Mini Project 1 Report

Maša Ćirković

I. Introduction

With the increasing number of fast food stores, and decreasing amount of time spent on oneself, many individuals struggle with making healthy dietary decisions, particularly when balancing nutritional needs, calorie control, and dietary restrictions such as gluten intolerance. With the increasing amount of health-conscious behaviors and dietary tracking tools, such as calorie-counting applications, there is a growing demand for efficient and personalized solutions that simplify meal planning. The question *What should I cook tonight?* can become complex for those with specific health goals, such as weight management, and dietary limitations like allergy, intolerance, etc.

Moreover, finding recipes that meet strict dietary requirements while still providing you with all the nutrients and keeping you full can be time-consuming. This is supported by the fact that many existing solutions provide static or limited filtering capabilities, failing to be helpful to users who need flexible, personalized criteria.

This report addresses this challenge by exploring how web scraping techniques can be applied to dynamically collect and filter recipes based on specific user-defined constraints, such as caloric content, Weight Watchers points, etc. The aim is to develop a tool that simplifies recipe search and selection, ultimately supporting healthier eating habits and saving time.

II. Data Collection

Data collection and storing is one of the most important parts of web scraping, with the former being one of the most challenging ones.

A. Approach

Web scraping is a method used to obtain information from web pages, and possibly simulate user interaction. There are many different libraries which offer web scraping, and for this project, Selenium was chosen. The particular website used for scrapping is Skinny Taste. The task was to scrape the first 50 pages with recipes, and it was decided to include *only recipes*, meaning that meal plans and other posts were not included. Information about the recipes consists of:

- Title
- Image url
- Recipe keys (tags)
- Personal Points (Weight Watcher Points)
- Calories
- Summary

After inspecting the web pages, common tags or classes are found and they are used in order to scrape the relevant information from them. All of the recipes are opened in a separate window, since some information about them is only available after clicking on the link. Driver is used to move to the next page after all of the recipes from the current page have been scraped.

B. Challenges

1) Website structure - web scraping is extremely subjective to the web page that is being scraped, so any modifications to the page structure or class names can lead to errors or incorrect data being scraped. For this project, the great majority of pages followed the same structure, but there were some which didn't, and so for them the data was not scraped. This can be solved with introducing additional complexities and checks in the code, but in turn that makes the code less readable and less clean, thus for this project it was decided not to do that and to simply filter out the values which weren't scraped.
2) Speed - websites which use JavaScript to dynamically load the content usually take some time to load it, and even though that time can be in the range of milliseconds, multiplied by the number of pages and number of recipes (in this case), it still amounts to a significant time. Scraping for this project took around two hours, and in the end there were 939 recipes.
3) Legal concerns - there are some websites which are illegal to scrape and which implement blocking mechanisms. Even though some can be bypassed, the ethical question still remains.

C. Storing the data

Given the fact that the scraping process takes a long time, after it has been done the data is stored in a pandas dataframe and then safely stored in a CSV file. This approach enables continuous use of the scraped data without needing to scrape again. If it is needed to update the file with new scraped data, the website can be scraped until the recipe which is the same as the first entry in the CSV file is found, and then new recipes can be inserted at the beginning of the dataframe and the CSV file.

III. Data Analysis

Most people don't understand really pure code and numbers, but everyone understands visualizations. Visualizing the data, trends, patterns, and correlations all contributes to the wider audience being able to understand what is being talked about and why. Plots like histograms, bar charts, pie charts, scatter plots, etc. are widely used in order to represent meaningful information extracted from the data. Based on these results, actions can be taken.

In this project, three different variables are explored: calories, personal points, and recipe keys. In the table I we can see the statistics on the numerical columns, which are personal points and calories.

Statistic	Personal Points	Calories
Count	933.000000	935.000000
Mean	4.961415	237.369412
Std Dev	2.794962	116.442076
Min	0.000000	8.000000
25%	3.000000	147.000000
50%	5.000000	225.000000
75%	7.000000	308.000000
Max	13.000000	608.000000

Summary statistics of personal points and calories

Visualizations for the distribution of the calories variable are given in the images 1 and 2. It can be seen that the majority of recipes have a caloric content in between 160 and 310, which is understandable given the fact this is a website with healthy recipes.

