UNIVERSITY OF SOUTHERN MAINE

Text Mining and Analytics, Spring 2024, Assignment 2

Instructor: Behrooz Mansouri (behrooz.mansouri@maine.edu)

Due: Feb 26, 2024

We are completing the first-class activity in this assignment. The objective is to train unigram and bigram language models, which will then be used to classify lyrics based on their genre.

Task 1: Finish the code for training both uni-gram and bi-gram language models using the files that were provided for the first class activity.

Task 2: Having finished task 1, now, for a given text, you can find the probability of it being generated by each genre, using unigram or bigram language models.

Create a third file, called MixedModel.py, where the probabilities from unigram and bigram language models are combined. For input text *T*, and genre *G*, the combined probability is:

$$P_{combined}(T|G) = \lambda P_{combined}(T|G) + (1 - \lambda)P_{combined}(T|G)$$

In this assignment, λ is a value ranging from 0 and 1. You should find the optimal value for λ on the validation set, using a step size of 0.1. The validation set should be constructed using a portion of the training data, likely in a 90:10 split, with 10% allocated for validation. It is important to split based on the genre, ensuring %10 of each genre's songs will be in the validation set.

Task 3: The next step is to evaluate your models. There is a test set provided for you (test.tsv), with gold labels. For unigram, bi-gram, and mixed models, measure F1-score. Provide a table of results in your report. You should also have a code to print the results in the console.

Task 4: Apply significant testing to determine whether the results from one model are significantly better than the others. In your report, discuss what test you have used and what are the results. You should also have a code to calculate and show this.

Task 5: Analyze your results, discussing cases where each model failed or succeeded. You should provide one example for each case and each model, and discuss the model's behavior. It is important to note that all these labels are generated by AI, which may introduce noise into the results.

Notes: Part of your grade is based on the achieved results, particularly the F1-Score. You can change the pre-processing steps or other techniques to improve your results. If you decide to do so, both your report and code should reflect what you have done.

Notes for submission:

Files to submit:

1. Python files (only .py is acceptable, not .ipynb) that are named accordingly for each language model.

2. A ReadME file, explaining how to run your code.

3. A .pdf file containing your answers to tasks 3 to 4.

Note 1: If you are using GitHub, you should still submit the files requested above.

Note 2: All your codes will be run from scratch, creating the language model and classifying the test cases. The test cases should never be seen during the training phase.