CSE250 HW4 Q3

October 30, 2018

In [18]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from numpy.linalg import *
import math
```

In [114]:
```
unigram = pd.read_csv('hw4_unigram.txt', sep=" ", header=None)   #df for unigram
unigram.columns = ['count(w)']
vocab = np.loadtxt(fname = "hw4_vocab.txt", dtype='str')   #array of vocabs
bigram = pd.read_csv('hw4_bigram.txt', sep="\t", header=None)   #df for bigram
bigram.columns = ['index(w1)', 'index(w2)','count(w1,w2)']
```

0.1 (a)

In [115]:
```
# compute unigram probabilities
unigram.loc[:, 'pu(w)'] = unigram['count(w)'].apply(lambda x: x/sum(unigram['count(w)
unigram.head()
```

Out[115]:

	count(w)	pu(w)
0	25223698	0.308490
1	3021866	0.036958
2	3021866	0.036958
3	3855375	0.047152
4	3667333	0.044852

In [86]:
```
letters = ['A']
output_names = [name for name in vocab if (name[0] in letters)]
```

In [87]:
```
def FindWordIndex(vocab, words):
    indexes = np.zeros((len(words),1))
    for i in range(len(words)):
        indexes[i] = np.argwhere(vocab == words[i] )
    return indexes

indexes = FindWordIndex(vocab, output_names)

def GetPu(indexes,unigram):
    Pu = np.zeros((indexes.shape[0],1))
```

```
        for i in range(indexes.shape[0]):
            Pu[i] = unigram.values[int(indexes[i,0]),1]
        return Pu

    Pu = GetPu(indexes,unigram)
```

In [88]:
```
# tokens start with A and their unigram probabilities
TableA = pd.DataFrame(data = output_names)
TableA.columns = ['word start with A']
TableA.loc[:,'Pu(w)'] = Pu
TableA
```

Out[88]:

	word start with A	Pu(w)
0	A	0.018407
1	AND	0.017863
2	AT	0.004313
3	AS	0.003992
4	AN	0.002999
5	ARE	0.002990
6	ABOUT	0.001926
7	AFTER	0.001347
8	ALSO	0.001310
9	ALL	0.001182
10	A.	0.001026
11	ANY	0.000632
12	AMERICAN	0.000612
13	AGAINST	0.000596
14	ANOTHER	0.000428
15	AMONG	0.000374
16	AGO	0.000357
17	ACCORDING	0.000348
18	AIR	0.000311
19	ADMINISTRATION	0.000292
20	AGENCY	0.000280
21	AROUND	0.000277
22	AGREEMENT	0.000263
23	AVERAGE	0.000259
24	ASKED	0.000258
25	ALREADY	0.000249
26	AREA	0.000231
27	ANALYSTS	0.000226
28	ANNOUNCED	0.000227
29	ADDED	0.000221
30	ALTHOUGH	0.000214
31	AGREED	0.000212
32	APRIL	0.000207
33	AWAY	0.000202

0.2 (b)

```
In [157]: #
          bigram['sum count(w1,w2)'] = bigram['count(w1,w2)'].groupby(bigram['index(w1)']).tra
          bigram['pb(w2|w1)'] = (bigram['count(w1,w2)']/bigram['sum count(w1,w2)'])
          bigram.head()
```

Out[157]:

	index(w1)	index(w2)	count(w1,w2)	sum count(w1,w2)	pb(w2|w1)
0	1	1	7355976	25223698	0.291630
1	1	3	5645	25223698	0.000224
2	1	4	647219	25223698	0.025659
3	1	5	2373160	25223698	0.094085
4	1	6	1801245	25223698	0.071411

```
In [153]: #find index of THE
          index_the = np.argwhere(vocab == "THE" )
          bigram_the = bigram[(index_the[0,0]-1)*496+1:(index_the[0,0])*499-33]
          bigram_the_5 = bigram_the.sort_values('pb(w2|w1)', ascending=False).head(5)
```

```
In [155]: word1 = vocab[0]
          word2 = vocab[69]
          word3 = vocab[78]
          word4 = vocab[72]
          word5 = vocab[60]
          likely_words = [word1,word2,word3,word4,word5]
          bigram_the_5['words to follow THE'] = likely_words
```

