DALHOUSIE UNIVERSITY
Inspiring Minds

Winter 2020 (Jan6-Apr6)
Faculty of Computer Science

CSCI 4152/6509 — Natural Language Processing

Assignment 3

Due: *Thursday, April 2, 2020 by midnight*
Worth: 121 marks (= 25 + 20 + 31 + 25 + 20)
Instructor: Vlado Keselj, CS bldg 432, 902.494.2893, vlado@dnlp.ca

---

Assignment Instructions:

The submission process for Assignment 3 is mostly based on the `submit-nlp` command on `bluenose` as discussed in the lab, or in the equvalent way by using the course web site, where you need to follow 'Login' and then the 'File Submission' menu option. Some questions may have some specific different submission instructions, in which case you should follow those.

Important: You must make sure that your course files on bluenose are not readable by other users. For example, if you keep your files in the directory `csci6509` or `csci4152` you can check its permission using the command:

```
ls -ld csci6509
```
or
```
ls -ld csci4152
```

and the output must start with `drwx------`. If it does not, for example if it starts with `drwxr-xr-x` or similar, then the permissions should fixed using the command:

```
chmod 700 csci6509
```
or
```
chmod 700 csci4152
```

1) (25 marks) Complete the Lab 6 as instructed. In particular, you will need to properly:

a) (5 marks) Submit the file '`hmm_tagger.py`' as instructed.

b) (5 marks) Submit the file '`crf_tagger.py`' as instructed.

c) (5 marks) Submit the file '`brill_demo.py`' as instructed.

d) (5 marks) Submit the file '`ne_chunker_exercise.py`' as instructed.

e) (5 marks) Submit the file '`first_notebook.ipynb`' as instructed.

2) (20 marks) Complete the Lab 7 as instructed. In particular, you will need to properly:

a) (10 marks) Submit the file '`tweets.csv`' as instructed.

b) (10 marks) Submit the file '`find_hashtags.py`' as instructed.

3) (31 marks) Complete the Lab 8 as instructed. In particular, you will need to properly:

a) (5 marks) Submit the file '`gcd.prolog`' as instructed.

b) (5 marks) Submit the file '`prog1.prolog`' as instructed.

c) (5 marks) Submit the file '`factorial.prolog`' as instructed.

d) (8 marks) Submit the file '`task.prolog`' as instructed.

e) (8 marks) Submit the file '`task-queries.txt`' as instructed.

4) (25 marks) Submit your answer using the command `nlp-submit` or the equivalent page on the web site either as file `a3q4.txt` or `a3q4.pdf`.

Consider the phrase 'woolloommooloo mall'. We will consider an alphabet of 26 lowercase English letters, and the space character used to separate the words, which makes the total vocabulary of 27 characters. Our main task it to create a uni-gram character model; i.e., a Markov Chain model, of this sentence, or in other words, to calculate probability estimates for each character.

a) (7 marks) What are probabilities of the 27 characters based on the given phrase if we do not use smoothing?

b) (8 marks) What are the probabilities if we use Laplace add-one smoothing method?

c) (10 marks) What are the probabilities if we use Witten-Bell smoothing method?

5) (20 marks, programming) Write and submit a program written in Perl, Python, C, C++, or Java, named `a3q5.pl`, `a3q5.py`, `a3q5.c`, `a3q5.cc`, or `a3q5.java`, which calculates the cosine similarity between two files.

The program will be given three filenames, for example `term-list`, `file1`, and `file2`, where `term-list` contains a list of words to be used in vectors for cosine similarity. The program will prepare a vector v_1 of counts of all words from the list in `term-list` in the file `file1` and another vector v_2 of the counts of the same words in the file `file2`, and then it will print the cosine similarity between those vectors. The words may consist of letters, digits, and the underscore character (`_`). The words should be counted in a case insensitive way.

The program is run from the command line as one of the options:

```
./a3q5.pl term-list file1 file2
./a3q5.py term-list file1 file2
./a.out   term-list file1 file2
java a3q5 term-list file1 file2
```

where `a.out` is an executable file obtained either from a C or C++ source code.

The output should be one line with a real number.

It is possible that one of the vectors is a zero-vector, in which case calculating cosine similarity would be impossible. If this is the case, your program should print one line of text as follows (ending with a new-line character):

```
Cosine similarity not defined! (Division by zero!)
```