

COSC 426 F24 Lab 6

Introduction

In this lab you will use bigram language models for Bayesian classification. By completing this lab, you will demonstrate that you:

- Are able to generalize your knowledge of n-gram models to new contexts.
- Understand the classification pipeline
- Understand the Bayes rule and can work with probabilities and log probabilities
- Can work with classes in python
- Can work with basic pandas dataframes

Provided files

- `BigramModel.py`
- `Lab6.py`
- `glove_vocab.txt`
- A folder, `sample_text_data` with sample data useful for debugging Part 2
- A folder, `sample_review_data` with sample data useful for debugging Part 3
- [A google doc template](#)

What to submit

- `BigramModel.py`
- `Lab6.py`
- A pdf of your google doc

Part 1

Say you are given two large text files, one with positive reviews and one with negative reviews. You are also given a set of test sentences that you want to classify as being either positive or negative. Describe how you will use Bayesian classification with bigram models to solve this task. After answering this question, talk me through your approach before continuing on.

Part 2

In this part, you will complete the class in `BigramModel.py`. You already implemented most of the methods in your previous lab, so you should be able to copy-paste most of your code. Note: If your code from before was incorrect, that is alright. Just get output in the correct format.

You will add one new function `evaluate`, which essentially generates by-word predictions from the model and saves it in a format similar to the output of `evaluate` for MinimalPair experiments in `NLP Scholar`.

Note, that in moving your code over you are moving them into a class. To refer to a function in the class, say `func`, you do `self.func`.

Part 3

In this part, you will complete all of the functions in `Lab6.py`. First, review the provided functions. Sketch out what will happen in `main` and what will happen in `calc_accuracy`. What functions will you call? What information will you need? Etc. Write your sketch in your google doc. **Once you have written your sketch, please talk me through it.**

Now code up the functions. **Make sure to write and test your code incrementally.**

Part 4

Answer the following question in the google doc

1. Say you didn't have a hypothesis about what the best `add-k` value was, describe how you could use the code you wrote to find the optimal `k` value. In your answer, reflect on the need for splitting the data into a `train`, `validation` and `test` split.