Naive Bayes Experiment (No Augmentation)

Real News and Fake News (~60k total)

Class: Label

Real: 1

Fake: 0

```
import pandas as pd
import numpy as np
import json


file = "combined_{}.csv"
dfs = []
for i in range(3):
    fp = file.format(i+1)
    read = pd.read_csv(fp)
    read = read[['label', 'clean_text']]
    dfs.append(read)

dfs[2] = dfs[2][:-13000]

data = pd.concat(dfs)
data.tail()
data.reset_index(inplace=True, drop=True)
print(data.shape)
print(data[data.label == 1].shape[0], "Real")
print(data[data.label == 0].shape[0], "Fake")

data.dropna(inplace=True)
data.head(10)
```

```
(59818, 2)
31514 Real
28304 Fake
```

	label	clean_text
0	0	house dem aide didnt even see comeys letter ja...
1	1	ever get feeling life circles roundabout rathe...
2	0	truth might get fired october tension intell...
3	0	videos civilians killed single us airstrike i...
4	0	print iranian woman sentenced six years prison...
5	1	trying times jackie mason voice reason weeks e...
6	0	ever wonder britains iconic pop pianist gets l...
7	1	paris france chose idealistic traditional cand...
8	1	donald trump scheduled make highly anticipated...
9	1	week michael flynn resigned national security ...

```
import nltk
from wordcloud import WordCloud
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score, accuracy_score, recall_score, precision_score


combined_text = data['clean_text'].values
labels = data['label'].values
data.groupby('label').agg('count')
```

label	clean_text
0	28010
1	31449

```
tfidf = TfidfVectorizer(ngram_range=(1,2), max_df= 0.85, min_df= 0.01)

combined_text = tfidf.fit_transform(combined_text)

print("tf idf vectorized text shape: {}".format(combined_text.shape))
print("original text shape: {}".format(data['clean_text'].shape))
```

```
tf idf vectorized text shape: (59459, 4252)
original text shape: (59459,)
```

```
train_text, test_text, train_labels, test_labels  = train_test_split(combined_text, labels, test_size=0.2)

import collections

print("size of train_text: {}".format(train_text.shape))

print("size of train_labels: {}".format(train_labels.shape))
collections.Counter(train_labels)
```

```
size of train_text: (47567, 4252)
size of train_labels: (47567,)
Counter({0: 22430, 1: 25137})
```

```
print("size of test_text: {}".format(test_text.shape))
```

```
print("size of test_labels: {}".format(test_labels.shape))
collections.Counter(test_labels)
```

```
size of test_text: (11892, 4252)
size of test_labels: (11892,)
Counter({0: 5580, 1: 6312})
```

```
from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()
```

```
nb.fit(train_text,train_labels)
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

```
train_pred = nb.predict(train_text)
```

```
print('Naive Bayes In Sample F1 and Accuracy Scores:')
print('F1 score {:.4}%'.format(f1_score(train_labels, train_pred, average='macro')*100 ))
print ('Accuracy score {:.4}%'.format(accuracy_score(train_labels, train_pred)*100))
```

```
Naive Bayes In Sample F1 and Accuracy Scores:
F1 score 79.98%
Accuracy score 80.31%
```

```
test_pred = nb.predict(test_text)
print('Naive Bayes Out of Sample F1 and Accuracy Scores:')
print('F1 score {:.4}%'.format(f1_score(test_labels, test_pred, average='macro')*100 ))
print ('Accuracy score {:.4}%'.format(accuracy_score(test_labels, test_pred)*100))
```

```
Naive Bayes Out of Sample F1 and Accuracy Scores:
F1 score 78.67%
Accuracy score 79.05%
```