SLOVENSKÁ ŠTATISTIKA a DEMOGRAFIA

SLOVAK STATISTICS and DEMOGRAPHY

3/2023
ročník/volume 33

Recenzovaný vedecký časopis so zameraním na prezentáciu moderných štatistických a demografických metód a postupov.

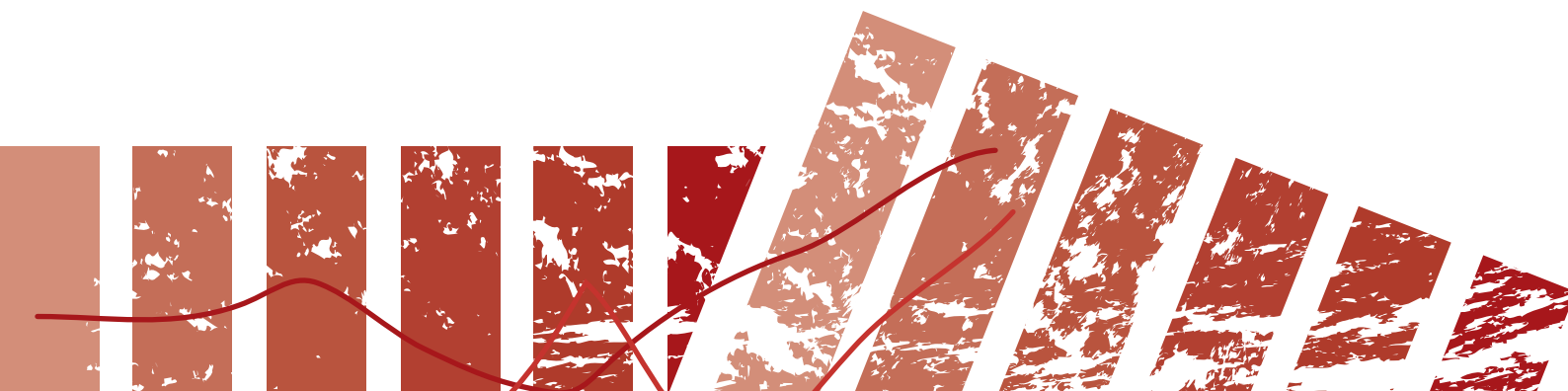
Scientific peer-reviewed journal focusing on the presentation of modern statistical and demographic methods and procedures.

Článok/Article: 5

Typ článku/Type of article: informatívny článok/informative article

Strany/Pages: 55 – 68

Dátum vydania/Publication date: 15. júl 2023/July 15, 2023



Informatívny článok/Informative article

Silvia KOMARA
Katedra štatistiky, Fakulta hospodárskej informatiky
Ekonomickej univerzity v Bratislave

Michal PÁLEŠ
Katedra matematiky a aktuárstva, Fakulta hospodárskej informatiky
Ekonomickej univerzity v Bratislave

VYUŽITIE JAZYKA PYTHON V OBLASTI WEB SCRAPINGU

THE USE OF THE PYTHON LANGUAGE IN WEB SCRAPING

ABSTRAKT
Príspevok sa zameriava na predstavenie základných atribútov web scrapingu v kontexte v súčasnosti tak skloňovaných pojmov, ako sú nové zdroje štatistiky veľké dáta, strojové učenie, umelá inteligencia, Business Intelligence a pod. Opisuje návrhy riešenia sťahovania údajov z internetu v jazyku Python a moduly, v ktorých možno tento proces realizovať. Špecificky sa venuje aj prepojeniu oblasti strojového učenia s web scrapingom. V praktickej ukážke predstavujeme funkcionalitu jazyka Python na získanie údajov z PDF dokumentov.

ABSTRACT
The paper focuses on presenting the basic attributes of web scraping in the context of currently used terms such as new sources of statistics, big data, machine learning, artificial intelligence, Business Intelligence, etc. It describes the Python language's options for downloading data from the Internet and modules in which this process can be executed. It is also specifically dedicated to connecting the field of machine learning with web scraping. In a practical demonstration, we present the functionality of the Python language for scraping data from the PDF documents.

KĽÚČOVÉ SLOVÁ
jazyk Python, web scraping, strojové učenie, PDF dokument

KEY WORDS
Python language, web scraping, machine learning, PDF dokument

1. ÚVOD
Dáta sú jednou z kľúčových hodnôt v dnešnej dobe a ich objem neustále rastie. Nové technológie prinášajú nové dáta, nové dáta generujú ďalšie dáta, vznikajú nové technológie a tento cyklus sa stále opakuje. Rozsiahla digitalizácia tento rast len urýchľuje. Ale s rastom objemu dát vzniká aj problém, a to zhoršenie ich kvality. Dáta sú cenné aktívum, ktoré je potenciálne schopné priniesť úžitok aj stratu. Kvalitné dáta s väčšou pravdepodobnosťou prinesú úžitok. Nekvalitné dáta v najlepšom prípade neprinesú nič. Aby teda dáta priniesli úžitok, je potrebné vedieť s nimi správne manipulovať a vedieť transformovať nekvalitné dáta na kvalitné. Rast objemu dát vo svete prispel k vytvoreniu konceptu Data Drive Decision Making, čo znamená rozhodovanie skôr na základe dát, faktov a metrík ako na základe intuície. Tento prístup umožňuje prijímať najlepšie rozhodnutie pre firmu. Na využitie daného

konceptu je však potrebné mať obrovské množstvo kvalitných dát. A pokiaľ firmy väčšinou nemajú problémy s objemom dát, tak s kvalitou je to čoraz zložitejšie. Získať znalosti z dát firmám pomáha proces, ktorý je známy ako Knowledge Discovery in Databases (KDD). KDD sa skladá z rôznych krokov,jedným je data mining (dolovanie dát) [12]. Jednou z oblastí data miningu je práve web scraping.

