**Craigslist Scraper Documentation**

**Goal**

Create a web scraper to collect data on Craigslists "gigs" listings and output to an excel for easy reading.

**Methodology**

* There is a main listings view that gives clickable urls for individual listings. 120 listings are available per page, and passing a url variable (eg. s=120) will get to the next page.
* Scrape through all these pages to make a master list of all listing urls
* Loop through the list of urls to scrape each individual listing page and scrape the data needed (title, posting date, compensation)
* Use some logic, post data collection, to parse out compensation values/rates
* Write this data to a CSV file

**Function descriptions**

**def main():**

Calls the buildListingList() function to build master list of urls to scrape. Then, calls postingExtractor() to scrape the data for the individual listings.

**def buildListingList():**

Output: returns a list of urls for all the listings

Uses a loop to cycle through the urls of listing pages, then stores the individual urls in the linksList variable.

**def postingExtractor(link):**

Input: url for the page to be scraped

Output: CraigsList\_Data.csv

Extracts important info from the listing page: title, compensation, posting date. Then it opens the output CSV and appends a row of data.

**def processRaw():**

Main function for the post processing. Calls the parseCompensation and parseTitle functions.

**def parseCompensation(compensation):**

Reads the text for the compensation and parses out the pay rate and whether it is a flat rate or hourly using regex.

**def parseTitle(title):**

Reads the title and tries to decide if it is a sketchy ad or not. A lot of craigslist ads are questionable in the type of work they ask for and sometimes specify that it is for women only. This flags these ads based on their title to make it easier to filter out.

**Future developments**

**Avoiding IP blocking**

I was able to scrape the first 500 or so listings without issue. However, when I tried for a larger corpus of data, I had my IP blocked. If I were to make this a more stable tool, I would implement two methods to avoid getting blocked again.

* Enforce wait times between requests
* Make use of private proxies to make requests.

**Allow user inputs for easier customization**

* Rewrite parts of the code so that user can specify numbers of reviews to scrape
* Currently hardcoded to scrape only Boston gigs. Allow for specifying of location in the future.