# nlp

October 16, 2023

```
[1]: # Loading libraries
     import pandas as pd
```

```
[2]: # Dataframe
     df = pd.read_csv('data.csv')
     df.head()
```

```
[2]:                            test  class
     0             I love Bangladesh      1
     1  Could you give me an iphone?      0
     2            Hello how are you?      1
     3             I want to talk you.      1
```

```
[3]: x = df.test
     y = df['class']
```

## 0.1 Count Vectorizer

```
[4]: from sklearn.feature_extraction.text import CountVectorizer
     cv = CountVectorizer()
```

```
[5]: result = cv.fit_transform(x)
     result
```

```
[5]: <4x14 sparse matrix of type '<class 'numpy.int64'>'
             with 16 stored elements in Compressed Sparse Row format>
```

```
[6]: result.toarray()
```

```
[6]: array([[0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
            [1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1],
            [0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1],
            [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1]], dtype=int64)
```

```
[7]: cv.get_feature_names()
```

```
D:\anaconda3\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning:
Function get_feature_names is deprecated; get_feature_names is deprecated in 1.0
```

```
and will be removed in 1.2. Please use get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

[7]:
```
['an',
 'are',
 'bangladesh',
 'could',
 'give',
 'hello',
 'how',
 'iphone',
 'love',
 'me',
 'talk',
 'to',
 'want',
 'you']
```

[8]:
```
len(cv.get_feature_names())
```

[8]:
```
14
```

[9]:
```
pd.DataFrame(result.toarray(),index=x,columns=cv.get_feature_names_out())
```

[9]:
```
                              an  are  bangladesh  could  give  hello  how  \
test
I love Bangladesh              0    0           1      0     0      0    0
Could you give me an iphone?   1    0           0      1     1      0    0
Hello how are you?             0    1           0      0     0      1    1
I want to talk you.            0    0           0      0     0      0    0

                              iphone  love  me  talk  to  want  you
test
I love Bangladesh                  0     1   0     0   0     0    0
Could you give me an iphone?       1     0   1     0   0     0    1
Hello how are you?                 0     0   0     0   0     0    1
I want to talk you.                0     0   0     1   1     1    1
```

## 0.2 TF-IDF Vectorizer

[10]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer()
```

[12]:
```
result = tf.fit_transform(x)
result.toarray()
```

[12]:
```
array([[0.        , 0.        , 0.70710678, 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.70710678, 0.        ,
        0.        , 0.        , 0.        , 0.        ],
       [0.43003652, 0.        , 0.        , 0.43003652, 0.43003652,
        0.        , 0.        , 0.43003652, 0.        , 0.43003652,
        0.        , 0.        , 0.        , 0.27448674],
       [0.        , 0.5417361 , 0.        , 0.        , 0.        ,
        0.5417361 , 0.5417361 , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.34578314],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.5417361 , 0.5417361 , 0.5417361 , 0.34578314]])
```

[13]:
```
pd.DataFrame(result.toarray(),index=x,columns=tf.get_feature_names_out())
```

[13]:
```
                                  an       are  bangladesh     could  \
test
I love Bangladesh           0.000000  0.000000    0.707107  0.000000
Could you give me an iphone? 0.430037  0.000000    0.000000  0.430037
Hello how are you?          0.000000  0.541736    0.000000  0.000000
I want to talk you.         0.000000  0.000000    0.000000  0.000000

                                give     hello       how    iphone  \
test
I love Bangladesh           0.000000  0.000000  0.000000  0.000000
Could you give me an iphone? 0.430037  0.000000  0.000000  0.430037
Hello how are you?          0.000000  0.541736  0.541736  0.000000
I want to talk you.         0.000000  0.000000  0.000000  0.000000

                                love        me      talk        to  \
test
I love Bangladesh           0.707107  0.000000  0.000000  0.000000
Could you give me an iphone? 0.000000  0.430037  0.000000  0.000000
Hello how are you?          0.000000  0.000000  0.000000  0.000000
I want to talk you.         0.000000  0.000000  0.541736  0.541736

                                want       you
test
I love Bangladesh           0.000000  0.000000
Could you give me an iphone? 0.000000  0.274487
Hello how are you?          0.000000  0.345783
I want to talk you.         0.541736  0.345783
```

## 0.3 Word2Vec Vectorizer

[15]:
```
pip install gensim
```

```
Requirement already satisfied: gensim in d:\anaconda3\lib\site-packages (4.2.0)
Requirement already satisfied: scipy>=0.18.1 in d:\anaconda3\lib\site-packages
(from gensim) (1.4.1)
Requirement already satisfied: Cython==0.29.28 in d:\anaconda3\lib\site-packages
(from gensim) (0.29.28)
Requirement already satisfied: smart-open>=1.8.1 in d:\anaconda3\lib\site-
packages (from gensim) (6.0.0)
Requirement already satisfied: numpy>=1.17.0 in d:\anaconda3\lib\site-packages
(from gensim) (1.18.1)
Note: you may need to restart the kernel to use updated packages.
```

[21]:
```
from gensim.models import Word2Vec, KeyedVectors
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Deadpool\AppData\Roaming\nltk_data…
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

[21]:
```
True
```

[25]:
```
tokenize_text = [nltk.word_tokenize(test) for test in x]
tokenize_text
```

[25]:
```
[['I', 'love', 'Bangladesh'],
 ['Could', 'you', 'give', 'me', 'an', 'iphone', '?'],
 ['Hello', 'how', 'are', 'you', '?'],
 ['I', 'want', 'to', 'talk', 'you', '.']]
```

[27]:
```
model = Word2Vec(tokenize_text,min_count=1)
```

[31]:
```
model.wv.most_similar('love')
```

[31]:
```
[('are', 0.2529045641422272),
 ('?', 0.17018887400627136),
 ('how', 0.15016479790210724),
 ('Bangladesh', 0.13887983560562134),
 ('iphone', 0.10852649062871933),
 ('Could', 0.03476495295763016),
 ('to', 0.016068339347839355),
 ('I', 0.004503019154071808),
 ('Hello', -0.005900928284972906),
 ('you', -0.027746984735131264)]
```

## 1 Word Scaling Techniques

### 1.1 Stemming

#### 1.1.1 Porter Stemmer

[34]:
```
from nltk.stem import PorterStemmer
ps = PorterStemmer()
```

[77]:
```
para = "Changing, changed & changes are from change"
```

[78]:
```
tokens = nltk.word_tokenize(para)
tokens
```

[78]:
```
['Changing', ',', 'changed', '&', 'changes', 'are', 'from', 'change']
```

[79]:
```
for word in tokens:
    print(ps.stem(word))
```

```
chang
,
chang
&
chang
are
from
chang
```

### 1.2 Lammatization

#### 1.2.1 Word Net Lemmatizer

[58]:
```
from nltk.stem import WordNetLemmatizer
lem = WordNetLemmatizer()
```

[57]:
```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Deadpool\AppData\Roaming\nltk_data…
```

[57]:
```
True
```

[59]:
```
print(lem.lemmatize('churches'))
```

```
church
```

[61]:
```
words
```

[61]:
```
['change', 'changing', 'changed', 'changes']
```

```
[60]: for word in words:
          print(lem.lemmatize(word))
```

```
change
changing
changed
change
```

```
[ ]:
```