Natural Language Processing (CS 4063)

Assignment 1

- Deadline to submit this assignment is: **Thursday 20th February 2025 11.59 p.m.**

Part 1: Regular Expressions and Preprocessing

Install nltk in Python by following steps

1. Go to pip folder in Python installation path (Find python installation path by running "where python" in command prompt (cmd) in windows)
   For example: cd C:\Program Files\Python\Python36-32\Scripts
2. Run "pip install –U nltk"
3. Run "nltk download()" (This command will download corpus etc. for nltk)
4. Run "pip install beautifulsoup4" (This software is needed for parsing html files)

You can get help on using nltk for this homework from following link

https://www.nltk.org/book/ch03.html

Q1) Describe the class of strings matched by the following regular expressions.

a. [a-zA-Z]+
b. [A-Z][a-z]*
c. p[aeiou]{,2}t
d. \d+(\.\d+)?
e. ([^aeiou][aeiou][^aeiou])*
f. \w+|[^\w\s]+

Test your answers using nltk.re_show(). (You will have to import libraries using "import nltk, re, pprint")

Q2) Write regular expressions to match the following classes of strings:

a. A single determiner (assume that *a*, *an*, and *the* are the only determiners).
b. An arithmetic expression using integers, addition, and multiplication, such as 2*3+8.

Q3) Write a utility function that takes a URL as its argument, and returns the contents of the URL, with all HTML markup removed. Use from urllib import request and then request.urlopen('https://www.csail.mit.edu/people/').read().decode('utf8') to access the contents of the URL. Use BeautifulSoup(html).get_text() to parse html. (incase the given url doesn't work, try another website)

Import the following for this question:

(from urllib import request

from bs4 import BeautifulSoup)

Q4) Tokenize text parsed from the above url using nltk. Find all phone numbers and email addresses from this text using regular expressions. (Do not tokenize text otherwise email addresses will be incorrectly tokenized)

Q5) Use the Porter Stemmer to normalize some tokenized text, calling the stemmer on each word. Do the same thing with the Lancaster Stemmer and see if you observe any differences.

Q6) For second part of this assignment, assume you have a shy friend who is hesitating to tell you something, so he/ she sent a long random text on WhatsApp that also contains his/ her message. Since you are a Regex Guru, your task is to extract the actual message from the random text using regular expressions and some rules.

Copy paste the below text (without quotes) on https://www.regexpal.com/ or sublime and follow the rules mentioned below to extract the actual message. (Provide answers for this part in a word file, with screenshots of the regex and its output)

"Pila Forfeited you engrossed but 1kometimes explained. Another 1kacokaco1 as studied it to evident. Merry sense 9given he be arisepila. Conduct at an replied removal an amongst.
Remainingzalima 0determine few her two cordially Zalima admitting old. Sometimes ctra*nger his pisdsdla ourselves her co*la depending you boy. Eat discretion cultivated possession far comparison projection pila considered. And few fat interested discovered inquietude insensible unsatiable increasing zalima eat."

Rules:

Message consists of five words.

1. First word starts with a letter 'Z' or 'z', followed by zero or more letters between 'a' and 'z' and ends with a letter 'a'.
    a) Write down the regular expression to extract first word.
    b) What is the frequency (count) of first word in random text?
    c) What's the first word?
2. Second word starts with a digit, followed by a letter 'k', followed by zero or more letters between 'a' and 'z' and ends with a digit.
    a) Write down the regular expression to extract the second word.
    b) Write down the word you extracted using above regular expression.
    c) Remove first and last three letters/ digits from word you get in part b) to get

actual second word. What's the second word?

3. Third word starts with a letter 'c', followed by zero or more letters between 'a' and 'z', followed by a star '*', followed by one or more letters between 'a' and 'z' and ends with a letter 'a'.
    a) Write down the regular expression to extract the third word.
    b) Write down the word you extracted using above regular expression.
    c) Remove star '*' from word you get in part b) to get actual third word. What's the third word?
4. Fourth word starts with a letter 'P' or 'p', followed by exactly two letters between 'a' and 'z' and ends with a letter 'a'.
    a) Write down the regular expression to extract the fourth word.
    b) What is the frequency (count) of fourth word in random text?
    c) What's the fourth word?
5. Well, if you have correctly extracted first four words, you can easily predict the fifth word. Write down the complete five word message that your shy friend sent you.

Submission

Submit your code file on Google Classroom. Write code for all questions 1 till 6 in one file. For Q6 you will provide a word/pdf file with output screenshots.

Part 2: Language Models and Smoothing

This question requires you to train some language models on a training corpus and to test them on two smaller corpora. Stater code is also provided with this assignment. Each sentence must be surrounded by a start of sentence and end of sentence marker (<s> ... </s>). Preprocess to add these sentence markers. These markers will allow your models to generate sentences that have realistic beginnings and endings.

Implement the following models:
1. **UnigramModel**: an unsmoothed unigram model
2. **SmoothedUnigramModel**: a unigram model smoothed using Laplace (add-one) smoothing,
3. **BigramModel**: an unsmoothed bigram model,
4. **SmoothedBigramModelLI**: a bigram model smoothed using Linear Interpolation

For each of the four language models, you need to implement the following methods:

generateSentence(self): returns a sentence sent that is generated by the language model. sent is a list of the form [<s> w1, ….., wn </s>]. You can assume that <s> starts each sentence (with probability 1). The following words (<s> w1, ….., wn </s>) are generated according to your language model's distribution. The number of words (n) is not fixed. Instead, you stop generating a sentence as soon as you generate the end of sentence symbol </s>.

getSentenceProbability(self, sen): returns the probability of the sentence sen (which is again a list of the form [<s> w1, …. , wn </s>]) according to the model.

Please use the provided generateSentencesToFile method and your unigram and bigram language models to generate 20 sentences (saved as unigram output.txt, smooth unigram output.txt, bigram output.txt, and smooth bigram kn output.txt).

Perplexity

You need to compute the perplexity (normalized inverse log probability) of the two test corpora according to all of your models. Evaluate the models on the test corpora. Do you see a difference between the two test domains?

Questions

1. When generating sentences with the unigram model, what controls the length of the generated sentences? How does this differ from the sentences produced by the bigram models?
2. Consider the probability of the generated sentences according to your models. Do your models assign drastically different probabilities to the different sets of sentences? Why do you think that is?
3. Generate additional sentences using your bigram and smoothed bigram models. In your opinion, which model produces better / more realistic sentences?
4. For each of the four models, which test corpus has a higher perplexity? Why? Make sure to include the perplexity values in the answer.

Submission

Submit your code file on Google classroom. Write code for Part 2 (all questions) in one file. Answer the 4 questions as comments at the end of your code file.