Machine Learning Fall 2016
HW5 Writeup Template

1.3

thetaPos

love:	0.0013032632122369557
wonderful:	0.000314082824001704,
best:	0.0010956797366036455,
great:	0.001256331296006816,
superb:	0.00010649934836839388,
still:	0.0007761816914984639,
beautiful:	0.00033032848731213694,
bad:	0.0005270815207384917,
worst:	6.137250583941343e-05,
stupid:	6.317757954057264e-05,
waste:	6.137250583941343e-05,
boring:	7.581309544868718e-05,
?:	0.001992801366079777,
!:	0.0008050628707170113,
UNK:	0.9912309519597685

thetaNeg

love:	0.001038760251718182
wonderful:	9.662886062494716e-05
best:	0.000744847467317301
great:	0.0006441924041663144
superb:	3.6235822734355186e-05
still:	0.0007307557584761629
beautiful:	0.00014494329093742075
bad:	0.0015541141750512334
worst:	0.0003623582273435518
stupid:	0.0003663844298695913
waste:	0.00034021411345033477
boring:	0.0003522927210284532
?:	0.003071992527368112
!:	0.0015219045548429176
UNK:	0.9889943753950711

*Note: “love” includes words love, loving, loves, loved

Explanation of how thetaPos and thetaNeg were computed (use equations if necessary):

Each feature corresponds to a specific word in the document. In this problem, they are "love, wonderful, best, great, superb, still, beautiful, bad, worst, stupid, waste, boring, ?, !, UNK" where UNK represents unknown words. To calculate **thetaPos**, the program would count the number of all feature-words (e.g. "love") on condition of "positive", and then divide them by total word number (on condition of "positive"). See the equation shown below. To calculate **thetaNeg**, I use the same idea, where just replace "*pos*" with "*neg*" in the equation below.

$$\text{thetaPos}_i = \frac{\sum_{i=0}^{15} word_i}{\sum pos\,\text{word}}$$

1.4

Multinomial Naive Bayes Classifier (MNBC) test set Accuracy:

0.701666666667

sklearn.naive_bayes.MultinomialNB test set Accuracy:

0.701666666667

Explanation of how you test using MNBC (use equations if necessary):

The thetaPos and thetaNeg have already been calculated before. In the predicting phrase, the program would count the number of each feature word's occurrence ($N_i = \sum word_i$) for every testing case, and let P(pos) multiples thetaPos[i] N times. The possibility P(neg) can be calculated the same way. After that, compare P(pos) and P(neg), the one with high possibility (bigger value) would win, which means the test label for this test case will be given in this way. Note that, the thetaPos[i] is a relatively small number, multiple several times would lead to an overflow in computer, thus we take logarithmic value of each thetaPos[i] and add them together to reflect the relative value of P(pos) and P(neg).

$$P(pos) = \prod thetaPos[i]^{\sum_{i=0}^{15} word_i}$$

****MNBC Extra Credit test set accuracy and discussion**:**

1.5

Multinomial Naive Bayes Classifier Testing with non-BOW test set Accuracy:

0.701666666667

Explanation of how you test using MNBC with non-BOW (use equations if necessary):

Given thetaPos and thetaNeg, the testing phrase is similar to MNBC with BOW. The difference is that we don’t calculate the occurrence of each feature word (BOW) before testing phrase, but to multiple thetaPos[i] whenever the program scans the document and meet with a word[i].

****MNBC non-Bow testing Extra Credit test set accuracy and discussion**:**

1.6

<u>thetaPosTrue</u>

love:	0.48714285714285716
wonderful:	0.1742857142857143
best:	0.5057142857142857
great:	0.49714285714285716
superb:	0.07142857142857142
still:	0.38571428571428573
beautiful:	0.1757142857142857
bad:	0.2814285714285714
worst:	0.04428571428571428
stupid:	0.03857142857142857
waste:	0.04428571428571428
boring:	0.05285714285714286
?:	0.5428571428571428
!:	0.27714285714285714
UNK:	1.0

<u>thetaNegTrue</u>

love:	0.4114285714285714
wonderful:	0.06142857142857143,
best:	0.37
great:	0.3028571428571429
superb:	0.024285714285714285
still:	0.3457142857142857
beautiful:	0.09142857142857143

bad:	0.53
worst:	0.19142857142857142
stupid:	0.18857142857142858
waste:	0.2042857142857143
boring:	0.19
?:	0.6885714285714286
!:	0.3985714285714286
UNK:	1.0

*Note: “love” includes words love, loving, loves, loved

Explanation of how thetaPosTrue and thetaNegTrue were computed (use equations if necessary):

In this case, the program will count how many files (on condition of positive) contain word[i] (no matter how many times the word[i] occurs in one document). Then divide it by the total number of files (on condition of positive), and assign it to thetaPos[i]. The thetaNeg would be calculated the same way.

The difference between BNBC and MNBC is that, BNBC would take a whole document as one sample in the learning phrase, while MNBC would take every word in all documents as one sample.

Multivariate Bernoulli Naive Bayes Classifier (BNBC) test set Accuracy:

0.686666666667

Explanation of how you test using BNBC (use equations if necessary):

Given thetaPos and thetaNeg, when a new document arrives, the program will check whether the document contains word[i], if there exists word[i], then let P(pos) multiples thetaPos[i] once. The P(neg) can be calculated the same way. After that, compare two different value, the bigger one would win.

****BNBC Extra credit test set accuracy and discussion**:**