Homework 5 (Due July 14)

CS4412: Data Mining
Kennesaw State University
Summer 2023

In this project, we will explore Wikipedia as a dataset. Our goal is to automatically scrape a small set of related articles from Wikipedia, and look for topical structure among the articles, using a topic model.

The task for this project is outlined as follows:

1. We will first collect a small dataset from Wikipedia. A script called `download.py` is included which allows you to specify an initial article on Wikipedia, and the script will automatically download that article, as well as all articles linked to by the first article. This constitutes a small to medium-sized dataset of articles that are related to the initially specified article.

2. We will next learn a topic model from the collected dataset. A script called `topic.py` is included which can learn such a topic model. This script is an implementation of a topic model from scratch based on the `NumPy` library for matrix math. The script is included for convenience, but also for you to see what an implementation of a topic model could look like. You are free to use any topic modeling software available for this and the next step.

3. Finally, we will examine the results of the topic model, and see what insights it might provide on the dataset, in terms of helping to understand the topical structures that appear in the articles that we downloaded from Wikipedia. The included script `topic.py` automatically prints out a visualization of all topics (the word-given-topic distributions) learned from data.

You are encouraged to select a specialized or esoteric article as the initial article used to collect data from Wikipedia (using the `download.py` script). In any case, it is suggested that you select a subject in which you are personally knowledgeable of, so that you would be capable of interpreting the resulting topics that are learned from data.

Downloading a Dataset from Wikipedia

The included `download.py` script can be used to download a dataset of articles from Wikipedia. One initially specifies a starting page, by setting the `main_page` variable at the top of the script. When you run the script, the script will then download the starting page, as well as all Wikipedia pages that are liked to by the starting page. All of the wikipedia articles will be downloaded as text files into a `data` directory which will be created by the script. If you want to download a new set of articles based on a different initial page, then you should remove or rename the `data` directory.

A dataset is included with the project, where the article for `Morty_Smith` (a character from the TV show "Rick and Morty") was used as the initial article. If you want to use a different dataset, one should again remove or rename the `data` directory first, since it already contains the `Morty_Smith` dataset. Note that, depending on how many links the initial article has, it may take minutes to hours to download the dataset from Wikipedia. It is suggested that you include an initial article with at least 100 links. An initial article with over 1,000 links may take a non-trivial amount of time to download. Note that the `download.py` script prints out the number of links that an article has, and asks for confirmation before downloading the dataset.

Training a Topic Model

The included `topic.py` script can be used to train a topic model from the Wikipedia articles downloaded by the `download.py` script. The `topic.py` script assumes that articles were downloaded in the `data` directory, which is the default location that the `download.py` script uses. Thus, the `topic.py` can be run without any initial changes to the script. At the top of the script, there are some options that can be changed:

```
count_limit = 20
topic_count = 10
max_iterations = 100
```

The `topic_count` variable is perhaps the most important variable. This variable specifies how many topics that the topic model should assume. If this count is too low, then topics may be merged together which would form overly general topics. If the count is too high, then topics may be split apart, leading to overly specialized or incoherent topics. In general, the smaller the corpus, the fewer the number of topics you should specify, although you may considering experimenting with this number.

The `count_limit` variable specifies the number of times that a word has to appear in the corpus to be included in the dataset. The idea is that if a word only appears a small number of times, in a very small number of different articles, then it may not be sufficiently important to form a topic around. The `max_iterations` variable specifies the number of iterations that the EM algorithm should run. This number should be increased if you observe that the EM algorithm is not converging, i.e., if the log likelihood (`ll`) that is printed by the script is still increasing by a significant amount. The variable can also be reduced to make the script run faster, although the quality of the resulting topics may become worse.

Visualizing a Topic Model

Once the `topic.py` script finishes running, it will print out a visualization of all topics, i.e., all of the word-given-topic distributions. For example, consider the following example of a topic that the script may print out:

```
========================================
== topic 0
========================================
                rick | 6.3235%
               morty | 4.7499%
             episode | 2.4159%
              season | 1.4778%
                beth | 1.1815%
               jerry | 1.1468%
              series | 1.0305%
              summer | 0.8498%
               smith | 0.7282%
             roiland | 0.6736%
           character | 0.6374%
              voiced | 0.6302%
                 one | 0.5826%
              family | 0.5569%
          characters | 0.5021%
              harmon | 0.4836%
                show | 0.4728%
              justin | 0.4315%
                 two | 0.4182%
               adult | 0.4125%
```

This is a topic that was found on a dataset based around the initial article of `Morty_Smith`, which is a character from the TV show "Rick and Morty". To visualize this topic, we list the top-20 most likely words appearing in this topic. Based on the words appearing in this topic, this is clearly a topic about the show "Rick and Morty."

Consider the following topic that was also learned by the topic model:

```
========================================
== topic 8
========================================
               adult | 2.8691%
                swim | 2.6502%
                 new | 1.2997%
                show | 1.2165%
            magazine | 1.1252%
             network | 0.9362%
              series | 0.9292%
                also | 0.7985%
               shows | 0.7932%
               block | 0.7849%
               aired | 0.7467%
             website | 0.6608%
                york | 0.6603%
               media | 0.6170%
            episodes | 0.5672%
             cartoon | 0.5569%
         programming | 0.5456%
               night | 0.5341%
               would | 0.5126%
             channel | 0.5114%
```

This topic appears to be about the cable station, Cartoon Network, which "Rick and Morty" appears on. The existence of this topic suggests that the Cartoon Network was a frequently appearing subject in the article `Morty_Smith` and its related articles.

Submit: Provide the following information about the dataset and topic model that you learned from it.

1. Provide the name of the initial article, and a short one-sentence summary of what the article is about.
2. Report the page count (the number of articles appearing in the dataset) and the word count (the number of unique words appearing in the dataset) as reported by the output of the `topic.py` script (if you are using a different topic modeling system, provide similar statistics about the dataset).
3. provide at least three examples of topics (like the table in the example above) that were learned from your dataset, and provide a one-sentence summary of what each topic could represent.
4. Were there any topics that appeared, that you did not expect to see? Were there any topics that did not appear, that you expected to see? Please provide any comments in either of these two cases.

Turn in: the answers to the above questions as a text file or as a pdf file (no Microsoft Word .doc's please), onto the course website under `Assignments` and `Homework 5`. Assignments are due Friday, July 14 by 11:59pm. Please start early in case you encounter any unexpected difficulties. Late projects are accepted without penalty until solutions are posted.

Included files:

- `hw5.pdf`: this document
- `download.py`: a python script that downloads a dataset from Wikipedia
- `topic.py`: a python script that learns a topic model from a dataset downloaded by `download.py`

- `dataset.py`: a python script for representing a text dataset (you do not need to run this script yourself)
- `stopwords`: a text file containing a list of "stopwords" or commonly occurring words that are dropped before learning the topic model (since they are unlikely to help provide better topics).
- `data`: a directory that contains the dataset downloaded by the `download.py` script. Initially, this directory contains a dataset based on the article `Morty_Smith`. If you download a new dataset, the `data` directory should be removed or renamed first.