Zisheng Jason Chang
Text Classification

In this lab we were tasked with calculating three different versions of naive bayes classifiers: raw, m-estimate, and tfidf. What I found in Version 1 was poor performance. My classifier had a 9.33% chance of classifying a post correctly. Randomly picking categories for a document would be more successful than a raw classifier.

Raw Scores:

category	n_correct	n_tot	Percent
alt.atheism	317	0 319.000000	0.99373
comp.graphics	26	389	0.066838
comp.os.ms-windows.misc	17	394	0.043147
comp.sys.ibm.pc.hardware	24	392	0.061224
comp.sys.mac.hardware	20	385	0.051948
comp.windows.x	23	392	0.058673
misc.forsale	41	390	0.105128
rec.autos	16	395	0.040506
rec.motorcycles	25	398	0.062814
rec.sport.baseball	20	397	0.050378
rec.sport.hockey	27	399	0.067669
sci.crypt	29	396	0.073232
sci.electronics	13	393	0.033079
sci.med	9	396	0.022727
sci.space	18	394	0.045685
soc.religion.christian	31	398	0.077889
talk.politics.guns	14	364	0.038462
talk.politics.mideast	12	376	0.031915
talk.politics.misc	11	310	0.035484
talk.religion.misc	9	251	0.035857

The reason for the poor performance was the decision regarding how the classifier treated words in the testing set that did not appear in the training set. From naive bayes we can calculate the probability of a category given words.

$$P(C|W1, W2, W3...) = P(C) * P(W1|C) * P(W2|C) * P(W3|C)$$

Note that above if any of the $P(W|C)$ is 0 the entire probability reduces to zero.

During my testing I've found that alt.atheism had significantly higher percentages of being classified correctly than any other category. My suspicion for this is that in my testing, alt.atheism has so far had the most success with finding common words found only in alt.atheism

Version 2 improves upon Version 1 by adding a smoothing factor - which accounts for cases where the test data does not exist in the training data. This is called the m-estimate. Out of all my testing methods this proved the most robust.

M-estimate Scores:

category	n_correc	n_tot	Percent
alt.atheism	244	319	0.76489
comp.graphics	317	389	0.81491
comp.os.ms-windows.misc	249	394	0.63198
comp.sys.ibm.pc.hardware	290	392	0.739796
comp.sys.mac.hardware	297	385	0.771429
comp.windows.x	305	392	0.778061
misc.forsale	265	390	0.679487
rec.autos	371	395	0.939241
rec.motorcycles	372	398	0.934673
rec.sport.baseball	365	397	0.919395
rec.sport.hockey	388	399	0.972431
sci.crypt	377	396	0.95202
sci.electronics	270	393	0.687023
sci.med	339	396	0.856061
sci.space	351	394	0.890863
soc.religion.christian	375	398	0.942211
talk.politics.guns	335	364	0.92033
talk.politics.mideast	335	376	0.890957
talk.politics.misc	180	310	0.580645
talk.religion.misc	99	251	0.394422
Average Correct:			0.80304125

From what I've found so far mest is the best on this task as it has the highest percentage. However I will concede that tfidf may result in lower scores as a result of improper execution of the idf calculation. On top of this, my function for calculating the idf takes too long, and I've run out of time to implement it correctly. The table that follows is a subset of categories that have idf calculated.

Tfidf Scores:

category	n_correct	n_tot	Percent
alt.atheism	48	319	0.15047
comp.graphics	87	389	0.22365
comp.os.ms-windows.misc	93	394	0.236041
comp.sys.ibm.pc.hardware	123	392	0.313776
comp.sys.mac.hardware	91	385	0.236364
comp.windows.x	68	392	0.173469
misc.forsale	72	390	0.184615
rec.autos	59	395	0.149367
rec.motorcycles	19	398	0.047739
rec.sport.baseball	71	397	0.178841
rec.sport.hockey	86	399	0.215539
sci.crypt	167	396	0.421717
sci.electronics	28	393	0.071247
sci.med	22	396	0.055556
sci.space	63	394	0.159898
soc.religion.christian	108	398	0.271357
talk.politics.guns	116	364	0.318681
talk.politics.mideast	81	376	0.215426
talk.politics.misc	62	310	0.2
talk.religion.misc	11	251	0.043825
			0.1933789