'parameter W2V: {model_w2v}')
```

```
ukuran vektor: 150
ukuran window: 25
jumlah kata vektor: KeyedVectors<vector_size=150, 4884 keys>
parameter W2V: Word2Vec<vocab=4884, vector_size=150, alpha=0.025>
```

[94]:
```
# untuk model wfasttext pre-trained
# inspeksi model
print(f'ukuran vektor: {model_fasttext.vector_size}')
print(f'ukuran window: {model_fasttext.window}')
print(f'jumlah kata vektor: {model_fasttext.wv}')
print(f'parameter FastText: {model_fasttext}')
```

```
ukuran vektor: 150
ukuran vektor: 25
jumlah kata vektor: FastTextKeyedVectors<vector_size=150, 4884 keys>
parameter FastText: FastText<vocab=4884, vector_size=150, alpha=0.025>
```

Kedua model, menggunakan ukuran vektor yang sama yaitu 150 dan ukuran *window* yang sama yaitu 25, ukuran *window* 25 ditetapkan agar model bisa lebih memahami konteks tanpa menangkap konteks yang terlalu lebar (dengan *window* yang terlalu besar)