Cameron Whistler

Friends of Cherry Valley Internship

findDonorGMaps

Purpose: The purpose of this program is to find potential donors to Friends of Cherry Valley Natural Wildlife Refuge. This program attempts to achieve this goal by using Google Maps API to find businesses local to the Friends of Cherry Valley Natural Wildlife Refuge address. The Google Maps API can then be used to return details about these businesses, including the URLs of these websites (if one exists). The websites of these businesses are then scraped to see if they contain links that contain certain keywords that may indicate that a website values charity.

Explanation of program:

The program is comprised of three files, and makes use of two classes.

One of these files is named regexs.py. This file contains a class named Regexs. This class is simply meant to contain regular expressions within a list. Most of these regular expressions are named legex, which is short for link-regular expression. These legexs are meant to find certain keywords which may indicate charitableness, within the hyperlinks of a webpage. This is an example: legex2 = '<a .\*? href="[^"]\*?.communit.\*?">.\*?</a>'. As this example shows, certain keywords are shortened to allow variation in spelling. This example legex could therefore return community or communities.

The class Regexs contains an initialization function which returns all the regular expressions within a list. The class also contains a list of the keywords that we are scraping for. This list is used to append these keywords to the context column of the excel file created at the end of the program.

Another of these files is named contextCreator.py. contextCreator.py contains a class named Contextor. The primary purpose of Contextor is to return a list of strings that add context to the URLs that the program scrapes. This list of strings will be output to excel in a column, next to their corresponding website URLs.

Contextor contains four functions. The first two of these functions are used mostly for debugging. increment\_char() can be used to increment the number of charitable websites that are found, and increment\_con() can be used to increment the number of context strings that need to be created. These numbers should be identical to each other, in order to ensure that the output to excel is working as expected. The third function within Contextor is named intake(string) and it simply allows a website URL to be appended immediately into the list of charitable websites within Contextor.

The final function within Contextor is addContext2(list of Strings). This function is responsible for most of Contextor’s functionality. The list of strings passed to AddContext2() will be a list of website URLs. AddContext2() makes use of a triple nested for loop. The first of these for loops is used to go through each of the items within the list of strings. Here a Boolean named notCharitable is set to true. If this Boolean remains true at the very end of the function, then a website’s context will be set to “No indication of being charitable.” Also within this for loop, the function uses python’s requests library to get the response and html text from a website. If the response isn’t request code 200 then a context will be created indicating that the website could not be accessed and a continue statement will be executed, advancing the for loop. The next for loop is used to go through every regular expression returned by the Regexs class. This loop checks whether the website accessed by the URL in the first for loop contains any hyperlinks containing any of these regular expressions. If even one of them is contained within a hyperlink, then the function creates the context “Possibly charitable. Keywords found: ”, the Boolean notCharitable is set to false, and the function enters the third for loop. The third for loop is used to check if, in addition to the previously found regular expression, the website contains hyperlinks containing any other keywords. If a hyperlink does, then the context is updated to include that keyword. After the third for loop has completed, the second for loop will execute a break statement to prevent adding more context than necessary, advancing the first for loop. When the first for loop reaches an end, the list of contexts will be returned to the caller of the function.

The final file is the main.py file. This file contains a function that converts miles to meters. Main makes use of a Google maps API key and an address (the one the user wants to search nearby). Main accepts input for the search term determining what businesses are returned and input for how many miles away from the address Google maps should search. Main uses the pandas library to create a data frame based on Google Maps search results. This data frame will then be output to an excel file named temp. Main then grabs the column of website URLs stored within temp and passes them to AddContext2(). Every URL returned by Google Maps is passed to AddContext2(). AddContext2() returns a list of contexts, which main then stores as a column, within temp. Finally, temp is copied to a master excel form which contains the result of every business type searched, listed within sheets named for that search term.

Other things to note: I ran this program in Replit, a free, web based compiler. <https://replit.com/> Using Replit requires creating a free account. Replit allows you to upload files to its projects, so these files will need to be uploaded to Replit if you wish to use it. The code in main expands upon code from this source: <https://learndataanalysis.org/source-code-search-nearby-businesses-with-google-maps-api-and-python/#google_vignette>

In order to use this program: You must use Google’s cloud platform (either with purchase or a free trial) and you must enable the Google Maps API. The video attached to this source displays how this can be done: <https://learndataanalysis.org/source-code-search-nearby-businesses-with-google-maps-api-and-python/#google_vignette> . Once that is done you must place your Google Maps API key into this program (line 18 of main) (line 17 can be removed if you wish, that is how the secret functionality works). I personally used the secret functionality on Replit to hide my API key so that others couldn’t access it.