Extrakcia informácii z webu pre analýzu dát pomocou Python*

Štefan Kučerák
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xkucerak@stuba.sk

16. december 2023

Abstrakt

Problém nastáva pri získavaní veľkého množstva dát z prostredia otvoreného webu. Vykonávanie tejto činnosti ľuďmi by bolo veľmi časovo náročné. Riešením tohto problému je proces, ktorý je v angličtine označovaný ako "Web scraping". Je to spôsob extrakcie dát z web stránky. Umožňuje získať presné dáta v krátkom čase za zlomok ceny oproti ľudskej sile. Dáta získané v tomto procese sú pripravené na ďalšie využitie. V tomto článku popisujeme knižnice, ktoré sa dajú použiť na získanie dát. Taktiež obsahuje aj jednoduchý príklad na prácu s jednou z knižníc.

Kľúčové slová - Web scrapping, Python, Web stránka, Extrakcia dát, Beautiful Soup

1 Úvod

Získavanie dát je zložitý proces obsahujúci viacej krokov začínajúc výberom aké dáta vlastne chceme zbierať, ich usporiadaním, odstraňovaním nesprávnych dát, opätovnou analýzou použitím algoritmov [12]. Najlepším miestom z ktorého sa dajú dáta získať je internet. Na internete sa nachádza veľké množstvo dát, ktoré sú predovšetkým zadarmo. Ak je náš projekt rozsiahli jednotlivé dáta by sa len s obtiažou dali získať klasickým kopírovaním a vkladaním do dokumentu. Pri použití ľudskej sily vzniká priestor na chybu z nepozornosti [1]. Ďalším problémom, ktorý môže nastať je samostatná čitateľnosťou textu, kvôli zlému dizajnu stránky alebo stránka môže obsahovať opatrenia proti priamemu kopírovaniu textu. Riešením tohto problému je web scraping, ktorý umožňuje automatizovať tento zber informácií a spraviť ho precíznejšie a rýchlejšie. Princíp web scrapingu je bližšie popísaný v časti 2. Tento proces je využívaný jednotlivcami ale i spoločnosťami, ktoré chcú využiť voľne dostupné informácie z webu [1]. Príklady jeho využitia sú napríklad: sledovanie ceny výrobku, návštevnosti stránky, čítanie komentárov na stránke, zistenie hypertextových odkazov na danej stránke a

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2023/24, vedenie: Ing. Mohammad Yusuf Momand, MSc.

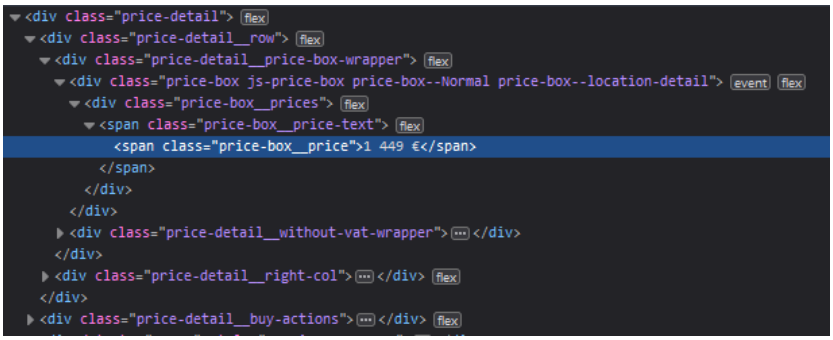
mnoho ďalších [4]. Jednotlivé príklady a knižnice, ktoré boli využité na extrakciu dát sú v časti 3. Keď už vieme aké dáta a odkiaľ ich chceme zbierať nastáva výber programovacieho jazyka v ktorom chceme napísať algoritmus na tento zber. Existuje mnoho programovacích jazykov ako napríklad: Python, R, PHP, Java [5], pre ktoré sú napísané knižnice na extrakciu dát. Pre tento článok sme sa rozdali vybrať programovací jazyk Python, pretože obsahuje rôzne dátové štruktúry, štandardné knižnice s implementáciu analýzy sentimentu a je používaný v dátovej vede [6]. Preto je Python vhodný programovací jazyk na prácu s dátami a dá sa využiť aj na ich extrakciu. Pre Python existuje mnoho knižníc špeciálne na extrakciu dát z webu, ktoré sú uvedené v časti 4. Ďalej článok obsahuje implementáciu knižnice Beautiful Soup v časti 5. Pri tejto implementácií je tiež zaujímavé porovnanie rýchlosti medzi strojom a človekom.

