```
!pip install datasets
```

```
Collecting datasets
  Downloading datasets-2.17.0-py3-none-any.whl (536 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 536.6/536.6 kB 3.4 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from datasets) (3.13.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from datasets) (1.23.5)
Collecting pyarrow>=12.0.0 (from datasets)
  Downloading pyarrow-15.0.0-cp310-cp310-manylinux_2_28_x86_64.whl (38.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 38.3/38.3 MB 17.8 MB/s eta 0:00:00
Requirement already satisfied: pyarrow-hotfix in /usr/local/lib/python3.10/dist-packages (from datasets) (0.6)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl (116 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 10.0 MB/s eta 0:00:00
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from datasets) (1.5.3)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.10/dist-packages (from datasets) (2.31.0)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.10/dist-packages (from datasets) (4.66.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.10/dist-packages (from datasets) (3.4.1)
Collecting multiprocess (from datasets)
  Downloading multiprocess-0.70.16-py310-none-any.whl (134 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.8/134.8 kB 14.0 MB/s eta 0:00:00
Requirement already satisfied: fsspec[http]<=2023.10.0,>=2023.1.0 in /usr/local/lib/python3.10/dist-packages (from datasets) (2023.6
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-packages (from datasets) (3.9.3)
Requirement already satisfied: huggingface-hub>=0.19.4 in /usr/local/lib/python3.10/dist-packages (from datasets) (0.20.3)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from datasets) (23.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from datasets) (6.0.1)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (4.0.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.19.4->
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets) (2.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets) (2024
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->datasets) (2023.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas->datasets)
Installing collected packages: pyarrow, dill, multiprocess, datasets
  Attempting uninstall: pyarrow
    Found existing installation: pyarrow 10.0.1
    Uninstalling pyarrow-10.0.1:
      Successfully uninstalled pyarrow-10.0.1
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
ibis-framework 7.1.0 requires pyarrow<15,>=2, but you have pyarrow 15.0.0 which is incompatible.
pandas-gbq 0.19.2 requires google-auth-oauthlib>=0.7.0, but you have google-auth-oauthlib 0.4.6 which is incompatible.
Successfully installed datasets-2.17.0 dill-0.3.8 multiprocess-0.70.16 pyarrow-15.0.0
```

```
import pandas as pd
import re
import nltk
```

```
from datasets import load_dataset

dataset = load_dataset("mystic-leung/medical_cord19")
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88: UserWarni
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public mo
  warnings.warn(
```

Downloading readme: 100% 194/194 [00:00<00:00, 9.01kB/s]

Downloading data: 100% 331M/331M [00:28<00:00, 14.3MB/s]

Downloading data: 100% 70.8M/70.8M [00:06<00:00, 11.1MB/s]

Downloading data: 100% 70.8M/70.8M [00:06<00:00, 11.1MB/s]

```
dataset
```

```
DatasetDict({
    train: Dataset({
        features: ['input', 'output'],
        num_rows: 210000
```

```
    })
    validation: Dataset({
        features: ['input', 'output'],
        num_rows: 45000
    })
    test: Dataset({
        features: ['input', 'output'],
        num_rows: 45000
    })
})
```

```
train_dataset = dataset['train']
```

```
train_dataset
```

```
Dataset({
    features: ['input', 'output'],
    num_rows: 210000
})
```

```
print("Features: ",train_dataset.features)
print("Number of Rows: ",train_dataset.num_rows)
```

```
Features:  {'input': Value(dtype='string', id=None), 'output': Value(dtype='string', id=None)}
Number of Rows:  210000
```

Data Preprocessing

Converting the dictionary to a Pandas Dataframe

```
train_df_full = pd.DataFrame(train_dataset)
```

```
train_df = train_df_full.head(1000)
```

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   input   1000 non-null   object
 1   output  1000 non-null   object
dtypes: object(2)
memory usage: 15.8+ KB
```

```
train_df.drop_duplicates(inplace = True)
```

```
<ipython-input-38-6f145123ae15>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
  train_df.drop_duplicates(inplace = True)
```

Text cleaning

Removing punctuations, Special characters, Email, Hashtags, Usernames,Leading white spaces,Case conversion,

```
def preprocessing_1(data:str):
  data = data.strip()
  data = data.lower()
  data = re.sub(r'\s+', ' ', data)
  url_pattern = re.compile(r"https?://\S+|www\.\S+")
  data = re.sub(url_pattern, "", data)
  username_pattern = re.compile(r"@\w+")
  data = re.sub(username_pattern, "", data)
  hashtag_pattern = re.compile(r"#\w+")
  data = re.sub(hashtag_pattern, "", data)
  data = re.sub(r"([a-zA-Z])\1{2,}", r'\1', data)
  data = re.sub(r'[^a-zA-Z\s]',"",data)#Remove special characters
  return data
```

```
#Assingning this way always converts to Series.Best to avaoid this method and instead use slicing
####input_data = train_df['input']# Similar to X
####output_data = train_df['output']# Similar to Y
X = pd.DataFrame()
Y = pd.DataFrame()
```

```
X['text'] = train_df['input'].apply(preprocessing_1)# Similar to X
Y['summary']=train_df['output'].apply(preprocessing_1)
```