Automatizované zhromažďovanie údajov z internetu je takmer také staré ako samotný internet. Hoci web scraping nie je nový pojem, v minulých rokoch sa táto prax častejšie označovala ako screen scraping, data mining, web harvesting alebo podobne. Zdá sa, že všeobecný konsenzus dnes uprednostňuje web scraping [7]. Web scraping je technologické riešenie, ktoré extrahuje údaje z webových stránok rýchlym, efektívnym a automatizovaným spôsobom a ponúka údaje vo formáte, ktorý je štruktúrovaný a ľahko použiteľný. [3] Teoreticky je web scraping zhromažďovanie údajov akýmkoľvek iným spôsobom ako prostredníctvom priamej komunikácie s rozhraním webového servera (alebo samozrejme prostredníctvom človeka používajúceho webový prehliadač). Najčastejšie sa to dosahuje napísaním automatizovaného programu, ktorý webovému serveru odosiela žiadosti o určité údaje (zvyčajne vo forme HTML a iných súborov, z ktorých sa skladajú webové stránky), následne tieto údaje analyzuje s cieľom získať potrebné informácie a dáta ukladá v určitom formáte napríklad do dátového skladu. V praxi web scraping zahŕňa širokú škálu programovacích techník a technológií, ako je analýza údajov, rozbor prirodzeného jazyka a zabezpečenie informácií [7].

Štúdia v [6] definuje web scraping ako prvý krok v procese dolovania dát. Samotné dolovanie dát sa považuje za súčasť Business Intelligence (BI), prvýkrát rozpracované Howardom Dresnerom z Gartner Group v roku 1989. Podľa jeho názoru je Business Intelligence súbor konceptov a metód na zlepšenie procesu rozhodovania v manažmente pomocou informačných systémov, využívajúcich obchodné dáta. Podľa Lifecycle Software Ltd. existujú dva prvky, ktoré odlišujú BI systémy od iných, a to integrácia dát, teda zlučovanie údajov z rôznych zdrojov a v rôznych formátoch, a poskytovanie koherentného prístupu k nim: poskytovanie techník na analýzu a vizualizáciu informácií novým spôsobom, zrozumiteľným pre používateľov [11].

Internet obsahuje nespočetné množstvo informácií rôzneho druhu. Či už ide o informácie o počasí, marketingové dáta, rešerše a kvalitatívne dáta, dáta týkajúce sa sociálnych sietí a mnoho ďalších, ich analýza môže napomôcť k hlbšiemu pochopeniu konkrétnej problematiky. Tieto informácie však málokedy majú formu, ktorú je možné využiť na analýzu dát. Každá webová stránka má svoju štruktúru inú, avšak spája ich značkovací jazyk HTML. Špecifická forma tohto jazyka umožňuje pomocou nástrojov vyhľadať konkrétne informácie a uložiť ich do čitateľnejšieho formátu. Spôsob, akým nástroj vykonáva tento úkon, je veľmi podobný spôsobu, aký by použil bežný používateľ. Na internetovej stránke sa vyberú potrebné dáta, tie sa následne skopírujú a vložia do tabuľky. Takýto proces je možný v prípade malého počtu dát, avšak ak sa jedná o počet presahujúci tisíce jednotiek, je tento proces časovo veľmi náročný. V tomto prípade je možná náhrada používateľa za robota, ktorý opakovane vykonáva ten istý úkon [4].

Klasickým príkladom je napríklad knižnica Selenium v programovacom jazyku Python alebo Puppeteer v programovacom jazyku Javascript. Obe tieto metódy

fungujú na princípe skrytého prehliadača. Robot je nastavený na určitú webovú stránku a z tej potom vyberie dôležité dáta. Ak nie je potrebná žiadna interakcia s webovou stránkou, metódy so skrytým prehliadačom sú zbytočne komplikované. Dáta je možné získať priamo z konkrétnej webovej stránky pomocou príkazu „request“. Manipulácia s dátami je následne uľahčená použitím knižnice Beautiful Soup programovacieho jazyka Python [10].

Programovať web scraping od samého začiatku je však často náročná cesta. Dnes už existuje niekoľko webových aplikácií a rozšírení, ktoré scraping dát zvládnu bez programovania. Konkrétnym príkladom je portál import.io [9].

Web scraping môže byť využitý na zhromaždenie dátového setu obsahujúceho informácie o online cenách na vytvorenie denného prehľadu cien. Web scrapingom možno preskúmať a vyhodnotiť cenové praktiky využívané spoločnosťami pôsobiacimi v elektronickom obchode, resp. na realitnom trhu. Banky a iné finančné inštitúcie používajú web scraping na analýzu svojej konkurencie. Banky frekventovane sťahujú dáta konkurentov, napríklad o tom, kde sa novo otvorili či zavreli pobočky alebo tiež napríklad na sledovanie aktuálnej úrokovej sadzby pri pôžičkách. Tieto informácie potom zakomponujú do svojich interných modelov a predpokladov. Niektoré spoločnosti sa tiež špecializujú na predaj pracovných profilov pomocou zberu a analýzy verejne dostupných dát napríklad z LinkedIn [13]. V oblasti aktuárstva to môže byť napríklad prehľadávanie cenových kalkulačiek v rámci havarijného poistenia.

2. PROCES WEB SCRAPING

Proces sťahovania dát z internetu je možné prvotne rozdeliť do dvoch po sebe idúcich krokov (pozri obrázok č. 1):

1. **získanie webových zdrojov (webové stránky zdrojov),**
2. **extrahovanie požadovaných informácií zo získaných zdrojov.**

Konkrétne sa proces web scrapingu začína odosielaním požiadavky HTTP na získanie zdrojov cielenej webovej stránky. Táto požiadavka môže byť naformátovaná buď ako adresa URL obsahujúca dotaz GET, alebo ako časť správy HTTP obsahujúca dotaz POST.