```
In [156]: # 5 most likely words to follow THE and their conditional probabilities
          bigram_the_5
```

Out[156]:

	index(w1)	index(w2)	count(w1,w2)	sum count(w1,w2)	pb(w2|w1)	\
993	4	1	2371132	3855375	0.615020	
1058	4	70	51556	3855375	0.013372	
1064	4	79	45186	3855375	0.011720	
1060	4	73	44949	3855375	0.011659	
1050	4	61	36439	3855375	0.009451	

	words to follow THE
993	<UNK>
1058	U.
1064	FIRST
1060	COMPANY
1050	NEW

0.3 (c)

```
In [122]: # calculate Lu
          words = ['LAST','WEEK','THE','STOCK','MARKET','FELL','BY','ONE','HUNDRED','POINTS']
          index_c = FindWordIndex(vocab, words)
```

```
Pu_c = GetPu(index_c,unigram)
l_u = np.log(np.prod(Pu_c))
print("Lu is:  ", l_u)
```

```
Lu is:   -64.50944034364878
```

```
In [212]: # calculate Lb
          def Pb(vocab,first,second,bigram):
              first_idx = np.argwhere(vocab == first)[0,0]
              second_idx = np.argwhere(vocab == second )[0,0]
              i1 = bigram.index[bigram['index(w1)'] == first_idx+1]
              i2 = bigram.index[bigram['index(w2)'] == second_idx+1]
              if bool(set(i1).intersection(i2)):
                  i = list(set(i1).intersection(i2))[0]
                  pb = bigram.values[i,4]
              else:
                  print("Not found in corpus", first, second)
              return pb

          l_b_c = np.log(Pb(vocab,'<s>','LAST',bigram)*Pb(vocab,'LAST','WEEK',bigram)* Pb(vocal
                     *Pb(vocab,'THE','STOCK',bigram)*Pb(vocab,'STOCK','MARKET',bigram)*Pb(vocal
                     *Pb(vocab,'FELL','BY',bigram)*Pb(vocab,'BY','ONE',bigram)*Pb(vocab,'ONE',
                     *Pb(vocab,'HUNDRED','POINTS',bigram))

          print("Lb is:  ", l_b_c)
```

```
Lb is:   -44.740469213403735
```

Lb > Lu, thus the bigram model yields higher log likelihood.

0.4 (d)

```
In [213]: # calculate Lu
          words_d = ['THE','NINETEEN','OFFICIALS','SOLD','FIRE','INSURANCE']
          index_d = FindWordIndex(vocab, words_d)
          Pu_d = GetPu(index_d,unigram)
          l_u_d = np.log(np.prod(Pu_d))
          print("Lu is:  ", l_u_d)
```

```
Lu is:   -41.64345971649364
```

```
In [214]: # calculate Lb
          l_b_d = np.log(Pb(vocab,'<s>','THE',bigram)*Pb(vocab,'THE','NINETEEN',bigram)* Pb(voc
                     *Pb(vocab,'OFFICIALS','SOLD',bigram)*Pb(vocab,'SOLD','FIRE',bigram)*Pb(voc
```

```
Not found in corpus NINETEEN OFFICIALS
```

```
---------------------------------------------------------------------------

UnboundLocalError                         Traceback (most recent call last)

<ipython-input-214-3927fc3f5c6e> in <module>()
  1 # calculate Lb
----> 2 l_b_d = np.log(Pb(vocab,'<s>','THE',bigram)*Pb(vocab,'THE','NINETEEN',bigram)* Pb(v


<ipython-input-212-ced86e3a348e> in Pb(vocab, first, second, bigram)
 10     else:
 11         print("Not found in corpus", first, second)
---> 12     return pb
 13
 14 l_b_c = np.log(Pb(vocab,'<s>','LAST',bigram)*Pb(vocab,'LAST','WEEK',bigram)* Pb(voc


UnboundLocalError: local variable 'pb' referenced before assignment
```

For the sentence in part (d), NINETEEN OFFICIALS is not observed in the training corpus, thus the Pb of this pair is zero. This will cause the log likelihood of bigram model to be minus infinity.