2 Princíp web scraping

V tejto časti bližšie priblížime princíp extrakcie dát z web stránky.

Obr. 1: Princíp extrakcie

Na začiatku procesu musíme poznať cieľovú stránku z ktorej chceme dáta získať. Následne potrebujeme získať HTML kód stránky (napríklad pomocou knižnice Requests 4). HTML kód využíva tagy ako sú napríklad: h1, div, span. Súčasťou týchto tagov je aj parameter trieda, ktorý odkazuje väčšinou na grafické úpravy tohoto tagu. Pre lepšiu predstavu sú tieto tagy zobrazené na Obr. 2 zo stránky[1]. Ak by sme v tomto prípade chceli zistiť cenu výrobku, hľadali by sme podľa tagu < span class=”price-box__price”> . Na vyhľadanie tagu sa použije niektorá z knižníc na extrakciu tagu. Tieto knižnice vyhľadávajú v HTML podľa tagov [11]. Výstupom knižnice je nami požadovaný tag s ktorým môžeme ďalej pracovať. Môžeme ho uložiť na neskoršiu analýzu. Výstup z ukážkového HTML by bol ”1 449 € ”.



Obr. 2: Ukážka HTML tagu spolu s triedou

[1] https://www.alza.sk/iphone-15-pro-max?dq=7927773

3 Príklady reálneho využitia

V tejto časti by sme chceli priblížiť niektoré z projektov, ktoré extrahujú dáta z web stránok pomocou Python.

Jedným z príkladov je článok "Phishing Web Page Detection using web Scraping" [3], v ktorom sa zaoberajú automatizovaným odhaľovaním phishing stránok. Ich riešenie na odhaľovanie takýchto stánok má presnosť detekcie viac ako 98%. Pri tomto projekte bol požitý Python a knižnica Beautiful Soup(časť 4).

Ďalším z príkladov je článok "Food Genie, Recipe Search Algorithm using Web Scraping" [9], ktorý využíva web scraping na porovnanie indegriencíi, ktoré majú ísť do rovnakého receptu. Rieši problém toho že naprieč rôznymi stránkami má recept iný postup alebo sú použité iné suroviny. Napríklad pri tomto projekte bol využitý Python a knižnica scrapy(časť 4).

Ako ďalší je článok "Web Scraping Methods on Odoo Framework to Collect Rupiah Exchange Rate From B ank Indonesia Website" [10], ktorý sledoval kurz cudzej meny voči Indonézskej rupie. Pri tomto projekte sa využíva Python a knižnica Beautiful Soup(časť 4).

Ako posledný príklad by sme uviedli článok "NEWSONE- AN AGGREGATION SYSTEM FOR NEWS USING WEB SCRAPING METHOD" [8], ktorý požíva web scraping na extrakciu správ z rôznych web stránok v rôznych jazykoch a sústreďuje ich na platformu "NewsOne". Pri tomto projekte sa taktiež využíva Python a knižnica Beautiful Soup(časť 4).

Z príkladov je vidieť že Web scraping je využívaný oblastiach ako ekonomika, novinárstvo ale i v gastronómii.

4 Knižnice pre získavanie dát

Zoznam knižníc, ktoré môžu byť použité na extrakciu dát z webu pre programovací jazyk Python:

- Requests[2] je jednoduchá knižnica ktorá umožňuje vytvoriť http alebo https požiadavky pomocou metódy POST a GET. Táto knižnica je najčastejšie používaná na získanie samotného HTML kódu zo stánky. Ak by sme chceli použiť len samotnú knižnicu museli by sme si na vyhľadávanie napísať vlastný algoritmus. To by bolo zbytočné, pretože existuje niekoľko knižníc, ktoré už obsahujú túto funkcionalitu a sú uvedené nižšie.

- Beautiful Soup[3] táto knižnica obsahuje nástroje na vyhľadávanie, upravovanie a iteráciu v HTML alebo XML kóde. Podmienkou pre fungovanie je vstup vo formáte HTML alebo XML a pre ten sa najčastejšie využíva vyššie spomenutá knižnica Requests.

