

## CSCE-470 Programming Assignment #2 Clustering: K-means

**Due: Sunday, April 21th, 2019 by 11:00pm**

In this programming assignment, you will implement the clustering algorithm K-means to divide a collection of documents into a set of clusters, exactly as we have described in class. Please refer to slide 23 of the file flat-clustering-k-means.pdf for the pseudo code.

**Note** that the pseudo code did not specify the stopping criterion, in this assignment, the K-means algorithm stops when the number of documents shifted into a different cluster in the step of vector reassignment is  $\leq E$ , with  $E$  being an integer to be tuned and set by you.

There is no starter code provided for this assignment other than a separate evaluation code used to evaluate the performance of your algorithm and generate a score based on clustering results. Therefore, you can choose a programming language you like to use.

But **note**: you should make sure that your implementation of the K-means algorithm outputs clustering results in the same format as specified in the later *Output Formatting* section. You will be provided with a real text data set used for developing your K-means algorithm, and for grading, we will run your clustering algorithm on a separate held-out text set and evaluate clustering results using the same evaluation code.

### Other Important Notes:

1. each document should be represented by a bag of word vector, and then you should normalize each vector using the euclidean normalization before you start to cluster documents.
2. you should calculate euclidean distance between a document vector and a cluster centroid vector, instead of cosine similarity.

### Key Steps:

**(2 Points!)** Building document representations: read in text contents and build a normalized bag of word vector for each document.

**(1 Points!)** Centroid Initialization: randomly select documents as the initial centroids.

**(3 Points!)** Reassignment of vectors: assign each document in the collection to its closest centroid.

**(3 Points!)** Recomputation of centroids: calculate new centroids for clusters.

---

## DATA

If you unzip the data.zip file, within the generated data folder, you will find a collection of cleaned documents in the folder “train”, and the file “train.txt” containing the gold class assignments formatted exactly as specified in the *Output Formatting* section. Your clustering algorithm will take the collection of documents as input and generate a set of clusters. Your clustering algorithm should save the clustering results in a .txt file and **name the result file “pred.txt”**. Then you will run the provided python evaluation code, “evaluate.py”, to obtain a score. By default, the evaluation code looks for the two files, “train.txt” and “pred.txt”, and takes them as the the gold clustering result file and the generated clustering

result file.

---

## The Evaluation Metric NMI

We will compare an algorithm generated clustering with the gold clustering and generate a score using the normalized mutual information (NMI) metric. The NMI score ranges from 0 (no mutual information) to 1 (perfect correlation), the higher the score is, the better the system generated clustering is. When the NMI score is 1, the system generated clustering completely matches with the gold clustering.

**Note**, this metric is independent of the absolute values of the cluster labels: a permutation of the class or cluster label values won't change the score value in any way. In other words, what matters is the generated clusters, not how you label those clusters.

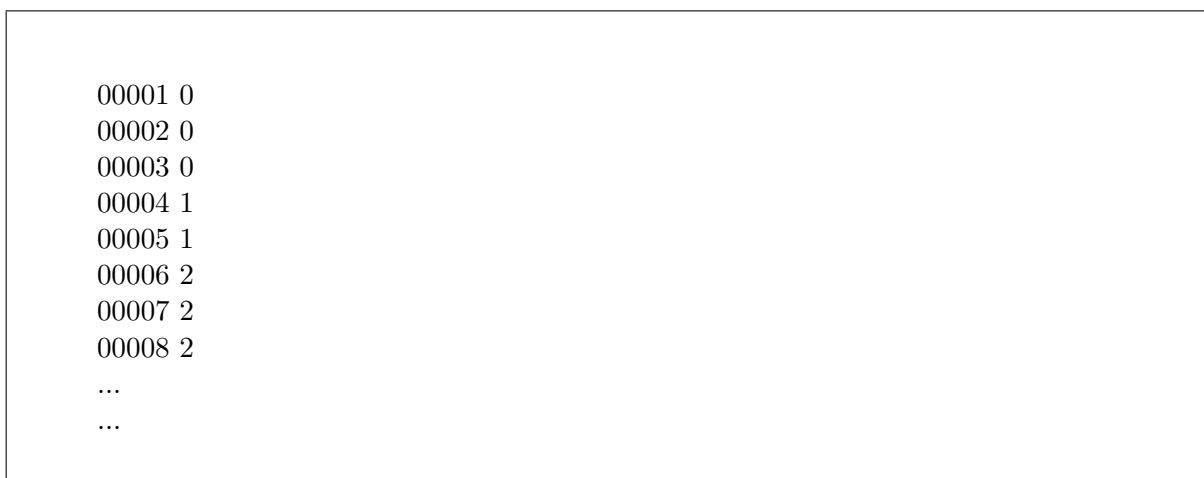
**Note**, the evaluation code depends on python packages numpy and scipy, please refer to the README file for instructions on how to install them, if your system does not already have these packages installed. You should not modify the evaluation code, and the same evaluation code will be used for grading.

---

## Output Formatting

Your K-means implementation should save the clustering results into a plain txt file named as "results.txt". The result file has a line for each document, with the document ID followed by its cluster ID, with a tab key in between separating the two values. The documents do not need to be ordered in the result file.

The results will look something as illustrated in figure 1. Your program output should follow exactly the same structure.



```
00001 0
00002 0
00003 0
00004 1
00005 1
00006 2
00007 2
00008 2
...
...
```

Figure 1: Sample Output

---

## The Report

**(6 Points!)** Please write up a 1-3 page report describing your observations and thoughts. In particular, you need to address the following questions in the report:

1. **Initial centroids (2 Point!)** Try a different set of documents as the initial centroids, does it change the clustering results and the NMI score? How many iterations did it take for the algorithm to converge and finish? Repeat this procedure several times.
  2. **Number of clusters (2 Point!)** Try to change the number of clusters (the  $K$  in K-means), consider numbers in the range (2, 20), report the NMI score with each value you tried for this parameter. For each parameter value you tried, how many iterations did it take for the algorithm to converge and finish?
  3. **Stopping criterion (2 Point!)** Try to change the parameter  $E$  used in the stopping criterion, report the NMI score with each value you tried for this parameter. For each parameter value you tried, how many iterations did it take for the algorithm to converge and finish?
- 

## GRADING CRITERIA

Your code will be graded mainly based on the correctness. You should tune your parameters using the provided training set. We will run your clustering algorithm on a separate held-out text set and make sure it runs properly. Your written report will be graded based on the grade allocations on individual items. Cheating on either code or report will be harshly penalized.

---

## ELECTRONIC SUBMISSION INSTRUCTIONS (a.k.a. “What to turn in and how to tun in”)

You only need to submit 2 things:

1. The source code files. **please compress the whole src folder into a .zip file and submit the .zip file.**
2. The written report in pdf format.

**NOTE:** Please do **NOT** include the data files or any other file in your submission.

How to turn in:

1. please submit the two files, the source code .zip file and the written report pdf file, via ecampus.