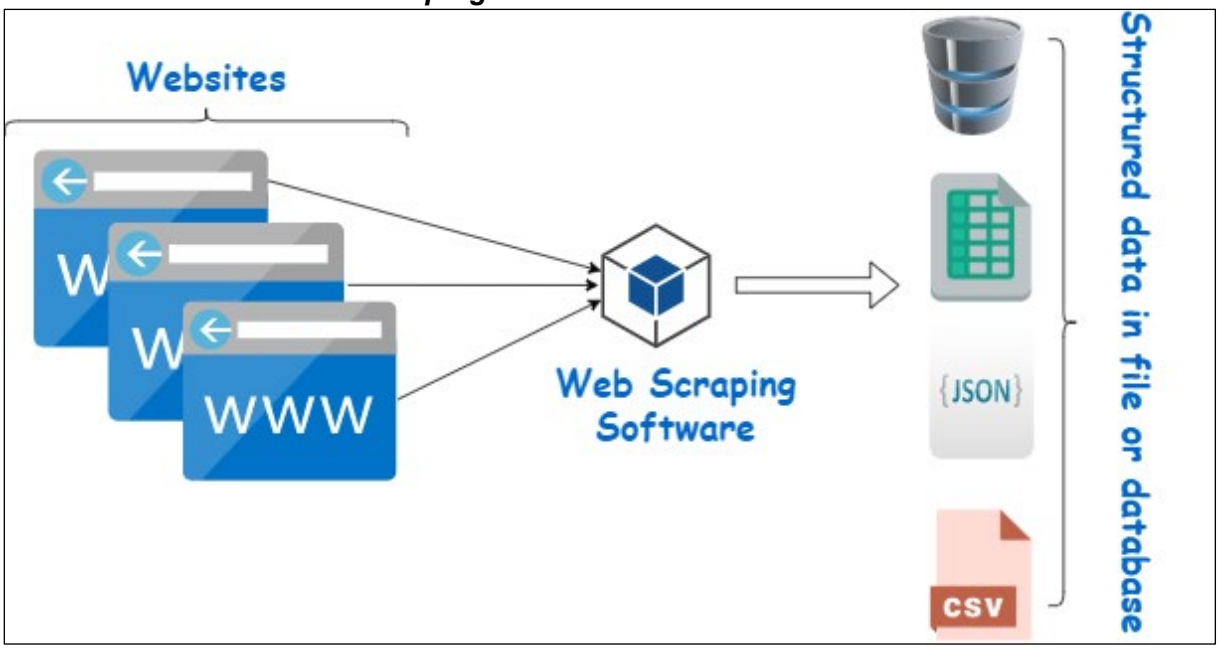
Len čo je požiadavka úspešne prijatá a spracovaná cielenou webovou stránkou, požadovaný zdroj bude z webovej stránky načítaný a potom odoslaný späť. Zdroj môže byť v niekoľkých formátoch, napríklad vo formáte HTML, XML alebo JSON. Zdroj môže tiež obsahovať multimediálne dáta, ako sú obrázky, audio alebo video súbory. Teda web scraping transformuje neštruktúrované dáta z webových stránok na štruktúrované.

Po získaní webových zdrojov proces extrakcie pokračuje v analýze, preformátovaní a usporiadaní dát štruktúrovaným spôsobom. Dôležitou súčasťou web scraperu sú dátové lokátory (čiže selektory) slúžiace na vyhľadanie dát, ktoré je potrebné z HTML súboru extrahovať – obvykle sa používa XPath, selektory CSS, regulárne výrazy alebo ich kombinácia.

Zjednodušený proces web scrapingu možno popísať nasledujúcimi krokmi (podľa [12], pozri tiež [1]):

1. **Identifikácia cieľovej webovej stránky.**
2. **Odosielanie požiadavky na URL adresu cieľovej webovej stránky.**
3. **Získanie HTML kódu stránky.**
4. **Vyhľadanie informácie v HTML kóde.**
5. **Uloženie dát do štruktúrovaného formátu (JSON, CSV a pod.).**

Obrázok č. 1: Proces web scrapingu



Zdroj: WebHarvy

Extrahované neštruktúrované údaje z webových lokalít je ďalej potrebné transformovať do potrebnej podoby hodnôt a štruktúry. Následne sa transformované a štruktúrované dáta uložia do databázy, resp. vyexportujú v príslušnom formáte. Transformácia údajov teda môže pozostávať z rôznych fáz, napríklad z čistenia údajov, prevodu kódovaných hodnôt, výpočtu nových hodnôt a pod. Významnú úlohu tu zohráva využitie rôznych techník machine learningu, databázových jazykov a štatistických metód (exploratívna analýza údajov). Pozri tiež kapitolu č. 4.

3. KNIŽNICE JAZYKA PYTHON PRE WEB SCRAPING

Jazyk Python umožňuje používanie širokej škály štatistických a grafických techník, ako napríklad regresná analýza, analýza časových radov, štatistické testy, zhluková analýza a i. Všetky tieto skutočnosti z neho robia ideálnu voľbu pre Data Science, analýzu Big Data a Machine Learning. Iinformácie o jazyku Python pozri v [8], resp. v inej zodpovedajúcej literatúre (tieto informácie tu neuvádzame).

