Annie Williams (maw3as)
André Zazzera (alz9cb)

CS 5010 Homework 3: Web Scraping Report

Introduction:

Many booksellers in the modern era rely on online traffic and sales for their livelihoods. We wanted to examine what a potential bookseller's website may contain as readily-available data, and analyze some relevant queries that a buyer or seller might use as a metric for performance.

The site we examined was http://books.toscrape.com/, a demo bookstore website built specifically for web scraping practice, and hence, according to the site itself, all of the prices and ratings "were randomly assigned and have no real meaning". That being said, we continued to perform our data analysis as if the site were a real one.

Approach to this assignment:

Before beginning this assignment, we both read the web scraping tutorial from GeeksforGeeks. We used BeautifulSoup to scrape the book website for relevant data including book title, price, availability, and rating, using code snippets from the tutorial to help us along the way with tasks like converting a list of dictionaries into a CSV file.

Kind of data:

We decided to scrape the data available on the website's book catalogue, pulling the title, price (in pounds sterling), availability (in stock or out of stock), rating (on a 5-star scale), and reference link.

Python algorithms and libraries:

The libraries we used for this assignment include BeautifulSoup, and python's request and csv libraries. As mentioned earlier, we also used the GeeksforGeeks tutorial to help learn how to read in the data from the site, and output it to a csv We first downloaded a physical copy of the website's HTML, and used BeautifulSoup to open the file in the python script. Our implementation next uses a for loop to move through all fifty pages of books in the catalogue. For each page of the book website, we used BeautifulSoup to collect the necessary data from the HTML, and append it to a universal directory of book data.

Reasons for use:

Our program rather neatly organizes all of the books in the site's catalogue, along with the most relevant information a buyer may want to know, and a link to the books' details page on the site. If a user wants to understand the performance of certain titles in the catalogue, or how quickly books of a certain price range or rating go out of stock, our program can help them sift through the analysis.

Analysis Results:

Contrary to expectation, it appears that the mean price for books does not significantly deviate according to the rating, implying that there are other factors at work when determining the price of a book.

```
Most expensive book =  59.99
Cheapest book =  10.0
Average Price for 1 star book =  34.56
Average Price for 2 star book =  34.81
Average Price for 3 star book =  34.69
Average Price for 4 star book =  36.09
Average Price for 5 star book =  35.37
```

Using the pandas .corr() method, we see that the correlation coefficient between the Price and Rating variables is only about 0.028, indicating that there is trivial correlation between these two variables. Again, this would be surprising if the data were from a real bookstore.

The average rating of books for sale on this site is about 2.92 stars out of 5, indicating that the store is less interested in selling “high-quality” books, and perhaps more on niche texts.

Potential Extensions:

On the seller site catalogue, the book availability status is marked simply as “In Stock” or “Out of Stock”, but if a user clicks on a particular title, the site will show them how many copies of that title remain in stock. It would be of tremendous help if our web scraper was able to relay that quantitative data alongside the binary categorical data of in and out of stock. Similarly, we could possibly extend the scraper to pull each book’s universal product code (UPC), which would enable buyers to quickly look at how other sellers are marketing the same item, and sellers to quickly look up relevant information about the title (stock pictures, reviews, stock information, etc).

Extra Credit:

We also sorted our catalogue list by book title and wrote a binary search function (to keep the complexity of O(log(1000)), much lower than O(1000)!) that will return whether or not the book you want to find is in the catalogue, as well as its price, availability, rating, and a link to the book’s detailed information page in the catalogue.