Objective

The purpose of this project is to write clustering algorithms based on k-means for clustering objects corresponding to
sparse high dimensional vectors. The project consists of multiple components that involve getting the dataset, selecting
the subset to cluster, pre-processing the dataset to convert it into a sparse representation, clustering the dataset, and
evaluating the quality of the clustering solution. The steps associated with subset selection and pre-processing need to
be done using scripts in any combination of Bash, Csh, Python, or Perl, whereas the actual clustering needs to be
written in either Java, C++, or C.

Getting the Dataset

The dataset for this assignment will be derived from the "Reuters-21578 Text Categorization Collection Data Set" that
is available at the UCI Machine Learning Repository (link). Download the reuters21578.tar.gz file from the "Data
Folder" (the file is 7.8MB compressed). Also, read the html file in the above folder to get an idea as to what the dataset
is all about.

Selecting the Subset of the Dataset for Clustering

Process the various SGML files (.sgm) extension and select the articles that contain only a single topic. The topic is
included within a pair of <TOPICS><D>topic</D></TOPICS> tags. Ignore the articles that do not contain any topic or
have more than a single topic. A back of the envelope calculation shows that are about 9494 such articles. From these
articles retain only the articles that correspond to topics that occur in the 20 most frequent topics.
From the above set of articles, extract the NEWID number (in the <REUTERS> tag attribute), the single topic, and the
text that is included within the <BODY>...</BODY> tags. The body of the article will form the text that you will be using
for clustering, the topic will act as a class label for evaluation, and the NEWID will be used as the ID of the article.
Hint: You can use modules such as beautifulsoup, HTMLParser, re or other regular expression tools to do preprocessing. Please specify in your report the modules/packages that you used and make sure that your script can be executed on a cselab machine.

Obtaining the Sparse Representation

Once you have extracted the text for each article, you need to derive a bag of words representation. In order to do that,
you need to first clean up the text. To do that perform the following steps in that sequence:
1. Eliminate any non-ascii characters.
2. Parse character entities such as "<" and "&ge" to corresponding characters "<" and ">", respectively.
3. Change the character case to lower-case.
4. Replace any non alphanumeric characters with space.
5. Split the text into tokens, using space as the delimiter.
6. Eliminate any tokens that contain only digits.
7. Eliminate any tokens from the stop list that is provided (file stoplist.txt).
8. Obtain the stem of each token using Porter's stemming algorithm; you can use any of the implementations
provided here: https://tartarus.org/martin/PorterStemmer/.
9. Eliminate any tokens that occur less than 5 times.

Vector Representations

Collect all the tokens that remained after step 9 (above) across all articles and use them to represent each article as a
vector in the distinct token space. For each document, derive three different representations by using the following
approaches to assign a value to each of the document's vector dimension that corresponds to a token *t* that it contains:
1. The value will be the actual number of times that *t* occurs in the document (*frequency).*
2. The value will be 1+sqrt(frequency) for those terms that have a non-zero frequency.
3. The value will be 1+log2(frequency) for those terms that have a non-zero frequency.
The above three representations will be referred to as *vector models* in the rest of this document.
Once you have obtained those vectors, normalize them so that they are of unit length. These unit-length vectors will be
the input to the clustering algorithms.

Clustering

Develop a partitional clustering algorithm that implements the (i) standard sum-of-squared-errors criterion function of kmeans, (ii) the I2 criterion function (i.e., spherical k-means). The formulas of the criterion functions are shown in slide #53 in the lecture's clustering slides. Your program should take as input the above vector-model representations of the
objects, the name of the clustering criterion function (i.e., SSE, I2), the number of clusters, the number of trials that it

will perform (each trial will be seeded with a different randomly selected set of objects), and the class labels of the
objects. The class label of an object is the topic that the corresponding article has. Upon completion, your program
should write the clustering solution to a file, and report the value of the criterion function for the best trial and that
solution will be used to analyze the characteristics of the clusters in terms of the class distribution of the objects that
they contain. To do that, your program should output a two dimensional matrix of dimensions (# of clusters)*(# of
classes) whose entries will be the number of objects of a particular class that belongs to a particular cluster. To
evaluate the quality of the clustering solution that you obtain, your program needs to compute the entropy and purity
of the clustering solution with respect to that class distribution (you can find the equations for these two measures in
the clustering slides).

Here is a sample command line for your program:

```
kcluster input-file criterion-function class-file #clusters #trials output-file
```

The format of the input file must be in the (i,j,v) format, where i is the article ID (i.e., the NEWID), j is the dimension #,
and v is the corresponding value (based on the vector model used). Note that the input file should be a commaseparated file.

The format of the class file must be in the (i, label) comma-separated format, where i is the article ID and label is the
topic.

The format of the output file must be in the (i, cluster#) comma-separated format, where i is the article ID and cluster#
is a number between 0 and #clusters-1 indicating the cluster in which article i belongs to.

In your experiments, you should set #trials to be 20. Please use these 20 seeds: [1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39].

Upon finishing, your program should print the value of the criterion function for the best trial in addition to both the
entropy and purity of the best clustering solution to the standard output.

Report

You need to prepare a report in which you describe if you did extra/different preprocessing steps than the ones
described here in order to clean the data. You should also evaluate the performance of your code in terms of its
runtime and also the quality of the clustering solutions (in terms of entropy and purity) for the different criterion

functions and vector models. To do that, report results for 20, 40, & 60 clusters for each of the 3X3 combinations.

Extra Credit

Develop a partitional clustering algorithm that implements for the E1 criterion function. Conduct all the experiments and
report the results. Based on the correctness and performance, it will worth up to an additional 20 points.

Submission

DO FOLLOW ALL OF THESE INSTRUCTIONS
1. You need to submit a directory that contains your code along with the report.
2. The report should be named "report.pdf" and should be in PDF format.
3. Your code should be in a single file named "kcluster.<extension>" and should run on the CSELabs machines.
4. The directory should include the various scripts that you developed to perform the text processing, that they read
the pristine dataset from a folder called "reuters21578". Make sure that your scripts can run "as is" on the CSE
lab machines and do not require any custom packages that you may have installed in your own directories. If
needed, and depending on how you structured those scripts, you might need to provide a "driver" script such that
the TAs can execute it using the makefile in a "pristine" version of the dataset as downloaded from UCI and
generate the four representations along with the associated files to perform the clustering.
5. The directory should contain a Makefile that will include a "kcluster" target and a "preprocess" target such that :
by simply doing "make kcluster" it will build the "kcluster" executable.
by doing "make preprocess", you should read the pristine dataset from a folder called "reuters21578" and
generate :
all the three input files, named as "freq.csv", "sqrtfreq.csv", and "log2freq.csv".
the class file, named as "reuters21578.class",
a label file named as "reuters21578.clabel" that stores a mapping between the dimension numbers and
the corresponding stems. The format of that file will just have one token per line sorted in increasing
order by dimension #.
6. The directory should be named as "<username>" and uploaded in as a "<username>.tar.gz" archive.