Web Scraping Using Python

C. D. V. P. Basnayake
S15025

Question 1

1. What are the possible ways of collecting alternative data?

    - Web scraping
    - Mobile device data
    - Public records
    - Experiments
    - Credit/Debit card data
    - Surveys
    - Satellite imagery

2. What are the advantages of picking web-scraping to gather alternative data compared to other methods?

    - It is an automated process
    - Cost effective
    - Updated data
    - Can collect large data amount in less time

3. What are the limitations/challenges you find in web-scraping?

    - Captcha in some sites blocks automated scrapers, so it makes scraping difficult
    - Some websites prohibit scraping in their terms and regulations
    - Technical challenges may occur while scraping, such as data removal from web site, dynamic web sites etc.
    - Difficult to use for the non-professional persons
    - Data quality vary with the sites used for scraping

4. Mention three major python libraries that are being used in web scraping and explain the use of them.

    - Beautiful Soup: This is a library for parsing HTML and XML documents. Then extract data from them.
    - Selenium: Use in automated web scraping processes , using this library we can run an automated web browser like chrome, firefox etc. Can also do things like clicking buttons.
    - Scrapy: This has the ability to work with data like XML and JSON. Can be used in large projects.

Question 2

a) Write a python script to get the required output into an excel file. Python code should be clean and well-structured with meaningful variable names etc.

```
from selenium import webdriver

import webbrowser

from bs4 import BeautifulSoup as soup


#1st page link

url="https://www.walmart.com/browse/cell-phones/apple-iphone/1105910_7551331_1127173?povid=GlobalNav_rWeb_Electronics_CellPhones_iPhone&affinityOverride=default&page=1"

edge_path = 'C:/Program Files (x86)/Microsoft/Edge/Application/msedge.exe %s'


#Start automated browser

automated_browser=webdriver.Edge(executable_path=r"G:\VI SEM\IS 3005 - Statistics in Practice I\Guest\msedgedriver.exe")

webbrowser.get(edge_path).open(url)


#test for the 1st page

#html=automated_browser.page_source

#print(html)

#phone=soup(html,'lxml')

#phone
```

```
#Genarate Url
url_list=[ ]
for i in range(1,26):
    url_list.append("https://www.walmart.com/browse/cell-phones/apple-iphone/1105910_7551331_1127173?povid=GlobalNav_rWeb_Electronics_CellPhones_iPhone&affinityOverride=default&page=" + str(i))
url_list
#create lists
item_names=[]
price_list=[]
item_shipping=[]

#obtain data
for url in url_list:

    result=automated_browser.get(url)
    html=automated_browser.page_source
    phone=soup(html,'lxml')

    product_name=phone.findAll('span',{'class':"w_V_DM"})
    product_shipping=phone.findAll('div',{'class':"mt2 mb2"})
    product_price=phone.findAll('div',{'data-automation-id':"product-price"})
    for names,shipping,price in zip(product_name,product_shipping,product_price):
        item_names.append(names.span.text.strip())
        item_shipping.append(shipping.span.text.strip())
        price_list.append(price.div.text.strip())

#create an excel file
import pandas as pd
df=pd.DataFrame({'product_Name':item_names,'Price':price_list,'Pickup/Checking':item_shipping})
filepath=r"G:\VI SEM\IS 3005 - Statistics in Practice I\Guest\walmart.xlsx"
df.to_excel(filepath)
```

b) Other data extraction methods.
   - Application Programming Interfaces
   - Machine learning

   These two methods can be also used in web scraping. But its easy to use python and its packages for it. Python and its packages being open source is another advantage.