Na rozšírenie možností jazyka sa v programovacích jazykoch používajú knižnice. Knižnica je súbor funkcií, tried, metód atď., zhromaždených na jednom mieste, ktoré sú potom využívané inými programami. Napríklad pre web scraping v jazyku Python sú to tieto knižnice (podľa [12]):

Requests (https://docs.python-requests.org)

Requests je jednoduchá knižnica, ktorá umožňuje odosielať HTTP požiadavky pomocou Pythonu. HTTP požiadavka potom vráti objekt odpovede so všetkými dátami (obsah, kódovanie, stav atď.). Nie je to knižnica výlučne určená pre web scraping, ale predstavuje pre neho osnovu.

BeautifulSoup (www.crummy.com/software/BeautifulSoup)

BeautifulSoup je jedným z najjednoduchších nástrojov pre web scrapingu v Pythone. Táto knižnica vyvinutá v roku 2004 poskytuje niekoľko jednoduchých metód na vyhľadávanie a extrahovanie potrebných dát. Niekedy funkcionality tejto knižnice úplne postačujú na vyriešenie problému, a zároveň výsledný skript nebude obsahovať mnoho kódov.

Tu však stojí za zmienku, že moderné webové stránky je možné rozdeliť na 2 typy: stránky so statickým obsahom a stránky s dynamickým obsahom. S druhým typom stránok sa BeautifulSoup nevysporiada. To znamená, že ak webová stránka obsahuje JavaScript alebo JQuery elementy, BeautifulSoup jednoducho nedokáže vyexportovať obsah vo vnútri nej.

Na druhú stranu má BeautifulSoup výhodu oproti iným nástrojom a tou je jeho schopnosť automaticky detegovať kódovanie, čo umožňuje spracúvať HTML dokumenty so špeciálnymi znakmi. Vie tak prevádzať prichádzajúce dokumenty na Unicode (medzinárodný štandard, ktorého cieľom je definovať kódovaciu schému schopnú reprezentovať väčšinu znakov používaných v písaných jazykoch spolu s inými symbolmi [15]) a odchádzajúce dokumenty na UTF-8 (8-bitový Unicode Transformation Format, je bezstratové kódovanie s variabilnou dĺžkou určené pre Unicode znaky [15]).

Selenium (https://www.selenium.dev)

Selenium je pôvodne automatizovaný testovací rámec používaný na overovanie webových aplikácií naprieč rôznymi prehliadačmi a platformami. Selenium umožňuje automatizovať webové prehliadače a má knižnice pre rôzne programovacie jazyky, vrátane Pythonu.

Selenium používa WebDriver na ovládanie webových prehliadačov, ako sú Chrome, Firefox alebo Safari. Postupom času sa však začala táto knižnica využívať nielen na testovanie aplikácií, ale aj na web scraping, a to vďaka svojej funkcionalite a kompatibilite s JavaScriptom.

Selenium je užitočný, keď je potrebné vykonať nejakú akciu na webe, napríklad na vypĺňanie polí alebo formulárov, rolovanie stránky, kliknutie na tlačidlá, vytvorenie snímky obrazovky. Ďalšou výhodou Selenia je možnosť fungovania s JavaScriptom. Vie napríklad načítať obsah vnorený do prvkov JavaScriptu. Selenium tiež podporuje tzv. *headless* prehliadače, čo sú prehliadače bez GUI, ktoré sa spúšťajú v príkazovom riadku. K výhodám podobných prehliadačov patrí väčšia rýchlosť a menšia spotreba pamäte.

Scrapy (https://www.scrapy.org)

Scrapy je open source a kolaboratívny rámec na extrahovanie dát z webových stránok rýchlym, jednoduchým a pritom rozšíriteľným spôsobom. V podstate ide

o najkomplexnejšie riešenie pre web scraping, ktoré poskytuje nástroje na prehliadanie webových stránok, sťahovanie dát, ich analýzu a ukladanie. Scrapy podporuje rozšírenie, čo prináša možnosť pridania proxy, spracovania súborov s cookies a ovládania hĺbky prehliadania.

Ďalšou vlastnosťou Scrapy je jeho asynchrónny spôsob spracovania požiadaviek. To umožňuje extrahovať dáta rýchlo aj z viacerých stránok naraz. Je však zrejmé, že keďže tento nástroj umožňuje viac, je tiež ťažšie ho nastaviť.