```
X
```

	text
0	cardiovascular disease is the leading cause of...
1	novel coronavirus disease covid continues to ...
2	ethnopharmacological relevance the subtribe hy...
3	we estimate the short to medium term impact of...
4	objectives to report epidemiological features ...
...	...
995	background nurses and midwives have a professi...
996	containment measures adopted to reduce the spr...
997	a global public health crisis caused by the n...
998	exposure from the dissolvedphase and through f...
999	the pandemic from covid causes a health threat...

1000 rows × 1 columns

Tokenization,Removing stopwords etc

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

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
True
```

```
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
def preprocessing_2(data:str):
  data = nltk.word_tokenize(data.lower())
  def get_pos(word):
        tag = nltk.pos_tag([word])[0][1].upper()
        tag_dict = {"N":"n","V":"v","R":"r","J":"a"}
        return tag_dict.get(tag,"n")
  lemma = nltk.stem.WordNetLemmatizer()
  data = [lemma.lemmatize(word,pos=get_pos(word))for word in data]
  data = [word for word in data if word.isalnum() and word not in stop_words]
  return data
```

```
X['text'] = train_df['input'].apply(preprocessing_2)# Similar to X
Y['summary']=train_df['output'].apply(preprocessing_2)
```

```
X['text'] = X['text'].apply(lambda x : " ".join(x))
Y['summary'] = Y['summary'].apply(lambda x : " ".join(x))
```

```
X
```

	text
0	cardiovascular disease leading cause death glo...
1	novel coronavirus disease 2019 continues affec...
2	ethnopharmacological relevance subtribe hyptid...
3	estimate medium term impact six major past pan...
4	objective report epidemiological feature clini...
...	...
995	background nurse midwife professional obligati...
996	containment measure adopted reduce spread coro...
997	global public health crisis caused 2019 novel ...
998	exposure contributes bioaccumulation polycycli...
999	pandemic cause health threat many country requ...

1000 rows × 1 columns

```
print(X['text'][1])
```

```
novel coronavirus disease 2019 continues affect pregnant woman concern adverse maternal fetal outcome rapidly spreading throughout r
```

```
print(Y['summary'][1])
```

```
consequence disease maternal perinatal neonatal outcome retrospective observational cohort study
```

```
Y[:1]
```

	summary
0	medication adherence cardiovascular medicine

```
#X['text'] = X['text'].apply(lambda x: " ".join(["".join(sentence) for sentence in x]))
#Y['summary'] = Y['summary'].apply(lambda x: " ".join([" ".join(sentence) for sentence in x]))
```

```
for i in range(5):
    print("Review:",X['text'][i])
    print("Summary:",Y['summary'][i])
    print("\n")
```

```
Review: cardiovascular disease leading cause death globally pharmacological advancement improved morbidity mortality associated carc
Summary: medication adherence cardiovascular medicine


Review: novel coronavirus disease 2019 continues affect pregnant woman concern adverse maternal fetal outcome rapidly spreading thro
Summary: consequence disease maternal perinatal neonatal outcome retrospective observational cohort study


Review: ethnopharmacological relevance subtribe hyptidinae contains approximately 400 accepted specie distributed 19 genus hyptis er
Summary: subtribe hyptidinae promising source bioactive metabolite


Review: estimate medium term impact six major past pandemic crisis co2 emission energy transition renewable electricity result show
Summary: impact past pandemic co emission transition renewable energy


Review: objective report epidemiological feature clinical characteristic outcome human rhinovirus hrv infection comparison community
Summary: impact seasonality human rhinovirus infection hospitalized patient two consecutive year
```

```
input = nltk.word_tokenize(X['text'][0])
```

```
input
```

```
['cardiovascular',
 'disease',
 'leading',
 'cause',
 'death',
```

```
'globally',
'pharmacological',
'advancement',
'improved',
'morbidity',
'mortality',
'associated',
'cardiovascular',
'disease',
'prescribed',
'treatment',
'remains',
'significant',
'barrier',
'improved',
'patient',
'outcome',
'variety',
'strategy',
'improve',
'medication',
'adherence',
'tested',
'clinical',
'trial',
'include',
'following',
'category',
'improving',
'patient',
'education',
'implementing',
'medication',
'reminder',
'testing',
'cognitive',
'behavioral',
'intervention',
'reducing',
'medication',
'cost',
'utilizing',
'healthcare',
'team',
'member',
'streamlining',
'medication',
'dosing',
'regimen',
'review',
'describe',
'specific',
```

```
output = nltk.word_tokenize(Y['summary'][0])
```

```
output
```

```
['medication', 'adherence', 'cardiovascular', 'medicine']
```

