Assignment One: Text preprocessing, N-grams and language models
CS918: 2019-20

Submission: 12pm Wednesday 6th November 2019

Notes:

a) This exercise will contribute towards 10% of your overall mark.
b) Submission should include Python code, a Jupyter notebook and a short report.

Preparation: Getting to know Python

Practice using the Jupyter notebooks from the module website.

For this exercise you will be using the "SIGNAL NEWS1" corpus provided on the module website, available through the following link:
https://warwick.ac.uk/fac/sci/dcs/teaching/material/cs918/signal-news1.tar.bz2

The corpus provides news stories formatted in JSON. Each line contains a JSON item with a news story. You should be using the "content" field of the news stories in this exercise.

You will be delivering the Python code that you developed, a Jupyter notebook that comments on your code and a short report (2-3 pages) that describes your work. For the Python code, you should make sure to develop code that runs on Python 3, and to deliver it as a single, standalone Python script that works on the corpus files as they were provided (i.e. we should be able to run it with Python 3 having the "SIGNAL NEWS1" corpus downloaded).

The exercise consists of three parts:

Part A: Text preprocessing (25 marks)

1. After lowercasing all the text, use regular expressions to parse and clean the texts:
   a) Remove all non-alphanumeric characters except spaces, i.e. keep only alphanumeric characters and spaces. [4 marks]
   b) Remove words with only 1 character. [4 marks]
   c) Remove numbers that are fully made of digits (e.g. you should remove the number '5', but in the case of '5pm', made of both digits and letters, you should keep it as is, without removing the digit that is part of the word). [4 marks]
   d) Remove URLs. Note that URLs may appear in different forms, e.g. "http://www.*", "http://domain", "https://www.*". [7 marks]

   NOTE: The preprocessing above may need to be processed in a different order, not necessarily as listed above.

2. Use an English lemmatiser to process all the words. Use of a POS tagger is optional, and you may instead assign each word the default POS tag when using the lemmatiser. [6 marks]

Part B: N-grams (20 marks)

With all the texts preprocessed as above, compute the following calculations:

1. Compute N (number of tokens) and V (vocabulary size). [5 marks]
2. List the top 25 trigrams based on the number of occurrences on the entire corpus. [5 marks]
3. Using the lists of positive and negative words provided with the corpus, compute the number of positive and negative word counts in the corpus. [5 marks]
4. Compute the number of news stories with more positive than negative words, as well as the number of news stories with more negative than positive words. News stories with a tie (same number of positive and negative words) should not be counted. [5 marks]

Part C: Language models (20 marks)

1. Compute language models for trigrams based on the first 16,000 rows of the corpus. Beginning with the bigram "is this", produce a sentence of 10 words by appending the most likely next word each time. [10 marks]
2. Compute the perplexity by evaluating on the remaining rows of the corpus (rows 16,001+). [10 marks]

Total: 65 marks

NOTE: we will NOT be giving marks for efficiency, but we will need to run your code to see the output, so you should make sure that your code runs in a reasonable time so that we can re-run it during marking, i.e. no more than 5 minutes.