4. VYUŽTIE TECHNÍK STROJOVÉHO UČENIA

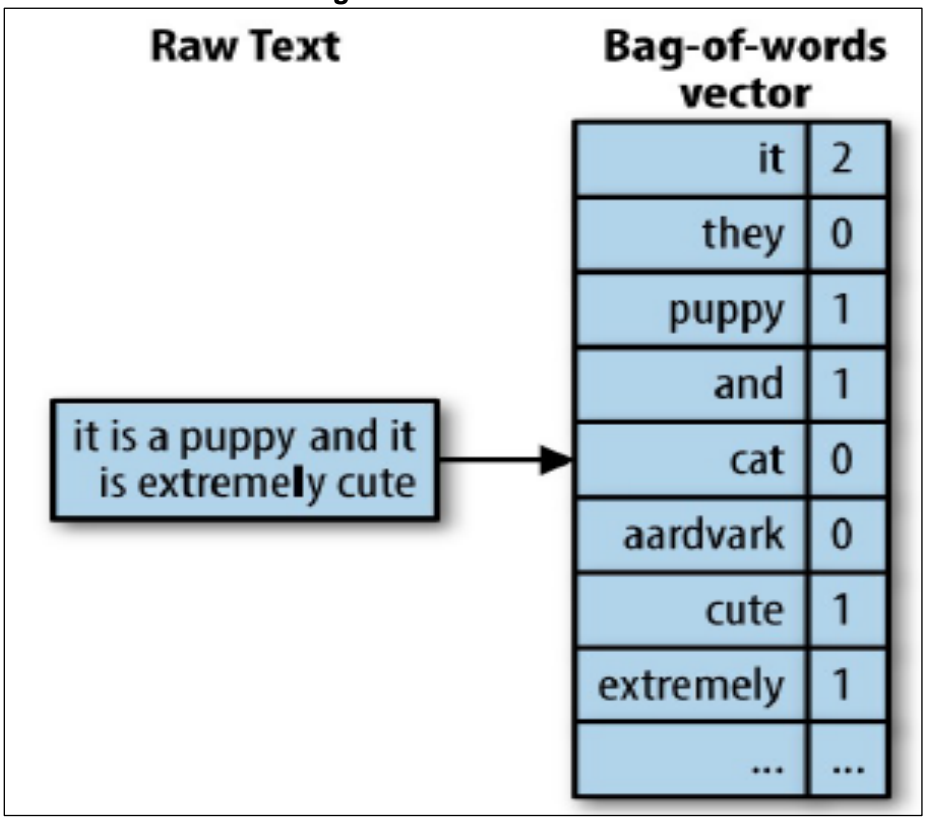
Autor v [2] skúma možnosti vytvorenia robustného algoritmu web scrapingu, ktorý je navrhnutý tak, aby neustále scrapoval konkrétnu webovú stránku, aj keď je HTML kód zmenený. Algoritmus je určený na použitie na webových stránkach, ktoré majú opakujúcu sa štruktúru HTML obsahujúcu údaje, pre ktoré je možné použiť web scraping. Opakujúca sa štruktúra HTML sa nachádza napríklad v spravodajských článkoch, videách, knihách a pod. V HTML tak vzniká kód, ktorý sa mnohokrát opakuje a líši sa tak napríklad len v nadpisoch. Dobrým príkladom môže byť napríklad Youtube. Scraper funguje pomocou klasifikácie textu slov v kóde HTML a trénuje podporný vektorový stroj na rozpoznávanie slov alebo názvov premenných. Klasifikácia slov obklopujúcich hľadané údaje sa vykonáva s predpokladom, že budúce HTML webovej stránky budú podobné súčasnej HTML, čo zase umožňuje vykonávať robustné scrapovanie. Na vyhodnotenie jeho výkonu sa používa webový archív, v ktorom sa výkon algoritmu spätne testuje na minulých verziách stránky, aby sme získali predstavu o tom, ako by mohol tento výkon v budúcnosti vyzerať. Algoritmus dosahuje rôzne výsledky v závislosti od veľkého množstva premenných v rámci samotných webových stránok, ako aj od minulých verzií webových stránok. Najlepšia výkonnosť bola napríklad dosiahnutá na Yahoo news s presnosťou 90 % z obdobia troch mesiacov od zastavenia scrapera.

Štandardný algoritmus scrapovania teda využíva statické cesty na navigáciu programu cez HTML k hľadaným údajom. Použitie statických ciest v nestatickom prostredí vedie k pokazeniu scrapera pri aktualizácii HTML vyžadujúcej aktualizáciu kódu, ktorý scrapovanie vykonáva. Frekvencia, s akou sa to musí robiť, do značnej miery závisí od webovej lokality a nemusí to byť príliš často. Ak sa firma spolieha na webový scraper pracujúci vo dne aj v noci, jeho pokazenie by mohlo spôsobiť narušenie procesov firmy. Keďže HTML je kód v textovom formáte, údaje musia byť pre model strojového učenia preformátované do číselnej formy. Predspracovanie údajov sa vykonáva pomocou extrakcie textových prvkov a využíva *bag-of-words model* s ďalšími úpravami na vytvorenie optimálneho výkonu. Predspracované údaje sa vložia do stroja Support Vector Machine (SVM, podporný vektorový stroj) na klasifikáciu údajov a extrahovanie údajov, ktoré sa pôvodne požadovali.

SVM sú trénované prostredníctvom učenia s učiteľom, čo znamená, že údaje, na ktorých je model trénovaný, musia byť označené. SVM môže byť použitý na klasifikáciu alebo regresiu. SVM na klasifikáciu v *N*-rozmernom kontexte klasifikuje dáta rozdelením *N*-rozmerného priestoru nadroviny tak, aby bolo správne klasifikovaných čo najviac bodov. [2] Pre optimálnu nadrovinu platí, že musí byť umiestnená v čo najväčšom odstupe (angl. Maximal margin) od krajných bodov, nazývaných podporné vektory (angl. Support vectors). Lineárny variant tohto algoritmu sa používa, keď sú dáta lineárne oddeliteľné, čiže oddeliteľné lineárnou nadrovinou. Algoritmus pracuje

v pôvodnej dvojrozmernej rovine dát. Nelineárna SVM sa používa, keď rovinu dát nie je možné rozdeliť lineárne. Tu sa uplatňuje funkcia nazývaná jadrová transformácia (angl. kernel transformation).