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

TF - IDF Vectorizer

```
tfidf = TfidfVectorizer()

result_1 = tfidf.fit_transform(input)
result_2 = tfidf.transform(output)
tfidf_embeddings_1 = result_1.toarray()
tfidf_embeddings_2 = result_2.toarray()
#print('\nInput Column : idf values:')
for ele1, ele2 in zip(tfidf.get_feature_names_out(), tfidf.idf_):
  print(ele1, ':', ele2)
#print(tfidf_embeddings_1)
#print(tfidf_embeddings_2)
#print('\nOutput Column :idf values:')
for ele1, ele2 in zip(tfidf.get_feature_names_out(), tfidf.idf_):
    print(ele1, ':', ele2)
print(tfidf_embeddings_1)
```

```
adherence : 3.970414465569701
advancement : 4.663561646129646
```

```
also : 4.663561646129646
associated : 4.663561646129646
barrier : 4.663561646129646
behavioral : 4.663561646129646
cardiovascular : 3.970414465569701
category : 4.258096538021482
cause : 4.663561646129646
clinical : 4.663561646129646
cognitive : 4.663561646129646
cost : 4.663561646129646
death : 4.663561646129646
describe : 4.663561646129646
disease : 3.970414465569701
dosing : 4.663561646129646
education : 4.663561646129646
examine : 4.663561646129646
following : 4.663561646129646
future : 4.663561646129646
globally : 4.663561646129646
healthcare : 4.663561646129646
highlight : 4.663561646129646
impact : 4.663561646129646
implementing : 4.663561646129646
improve : 4.663561646129646
improved : 4.258096538021482
improving : 4.258096538021482
include : 4.663561646129646
inquiry : 4.663561646129646
intervention : 4.663561646129646
leading : 4.663561646129646
line : 4.663561646129646
medication : 3.4107986776342782
member : 4.663561646129646
morbidity : 4.663561646129646
mortality : 4.663561646129646
ongoing : 4.663561646129646
outcome : 4.663561646129646
patient : 3.970414465569701
pharmacological : 4.663561646129646
prescribed : 4.663561646129646
reducing : 4.663561646129646
regimen : 4.663561646129646
remains : 4.663561646129646
reminder : 4.663561646129646
review : 4.663561646129646
significant : 4.663561646129646
specific : 4.663561646129646
strategy : 4.663561646129646
streamlining : 4.663561646129646
team : 4.663561646129646
tested : 4.663561646129646
testing : 4.663561646129646
treatment : 4.663561646129646
trial : 3.970414465569701
utilizing : 4.663561646129646
variety : 4.663561646129646
```

```
!pip install -U altair
```

```
Requirement already satisfied: altair in /usr/local/lib/python3.10/dist-packages (4.2.2)
Collecting altair
  Downloading altair-5.2.0-py3-none-any.whl (996 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 996.9/996.9 kB 4.2 MB/s eta 0:00:00
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from altair) (3.1.3)
Requirement already satisfied: jsonschema>=3.0 in /usr/local/lib/python3.10/dist-packages (from altair) (4.19.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from altair) (1.23.5)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from altair) (23.2)
Requirement already satisfied: pandas>=0.25 in /usr/local/lib/python3.10/dist-packages (from altair) (1.5.3)
Requirement already satisfied: toolz in /usr/local/lib/python3.10/dist-packages (from altair) (0.12.1)
Requirement already satisfied: typing-extensions>=4.0.1 in /usr/local/lib/python3.10/dist-packages (from altair) (4.9.0)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair) (23.2.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair) (0.33.0
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair) (0.17.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.25->altair) (2.8.2
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.25->altair) (2023.4)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->altair) (2.1.5)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas>=0.25->alta
Installing collected packages: altair
  Attempting uninstall: altair
    Found existing installation: altair 4.2.2
    Uninstalling altair-4.2.2:
      Successfully uninstalled altair-4.2.2
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sou
lida 0.0.10 requires fastapi, which is not installed.
lida 0.0.10 requires kaleido, which is not installed.
lida 0.0.10 requires python-multipart, which is not installed.
lida 0.0.10 requires uvicorn, which is not installed.
Successfully installed altair-5.2.0
```

```
print(len(tfidf_embeddings_1))
```

```
77
```

```
print(len(tfidf_embeddings_2))
```

```
4
```

Bag of Words

```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer()
bow_matrix_1 = vectorizer.fit_transform(input)
input_bow = vectorizer.transform()
bow_embeddings_1 = bow_matrix_1.toarray()
print(bow_embeddings_1)
from sklearn.feature_extraction.text import CountVectorizer
bow_matrix_2= vectorizer.fit_transform(output)
bow_embeddings_2 = bow_matrix_2.toarray()
print(bow_embeddings_2)
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 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 0]
 [1 0 0 0]
 [0 1 0 0]
 [0 0 0 1]]
```

```
from  sklearn.metrics.pairwise import cosine_similarity
```

```
cs = cosine_similarity(tfidf_embeddings_1, tfidf_embeddings_1)
```

```
cs
```

```
array([[1., 0., 0., ..., 0., 1., 0.],
       [0., 1., 0., ..., 0., 0., 1.],
       [0., 0., 1., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 1., 0., 0.],
       [1., 0., 0., ..., 0., 1., 0.],
       [0., 1., 0., ..., 0., 0., 1.]])
```

