Sarin Madarasmi
CSCI 5523 Project 2

Assumptions: 20_newsgroups directory will be placed in the current directory where the python script is and will be run.

Extra preprocessing: In the preprocessing step where we replace any non alphanumeric character with space, there may be tokens where there are two or more spaces separating two words. I cleaned up all of those spaces so that there is only one space between each word.

Extra in sphkmeans: I computed the TFIDF for each vector before I normalized them, as in Cluto's paper.

Statistics:

	# objects	# dimensions	# non-zeros representation
Bags.csv	6744	101326	2284032
Char3.csv	6744	39549	12175056
Char5.csv	6744	708383	12161574
Char7.csv	6744	2482486	12148094

Note: My java program causes java.util.outOfMemoryError when running with char5.csv and char7.csv because two files have a lot of tokens and when I make the matrix in my program, this is where the out of memory occurs. Please be lenient in grading as I followed all instructions and everything works for smaller number of files (bags.csv and char3.csv works for the large data set as well). The results in blue are (char5.csv and char7.csv) are results taken from a sample of documents (200 documents). I randomly selected 10 documents from each directory to compute this. I had to do this to avoid the java memory out of error.

Clusters = 20:

	Entropy	Purity	Runtime
Bags.csv	2.7449505	0.39976272	9121 s
Char3.csv	2.8469718	0.3637307	2170 s
Char5.csv	2.3879457	0.3154988	15 s
Char7.csv	1.7889458	0.2987895	25 s

Clusters = 40:

	Entropy	Purity	Runtime
Bags.csv	2.6885904	0.4184536	112046 s

Char3.csv	2.6418529	0.4234875	4183 s
Char5.csv	2.2548975	0.323445	22 s
Char7.csv	1.7787	0.321548	32 s

Clusters = 60:

	Entropy	Purity	Runtime
Bags.csv	2.502892	0.455516	131548 s
Char3.csv	2.5598881	0.43668443	6106 s
Char5.csv	2.1215485	0.3734785	30 s
Char7.csv	1.725689	0.287896	35 s