Applied probability models for CS
Exercise 3

General

In this exercise you will implement the EM clustering algorithm for unsupervised classification of articles into clusters. You need to read and understand the material in the document *Underflow Scaling and Smoothing in EM*, provided with this exercise. This document is part of the course material.

The input

Since the classification task is unsupervised, you should use the develop.txt file only in this exercise (same file as provided with the previous exercise). This time you should use the topic information for your evaluation (the header of each article contains a list of topic labels for this article) as explained below.

Unsupervised classification

Your goal is to split the articles in develop.txt into clusters based on the unigrams (words) that they contain. You should ignore the topic labels when clustering the articles and only use them for evaluation after clustering is complete (see explanations below). You will see that good clustering algorithms will cluster together articles that have more in common with each other than with articles in other clusters. Since we assume that the topic labels correspond to what the article is about, we expect each cluster to have one or few dominant topics.

Your program should first use EM as shown in class to cluster the articles into 9 clusters. Then, it should make a hard assignment of every article to its most probable cluster, i.e. the cluster with the highest likelihood for this article.

Implementation instructions

Underflow - In this exercise you will easily fall into underflow. Read the “Underflow and Smoothing in EM” file to learn how to avoid it.

Time and place complexity - In order to reduce time and place complexity you should filter rare words. A rare word, for this exercise, is a word that occurs 3 times or less in the input corpus (develop.txt).

Choosing λ - In the smoothing part of the M step you should choose a λ. For this exercise, no need to test all possible values of lambda, but do experiment with a few options to make sure that you get reasonable performance.

EM initialization - As you learned in class, one option to initialize the EM algorithm is to do it randomly. In this exercise you will implement another initialization procedure:

1. There are 2124 articles in your input (develop.txt only). Split them into 9 initial clusters in a modulo-9 manner (i.e. the 1^{st} article to cluster 1, the 2^{nd} to cluster 2, ... the 9^{th} to cluster 9, the 10^{th} to cluster 1 and so on). Note that you are splitting according to the ordinal number of the article and not its id.
2. Compute α_i and P_{i_k} as in M-step.

Report
In your report you should analyze the structure of the 9 clusters you have found. Please enumerate your results as listed here.

1. Decide what would be your threshold to stop the EM iterations. Report this threshold.
2. Remember that after each iteration of the algorithm the likelihood should increase (or remain the same). This is a great debugging tool. Print out the log likelihood after each iteration and plot a graph in which the X axis is iteration number and the Y axis is the log likelihood. Calculate the mean perplexity per word and plot those values as well (the perplexity value should decrease or remain the same after each iteration). Add these two graphs to your report.
   - Perplexity is a measurement of how well a probabilistic model predicts a sample, and it is used to compare between probabilistic models. Here, you need to use it to measure how well your model predicts articles' topics. Compute it using the model's log likelihood and normalize it by the size of the dataset (the number of words): $2^{-\frac{1}{N}\times LogLikelihood}$ (or e if you've used ln for the likelihood computation).
3. Create a *confusion matrix* M of $9X9$. The rows should be the 9 clusters you find and the columns are the 9 topics from Reuters (according to topics.txt). Columns should be ordered from left to right in the same order as the topics appear in topics.txt. Add another column that indicates the size of each cluster (i.e. the number of articles that were assigned to this cluster). Rows should be sorted by cluster size in descending order.
   - In each cell M_{ij} you should write the number of articles from the j^{th} topic in the i^{th} cluster.
   - In your report, print the confusion matrix in *human-readable* format, and include column and row names. The exercise grader is (unfortunately) still human.
4. Use this matrix and an application like Excel to create 9 histograms of topics, one for each cluster. The X axis should be the 9 topics (in the same order as in the matrix's columns) and the Y axis should be the number of articles from that topic in the cluster. In the histogram title write the number of the cluster (corresponding to row number in the matrix).
   - Label each cluster with its most dominant topic (the topic that has the largest number of articles in this cluster). Add this label to the title of its histogram.
5. Calculate and report the accuracy of your classification, taking as an article's class the dominant topic of its most likely cluster. It is enough that the topic you assigned to an article is one of its topics, as assigned by Reuters, to consider this assignment correct. If we define *assignment* as labeling an article with a topic then we define accuracy as follow:

   $$\text{Accuracy} = \frac{\text{number of correct assignments made by your model}}{\text{total number of assignments made by your model}}$$

   Your model's accuracy should be greater than 0.5.
6. Clearly report the following numbers: k (the constant from the underflow treatment in EM), vocabulary size after filtering (should be 6,800) and the λ you use for smoothing the M step.

The code
Observe the following directions when writing your code:

1. Write your code in Python.

2. Please do not use Python packages that would compute major chunks of this exercise for you, or that depend on specific Python versions.
3. Write clear code with informative comments.
4. The first line of each code file should include *your_names your_ids*.
5. The name of the main source code file that runs your program should be ex3.py.
6. Use double precision floating points for real numbers' arithmetics.

Submission

You may work either in pairs or individually.

Submit your code files(s) and your report (either in PDF or MS doc format), all in a single zip file, via Moodle. When working as a pair, one student should submit the zip file, and the other should submit a text file called 'pair.txt' (the content of this file does not matter).

Note: in this exercise you will probably encounter tiny details that were not fully defined here. This is the nature of bringing the equations from class to practice. This is the best way to fully understand the material. When you find an undefined detail you should think for yourself how to cope with it and implement your solution. Don't forget to report any such detail that you have found and how you chose to solve your problems.