Word2Vec

```
from gensim.models import Word2Vec

word2vec_model_1 = Word2Vec([input], vector_size=200, window=7, min_count=1, workers=4)
word2vec_embeddings_1 = [word2vec_model_1.wv[word] for word in input]
print(word2vec_embeddings_1)
print("CBOW input Word Vectors:")
for word in word2vec_model_1.wv.index_to_key:
    print(word, ':', word2vec_model_1.wv[word])

word2vec_model_2 = Word2Vec([output], vector_size=200, window=7, min_count=1, workers=4)
word2vec_embeddings_2 = [word2vec_model_2.wv[word] for word in output]
print(word2vec_embeddings_1)
print("CBOW input Word Vectors:")
for word in word2vec_model_1.wv.index_to_key:
    print(word, ':', word2vec_model_1.wv[word])
```

```
Streaming output truncated to the last 5000 lines.
        2.73144874e-03, -4.88970662e-04,  1.46124672e-04,  1.07236300e-03,
        3.64232506e-03,  8.67599389e-04,  1.49445295e-05,  2.62066396e-03,
       -1.14890805e-03,  4.91765328e-03,  1.44274859e-03,  1.43793947e-03,
        1.79356057e-03, -1.33768225e-03,  3.00990720e-03, -3.23401368e-03,
        3.67726176e-03, -3.96257453e-03,  2.25142910e-04,  1.90282718e-03,
       -4.27433476e-03, -2.45355093e-03, -3.12529551e-03, -2.21793377e-03,
        4.85354755e-03, -6.03405351e-04,  6.59228361e-04,  3.02102242e-04,
        3.71419312e-03, -8.69793002e-04,  2.62665120e-03,  4.52925358e-03,
        3.51981423e-03, -2.01778719e-03, -1.87341985e-03,  1.01616792e-03,
```

```
         -4.45311842e-03, -4.84207738e-03, -1.65213854e-03, -4.38045943e-03,
          1.89710304e-03, -3.00770835e-03,  1.61952735e-03, -2.18944717e-03,
          3.62433121e-03, -3.28776572e-04, -9.08664835e-04,  7.25332531e-04,
         -1.03188970e-03, -3.35709797e-03,  1.42141664e-03,  7.82564515e-04,
         -8.42687616e-04, -4.19954071e-03, -1.89533562e-03, -1.07767619e-03,
          4.53935796e-03,  2.98528094e-03,  4.05227952e-03, -2.24618649e-04,
         -4.42429818e-03,  1.68785686e-03, -2.01124093e-03, -2.73194863e-03,
          4.38377232e-04,  2.08066963e-03, -4.13119141e-03,  4.84894961e-03,
          3.60024883e-03,  1.48247648e-03, -3.89444572e-03,  4.79171844e-03,
         -1.82902766e-03,  3.34102544e-03, -5.82411340e-05,  2.69562867e-03,
         -4.97974595e-03, -2.26341726e-04, -1.68913603e-03,  7.52793101e-04,
         -1.85428374e-03, -2.52061035e-03,  1.81353756e-03,  3.76515463e-03,
         -1.23920932e-03, -4.85930406e-03,  4.75250231e-03,  2.98563298e-03,
         -8.56885046e-04,  4.39230632e-03,  8.67392519e-04, -1.66126108e-03,
          1.24680507e-03,  3.01609514e-03, -4.82995575e-03, -7.08100619e-04,