- Scrapy[4] je rýchly framework pre prehľadávanie webu na vysokej úrovni, ktorý sa používa na prehľadávanie webových stránok a extrahovanie štruktúrovaných údajov. Dá sa použiť na širokú škálu účelov, od získavania dát až po monitorovanie a automatizované testovanie. Výhodou oproti ostatným knižniciam je že obsahuje nástroj na extrakciou HTML kódu zo stránky.

[2] `https://pypi.org/project/requests/`
[3] `https://pypi.org/project/beautifulsoup4/`
[4] `https://pypi.org/project/Scrapy/`

- Selenium[5] je knižnica, ktorá umožňuje priamo pomocou prehliadača interagovať so stránkami. Podporované prehliadače: Firefox, Chrome, Internet Explorer. V základe je to script, ktorý sa vykonáva v internetovom prehliadači. Keďže sa jedná priamo o prehliadač dokáže uchovávať súbory cookies a taktiež podporuje JavaScript.
- MechanicalSoup[6] je knižnica na automatizáciu interakcie s webovými stránkami. MechanicalSoup automaticky ukladá a odosiela súbory cookie, sleduje presmerovania a môže sledovať odkazy a odosielať formuláre. Avšak nepodporuje JavaScript.

Toto boli najpoužívanejšie knižnice na extrakciu dát z web stránok. Všetky uvedené knižnice sú zadarmo a Open-source.

5 Jednoduchý príklad požitím knižnice Beautiful Soup

5.1 Kód

```
1  import requests
2  from bs4 import BeautifulSoup
3
4  page = requests.get("https://www.google.com/")
5  soup = BeautifulSoup(page.content, "html.parser")
6
7  print(soup.title.string)
8
```

Obr. 3: Program (výstup: Google)

Na Obr. 3 je príklad kódu, ktorý zo stránky `https://www.google.com/` extrahuje názov stánky a teda tag <title> . Ako prvé sa spraví požiadavka pomocou knižnice Requests, ktorá vráti kód web stránky vo formáte HTML. Na HTML sa použije knižnica Beautiful Soup, ktorá vie pracovať s jednotlivými tagmi v HTML kóde. Knižnici dáme požiadavkou na vypísanie názvu stránky v reťazci. Následne dostaneme výstup "Google". Tento princíp je uvedený v Obr. 1. Podobný postup sa dá využiť aj napríklad na sledovanie ceny produktu a to tak že nájdeme html tag, ktorý obsahuje symbol meny a teda aj číselnú hodnotu.

5.2 Časová náročnosť

V tejto časti si priblížime množstvo času, ktorý je potrebný na splnenie rovnakej úlohy pomocou knižnice Beautiful Soup a človeka. Z tabuľky 1 vyplýva že spracovanie tejto požiadavky teda vypísanie názvu stránky je v priemere o rýchlejšie asi 100-krát.

[5] `https://pypi.org/project/selenium/`
[6] `https://pypi.org/project/MechanicalSoup/`

Tabuľka 1: Porovnanie času pri výpise knižnicou a človekom

Vstup	Výstup	Čas (ms)	
		Knižnica	Človek
google.com	Google	31	933
instagram.com	Instagram	5	1433
youtube.com	YouTube	215	1433
wikipedia.org	Wikipedia	30	3133
www.stuba.sk	Slovenská technická univerzita v Bratislave (STU)	50	16100
www.fiit.stuba.sk	Fakulta informatiky a informacnych technologii v Bratislave	43	14766

Pri tejto ukážke sme zaznamenali len čas extrakcie tagu <title> .

6 Záver

Výsledkom našej práce je článok ktorý približuje tému extrakcie informácii z webu pomocou programovacieho jazyka Python. Vysvetľuje základný princíp web scrapingu a taktiež ukazuje na aktuálnosť tejto témy. Boli vymenované najznámejšie knižnice, ktoré umožňujú efektívne prehliadanie HTML web stránok. Ďalej bol uvedený jednoduchý príklad použitím knižnice Beautiful Soup, ktorý ukazuje rýchlosť a ľahkosť naučenia sa tejto knižnice. Verím že tento článok by mohol byť dobrým úvodom do problematiky extrakcie dát z webu aj s príkladmi. Článok je taktiež semestrálna práca na predmet Metódy inžinierskej prace.

