NLP Homework 01: Corpus Analysis

Winter 2024 Semester

1 Overview

The goal of this assignment is to gain experience analyzing a corpus of your choosing using methods covered in the class.

2 Requirements

You should perform the following steps:

1. Collect a dataset,
2. Convert to bag-of-words format,
3. Compute Naïve Bayes probabilities for terms in each category,
4. Run topic modeling, and
5. Experiment with the effects of text normalization on the entire process.

More details about each step are below.

2.1 Dataset

You should find a text dataset that you are interested in learning more about. *Your dataset/corpus must be organized into two or more distinct* ***categories****.* Some ideas include:

- Books from Project Gutenberg from different authors, genres, decades, etc.
- Subtitles from Open Subtitles or The Internet Movie Script Database. Categories might be genre, which TV series episodes/scenes come from, time period, etc.
- Posts from two or more different subreddits from Reddit.com, or posts from one subreddit from different time periods.

- Congressional Hearings from different years, presidential terms, committees, or political parties.
- Chat logs from different Twitch streamers. One example is this dataset.
- A Text Classification Dataset from the huggingface datasets hub (note these are uploaded by the community and the quality may vary).
- Any other dataset you are interested in – try to choose something you will enjoy learning more about!

A few other notes:

Size. You should aim to have *at least* 100 documents per category (though you may have 1000s or more). Remember that "document" is a lose concept here: a book can be converted into many documents by treating each chapter as its own document, a tv series could be split into episodes, etc.

Format. The dataset should *not* already be processed, i.e., you should not use a dataset that is already tokenized and in bag-of-words format.

Language. Your dataset can be in any language(s) but you should be able to describe/analyze what you find in English for the report, and you may need to research and apply specialized preprocessing steps on your own if you work with a non-English language.

Exceptions. If there is a dataset that you are passionate about studying further that does not appear to meet the requirements above, please reach out to Prof. Wilson to discuss further and there is likely a way that we can make it work for the assignment.

2.2 Bag-of-words

After you have your corpus, you should process it to convert it to a BOW format. You do not need to provide any specific output to prove that you did this, but it will be required for the subsequent steps. You may use your code from Homework 00 to preprocess the data, write new preprocessing code, or use another 3rd-party library if you like (though you will need to understand what that library is doing). If you want to store your data as a document-term matrix and are having trouble with the amount of memory this requires, you may find the SciPy Sparse Matrix Representations useful. These methods do not require you to store all of the 0s in your matrix and can lead to a much smaller memory footprint.

2.3 Naïve Bayes

Next, use a Naïve Bayes model of probability to compute the probabilities of word belonging to the categories you defined earlier. Our goal here is to produce lists of words that represent what is unique about each category compared to the other(s). You don't need to split into training and testing sets and experiment with different approaches to maximize your f-1 score (there will be another

assignment about that in the future). In this assignment, we won't actually use the *classifier* part of Naïve Bayes, we will just use this type of model as a way to estimate probabilities and associate words with classes.

You should use the data that you have collected to compute the probabilities of each word belonging to a given category, $P(w|c)$ as you would in a Naïve Bayes model. Then, to find out which words are *most* associated with a category, c, we want to know *how much higher* this probability is compared to $P(w|c_o)$ for the other categories $c_o \in C_o$ where C_o is the set of other categories you are considering (not including c). To determine this we can compute the **log likelihood ratio**:

$$llr(w, c) = log\left(\frac{P(w|c)}{P(w|C_o)}\right) = log(P(w|c)) - log(P(w|C_o)),$$

basically, *how much more likely* is it that we observe the word given that the document belongs to class c than it is to observe the same word given a document that is *not* in class c. If you have only 2 classes, $P(w|C_o)$ is equal to $P(w|c_o)$ where c_o is the single class other than c, and so $llr(w, c_o) = -llr(w, c)$. Otherwise, more generally, it will be:

$$P(w|C_o) = \frac{\sum_{c_o \in C_o} count(w, c_o)}{\sum_{c_o \in C_o} \sum_{w' \in V} count(w'|c_o)}.$$

where C_o is the *set* of classes that are not c and V is the vocabulary for the entire corpus. You may also want to use add-one smoothing for all of the probabilities.

Given this, you should produce a list of the top 10 words sorted by their log-likelihood ratios for each class $c \in C$, the set of all classes (*i.e.*, the categories from the dataset you selected). When computing your probabilities, you may use count data, binary, or even tf-idf transformed data.

Optional bonus analysis: if you find these results include too many rare words with high probabilities or frequent words have too low of probabilities, you may try using the "log odds ratio informative Dirichlet prior" method shown in section 2 of this blog post to account for that.