Obrázok č. 2: Príklad bag-of-words vektora

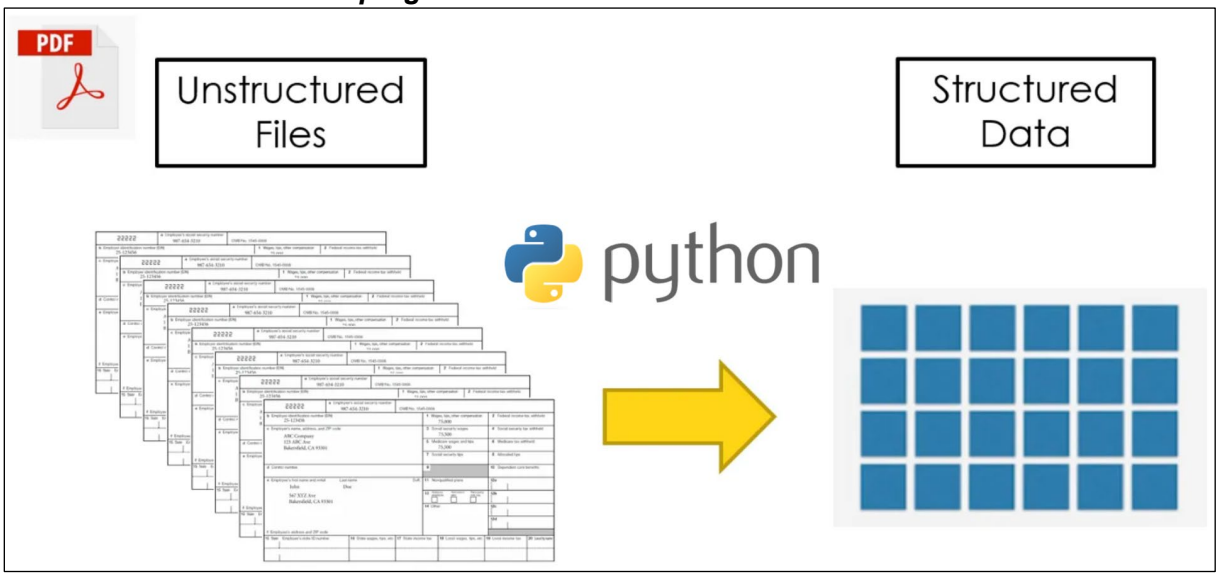


Zdroj: [2]

5. PRAKTICKÁ UKÁŽKA ZÍSKAVANIA ÚDAJOV Z PDF DOKUMENTOV

Väčšinou je web scraping založený na analýze štruktúry webovej stránky, a preto je dôležité pochopiť jej princíp. Existujú tzv. značkovacie jazyky, ktoré pomocou špeciálnych značiek vysvetľujú význam (sémantiku) rôznych častí textu alebo určujú vzhľad (formát) jednotlivých častí textu. K najvýznamnejším značkovacím jazykom patrí HTML a XML. HTML a XML sa používajú na ukladanie štruktúrovaných dát, prípadne na tvorbu vizuálneho pohľadu webových stránok. Tieto formáty definujú obsah celej webovej stránky, zatiaľ čo jedným z krokov web scrapingu je vyhľadanie konkrétnej informácie v zdrojovom kóde. Na tento krok je možné použiť XML Path Language (angl. skratka XPath). XPath je dotazovací jazyk, ktorý je užitočný pre identifikáciu a extrahovanie častí z dokumentov HTML/XML. Každý HTML alebo XML dokument si možno predstaviť ako strom. XPath potom umožňuje vyhľadávanie v podobných dokumentoch pomocou dotazu. [12] Keďže ide o širokú problematiku na ukážku jednoduchého cenového web scrapera v jazyku Python pozri napr. [5].

Obrázok č. 3: Proces scrapingu z PDF dokumentov



Zdroj: [14], upravené autormi

Niekedy však môžu byť údaje uložené aj v nekonvenčnom formáte, ako je napríklad PDF (obrázok č. 3). Ďalej teda budeme prezentovať prístup scrapingu z PDF pomocou Python modulu tabula-py. Pre viac informácií odporúčame dokumentáciu na webovej stránke: https://tabula-py.readthedocs.io/en/latest/.