7 Reakcia na témy z prednášok

Spoločenské súvislosti. V aktuálnej dobe pri rýchlom rozvoji tento proces tvorí základ získavania dát pre učenie umelej inteligencie. Najznámejšia umelá inteligencia ChatGPT[7] využíva tento proces na extrakciu človekom napísaným textov napríklad z online fór [7].

Technológia a ľudia. Technológia slúži na uľahčenie života ľudom. V súvislosti s našim článkom priamo i nepriamo uľahčuje ľudom život. Nepriamo napríklad vo forme umelej inteligencie. Pri tvorbe modelu pre umelú inteligenciu museli byť použité dáta z web stránok. A priamo tým že ľudia nemusia zložito prepisovať alebo kopírovať údaje zo stránky ale jednoduchým programom môžu získať rovnaké informácie za kratší čas. Asi jedinou zápornou stránkou je že nástupom tejto technológie určite zanikli niektoré pracovné pozície.

Udržateľnosť a etika. Udržateľnosť tejto technológie je veľmi dôležitá v dnešnej dobre pri rýchlom rozvoji umelej inteligencie, pretože potrebujeme získavať veľké množstvo dát v krátkom čase. Internet a celkovo web stránky predstavujú najrýchlejší spôsob získavania informácii. Každý deň ho využíva veľký počet pravidelných používateľov. Tento spôsob získavania informácii môže byt neetický,

[7] `https://chat.openai.com`

pretože stránka môže predávať svoje dáta za peniaze a tak obchádzame túto peňažnú bariéru. Týmto problémom sa zaoberajú mnohý ľudia a jedným z príkladov je článok [2], ktorý navrhuje rôzne spôsoby ako zabrániť extrakcii dát. Napríklad používaním Captcha alebo registrovaním sa. Niektoré stránky používajú obmedzenie počtu návštev v určitom čase.

Literatúra

[1] Ayat Abodayeh, Reem Hejazi, Ward Najjar, Leena Shihadeh, and Rabia Latif. Web scraping for data analytics: A beautifulsoup implementation. In *2023 Sixth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU)*, pages 65–69, 2023.

[2] Chandan Biswas, Rahul Mallick, Subrata Paul, and Dipta Mukherjee. Solution to web scraping. In *2023 11th International Conference on Internet of Everything, Microwave Engineering, Communication and Networks (IEMECON)*, pages 1–5, 2023.

[3] Mallika Boyapati and Ramazan Aygun. Phishing web page detection using web scraping. In *SoutheastCon 2023*, pages 167–174, 2023.

[4] Rahul Chauhan, Ayush Negi, and Mahesh Manchanda. An extensive review on web scraping technique using python. In *2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*, pages 1134–1138, 2023.

[5] Mihai Gheorghe, Florin Mihai, and Marian Dârdal. Modern techniques of web scraping for data scientists. 2019.

[6] Arun Kumar and Supriya.P. Panda. A survey: How python pitches in it-world. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 248–251, 2019.

[7] Natalie Staudacher. What is ChatGPT? — OpenAI Help Center — help.openai.com. `https://help.openai.com/en/articles/6783457-what-is-chatgpt`, 2023. [16-12-2023].

[8] K. Sundaramoorthy, R. Durga, and S. Nagadarshini. Newsone — an aggregation system for news using web scraping method. In *2017 International Conference on Technical Advancements in Computers and Communications (ICTACC)*, pages 136–140, 2017.

[9] Nikhil Suwalka, Nishit Shanbhag, Sakina Salmani, and Pooja Raundale. Food genie, recipe search algorithm using web scraping. In *2023 3rd Asian Conference on Innovation in Technology (ASIANCON)*, pages 1–6, 2023.

[10] Masmur Tarigan, Ford Lumban Gaol, and Tukoro Matsuo. Web scraping methods on odoo framework to collect rupiah exchange rate from bank indonesia website. In *2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI)*, pages 697–702, 2021.

[11] S Thivaharan., G Srivatsun., and S Sarathambekai. A survey on python libraries used for social media content scraping. In *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, pages 361–366, 2020.

[12] David Mathew Thomas and Sandeep Mathur. Data analysis by web scraping using python. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 450–454, 2019.