Mini Project 1 – Where do I book an accommodation?

Project Report

Name: Aicha Moussaid

Email: Aicha.Moussaid@abo.fi

Introduction

The project at hand offers a system for the user to book an accommodation. The main idea consists of web scraping multiple websites and collecting enough relevant data to be used in our system, handling the data and making it usable, visualizing important features, and offering a user interaction system.

The goal of this approach is to get important information related to various accommodations in a target city and make it useful and informative, in a way that gives the general user an idea of the best deals available, saving them time from the tedious task that is browsing manually multiple websites.

Although there are websites that include a built-in comparison system, some of them host multiple offers from unreliable websites, which can lead the user to a bad experience and sometimes even risk being scammed.

The general approach to the problem seems simple on paper but gets more complex when implementing it. Multiple challenges arise at different levels, but the most prominent one stays the web scraping approach.

In this report, we will investigate these challenges and their workarounds and discuss the implementation and analysis of the problem.

Implementation

1- Data Collection

For this first part of the project implementation, the requirements were to collect data from 3 different websites by web scraping. The target data was the available accommodations in a city that offered at least 50 housing options. The city chosen for this case is Marrakech, Morocco, for the time frame of 1st to 8th of October 2022.
The chosen websites were Booking.com, ZenHotels.com, and CheapOair.com. The selection of these three websites was not easy because, to use them in one system, they have to harbor some sort of conformity with each other when it comes to the information we want to collect from them. Some websites are also a lot stricter in their “anti-scraping” policy, which aborted multiple tries at different other websites.
The main goal of web scraping is that, instead of a human end-user clicking away in their web browser and copying interesting bits into a spreadsheet, web scraping passes this process to a computer program that can perform it considerably faster and more accurately than a human can.
The general implementation in python that is easy and beginner-friendly stays Beautiful Soup. Because of its basic capabilities, it is a truly “beautiful” tool for web scrappers. It can assist the programmer in swiftly extracting data from a certain web page, from HTML and XML files.

But, since it can only handle static scraping as it retrieves web pages from the server without using a browser, often times web scraping with python requires more than this library.
When the target data involves interaction with JavaScript links, as in clicking buttons, scrolling, finding hidden content, or pagination in general, the use of dynamic scraping becomes crucial. This is why, in the implementation of this project, we use an optimal technique that is combining Selenium and Beautiful Soup. Selenium is a Python library that automates web browser interaction. As a result, the data rendered by JavaScript links can be made available by Selenium, by automating button clicks and later on extracted using Beautiful Soup.
In all the implementations of the web scraping section for the three websites, Beautiful Soup was preferred as it is generally simpler, but for the ZenHotels.com website, Selenium was added to the equation.
Multiple challenges were encountered in this step because even with the use of Selenium, some websites like Hotels, Trivago and Agoda would not showcase the target classes we find in the "Inspect" section of the page, and on other times, even when the correct class is found, multiple NoneType errors arise, despite try/except segments to handle them. There also were different price currencies for different runs, which necessitated repetitive rerunning to get target results. This complicated the process of web scraping and aborted multiple time-consuming implementations.
By opting for the mentioned websites, this step was a success, and the necessary data was available and ready to be cleaned.

2- Data Processing

The information that we aimed to collect consisted of the *name* of the hotel, the *distance from the city center*, the *price for the entire stay*, the *average price for a night*, the *review score*, the *hotel rating*, the *summary*, the *address*, and multiple *pictures of the accommodation*. After getting the raw data of these target elements and storing them in arrays, it was time for this data to get cleaned to conform with the .csv files we want to store them in.

To make this data usable, we had to get rid of the extra elements such as currencies when it comes to prices, getting the average price per night, conversions of some distances from meters to Kms, then adding them to global variables that can be stored in the .csv file. There were some other handlings of scraped data inside of the scraping script like opting for only the number of reviews from values like "400 reviews", or the distance from values like "3 Kms from city center". After all of this, the data were grouped into a Hotels.csv file, being the global file containing 375 rows and 10 columns. Later on, during the visualization step, more handling was involved to drop duplicate rows for a specific plot.

This step is crucial because it will ease out the visualization step and getting insights from the graphs and the user interaction part as well since there has to be comparing and respecting user preferences selected.

