Course Project Progress Update (11/28/20)

pmanden2@illinois.edu

Objectives

Objective of my project is to improve the ExpertSearch system. I will attempt to achieve the following enhancements in this project, as mentioned in the project proposal.

1. Given a URL, use Naïve Bayes classifier to classify it as a directory page or a non-directory page
2. Given a URL use Naïve Bayes classifier to classify it as a faculty page or a non-faculty page

Which tasks have been completed?

Getting the baseline code to work

I have already spent quite a bit of time trying to get the baseline ExpertSearch code on Git to work. After trying different python versions, python library versions, trying on Linux and Windows and various experiments and debugging, I got it to work on Python 2.7 on Windows with some code changes. Now I have the baseline to do the actual implementation.

Generating negative samples for directory and faculty pages

I have written web scraping code to achieve the following:

1. Wrote code to get a list of all universities in US (from here - http://doors.stanford.edu/~sr/universities.html)
2. Wrote code to scrape each of the university in the list above, and identified 10 links
3. Wrote code to clean up the list to exclude directory/faculty URLs, so that it can be used as "negative" samples for directory and faculty classes.

Now I have the positive and negatives samples for the directory and faculty URLs.

Core Naïve Bayes classifier code

Core code that implements Naïve Bayes classifier has been written. It can accept file names of positive and negative samples, load data, create term document matrix etc. Also provides a function for classification.

Which Tasks are pending?

1. Need to get all the code to run on one version of Python, that I haven't been able to do so far.
2. Clean up code, better documentation

I also will attempt to do the following (I am not very familiar with javascript/UI, so I may not be able to do this):

1. Change the UI so that an additional option can be added to the UI to type in a directory page or a faculty page so that classification results can be seen visually in UI

Are you facing any challenges?

As indicated above, I have managed to get the baseline ExpertSearch code to work on Python 2.7. However, my code (classifier etc.) doesn't run on 2.7 (it runs on 3.8). Need to figure out a way to get all code to run on one version. Dealing with Python versions and libraries continue to be a pain.

Thoughts on work beyond the scope of this project

In order to achieve full automation of identifying and extracting faculty pages, we will first need to automatically identify the directory and faculty pages on a university website, given a root URL for the university. This can be done if a list of university websites is available (one such list of universities in US is available here - http://doors.stanford.edu/~sr/universities.html)

With full automation of identifying faculty web pages will look as below:

1. For each university
    a. Get list of all URLs on the website by:
        i. Finding the sitemap file (sitemap.xml, …)
        ii. Or by crawling the website and generating a full list of all URLs on the website
2. From the list of URLs generated above
    a. Create 2 training sets
        i. One for directory pages classification (using the directory pages listing from Coursera?)
        ii. One for faculty pages classification (using faculty URLs provided)
3. Use Naïve Bayes Classifier to identify URLs that are directory pages (If the ultimate objective is to find the faculty pages, there is really no need to find the directory pages, as the sitemap will contain the full faculty pages. And hence this step can be eliminated)
4. Use Naïve Bayes classifier to identify URLs that are faculty pages
5. Scrape each faculty page classified as a faculty page

My project implements part of this work. Maybe the rest can be done by a future student of this class!

References

https://sebastianraschka.com/Articles/2014_naive_bayes_1.html

https://medium.com/analytics-vidhya/naive-bayes-classifier-for-text-classification-556fabaf252b

https://towardsdatascience.com/implementing-a-naive-bayes-classifier-for-text-categorization-in-five-steps-f9192cdd54c3

https://www.xml-sitemaps.com/

https://code.google.com/archive/p/sitemap-generators/wikis/SitemapGenerators.wiki

http://doors.stanford.edu/~sr/universities.html

https://pagedart.com/blog/how-to-find-the-sitemap-of-a-website/