          2.57068477e-03,  3.53983673e-03,  3.45219905e-03,  1.29821431e-03,
          3.40202660e-03, -4.47224127e-03,  2.68770941e-03, -1.27026055e-03,
          1.86251744e-03, -4.81107691e-03, -4.49218322e-03,  2.48919544e-03,
         -5.87784511e-04,  3.69686796e-03, -4.67617344e-03,  4.92656091e-03,
         -2.98282481e-03,  2.78733409e-04,  2.31816573e-03, -1.11581967e-03,
          3.65994871e-03,  2.49042292e-04, -1.46448624e-03, -5.61784487e-04,
         -4.72368859e-03, -2.06430606e-03, -1.96062727e-03,  4.09401581e-03,
         -4.98993974e-03, -1.07052387e-03,  3.68539547e-03,  3.98031808e-03,
         -4.24422696e-03,  6.54536183e-04, -4.64052474e-03,  3.46003706e-03,
          4.82037710e-03,  1.63309590e-03, -4.70119622e-03, -2.97356077e-04,
         -8.87321483e-04,  2.17407406e-03,  3.47646023e-03,  2.24337494e-03,
         -4.60485881e-03,  3.68782342e-03,  1.38792337e-03, -4.00797435e-04,
         -1.43469125e-03,  3.04089452e-04, -2.86902022e-03,  2.66611180e-03,
         -4.51685628e-03,  1.62172801e-04,  1.15681719e-03,  1.21060398e-03,
          4.55438253e-03, -2.62863375e-03,  4.79964353e-03,  4.76399343e-03,
         -3.56088090e-03,  1.37622270e-03,  4.84586321e-03,  1.50215591e-03,
         -1.09111203e-03,  2.43721413e-03,  2.44635786e-03,  2.87179183e-03,
         -1.66354550e-03, -4.02107788e-03, -8.16404994e-04, -1.28580141e-03,
         -1.61792408e-03,  1.00198819e-03, -4.79829591e-03,  4.95467428e-03,
         -8.83693690e-04,  1.70416388e-04,  2.20571505e-03, -4.97141061e-03,
          2.16904702e-03,  4.70774528e-03,  4.46570385e-03,  5.12496277e-04,
         -1.04521867e-03, -4.36690124e-03,  3.15035158e-03, -1.51690177e-03,
          4.32502152e-03,  4.57450189e-03, -3.82153108e-03, -1.67198642e-03,
          2.66192621e-03,  5.44470851e-04, -1.20554038e-03, -1.74746756e-03],
        dtype=float32), array([-2.43397895e-04,  1.17722622e-04,  2.54957401e-03,  4.51032910e-03,
         -4.63713100e-03, -3.56551376e-03,  3.25163547e-03,  4.51285159e-03,
         -2.51505920e-03, -1.87524175e-03,  3.68385087e-03, -7.91448576e-04,
         -2.28501554e-03,  3.30337812e-03, -2.45855120e-03, -8.97895603e-04,
          1.44319597e-03,  5.16223197e-04, -4.14273888e-03, -4.76494711e-03,
          3.67238838e-03,  2.54630856e-03,  3.39660444e-03,  3.99898883e-04,
          3.17395572e-03, -1.69626612e-03, -4.67148609e-04,  2.87643191e-03,
         -3.77776939e-03, -1.97092374e-03, -3.76178417e-03, -4.60139854e-04,
          4.80629224e-03, -3.69808194e-03, -1.16402062e-03, -9.47833061e-04,
```

