Srinath Narayanan - A53213478

Code for 250A - HW1

10-10-2017

Hangman

```
__author__ = "SRINATH NARAYANAN"
import string
import operator
from collections import defaultdict
```

Reads data

```
r = open('hw1_word_counts_05.txt','r')
r.seek(0)
corpus = r.readlines()
```

Computes prior probabilities

```
cosplit = [i.strip().split() for i in corpus]
word_count = defaultdict(int)
for i,j in cosplit:
    word_count[i]=int(j)
```

```
word_probabilities = word_count.copy()
sumcounts = sum(word_count.values())
for i in word_probabilities:
    word_probabilities[i]=word_probabilities[i]*1.0/sumcounts
```

```
print "Total #unique words : ",len(word_count),"\nTotal words : ",sumcounts
```

```
Total #unique words :  6535
Total words :  7664857
```

```
sorted_dict = sorted(word_count.items(), key=operator.itemgetter(1))
print "14 least frequent 5-letter words along with their counts:"
for i in sorted_dict[:14]:
    print i[0],i[1]
print "\n15 most frequent 5-letter words along with their counts :"
for i in sorted_dict[-15:]:
    print i[0],i[1]
```

```
14 least frequent 5-letter words along with their counts:
TROUP 6
MAPCO 6
CAIXA 6
OTTIS 6
BOSAK 6
NIAID 7
YALOM 7
SERNA 7
CLEFT 7
CCAIR 7
FOAMY 7
PAXON 7
TOCOR 7
FABRI 7

15 most frequent 5-letter words along with their counts :
SIXTY 73086
THERE 86502
YEARS 88900
FORTY 94951
OTHER 106052
FIFTY 106869
FIRST 109957
AFTER 110102
```

```
WHICH 142146
THEIR 145434
ABOUT 157448
WOULD 159875
EIGHT 165764
SEVEN 178842
THREE 273077
```

The above results do make sense, since we see that the least frequent words are mostly typos or proper nouns, and the most frequent words are numbers and pronouns, which is exactly what we can expect in a large corpus of words.

Creates place-holder for included and excluded words

```
def create_given():
    for word in word_count :
        for i in range(5) :
            if ((correct[i]!= word[i]) and (correct[i]!=' ')):
                exclude[word] = word_count[word]
                break

    for word in word_count :
        dele = 0
        for a in range(5) :
            for b in range(len(inc[a])) :
                if inc[a][b]==word[a] :
                    exclude[word] = word_count[word]
                    dele = 1
                    break
            if dele == 1:
                dele = 0
                break

    for z in word_count :
        if z not in exclude :
```

```
            given[z] = word_count[z]
```

Filtering out unwanted words for ease in probability calculation

```
def create_incorrect():
    for x in range(len(incorrect)):
        for y in range(5) :
            if y in inc:
                inc[y].append(incorrect[x])
            else :
                inc[y] = [incorrect[x]]
    for m in range(len(correct)) :
        for n in range(5) :
            if m!=n and correct[m]!=correct[n] :
                if n in inc:
                    if correct[m]!= ' ':
                        inc[n].append(correct[m])
                else :
                    if correct[m]!= ' ':
                        inc[n] = [correct[m]]
```

Next guess probability computation

```
def best_next_guess():
    prob_evi_given_word = {}
    for word in word_count :
        if word in given :
            prob_evi_given_word[word] = 1
        else :
            prob_evi_given_word[word] = 0

    denominator_1 = sum((prob_evi_given_word[word] * word_probabilities[word]) for wor
d in word_count)
    prob_word_given_evi = {}
```

```
    for word in word_count :
        prob_word_given_evi[word] = float((prob_evi_given_word[word]*word_probabilitie
s[word])/denominator_1)

    alphabet = {'A':0, 'B':0, 'C':0, 'D':0, 'E':0, 'F':0, 'G':0, 'H':0, 'I':0, 'J':0,
'K':0, 'L':0, 'M':0, 'N':0, 'O':0, 'P':0, 'Q':0, 'R':0, 'S':0, 'T':0, 'U':0, 'V':0,
'W':0, 'X':0, 'Y':0, 'Z':0}
    for word in word_count:
        for letter in alphabet :
            if letter in word :
                alphabet[letter]+=prob_word_given_evi[word]

    for x in correct :
        if x in alphabet :
            del alphabet[x]

    alphabet_sort = sorted(alphabet.items(), key=operator.itemgetter(1))

    print("Next Best Guess is ",alphabet_sort[-1][0])
    print("Probability of that is ",float(alphabet_sort[-1][1]))
```

Results for 1.9(b)

```
for i in range(0,9):
    inc = {0:[],1:[],2:[],3:[],4:[]}
    exclude,given = {},{}

    correct = []
    incorrect=[]
    while len(correct)!=5 :
        correct = raw_input ('\n Enter correct characters (Enter 5 characters and use
 Space to denote if character not filled till now :)')
        if len(correct)!=5 :
            print("Correct Character Length not 5, Please try again")
    incorrect = raw_input('Enter incorrect characters : ')
```

```
    create_incorrect()
    create_given()
    best_next_guess()
```

```
 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)
Enter incorrect characters :
('Next Best Guess is ', 'E')
('Probability of that is ', 0.5394172389647942)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)
Enter incorrect characters : EA
('Next Best Guess is ', 'O')
('Probability of that is ', 0.5340315651557651)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)A   S
Enter incorrect characters :
('Next Best Guess is ', 'E')
('Probability of that is ', 0.7715371621621622)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)A   S
Enter incorrect characters : I
('Next Best Guess is ', 'E')
('Probability of that is ', 0.7127008416220354)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)  O
Enter incorrect characters : AEMNT
('Next Best Guess is ', 'R')
('Probability of that is ', 0.7453866259829716)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)
```

```
Enter incorrect characters : EO
('Next Best Guess is ', 'I')
('Probability of that is ', 0.6365554141009606)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)D  I
Enter incorrect characters :
('Next Best Guess is ', 'A')
('Probability of that is ', 0.8206845238095238)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :)D  I
Enter incorrect characters : A
('Next Best Guess is ', 'E')
('Probability of that is ', 0.7520746887966804)

 Enter correct characters (Enter 5 characters and use Space to denote if character not filled till now :) U
Enter incorrect characters : AEIOS
('Next Best Guess is ', 'Y')
('Probability of that is ', 0.6269651101630526)
```