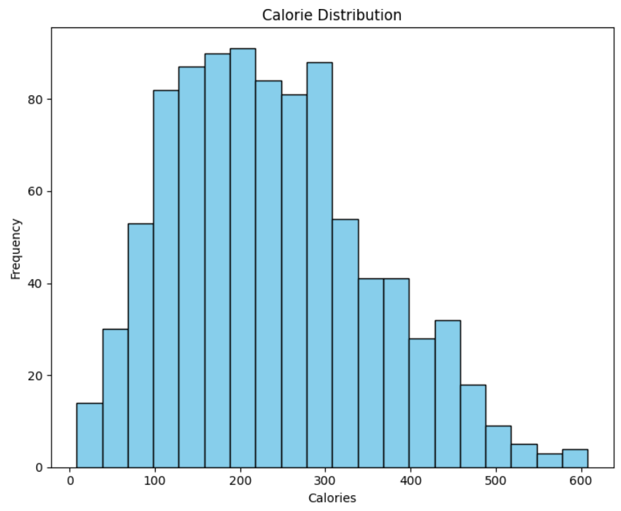

Fig. 1: *Calories distribution - histogram*

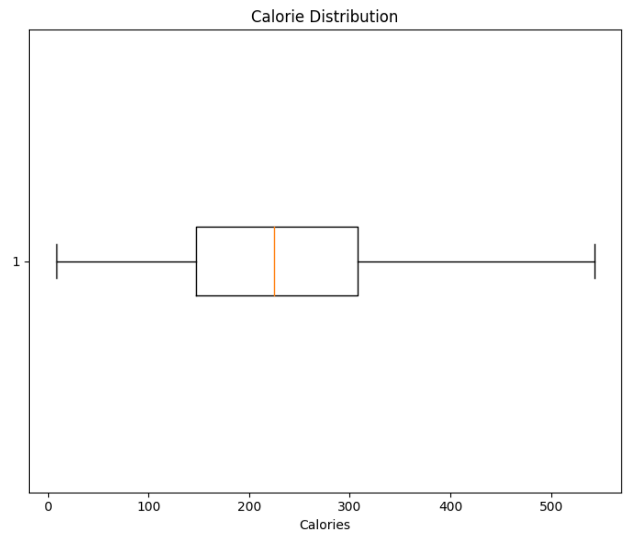

Fig. 2: *Calories distribution - box plot*

Visualization for the distribution of the personal points variable is given in the image 3. It can be seen that the majority of recipes have personal points in the range from two to seven. Smaller number for personal points is better, it means more fiber and protein, and less added sugars and saturated fats.

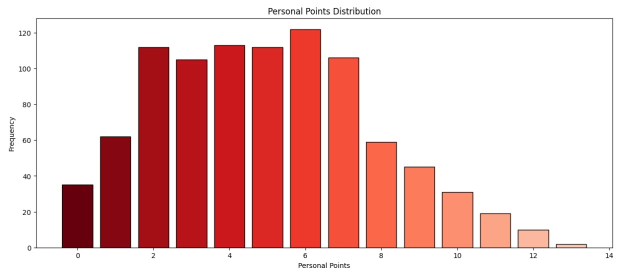

Fig. 3: *Personal points distribution - bar plot*

Visualizations for the distribution of the recipe keys variable are given in the images 4 and 5. It can be observed that the majority of recipes are gluten free, kid friendly and under 30 minutes. This might suggest a growing number of people with gluten intolerance (can be further explored) and also that majority of people prefer something which is quick to make.