3- Data Analysis

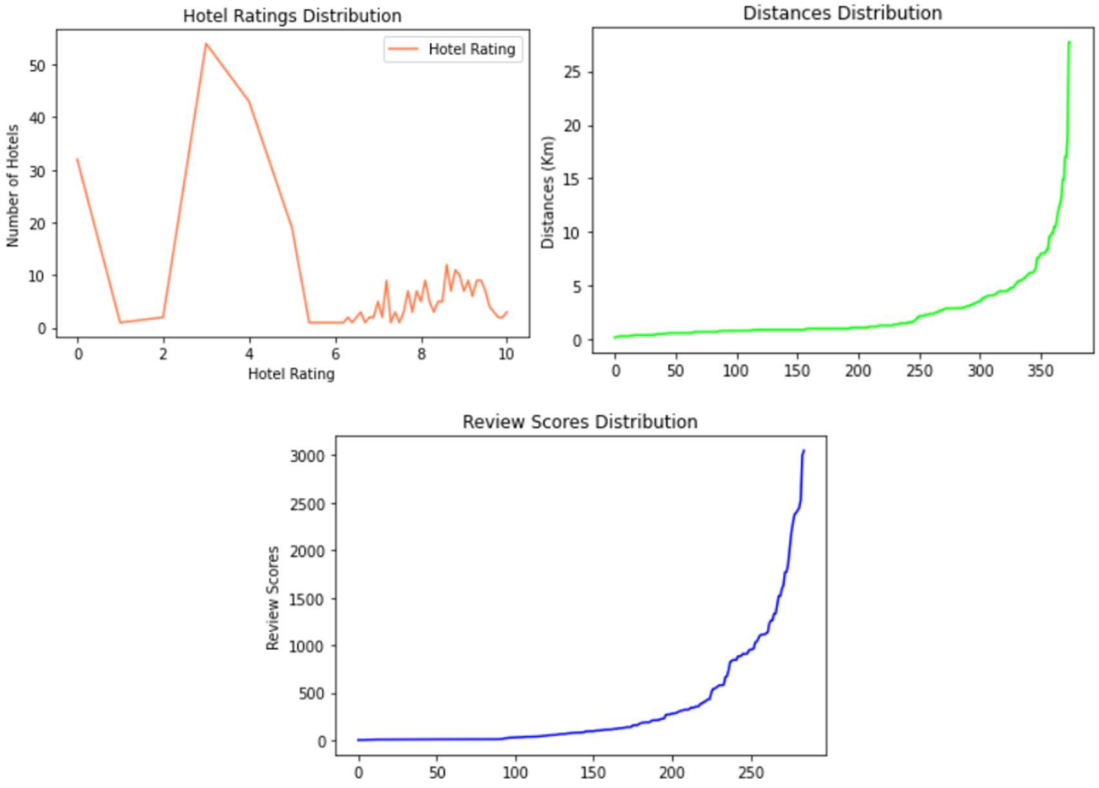
In the visualization part of this project, we were given to choose relevant parameters to plot and get useful insights from them and a general idea of the accommodation in Marrakech, Morocco.

The plots below show respectively the Distributions of the Hotel Ratings, the distances from the city center, and the review scores. The general insight that can be sought from them could be that the accommodations in Marrakech are scattered at different ranges from the city center and this could be explained by the multiple canon spots around it, and the spaces needed for the relatively bigger resorts, hotels, and Spas.

The review scores and hotel ratings are equally scattered:

From the Hotel Ratings Distribution plot, we can see that the accommodations are mid-ranged. This shows the variety of the data from these websites, getting new offers that have 0 ratings, offers that are very popular, to accommodations that are building their client satisfaction, which is the most prominent in our data.

