```
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer()
#
# Create sample set of documents
#
docs = np.array(['Mirabai has won a silver medal in weight lifting in
Tokyo olympics 2021',
                 'Sindhu has won a bronze medal in badminton in Tokyo
olympics',
                 'Indian hockey team is in top four team in Tokyo
olympics 2021 after 40 years'])
#
# Fit the bag-of-words model
#
bag = vectorizer.fit_transform(docs)
#
# Get unique words / tokens found in all the documents. The unique
words / tokens represents
# the features
#
# print(vectorizer.get_feature_names())
print(vectorizer.get_feature_names_out())
#
# Associate the indices with each unique word
#
# print(vectorizer.vocabulary_)
#
# Print the numerical feature vector
#
print(bag.toarray())

import numpy as np
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer()
#
# Create sample set of documents
#
docs = np.array(['This movie is very scary and long',
                 'This movie is not scary and is slow',
                 'This movie is spooky and good'])
#
# Fit the bag-of-words model
#
bag = vectorizer.fit_transform(docs)
#
# Get unique words / tokens found in all the documents. The unique
words / tokens represents
```

```
# the features
#
# print(vectorizer.get_feature_names())
print(vectorizer.get_feature_names_out())
#
# Associate the indices with each unique word
#
print(vectorizer.vocabulary_)
#
# Print the numerical feature vector
#
print(bag.toarray())
```

#TF-IDF

```
# TfidfVectorizer
# CountVectorizer
from sklearn.feature_extraction.text import
TfidfVectorizer,CountVectorizer
import pandas as pd
# set of documents
train = ['The sky is blue.','The sun is bright.']
test = ['The sun in the sky is bright', 'We can see the shining sun,
the bright sun.']
# instantiate the vectorizer object
countvectorizer = CountVectorizer(analyzer= 'word',
stop_words='english')
tfidfvectorizer = TfidfVectorizer(analyzer='word',stop_words=
'english')
tfidfvectorizer = TfidfVectorizer() #does not remove stopwords
# convert th documents into a matrix
count_wm = countvectorizer.fit_transform(train)
tfidf_wm = tfidfvectorizer.fit_transform(train)
#retrieve the terms found in the corpora
# if we take same parameters on both Classes(CountVectorizer and
TfidfVectorizer) , it will give same output of get_feature_names()
methods)
#count_tokens = tfidfvectorizer.get_feature_names() # no difference
count_tokens = countvectorizer.get_feature_names_out()
tfidf_tokens = tfidfvectorizer.get_feature_names_out()
df_countvect = pd.DataFrame(data = count_wm.toarray(),index =
['Doc1','Doc2'],columns = count_tokens)
df_tfidfvect = pd.DataFrame(data = tfidf_wm.toarray(),index =
['Doc1','Doc2'],columns = tfidf_tokens)
print("Count Vectorizer\n")
print(df_countvect)
print("\nTF-IDF Vectorizer\n")
print(df_tfidfvect)
```

```
# TfidfVectorizer
# CountVectorizer
from sklearn.feature_extraction.text import
TfidfVectorizer,CountVectorizer
import pandas as pd
# set of documents
train = ['The sky is blue.','The sun is bright.']
test = ['The sun in the sky is bright', 'We can see the shining sun,
the bright sun.']
# instantiate the vectorizer object
#countvectorizer = CountVectorizer(analyzer= 'word',
stop_words='english')
countvectorizer = CountVectorizer()
#tfidfvectorizer = TfidfVectorizer(analyzer='word',stop_words=
'english')
tfidfvectorizer = TfidfVectorizer() #does not remove stopwords
# convert th documents into a matrix
count_wm = countvectorizer.fit_transform(train)
tfidf_wm = tfidfvectorizer.fit_transform(train)
#retrieve the terms found in the corpora
# if we take same parameters on both Classes(CountVectorizer and
TfidfVectorizer) , it will give same output of get_feature_names()
methods)
#count_tokens = tfidfvectorizer.get_feature_names() # no difference
count_tokens = countvectorizer.get_feature_names_out()
tfidf_tokens = tfidfvectorizer.get_feature_names_out()
df_countvect = pd.DataFrame(data = count_wm.toarray(),index =
['Doc1','Doc2'],columns = count_tokens)
df_tfidfvect = pd.DataFrame(data = tfidf_wm.toarray(),index =
['Doc1','Doc2'],columns = tfidf_tokens)
print("Count Vectorizer\n")
print(df_countvect)
print("\nTF-IDF Vectorizer\n")
print(df_tfidfvect)
```

#Similarity Measures

```
def jaccard_similarity(s1, s2):
    s1=s1.lower()
    s2=s2.lower()
    set1 = set(s1.split())
    set2 = set(s2.split())
    intersection = set1.intersection(set2)
    print(intersection)
    union = set1.union(set2)
    print(union)
    return len(intersection) / len(union)
```

```python
s1 = "The cat sat on the mat"
s2 = "The kitten rested on the rug"
print(jaccard_similarity(s1, s2))
```