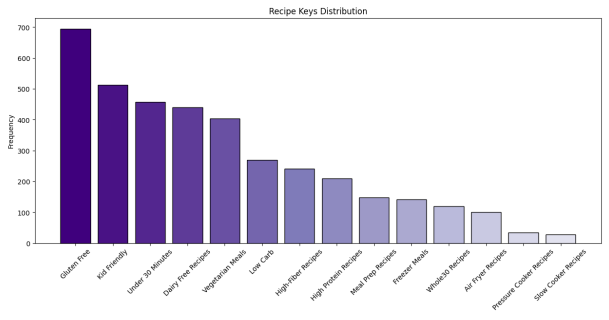

Fig. 4: *Recipe keys distribution - bar plot*

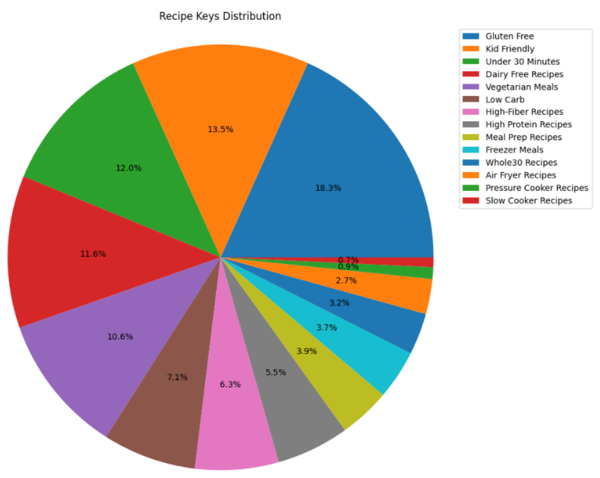

Fig. 5: *Recipe keys distribution - pie chart*

Visualization for the average calories per recipe key is given in the image 6. This was an interesting observation, as it was slightly unexpected that high protein and high fiber recipes would have the highest average calories.

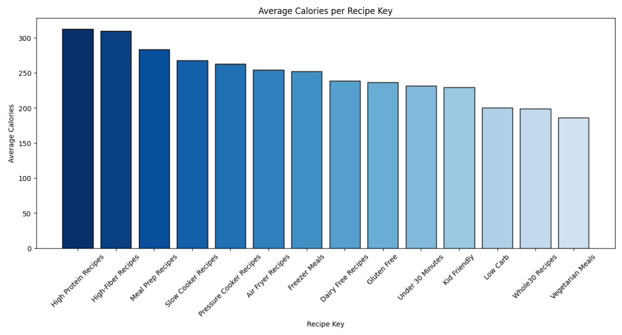

Fig. 6: *Average calories per recipe key - bar chart*

Lastly, correlation is explored. Correlation refers to the relationship between two variables. Correlation is performed on numerical data, so here it is explored only for personal points and calories. Pearson and Spearman correlation methods were tested and the results are given in table II. Correlation matrix is given in the image 7. Conclusion here is that we have a moderately positive and linear correlation between the two, which makes sense because higher amount of calories usually leads to higher amounts of added sugars and saturated fats, which leads to higher personal points.

Method	Result
Pearson	0.6470718864201456
Spearman	0.64182077725037492

Correlation between personal points and calories

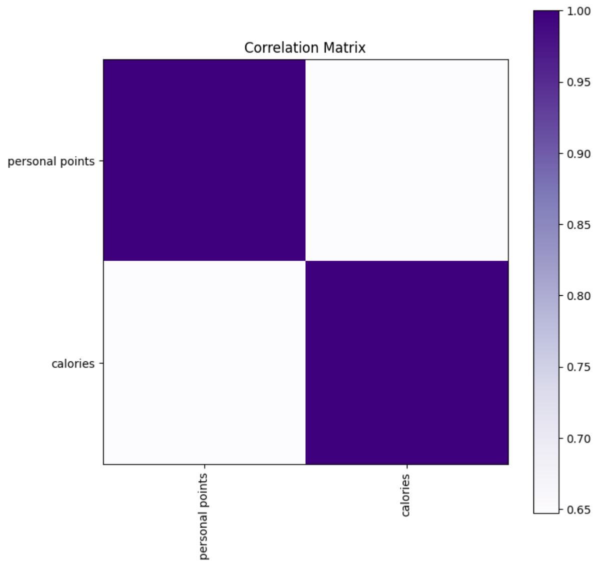

Fig. 7: *Correlation matrix*

IV. Conclusion

Although web scraping is very useful and bring benefits such as flexibility and filtering, it is very susceptible to changes in the structure of website being scraped, and thus requires constant modifications. One problem with Selenium is the speed, it takes a while to scrape because of the additional window opening. Many different exceptions occur because the page is dynamically loaded, with the most common one being stale element exception. This was solved by opening individual pages of recipes in separate tabs. Additionally, it happens that the program just loops infinitely in the part of the code which is unrelated to the scraping itself, and so the scraping was divided into two parts with 25 pages each, which worked well. Possible cause could be some time out exception, or driver needs refreshing, so the conclusion is that it is better to divide into smaller workloads. Web scraping has many benefits, but many challenges as well.