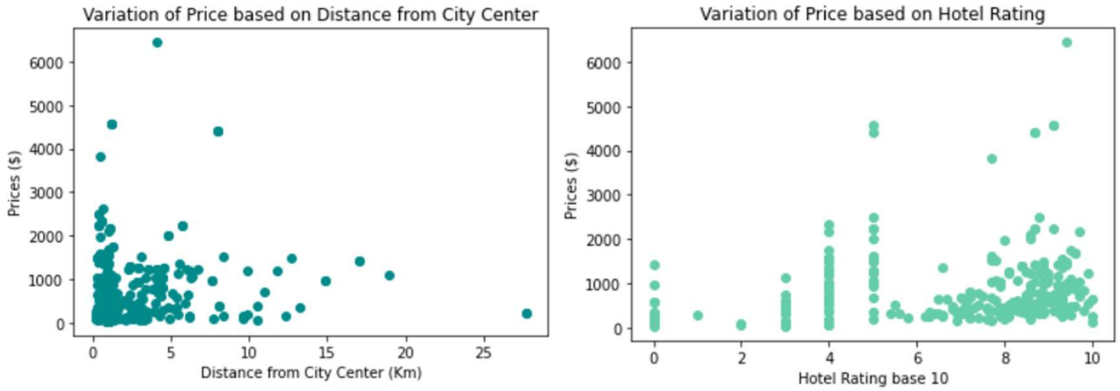
Meanwhile, the Review Scores Distribution graph, shows that a major number of the housing offers collected are ones with a responsive client review base, that interacted with the websites enough to leave multiple reviews per housing offer.



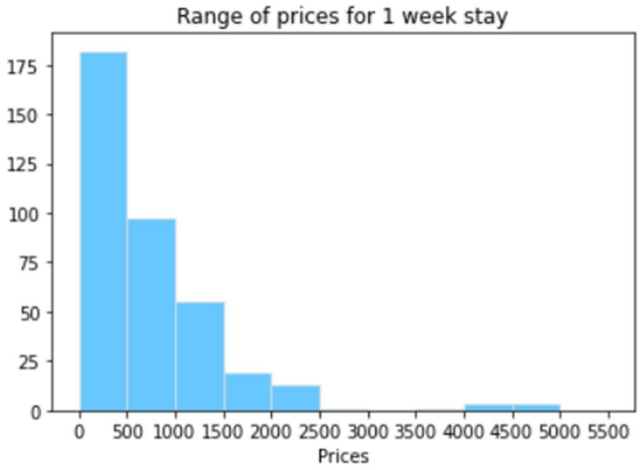
The next graphs show respectively the variation of price based on the distance from the city center and the variation of the price based on hotel reviews.

The first one shows that the prices are at a maximum of 1500 dollars for a 1-week stay in the city, with a maximum distance of 20 Km from the city center. Although, most of the options in that price range are mostly available from 1 to 6 Km.

The second scatter plot illustrates the importance of the hotel rating in the variation of the price for the stay. Most of the housing offers available are well rated and are still within an affordable price range per week.



The last histogram shows how the aforementioned price ranges for a 1-week stay are scattered in conformity with what was shown in the previous graphs, and how the price ranges up until 1500 dollars, and few housings of luxurious offers that go up until 4000-5000 dollars.



4- Limitations

Since the web scraping part was cumbersomely tedious and a challenge on its own, the implementation ended up having a couple of obstructions. The first limitation is that running the entire final implementation is time-consuming because, to get around failures in getting correct information from the websites, multiple for-loops were used which is not optimal. The entire Mini_Project1.ipynb Google Colaboratory notebook takes, from scraping to user interaction, about 30 minutes. Another limitation is in the user interaction part, which has no input validation. This means that on times, the notebook will exit the execution when given a number that is not expected (e.g., for criteria, Hotel name). Not to forget that the user interaction output is open to improvements and an overall better interface. The grasp of Selenium and wider usage of it is also up to improvement to be able to compare it with Beautiful Soup time and performance-wise. This, of course, could be improved given more time before the deadline, by introducing functions, limiting the use of for-loops, having a solid input validation, and involving a GUI or a static user-friendly website.

Conclusion

Through this project, we have been able to implement a system that collects data from multiple websites by static and dynamic web scraping, visualize, get relevant insights on the overall parameters collected, and built a user interaction system to provide a simple usage for the approach. The learning scope of the problem took us through multiple concepts and new practices that could become handy as data scientists, since the implementation of every intelligent system starts with data handling, from collecting to processing, cleaning, and analyzing. Although, multiple limitations are still open for improvement that could take this system from a basic functioning one to a more robust version.