Continuous Bag of Words (CBOW) (CountVectorizer)

```
from gensim.models import Word2Vec

cbow_model_1 = Word2Vec(sentences=[input], vector_size=200, window=7, sg=0, min_count=1, workers=4)
cbow_embeddings_1 = [cbow_model_1.wv[word] for word in input]
print("CBOW input Word Vectors:")
for word in cbow_model_1.wv.index_to_key:
    print(word, ':', cbow_model_1.wv[word])


cbow_model_2 = Word2Vec(sentences=[output], vector_size=200, window=7, sg=0, min_count=1, workers=4)
cbow_embeddings_2 = [cbow_model_2.wv[word] for word in output]
print("CBOW output Word Vectors:")
for word in cbow_model_2.wv.index_to_key:
    print(word, ':', cbow_model_2.wv[word])
```

```
CBOW input Word Vectors:
medication : [-2.43397895e-04  1.17722622e-04  2.54957401e-03  4.51032910e-03
 -4.63713100e-03 -3.56551376e-03  3.25163547e-03  4.51285159e-03
 -2.51505920e-03 -1.87524175e-03  3.68385087e-03 -7.91448576e-04
 -2.28501554e-03  3.30337812e-03 -2.45855120e-03 -8.97895603e-04
  1.44319597e-03  5.16223197e-04 -4.14273888e-03 -4.76494711e-03
  3.67238838e-03  2.54630856e-03  3.39660444e-03  3.99898883e-04
  3.17395572e-03 -1.69626612e-03 -4.67148609e-04  2.87643191e-03
 -3.77776939e-03 -1.97092374e-03 -3.76178417e-03 -4.60139854e-04
  4.80629224e-03 -3.69808194e-03 -1.16402062e-03 -9.47833061e-04
  4.05259524e-03 -2.99035665e-03  1.09780931e-05 -2.39576166e-03
 -4.80683148e-03  2.50480813e-03 -4.38536284e-03 -2.20403029e-03
  1.88924969e-05 -1.42462290e-04 -3.85450828e-03  4.78526717e-03
  2.51330575e-03  4.61710291e-03 -4.09376854e-03  2.24235514e-03
 -2.07241438e-03  4.17237519e-04  4.27090051e-03 -2.24921876e-03
  2.25209980e-03 -3.40690790e-03 -1.78425375e-03  4.69333772e-03
 -7.88534817e-04  1.37734110e-04 -2.06553284e-03 -3.83562851e-03
 -7.78124609e-04  1.23975298e-03 -4.38545278e-04  2.78540351e-03
 -1.38571335e-03  1.15438027e-03  2.73136212e-03  4.19077557e-03
 -7.14538153e-04 -4.59928019e-03  2.20764824e-03  2.86642520e-04
  3.75571800e-03 -4.28719999e-04 -1.32302812e-03 -4.38319100e-03
```

```
 -4.39121126e-04  1.41345477e-03  2.68909056e-03  3.53308907e-03
 -2.87524308e-03  9.09364200e-04  3.05670057e-03 -2.38309987e-03
 -1.56574394e-03  3.40214814e-03  8.28593154e-04  1.11142916e-04
  1.75784691e-03  1.10478606e-04  4.83453181e-03  2.53630080e-03
 -4.46916185e-03 -3.52568692e-03  4.57892282e-04  3.22038820e-03
 -4.31144377e-03  1.84189912e-03  2.60584126e-03  2.87447963e-03
  3.74397356e-03 -3.10777430e-03  5.74485632e-04  3.02756508e-03
 -1.44191715e-03 -3.11041973e-03 -2.11135295e-04 -4.20971727e-03
 -2.80767889e-03  3.55624827e-03  1.69188564e-03  3.60373151e-03
  3.38935899e-03  3.75725096e-03 -1.89743156e-03 -3.04426212e-04
  1.17409974e-03 -2.23785383e-03  4.19765199e-03 -4.95673670e-03
  3.37124383e-03  1.46347238e-03 -2.45913374e-03  2.20782938e-03
 -8.72436387e-04  3.35487491e-03  5.00583043e-03 -2.18794867e-03
 -2.98978674e-04 -2.85634911e-03  1.92003627e-03  1.40955846e-03
  3.44955968e-03  3.03705735e-03  4.77960985e-03  4.63266019e-03
  3.97557020e-03 -3.51917115e-03 -4.59417980e-03 -1.69987426e-04
 -1.54050183e-03  3.95821454e-03  2.95688468e-03 -7.84859119e-04
  7.73602689e-04  8.82107997e-04  3.92382313e-03 -4.76300390e-03
 -8.56025654e-05  1.74036995e-03 -4.88541089e-04  4.21383325e-03
  4.51653032e-03  3.27931903e-03 -3.58422607e-04  3.86370439e-03
 -4.25695069e-03  1.61562045e-03 -2.33184779e-03 -2.54722708e-03
  1.81438320e-03  2.69636628e-03  3.87762417e-03 -2.88636587e-03
  3.70907574e-03  3.30382166e-03 -1.87198480e-03 -4.37592203e-03
  2.72473064e-03  3.26694781e-03 -3.79896461e-04 -3.34426272e-03
 -3.55738821e-03 -1.23994565e-03  2.58348254e-03 -1.82270515e-03
 -4.69964137e-03  1.91476638e-03  2.43521831e-03 -3.20625957e-03
  6.11510535e-04 -1.03400263e-03  3.57953504e-05 -4.93346481e-03
  1.36854884e-03 -2.38839886e-03  5.61250141e-04 -7.58533657e-04
  1.09596422e-03 -3.95021867e-03 -1.34896149e-03  1.35245186e-03
  2.69268872e-03 -1.20098062e-03 -4.74680401e-03  2.23622331e-03]
cardiovascular : [ 5.5819572e-05  1.5382001e-03 -3.4035724e-03 -6.8600645e-04
  3.8408015e-03  3.6693064e-03 -1.8291188e-03  1.3336225e-03
 -4.1608596e-03  3.1034756e-03 -2.3193690e-03 -1.5921145e-03
  4.6495600e-03  4.4818191e-04  3.7358576e-03 -3.0309972e-03
  2.5796790e-03  4.9703326e-03 -4.2296192e-03 -2.5848746e-03
 -3.5276520e-03 -2.4290096e-03 -1.8822814e-03 -4.2624054e-03
  3.9766789e-03 -2.4183090e-03  4.2147082e-03  2.6268591e-03
```

