Portfolio Chapter 8: Ngrams

Individual or Pair Assignment

Objectives:

- Gain experience creating ngrams from text
- Build a language model from ngrams
- Reflect on the utility of ngram language models

Turn in:

- Upload your .py code to eLearning for grading
- Upload your .py code to your portfolio, and create a link to on your index page
- This program should be written with an IDE

Overview:

- In this homework you will create bigram and unigram dictionaries for English, French, and Italian using the provided training data where the key is the unigram or bigram text and the value is the count of that unigram or bigram in the data. Then for the test data, calculate probabilities for each language and compare against the true labels.
- You may use an IDE or a notebook for this assignment
- create a pdf of your output interspersed with commentary

Instructions:

1. Program 1: Build separate language models for 3 languages as follows.
    a. create a function with a filename as argument
    b. read in the text and remove newlines
    c. tokenize the text
    d. use nltk to create a bigrams list
    e. use nltk to create a unigrams list
    f. use the bigram list to create a bigram dictionary of bigrams and counts, [‘token1 token2’] -> count
    g. use the unigram list to create a unigram dictionary of unigrams and counts, [‘token’] -> count
    h. return the unigram dictionary and bigram dictionary from the function
    i. in the main body of code, call the function 3 times for each training file, pickle the 6 dictionaries, and save to files with appropriate names. The reason we are pickling them in one program and unpickling them in another is that NLTK ngrams is slow and if you put this all in one program, you may waste a lot of time waiting for ngrams() to finish.

Caution: All course work is run through plagiarism detection software comparing students’ work as well as work from previous semesters and other sources.

2. Program 2.
    a. Read in your pickled dictionaries.
    b. For each line in the test file, calculate a probability for each language (see note below) and write the language with the highest probability to a file.
    c. Compute and output your accuracy as the percentage of correctly classified instances in the test set. The file LangId.sol holds the correct classifications.
    d. output your accuracy, as well as the line numbers of the incorrectly classified items
3. Narrative. Write a one-page or more narrative about Ngrams:
    a. what are n-grams and how are they used to build a language model
    b. list a few applications where n-grams could be used
    c. a description of how probabilities are calculated for unigrams and bigrams
    d. the importance of the source text in building a language model
    e. the importance of smoothing, and describe a simple approach to smoothing
    f. describe how language models can be used for text generation, and the limitations of this approach
    g. describe how language models can be evaluated
    h. give a quick introduction to Google's n-gram viewer and show an example
4. Create a link to your programs and narrative on your index page

Hints for the programs:

Creating the dictionaries in Program 1:
You can use the NLTK ngrams() function to create a bigrams and a unigrams generator object. Then you can iterate over each to create the dictionary using Python's .count() string method to extract counts from the text you read in.

Calculating probabilities in Program 2:
The probabilities will be large enough so that you don't need to use logs, we will simply multiply the probabilities together. Each bigram's probability with Laplace smoothing is: (b + 1) / (u + v) where b is the bigram count, u is the unigram count of the first word in the bigram, and v is the total vocabulary size (add the lengths of the 3 unigram dictionaries).

Grading Rubric for the code:

Element	Points
Program 1	50
Program 2	50

Caution: All course work is run through plagiarism detection software comparing students' work as well as work from previous semesters and other sources.

Narrative	50
Total	150

Caution: All course work is run through plagiarism detection software comparing students' work as well as work from previous semesters and other sources.