Coursework	**Fall 2016**
Statistical Pattern Recognition (COMP136)	**Roni Khardon**
Department of Computer Science, Tufts University	**Rishit Sheth**

Programming Project 4

This programming project is due by noon on Monday, December 12 in hardcopy (in class) and electronically (via provide).

Latent Dirichlet Allocation (LDA)

In this programming project we implement LDA and inspect its performance both in an unsupervised manner and when used as a preprocessing step for supervised learning. Your goals in this assignment are to (i) implement the collapsed Gibbs sampler for LDA inference, and (ii) compare the LDA topic representation to a "bag-of-words" representation with respect to how well they support document classification.

We will use a subset of the *20 newsgroups* dataset[1], available through the course web page. The subset consists of 200 documents that have been pre-processed and "cleaned" so they do not require *any* further manipulation, i.e., you only have to read the space-separated strings from the ASCII text files. Each document belongs to one of two classes. The file *index.csv* holds the true labels of each document.

Task 1: Gibbs Sampling

In this portion, your task is to implement the collapsed Gibbs sampler for LDA. Recall that Gibbs sampling is a type of inference method that can be used in cases where the exact posterior distribution is intractable. The output of this process is a sample from the true posterior distribution. In the case of LDA, the output represents a sample of the (hidden) topic variables for each word. This sample of topic variables can be used to calculate topic representations per document. Algorithm 1 describes one version of a collapsed Gibbs sampler.

For this project, fix the number of iterations to run the sampler at $N_{iters} = 500$. The Dirichlet parameter for the topic distribution is $\alpha\mathbf{1}$ where $\mathbf{1}$ is a vector of ones with K entries (K is the number of topics), and $\alpha = \frac{50}{K}$. The Dirichlet parameter for the word distribution is $\beta\mathbf{1}$ where $\mathbf{1}$ is a vector of ones with V entries (V is the size of the vocabulary), and $\beta = 0.1$.

We have provided an additional smaller dataset, *artificial*, for developing your implementation. Running your sampler on this dataset with $K = 2$ and the above parameters, you should find the three most frequent words per topic to be $\{$*bank, river, water*$\}$ and $\{$*dollars, bank, loan*$\}$ (not necessarily in those orders).

[1] `http://ana.cachopo.org/datasets-for-single-label-text-categorization`

Algorithm 1 Collapsed Gibbs sampler for LDA

Require: Number of topics K, Dirichlet parameter for topic distribution α, Dirichlet parameter for word distribution β, number of iterations to run sampler N_{iters}, array of word indices $w(n)$, array of document indices $d(n)$, and array of initial topic indices $z(n)$, where $n = 1 \dots N_{words}$ and N_{words} is the total amount of words in the corpus.

Generate a random permutation $\pi(n)$ of the set $\{1, 2, \dots, N_{words}\}$
Initialize a $D \times K$ matrix of topic counts per document C_d, where D is the number of documents
Initialize a $K \times V$ matrix of word counts per topic C_t, where V is the number of words in the vocabulary
Initialize a $1 \times K$ array of probabilities P (to zero)
for $i = 1$ to N_{iters} **do**
    for $n = 1$ to N_{words} **do**
        $word \leftarrow w(\pi(n))$
        $topic \leftarrow z(\pi(n))$
        $doc \leftarrow d(\pi(n))$
        $C_d(doc, topic) \leftarrow C_d(doc, topic) - 1$
        $C_t(topic, word) \leftarrow C_t(topic, word) - 1$
        for $k = 1$ to K **do**
            $P(k) = \frac{C_t(k, word) + \beta}{V\beta + \sum_j C_t(k, j)} \frac{C_d(doc, k) + \alpha}{K\alpha + \sum_l C_d(doc, l)}$
        end for
        $P \leftarrow$ normalize P
        $topic \leftarrow$ sample from P
        $z(\pi(n)) \leftarrow topic$
        $C_d(doc, topic) \leftarrow C_d(doc, topic) + 1$
        $C_t(topic, word) \leftarrow C_t(topic, word) + 1$
    end for
end for
return $\{z(n)\}, C_d, C_t$

Once you have verified that your implementation works correctly, run your sampler with $K = 20$ on the *20 newsgroups* dataset. After the sampler has finished running, output the 5 most frequent words of each topic into a CSV file, *topicwords.csv*, where each row represents a topic. Include these results in both your report and submission.

Finally, you'll need the topic representations for the next part. For a document *doc*, this will be a vector of K values, one for each topic, where the kth value is given by $\frac{C_d(doc,k)+\alpha}{K\alpha+\sum_l C_d(doc,l)}$ and C_d is output from the sampler.

Task 2: Classification

In this portion we will evaluate the dimensionality reduction accomplished by LDA in its ability to support document classification and compare it to the bag of words representation.

The first step is to prepare the data files for the two representations. The first is given by the topic representation of the previous section, where each document is represented by a feature vector of length K. The second representation is the "bag-of-words" representation. This representation has a feature for each word in the vocabulary and the value of this feature is the number of occurrences of the corresponding word in the document divided by the total number of words in the document.

For the evaluation we will reuse the logistic regression implementation from project 3, in particular your implementation of Newton's method for this problem. You should use the value $\alpha = 0.01$ for the regularization parameter of logistic regression in this part.

Your task is to generate learning curves in the same way you did there: Step 1) Set aside 1/3 of the total data (randomly selected) to use as a test set. Step 2) Record performance as a function of increasing training set size (with each training set randomly selected from the other 2/3 of the total data). Repeat Steps 1 & 2 a total of 30 times to generate learning curves with error bars (i.e., $\pm 1\sigma$). Performance is defined as classification accuracy on the test set.

Plot the learning curve performance of the logistic regression algorithm on the two representations. Then discuss your observations on the results obtained.

Additional Notes

- Aside from standard I/O, math, and plotting libraries **NO** external libraries should be used for this assignment.
- Please submit the logistic regression implementation with this project so that your code can be used and tested without further manipulation. You do not need to print this portion of the code for paper submission.
- A random sample drawn from a probability mass function P over K values can be implemented in the following way:
  1. Divide the unit interval $[0, 1]$ into K distinct sub-intervals whose lengths match the probabilities specified by P
  2. Draw a random number r from $[0, 1]$ with respect to the uniform distribution
  3. Output the index of the sub-interval that r falls into
- Please plan your time and run time for your program. A Matlab implementation of the collapsed Gibbs sampler took ≈ 20 minutes to run on the *homework* server.

Submission

- You should **submit** the following items **both electronically and in hardcopy**:
  (1) All your source code for the assignment. Please write clear code with documentation as needed. The source code should (i) run on *homework.eecs.tufts.edu*, (ii) run from the command line *without editing* with a single command (if there is more than one execution command required, include those commands in a Bash script which we can run), and (iii) output the requested results.
  You can assume the data files will be available in the same directory as where the code is executed. Please use filenames as provided for the data. Please include a short README file with the code execution command.
  (2) A PDF report on the experiments, their results, and your conclusions as requested above.

- For electronic submission, put all the files into a zip or tar archive, for example `myfile.zip` (you do not need to submit the data we give you). Please do not use another compression format such as RAR. Then submit using `provide comp136 pp4 myfile.zip`.
- Your assignment will be graded based on the clarity and correctness of the code, and presentation and discussion of the results.