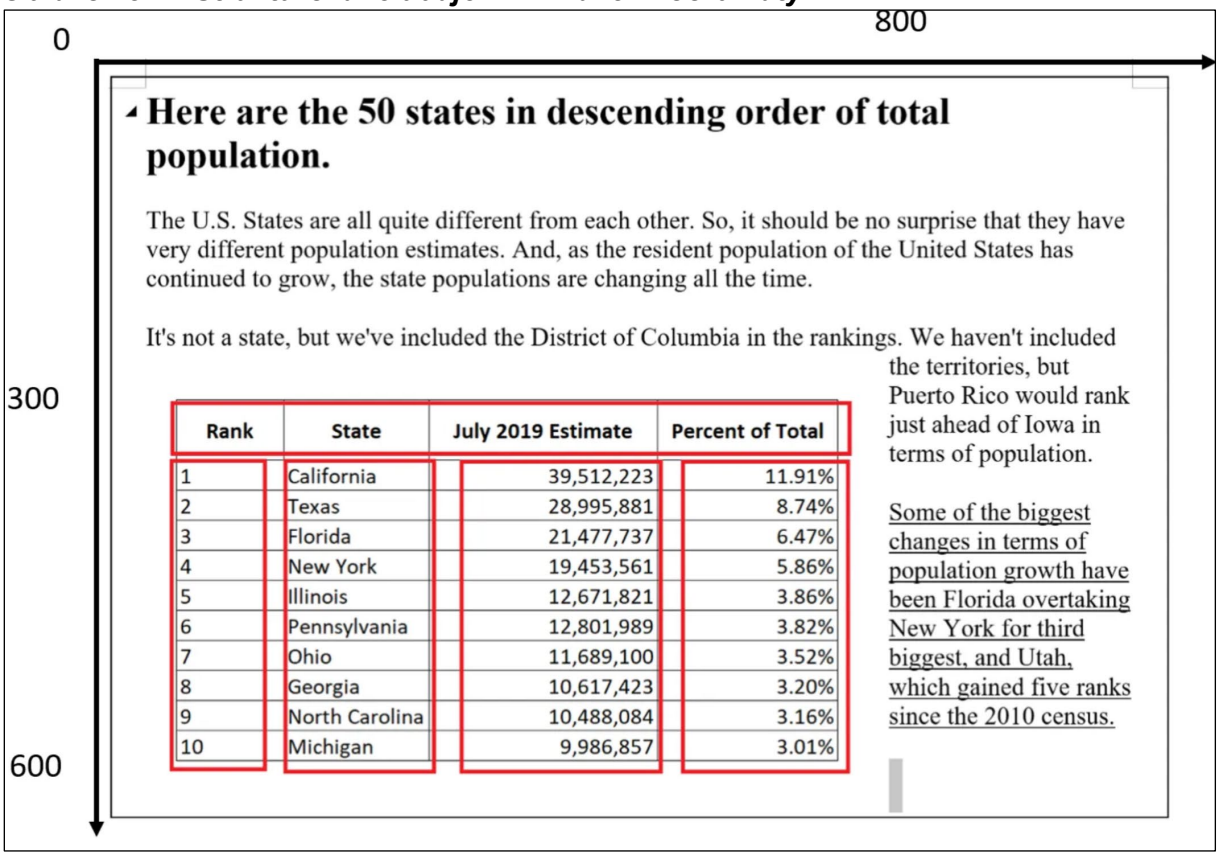
Jednoduchší prípad je, ak extrahujeme údaje z PDF v štruktúrovanom formáte. Ak napríklad chceme prehľadať tabuľku zvýraznenú na obrázku č. 4 – štruktúrované tabuľkové údaje, v ktorých sú prehľadne definované riadky a stĺpce. Extrahovanie takýchto údajov z PDF v štruktúrovanej forme je jednoduché využitím práve Python modulu tabula-py. Potrebujeme len zadať umiestnenie tabuľkových údajov (koordináty) na stránke PDF zadaním (hore, vľavo, dole, vpravo) súradníc danej oblasti. Ak stránka PDF obsahuje iba cieľovú tabuľku, potom ani nemusíme špecifikovať oblasť, funkcia tabula-py by mala byť schopná automaticky rozpoznať riadky a stĺpce danej tabuľky. Nižšie uvádzame kód v jazyku Python na extrahovanie údajov z PDF v štruktúrovanej forme:

```
pip install tabula-py
pip install pandas

import tabula as tb
import pandas as pd
import re

file = 'moje_pdf_1.pdf'
data = tb.read_pdf(file, area = (300, 0, 600, 800), pages = '1')
```

Obrázok č. 4: Štruktúrované údaje v PDF a ich koordináty



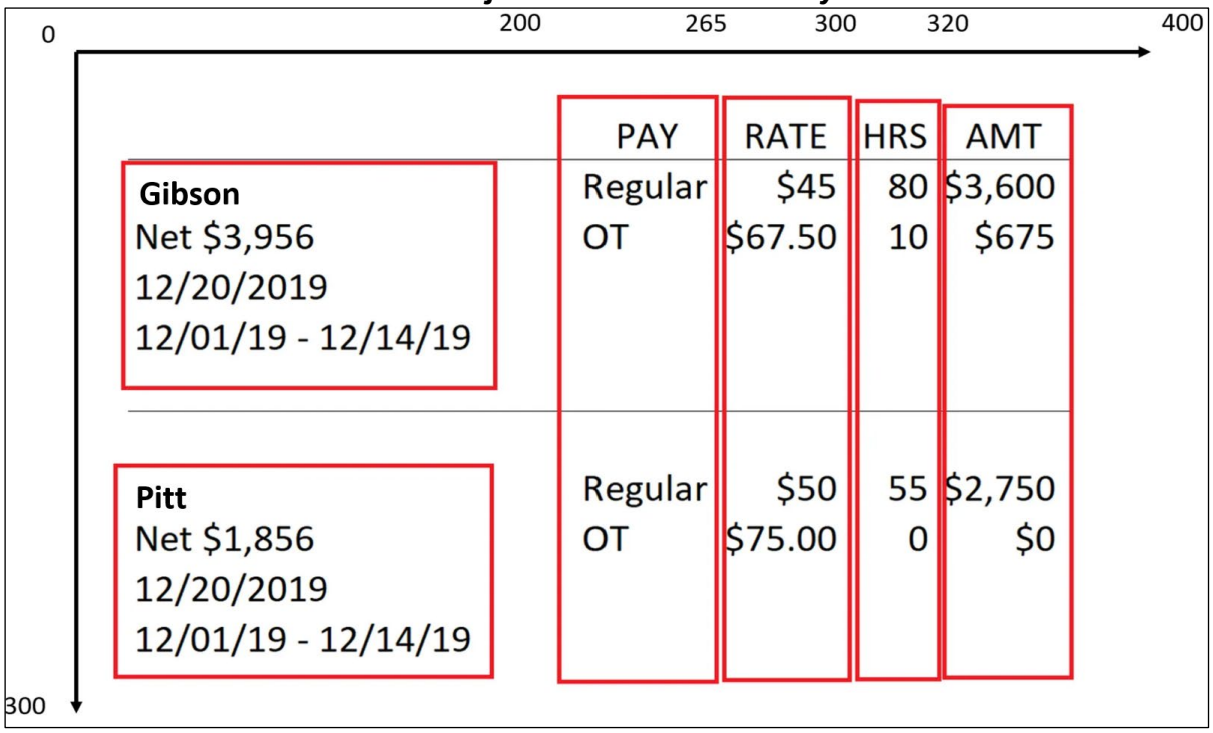
Zdroj: [14]

Na implementáciu štatistických analýz, vizualizáciu údajov a aplikáciu modelov strojového učenia potrebuje analytik údaje obvykle v tabuľkovej forme (panelové údaje). Mnohé údaje sú však dostupné iba v neštruktúrovanom formáte (pozri obrázok č. 5).

