Doc2vec vectorization

Created by: Aditi Das

From: SVNIT, Gujarat

Mtech Data Science - p23ds008 (2023-25)

Subject: NLP Project

Last Updated: 29/03/2024

Importing Libraries

In [1]:
```
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
import string
```

Downloading NLTK Resources for performing Text Preprocessing

In [2]:
```
# Download NLTK resources
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[2]:
```
True
```

Loading the "train.csv" Dataset

In [3]:
```
# Load dataset
df = pd.read_csv("train.csv")
df.head()
```

Out[3]:

	tweets	class
0	Be aware dirty step to get money #staylight ...	figurative
1	#sarcasm for #people who don't understand #diy...	figurative
2	@IminworkJeremy @medsingle #DailyMail readers ...	figurative
3	@wilw Why do I get the feeling you like games?...	figurative
4	-@TeacherArthurG @rweingarten You probably jus...	figurative

Applying Text Preprocessing Function

In [4]:
```
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import string

# Download NLTK resources
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

# Load dataset
df = pd.read_csv("train.csv")

# Text preprocessing function
def preprocess_text(text):
    # Lowercasing
    text = text.lower()

    # Tokenization
    tokens = word_tokenize(text)

    # Removing punctuation and special characters
    table = str.maketrans('', '', string.punctuation)
    tokens = [token.translate(table) for token in tokens]
    tokens = [token for token in tokens if token.isalnum()]

    # Removing numbers
    tokens = [token for token in tokens if not token.isdigit()]

    # Removing stopwords
    stop_words = set(stopwords.words('english'))
    tokens = [token for token in tokens if token not in stop_words]

    # Lemmatization
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(token) for token in tokens]

    # Join tokens back into a string
    preprocessed_text = ' '.join(tokens)

    return preprocessed_text

# Apply preprocessing to the 'tweets' column
df['tweets_preprocessed'] = df['tweets'].apply(preprocess_text)

# Save preprocessed tweets and class labels to a new CSV file
df[['tweets_preprocessed', 'class']].to_csv("preprocessed_train_data.csv", ind

# Display the preprocessed data
print("Preprocessed data saved successfully!")
```

```
[nltk_data] Downloading package punkt to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to C:\Users\Harsh
[nltk_data]     Bari\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!

Preprocessed data saved successfully!
```

Displaying the Preprocessed Saved Train csv File

In [5]:
```
# Load the saved CSV file
preprocessedTrainData_df = pd.read_csv("preprocessed_train_data.csv")

# Display the content of the DataFrame
preprocessedTrainData_df.head()
```

Out[5]:

	tweets_preprocessed	class
0	aware dirty step get money staylight staywhite...	figurative
1	sarcasm people nt understand diy artattack htt...	figurative
2	iminworkjeremy medsingle dailymail reader sens...	figurative
3	wilw get feeling like game sarcasm	figurative
4	teacherarthurg rweingarten probably missed tex...	figurative

Doc2Vec Embedding

In [6]:
```
import pandas as pd
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
from nltk.tokenize import word_tokenize

# Load the preprocessed data from CSV
df = pd.read_csv("preprocessed_train_data.csv")

# Define a function to tokenize the preprocessed text
def tokenize_text(text):
    return word_tokenize(text)

# Tokenize the preprocessed text
df['tokens'] = df['tweets_preprocessed'].apply(tokenize_text)

# Create TaggedDocuments from tokenized text
tagged_data = [TaggedDocument(words=doc, tags=[i]) for i, doc in enumerate(df[

# Train Doc2Vec model
vector_size = 150
model = Doc2Vec(vector_size=vector_size, min_count=1, epochs=40)
model.build_vocab(tagged_data)
model.train(tagged_data, total_examples=model.corpus_count, epochs=model.epoch

# Extract Doc2Vec embeddings
doc2vec_embeddings = [model.infer_vector(doc) for doc in df['tokens']]

# Add Doc2Vec embeddings to DataFrame
for i in range(vector_size):
    df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]

# Display the DataFrame with Doc2Vec embeddings
df.head()
```

```
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
```

```
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
```

```
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
```

```
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
```

```
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
```

```
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
```

```
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
```

```
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
C:\Users\Harsh Bari\AppData\Local\Temp\ipykernel_19952\4286379589.py:29: Perf
ormanceWarning: DataFrame is highly fragmented.  This is usually the result o
f calling `frame.insert` many times, which has poor performance.  Consider jo
ining all columns at once using pd.concat(axis=1) instead. To get a de-fragme
nted frame, use `newframe = frame.copy()`
  df[f"doc2vec_{i+1}"] = [embedding[i] for embedding in doc2vec_embeddings]
```

Out[6]:

	tweets_preprocessed	class	tokens	doc2vec_1	doc2vec_2	doc2vec_3	doc2vec_4
0	aware dirty step get money staylight staywhite...	figurative	[aware, dirty, step, get, money, staylight, st...	0.094989	-0.130547	0.209554	-0.361079
1	sarcasm people nt understand diy artattack htt...	figurative	[sarcasm, people, nt, understand, diy, artatta...	0.141715	-0.156211	0.124129	-0.033891
2	iminworkjeremy medsingle dailymail reader sens...	figurative	[iminworkjeremy, medsingle, dailymail, reader,...	0.071912	-0.202637	0.029908	-0.157918
3	wilw get feeling like game sarcasm	figurative	[wilw, get, feeling, like, game, sarcasm]	-0.180418	0.049932	-0.014645	0.068700
4	teacherarthurg rweingarten probably missed tex...	figurative	[teacherarthurg, rweingarten, probably, missed...	0.315940	-0.020790	-0.059559	-0.123925

5 rows × 153 columns

Saving the Resultant Doc2Vec Embedded Text into a csv File

In [7]:
```
# Save the DataFrame to a CSV file
df.to_csv("resultant_train_dataframe_with_doc2vec.csv", index=False)
```

Displaying the Saved csv File

In [8]:
```
# Load the saved CSV file
Doc2Vec_df = pd.read_csv("resultant_train_dataframe_with_doc2vec.csv")

# Display the content of the DataFrame
Doc2Vec_df.head()
```

Out[8]:

	tweets_preprocessed	class	tokens	doc2vec_1	doc2vec_2	doc2vec_3	doc2vec_
0	aware dirty step get money staylight staywhite...	figurative	['aware', 'dirty', 'step', 'get', 'money', 'st...	0.094989	-0.130547	0.209554	-0.36107
1	sarcasm people nt understand diy artattack htt...	figurative	['sarcasm', 'people', 'nt', 'understand', 'diy...	0.141715	-0.156211	0.124129	-0.03389
2	iminworkjeremy medsingle dailymail reader sens...	figurative	['iminworkjeremy', 'medsingle', 'dailymail', '...	0.071912	-0.202637	0.029908	-0.15791
3	wilw get feeling like game sarcasm	figurative	['wilw', 'get', 'feeling', 'like', 'game', 'sa...	-0.180418	0.049932	-0.014645	0.06870
4	teacherarthurg rweingarten probably missed tex...	figurative	['teacherarthurg', 'rweingarten', 'probably', ...	0.315940	-0.020790	-0.059559	-0.12392

5 rows × 153 columns