Skip-gram

```
from gensim.models import Word2Vec

skipgram_model_1 = Word2Vec(sentences=[input], vector_size=200, window=7, sg=1, min_count=1, workers=4)
skipgram_embeddings_1 = [skipgram_model_1.wv[word] for word in input]

print("\nSkip-gram Word Vectors:")
for word in skipgram_model_1.wv.index_to_key:
    print(word, ':', skipgram_model_1.wv[word])

skipgram_model_2 = Word2Vec(sentences=[output], vector_size=200, window=7, sg=1, min_count=1, workers=4)
skipgram_embeddings_2 = [skipgram_model_2.wv[word] for word in output]

print("\nSkip-gram Word Vectors:")
for word in skipgram_model_1.wv.index_to_key:
    print(word, ':', skipgram_model_1.wv[word])
```

```
Streaming output truncated to the last 5000 lines.
  2.52359943e-03  3.26795201e-03  2.15621642e-03 -4.16612579e-03
  1.44511426e-03 -5.81645581e-04 -1.24258036e-03  2.78728991e-03
 -6.80438359e-04  5.10229776e-03 -3.28734022e-04 -1.21548066e-04
 -1.02306413e-03 -4.19426151e-03 -4.80649900e-03  2.58191326e-03
 -3.30611854e-03  1.54280942e-03  1.43309624e-03 -3.59228114e-03
 -1.69031660e-03 -2.96740397e-03 -4.30572871e-03  3.84130180e-05
  5.13059273e-03 -3.57472035e-03 -2.14961520e-03  4.31996258e-03
  4.76301834e-03  4.77772998e-03 -5.03549585e-03  2.38854392e-03
 -3.38240224e-03  3.45636043e-03  3.04419384e-03 -1.73290388e-03
  3.70362308e-03 -3.88695113e-03  2.09534029e-03  2.15477799e-03
  3.08935507e-03  2.97126803e-03 -1.43272954e-03  1.93556119e-03
 -3.06088151e-03 -3.22365202e-03  1.52587309e-03 -4.87774843e-03
 -1.63232777e-04  4.72362532e-04  3.94014409e-03 -3.53522715e-03
 -1.14526448e-03 -2.17681774e-03 -2.57988530e-03 -2.31732687e-04
  2.16885586e-03 -2.46788329e-03 -8.03708390e-04  4.47648298e-03
 -7.39487470e-04 -1.66929315e-03  3.70274158e-03  7.38279894e-04
 -1.17875054e-04  4.78436705e-03  4.23447369e-03  1.70411274e-03
  1.89667800e-03 -1.29248330e-03  4.00650268e-03  2.45350855e-03
 -2.78733368e-03 -3.06073623e-03 -3.95584939e-04 -3.93713638e-03
  1.53752987e-03 -4.73904656e-03 -6.71714195e-04  1.75296923e-03
  1.43463269e-03  1.56716770e-03 -4.45947144e-03  1.82660914e-03
  2.81956908e-03  1.78901502e-03  4.07189876e-03  8.93058721e-04
  3.06205009e-03  4.37871786e-03 -4.53186512e-04 -2.66135728e-04
  1.09547668e-03 -3.71676474e-03  2.86919624e-03 -2.54350482e-03
 -1.55348214e-03 -1.12105394e-03  4.25003935e-03  1.19283912e-03
 -6.07984664e-04 -1.61452964e-03 -1.06016721e-03 -8.51000834e-04
 -2.58320803e-03 -2.02282495e-03  2.01815201e-04  2.20483961e-03
  8.34827835e-04  3.41231981e-03  3.33503034e-04 -1.75742176e-03
 -3.70453321e-03  3.09954002e-03  3.47817712e-03 -3.22162313e-03
```

```
  2.26064585e-03  4.76011168e-03 -3.11835273e-03 -1.40559638e-03
 -4.55547171e-03  1.55825284e-03 -1.58850278e-03  9.66359279e-04
  4.72147670e-03 -4.24059760e-03 -4.66729235e-03  9.63725324e-04
 -3.98625759e-03 -4.63391142e-03 -4.63720877e-03 -1.79253489e-04
 -3.75555974e-04  2.24391813e-04 -7.78056798e-04  4.58918186e-03
  4.45333496e-03 -3.10695660e-03  4.27327910e-03 -4.58967220e-03
 -5.65015129e-04  1.20668125e-03  5.75605009e-05 -1.95428752e-03
  3.31026735e-04  5.63134032e-04 -4.84150276e-03 -2.30588671e-03
 -3.80731095e-03  1.93056825e-03 -4.53185843e-04 -3.71493935e-03
  2.88090901e-03  1.96302964e-04  1.24558900e-03  2.69120978e-03
 -2.00923975e-03  4.02055634e-03 -3.94624891e-03 -1.74556917e-03
 -3.72356270e-03  4.38054994e-04 -3.79773439e-03  4.50119143e-03
  2.81994417e-03 -3.53641901e-03  3.20350332e-03  3.40460823e-03
 -3.78236338e-03  2.38672178e-03 -3.80887673e-03  4.77143936e-03
  2.25756317e-03 -7.39999596e-05 -4.54879785e-03 -4.20057564e-04
 -9.58975070e-05  4.72341577e-04  4.47895704e-03 -1.33679202e-03
 -4.18825261e-03 -4.46341420e-03 -4.06744075e-04  8.76497361e-04
 -9.13901204e-06 -1.14589732e-03  4.35298542e-03  1.95895336e-04
  3.33761564e-03  1.95616996e-03 -3.53504601e-03 -1.16232724e-03]
remains : [-3.8556790e-03 -3.3799619e-03 -1.5693560e-03  3.2881303e-03
 -3.7658264e-04  4.4078762e-03 -1.1062599e-03 -2.5894120e-03
  1.9418295e-03  1.0957106e-03 -1.5454009e-04 -1.2615781e-03
 -3.0327898e-03  2.0590588e-03 -3.8543362e-03 -4.4230875e-03
 -4.5981179e-03  4.2327400e-03  2.0038730e-03  3.4631814e-03
  4.9613966e-03 -3.6027736e-03  2.0696847e-03  1.4275101e-03
  3.0600044e-03 -1.6664518e-03  4.6823416e-03 -2.9241620e-03
  3.4873618e-03  3.4618550e-03 -1.9533971e-04  2.5304654e-03
```

