Homework #1

CS 525/DS 595, Spring 2019

---

100 points total [6% of your final grade]

Due: January, 31, 2019 by 11:59pm
[no submission will be accepted after Feburary 3, 2019 at 11:59pm]

Delivery: Submit via Canvas

For this assignment, you will:
(0 pts) Get started with Python
(100 pts) Implement boolean retrieval with Python

Part 0: Getting Started with Python
As in all the homeworks this semester, you will be using Python. So let's get started first with our Python installation.

Python Installation and Basic Configuration
First off, you will need to install a recent version of python 3 in here. There are lots of online resources for help in installing Python. Here are several depending on your platform:

Windows: http://docs.python-guide.org/en/latest/starting/install3/win/
https://docs.python.org/3/using/windows.html

Linux: https://www.digitalocean.com/community/tutorials/how-to-install-python-3-and-set-up-a-local-programming-environment-on-ubuntu-16-04

Mac OSX: http://docs.python-guide.org/en/latest/starting/install3/osx/

Alternatively, there is a nice collection called Anaconda, that comes with Python plus tons of helpful packages that we may use down the line in this course:

Anaconda: which has install instructions for Windows, Linux, and Mac OSX.

Part 1: Boolean retrieval with Python
For this assignment, we will build a basic boolean search engine, including the tokenization, stemming, indexing, and basic boolean query components. Begin by downloading hw1.zip and decompressing it; you should find three python files and five sample document files.

a) cs525.py - This helper class will be used to represent a student's identifying information in all programming assignments. In this assignment, you don't need to modify it. But, you must instantiate a student object in hw1.py. Any assignments without an instantiated student object of type Student will not be graded.

b) PorterStemmer.py – Implemenation of Porter Stemmer. Just use it for stemming. Do NOT modify it.

c) hw1.py - This is where you will fill out four functions: index_dir(...), tokenize(...), stemming(...) and boolean_search(...). Your overall goal is to build a basic index that can support simple boolean queries (AND, OR queries only).

- index_dir(self, base_path): This function should crawl through a nested directory of text files and generate an inverted index of the contents of these files. We give a simple example in the comments of hw1.py to illustrate what the resulting index should look like. This function should call tokenize and stemming as helper functions.

- tokenize(self, text): This helper function should take a raw string (either extracted from each file or input by the user as a query) and convert it to a list of valid tokens. For the purposes of this assignment, a valid token consists of only English alphabet characters (a-z, A-Z) or numbers (0-9). All other characters are considered as token delimiters. For example, the text 'http://www.cnn33.com' should result in the tokens: 'http', 'www', 'cnn33' and 'com'. The text: 'This 3rd class is cool, right?' should result in the tokens: 'this', '3rd', 'class', 'is', 'cool' and 'right'. Note that you should convert all letters to lower case.

- stemming(self, tokens): This helper function should take a list of tokens returned from the tokenize function. Convert the list of tokens to a list of stemmed tokens. For the stemming task, you should use Potter Stemmer, importing PorterStemmer.py.

- boolean_search(self, text): This function search for the terms in the string text in the inverted index. This function should call tokenize and stemming as helper functions to tokenize the string text and do stemming, respectively. In order to make search function simple, we assume "text" contain either single term or three terms. If "text" contains only single term, search the term from the inverted index and return document names containing the term. If "text" contains three terms including "or" or "and", do OR or AND search depending on the second term ("or" or "and") in the "text". For example, if the boolean_search is called with ‘mike or sherman’, then the return value should be a list of documents that include ‘mike’ OR ‘sherman’. If the boolean_search is called with ‘mike and sherman’, then the return value should be a list of documents that include ‘mike’ AND ‘sherman’.

We have provided a sample set of five documents. Feel free to test your code over these five documents. You should be able to manually check whether your code is doing what you think it should. Once you submit your final code, we will evaluate it by constructing an index over our own set of 1,000 documents and issuing several queries (single term and three terms containing either “or” or “and”).

What to turn in:

- Submit to Canvas your hw1.py file.
- This is an individual assignment, but you may discuss general strategies and approaches with other members of the class (refer to the syllabus for details of the homework collaboration policy). At the top of hw1.py you will see a list of COLLABORATORS. Please fill this out with the names of classmates you consulted and the nature of your discussion.