Vectorizing Raw Data: Count Vectorization

Count vectorization

Creates a document-term matrix where the entry of each cell will be a count of the number of times that word occurred in that document.

Read in text

```
import pandas as pd
import re
import string
import nltk
pd.set_option('display.max_colwidth', 100)

stopwords = nltk.corpus.stopwords.words('english')
ps = nltk.PorterStemmer()

data = pd.read_csv("SMSSpamCollection.tsv", sep='\t')
data.columns = ['label', 'body_text']
```

Create function to remove punctuation, tokenize, remove stopwords, and stem

```
def clean_text(text):
    text = "".join([word.lower() for word in text if word not in string.punctuation])
    tokens = re.split('\W+', text)
    text = [ps.stem(word) for word in tokens if word not in stopwords]
    return text
```

Apply CountVectorizer

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
count_vect = CountVectorizer(analyzer=clean_text)
X_counts = count_vect.fit_transform(data['body_text'])
print(X_counts.shape)
print(count_vect.get_feature_names())
```

```
(5567, 8104)
['', '0', '008704050406', '0089mi', '0121', '01223585236', '01223585334', '0125698789', '02', '020603', '0207', '02070836089', '0
```

Apply CountVectorizer to smaller sample

```
data_sample = data[0:20]

count_vect_sample = CountVectorizer(analyzer=clean_text)
X_counts_sample = count_vect_sample.fit_transform(data_sample['body_text'])
print(X_counts_sample.shape)
print(count_vect_sample.get_feature_names())
```

```
(20, 192)
['08002986030', '08452810075over18', '09061701461', '1', '100', '100000', '11', '12', '150pday', '16', '2', '20000', '2005', '21s
```

Vectorizers output sparse matrices

***Sparse Matrix**: A matrix in which most entries are 0. In the interest of efficient storage, a sparse matrix will be stored by only storing the locations of the non-zero elements.*

```
X_counts_sample
```

```
<20x192 sparse matrix of type '<class 'numpy.int64'>'
        with 218 stored elements in Compressed Sparse Row format>
```

```
X_counts_df = pd.DataFrame(X_counts_sample.toarray())
X_counts_df
```

	0	1	2	3	4	5	6	7	8	9	...	182	183	184	185	186	187	188	189	190	191
0	0	1	0	0	0	0	0	0	0	0	...	0	1	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	1	0	0	...	0	0	1	0	0	0	0	0	0	0
6	1	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0	1	1	...	0	1	0	0	0	0	0	0	0	0
9	0	0	0	1	0	1	0	0	0	0	...	0	0	0	0	1	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	1	0	0
11	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	1	0
13	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
18	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

```
X_counts_df.columns = count_vect_sample.get_feature_names()
X_counts_df
```

	08002986030	08452810075over18	09061701461	1	100	100000	11	12	150pday	16	...	wet	win	wi
0	0	1	0	0	0	0	0	0	0	0	...	0	1	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	
5	0	0	1	0	0	0	0	1	0	0	...	0	0	
6	1	0	0	0	0	0	1	0	0	0	...	0	0	
7	0	0	0	0	0	0	0	0	0	0	...	0	0	
8	0	0	0	0	1	0	0	0	1	1	...	0	1	
9	0	0	0	1	0	1	0	0	0	0	...	0	0	
10	0	0	0	0	0	0	0	0	0	0	...	0	0	
11	0	0	0	0	0	0	0	0	0	0	...	0	0	
12	0	0	0	0	0	0	0	0	0	0	...	1	0	
13	0	0	0	0	0	0	0	0	0	0	...	0	0	
14	0	0	0	0	0	0	0	0	0	1	...	0	0	
15	0	0	0	0	0	0	0	0	0	0	...	0	0	
16	0	0	0	0	0	0	0	0	0	0	...	0	0	
17	0	0	0	0	0	0	0	0	0	0	...	0	0	
18	0	0	0	0	0	0	0	0	0	0	...	0	0	
19	0	0	0	0	0	0	0	0	0	0	...	0	0	

Colab paid products - Cancel contracts here