2.4 Topic Modeling

Run Latent Dirichlet Allocation using all of the documents in your corpus. You may choose the number of topics that you feel is most appropriate and gives the results that either look most reasonable to you or optimize a metric like coherence. You do not need to implement LDA yourself, and should use a 3rd-party library like gensim or mallet (or any other reputable library you find for LDA topic modeling) to do the topic modeling and may use libraries like PyLDAvis to help present your results. You may also use jsLDA for both topic modeling and generating some interesting visualizations (but you may *not* use the example corpus presented there).

You should present your topics (or a selection of the topics you think are most interesting/useful) in a table where the first column contains your own manually

assigned label for the topic (e.g., "school"), and the subsequent columns contain a the top 10 (or more) terms from that topic, sorted by their probability of belonging to that topic, along with their probabilities (e.g. "homework" (0.02), "class" (0.01), and so on).

Finally, for each category, find the average distribution of all topics for documents in that category and report the top 3-5 topics for each category. You can determine this by taking the topic distribution for each document in a given category and averaging the probabilities for each topic across all documents in the category.

2.5 Experimentation

After your code is written and you are able to complete all of the steps, try at least 1 additional variation of text normalization (e.g., using the options available from hw0) and 1 additional variation in the bag-of-words representation (counts, binary, tf-idf). Take note of how the results from all steps changed, and try to decide which of the configurations gives you the most insightful results in the end.

More than just writing the code and getting it to run, your overall goal with the entire assignment should be to produce an insightful analysis of your dataset, comparing and contrasting the documents from each of your categories. This does not mean you results need to be "surprising" – many of your results may make perfect sense to you, but you should be able to see that simply quantifying these results across a large set of documents is meaningful.

3 Deliverables

You should submit the code you used as well as a PDF report documenting your approach and findings.

3.1 Code

Your code can be written in any language but should include enough documentation/instructions for someone else to be able to run. You may include the code directly in your submission on Moodle (as a compressed archive) or provide a link to a GitHub repository. Your code should include a README file that explains the files/directories and how to set up and run the code.

You are welcome to use code snippets from examples in class, things you find online, or from AI code generation tools, just make sure to give proper attribution to code you didn't write. However, if you happen to find a codebase that does the entire assignment already (e.g., a student's project from a past semester at OU or elsewhere), you may not just copy it.

It is okay if some steps were done manually – it is not required that you automate the entire process from dataset collection to generation of your figures and tables from the report. For example, it is not a problem if you don't

generate things like tables and figures using code but instead did this using another tool like MS Excel. The code should just showcase how you did things like preprocessing, computing probabilities, and topic modeling.

3.2 Report

Your report should have the following structure (or something similar that captures these main elements). To get full credit for the assignment, please make sure you include everything below.

1. **Dataset.** Describe the dataset you chose and why you think it is interesting/useful to analyze. How did you collect the data, choose the categories, and split it into "documents"? Include tables/figures to show the size of your dataset, e.g., a table with the number of documents and the average number of tokens per document, broken down by category.

2. **Methodology.** Describe the steps that you performed and what informed your decisions along the way. For example, if you decided *not* to lowercase the text because it gave your better results at some later stage, include that decision and your reasoning for doing that. This should include your preprocessing steps (in detail, e.g., lowercasing, stemming, etc., do not just say "each document was preprocessed") and the kinds of analysis that you performed. For any steps that you didn't implement yourself (e.g., topic modeling), mention which package/library you used.

3. **Results and Analysis.** Present your results as formatted tables/figures (they should not just be listed in the body of a paragraph). This must include at least the results of the required steps, but may also include any other interesting findings you came across (for example, you could show the results of topic modeling both with and without a certain preprpocessing step that you noticed made a large difference in the quality of the results). For each table/figure, include a description of your main takeaways or findings.

4. **Discussion.** Include 2 subsections in your discussion. The first should cover what you learned about your dataset – you might imagine that you are describing what your results showed, at a high level, to a friend who doesn't have NLP experience but is interested in the corpus that you chose. The second subsection should cover what lessons you personally learned during the completion of the assignment. You might write about finding and processing data, preprocessing and its effects on topic modeling results, limitations you noticed with the approaches used, or anything else.

Formatting: The specific formatting of the report is up to you, but part of your grade will be based on having a well-organized and professionally presented document. Please avoid things like blurry, low-resolution/poorly-cropped screenshots, submitting one long paragraph with no subsections/formatting, or

copying and pasting long strings from your program output that have no formatting applied. There is no minimum/maximum page limit for the report, but in a single-column format similar to the one that this document is written in, around 3-5 pages is the expected length (including figures and tables).

4 Help/Questions

Please ask at any time on Discord or stop by office hours (in person or on Zoom) if you need advice/guidance/pointers on any aspect of the homework. You are also free to discuss your approach and ideas with your classmates, but you should not share code or reuse data. As always, you should take full responsibility for the deliverables that you submit.