BERT - Capturing contextual relatioships & Understand complex sentences

```
output
```

```
['medication', 'adherence', 'cardiovascular', 'medicine']
```

```
from transformers import BertTokenizer, BertModel
import torch

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
tokens = tokenizer(input, padding=True, truncation=True, return_tensors='pt')
model = BertModel.from_pretrained('bert-base-uncased')
with torch.no_grad():
    out = model(**tokens)
bert_embeddings = out.last_hidden_state
for i, word in enumerate(input):
        print(word, ':', bert_embeddings[i].numpy())
```

tokenizer_config.json: 100% 28.0/28.0 [00:00<00:00, 1.65kB/s]

vocab.txt: 100% 232k/232k [00:00<00:00, 4.46MB/s]

tokenizer.json: 100% 466k/466k [00:00<00:00, 16.6MB/s]

config.json: 100% 570/570 [00:00<00:00, 28.4kB/s]

model.safetensors: 100% 440M/440M [00:02<00:00, 165MB/s]

```
cardiovascular : [[-0.42869398  0.09234727 -0.16442406 ... -0.05152863 -0.06069818
   0.21705772]
 [-0.5257284  -0.5550021  -0.5810955  ...  0.11191282 -0.09953576
  -0.08235998]
 [ 1.0284959   0.0464237  -0.29919413 ...  0.26279944 -0.8247318
  -0.13545413]
 [-0.54896575 -0.21392138 -0.12034225 ...  0.1214065   0.13039722
   0.31229612]
 [-0.574981   -0.33561528 -0.11358792 ...  0.1994264   0.14108557
   0.3021204 ]
 [-0.426867   -0.18331195 -0.12883897 ...  0.09023194  0.1254773
   0.36808056]]
disease : [[-0.2605581   0.43561712 -0.40097576 ... -0.1130892  -0.203854
   0.5377211 ]
 [-0.48184448 -0.11940151 -0.55486214 ...  0.19909623 -0.34638098
  -0.08100349]
 [ 0.9990827   0.18900774 -0.41556394 ...  0.06863718 -0.9453972
  -0.13199641]
 [-0.23479742  0.22771095 -0.3019244  ... -0.15019858 -0.3853428
   0.33261546]
 [-0.3025167   0.0975984  -0.37986338 ... -0.05640217 -0.31819922
   0.33996874]
 [-0.16759847  0.18787096 -0.34392974 ... -0.10083274 -0.3139153
   0.37029734]]
leading : [[-0.4283086   0.18138066 -0.01238894 ...  0.0672982   0.15431872
   0.21584694]
 [-0.93106544 -0.53472626 -0.10894294 ...  0.21810368  0.28883785
  -0.12888125]
 [ 1.0767456   0.19219212 -0.22336976 ...  0.19945161 -0.9654406
  -0.09261026]
 [-0.66597277 -0.29381213  0.49116606 ...  0.05045847  0.04457827
   0.10640617]
 [-0.7143598  -0.38756338  0.4517702  ...  0.09157293  0.08955105
   0.1058    ]
 [-0.46248835 -0.19515252  0.39271232 ... -0.02982709 -0.00894444
  -0.0072077 ]]
cause : [[-0.24466427  0.2531879   0.02657413 ... -0.3774786   0.01022164
   0.3219284 ]
 [ 0.07396542 -0.22556338 -0.12081281 ...  0.28233612  0.1746518
   0.5236475 ]
 [ 0.9240065   0.02732413 -0.38356456 ...  0.08922099 -0.9623033
  -0.15972036]
 [ 0.01669697 -0.3449486   0.4611239  ... -0.27943298  0.0541884
   0.49518254]
 [-0.0315476  -0.517021    0.43147466 ... -0.24772081  0.0361188
```

```
from transformers import BertTokenizer, BertModel
import torch

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
tokens_1 = tokenizer(output, padding=True, truncation=True, return_tensors='pt')
model_1 = BertModel.from_pretrained('bert-base-uncased')
with torch.no_grad():
    out_1 = model_1(**tokens_1)
bert_embeddings_1 = out_1.last_hidden_state
for i, word in enumerate(output):
        print(word, ':', bert_embeddings_1[i].numpy())
```

```
medication : [[-0.11821673  0.16197571 -0.07180474 ... -0.22547865 -0.10284764
   0.25458854]
 [-0.29588208  0.24661656 -0.3725784  ... -0.6278242  -0.33135694
   0.02605976]
 [ 0.8551517   0.12936395 -0.42021742 ...  0.05498876 -0.6903206
  -0.3861094 ]]
adherence : [[-0.38320982  0.27525085  0.03545895 ...  0.1297458   0.09269787
   0.25569412]
 [-0.21068221 -0.12006158  0.2886017  ...  0.35808176  0.4607638
   0.4692966 ]
 [ 0.97147644  0.08883908 -0.25682253 ...  0.29493722 -0.8092183
  -0.22931008]]
cardiovascular : [[-0.42869335  0.09234771 -0.16442394 ... -0.05152808 -0.06069821
   0.21705684]
 [-0.52572745 -0.5550029  -0.5810933  ...  0.11191379 -0.09953394
  -0.08235869]
 [ 1.0284969   0.04642388 -0.2991941  ...  0.26279926 -0.8247325
  -0.13545401]]
medicine : [[ 0.03120169  0.42934826 -0.44671717 ... -0.27408144  0.13439642
```

```
  0.64538777]
[ 0.1565493   0.38183063 -0.50650716 ... -0.5074962   0.3653871
 -0.30214646]
[ 1.023633    0.08833586 -0.2898354  ... -0.02793229 -0.9207077
 -0.1900469 ]]
 -9.3540531e-01 -1.6282436e-01]]
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size=0.2,random_state=42)
#train_df,test_df = train_test_split(df,test_size=0.2,random_state=42)
```

```
[ 1.0812808   0.02938276 -0.16138354 ...  0.1945422  -0.8669293
```

```
text_word_count = []
summary_word_count = []
```

```
[-0.58806217 -0.01078176  0.3300765  ... -0.0669221  -0.06996652
```

```
for i in X['text']:
      text_word_count.append(len(i.split()))
for i in Y['summary']:
      summary_word_count.append(len(i.split()))
```

```
[-0.7527598  -0.1392701  -0.32808584 ... -0.25329632  0.71315277
```

```
import matplotlib.pyplot as plt
length_df = pd.DataFrame({'text':text_word_count, 'summary':summary_word_count})
length_df.hist(bins = 30)
plt.show()
```