CSC 411: Assignment #3

Xiangyu Kong **Yun Lu**
kongxi16 **luyun5**

March 15, 2018

Problem 1

The dataset contains headlines including the word "Trump". The quality of the dataset is generally good since there is a large variety of vocabularies included.

By observing, we can see that the fake news' headlines are generally longer than the real news' headlines.

Except for "Donald" and "Trump", the most frequent words are "to", "for", etc, but they do not mean a lot. The top meaningful words in total are "Clinton", "election" and "president". This infers that most news are regarding the 2017 presidential election and especially between Donald Trump and Hilary Clinton.

The statistics for the three words are generated through part1 in fake.py and the results are given below in Listing 1. We can see that although "Clinton" has the largest word count, most of them appear in the fake news. Reports that include "election" are more likely to be real news.

Listing 1: statistic results

```
[clinton]:
        real: 83
        fake: 132
        total: 215
[election]:
        real: 87
        fake: 74
        total: 161
[president]:
        real: 66
        fake: 64
        total: 130
```

Problem 2

The Naive Bayes algorithm is implemented in naive_bayes in util.py. To tune the parameters m and $\hat{p}$, we try using naive bayes on different value of m and $\hat{p}$ and pick one that has the best performance on the validation set. The range of test values for m was from 1 to 10, and for $\hat{p}$ was from 0.05 to 0.95. The returned optimum values were $m = 1$ and $\hat{p} = 0.05$.

To deal with small multiplication, a small_product function was implemented in util.py. It takes in a list of small values and uses the fact that $\prod_{i=1}^{N} a_i = \exp(\sum_{i=1}^{N} \log(a_i))$ to compute the value of $p(a_1, a_2, \ldots, a_n) = p(a_1)p(a_2)\ldots p(a_1)$

The performance on training set, validation set and test set is given below:

Listing 2: Performance

```
train performance = 0.954108391608
validation performance = 0.78936605317
test performance = 0.775051124744
```

Problem 3

The results are produced by part3 in fake.py and are listed below.

$P(c|word) = \frac{P(word|c) \times P(c)}{P(words)}$ where
$P(c) = \frac{count(c)}{count(total)}$, $P(word|c) = \frac{count(word\ in\ c)}{count(c)}$ and $P(word) = \frac{count(word\ in\ c)}{count(total)}$

$P(c|not\ word)$ follows with similar calculations.

The most important presence for predicting a class means $P(c|word)$ must be high and the most important absence for predicting a class means $P(c|not\ word)$ must be high.

After removing the stop words like "be", we get the results in b.

Listing 3: top results

```
a:
Real:
top 10 important presence:
        ['trump', 'donald', 'to', 'us', 'trumps', 'in', 'on',
        'of', 'says', 'for']
top 10 important absence:
        ['kommonsentsjane', 'lord', 'tired', 'miller', '270',
         'elegant', 'battleground', 'fingers', 'salbuch', 'cult']
Fake:
top 10 important presence:
        ['trump', 'to', 'the', 'donald', 'in', 'of', 'for', 'a', 'and', 'on']
top 10 important absence:
        ['hanging', 'marching', 'regional', 'hearin', 'piling',
        'jennett', 'loathing', 'deferred', 'decry', 'lgbt']

b:
Real:
top 10 important presence:
        ['trump', 'donald', 'trumps', 'says', 'election',
        'clinton', 'north', 'korea', 'ban', 'president']
Fake:
top 10 important presence:
        ['trump', 'donald', 'hillary', 'clinton', 'election',
        'just', 'new', 'president', 'obama', 'america']
```

Problem 4

Problem 5

Problem 6

Problem 7

Problem 8