Najskôr importujeme údaje rovnakým spôsobom ako údaje v štruktúrovanom formáte, pričom musíme špecifikovať ďalšie atribúty na správny import údajov:

```
file = 'moje_pdf_2.pdf'
df= tb.read_pdf(file, pages = '1', area = (0, 0, 300, 400), columns = [200, 265, 300, 320], pandas_options={'header': None}, stream=True)[0]
```

Obrázok č. 5: Neštruktúrované údaje v PDF a ich koordináty



Zdroj: [14], upravené autormi

Tu používame aj nastavenie na identifikáciu umiestnení všetkých relevantných stĺpcov. Niekedy sa využíva metóda *pokus-omyl*. Ak existujú vo vstupe mriežkové čiary, ktoré oddeľujú bunky, môžeme použiť lattice = True na automatickú identifikáciu každej bunky. Ak nie, môžeme použiť stream = True a columns na manuálne určenie každej bunky. Režim stream bude hľadať medzery medzi stĺpcami. Tieto nastavenia sú často veľmi relevantné a môžu výrazne vylepšiť celkový scraping.

Obrázok č. 6: Získané neštruktúrované údaje z PDF

0	nan	PAY	RATE	HRS	AMT
1	Gibson	Regular	$45	80	$3,600
2	Net $3,956	OT	$67.50	10	$675
3	12/20/2019	nan	nan	nan	nan
4	12/01/19 - 12/14/19	nan	nan	nan	nan
5	Pitt	Regular	$50	55	$2,750
6	Net $1,856	OT	$75.00	0	$0
7	12/20/2019	nan	nan	nan	nan
8	12/01/19 - 12/14/19	nan	nan	nan	nan

Zdroj: vlastný, podľa údajov [14]

Následne získame údaje (obrázok č. 6), s ktorými môžeme ďalej pracovať. Na manipuláciu s dátovým rámcom môžeme využiť v jazyku Python napríklad knižnicu Pandas a naprogramovať kód (tento postup tu pre rozsah a náročnosť neuvádzame), ktorý údaje upraví do finálnej podoby (obrázok č. 7).

Obrázok č. 7: Získané štruktúrované údaje z neštruktúrovaných údajov z PDF

Index	row	Surname	net_amount	pay_date	pay_period
0	1	Gibson	$3,956	12/20/2019	12/01/19 - 12/14/19
1	2	Pitt	$1,856	12/20/2019	12/01/19 - 12/14/19

Index	row	OT_Rate	Regular_Rate	OT_Hours	Regular_Hours	OT_Amt	Regular_Amt
0	1	$67.50	$45	10	80	$675	$3,600
1	2	$75.00	$50	0	55	$0	$2,750

Zdroj: vlastný, podľa údajov [14]

6. ZÁVER

Web scraping je populárny spôsob získavania dát, ktorý má však má aj slabé stránky. Niektoré weby sa snažia zbaviť web scraperov a používajú tak rôzne techniky, ako sa vyrovnať s nežiaducim scrapovaním webu. Moderné riešenia umožňujú takéto nástroje detegovať a blokovať. Sem napríklad patria: CAPTCHA, Honeypot, dynamický obsah, zmeny štruktúry webový stránok, mnoho častých HTTP požiadaviek, IP blokovanie.

Niekedy vlastníci webových stránok píšu o tom, či je možné kopírovať obsah ich webu priamo na stránkach, a to spravidla dole, v tzv. footeri stránky. Ak tam nie je nič napísané, ďalším vhodným krokom je otvoriť súbor robots.txt.

Keďže web scraping je extrakcia dát z verejných webových stránok, tieto stránky niekedy obsahujú dáta, ktoré patria práve ich vlastníkom, na základe čoho vznikla diskusia, či je web scraping legálny. Na túto otázku stále neexistuje definitívna odpoveď. Na jednej strane sú údaje zverejnené na stránkach verejne dostupné, na druhej strane to, že sú tieto dáta zverejnené na internete, neznamená, že ich môže ktokoľvek používať. To platí najmä pre citlivé dáta alebo osobné údaje, ktoré podliehajú GDPR (General Data Protection Regulation) [12].

Práve prezentácie ukážky extrakcie údajov z PDF sa nemusia významne dotýkať týchto problémov pretože používateľ môže využiť scraping na získanie údajov, ktoré sa môžu týkať vlastnej firmy, resp. organizácie (faktúry, vlastné zoznamy a dokumenty, sprístupnené a schválené zdroje a pod.).

V súčasnosti veľa spoločností stále manuálne spracúva údaje z PDF. Pomocou predstavenej procedúry môžu ušetriť čas aj zdroje automatizáciou tohto procesu získavania údajov zo súborov PDF a konverziou neštruktúrovaných údajov na údaje panelové. Uveďme však, že aj tu je potrebné poznať zmluvné podmienky, resp. povolenia autora PDF dokumentu na takéto operácie.

Príspevok bol vytvorený v rámci projektov VEGA č. 1/0431/22; 1/0561/21.

LITERATÚRA

[1] BIHÁNY, D.: Cenové stratégie za využitia analýzy veľkých dát. Bakalářská práce: FPH, VŠE v Praze, 2022.
[2] CARLE, V.: Web Scraping using MachineLearning. 2020. [online]. [cit. 19. 4. 2023]. Dostupné na: https://www.diva-portal.org/smash/get/diva2:1468583/FULLTEXT01.pdf

[3] CASTRILLO-FERNÁNDEZ, O.: Web Scraping: Applications and Tools. 2015. [online]. [cit. 11. 4. 2022]. Dostupné na: https://data.europa.eu/sites/default/files/report/2015_web_scraping_applications_and_tools.pdf
[4] DENSMORE, J.: Ethics in Web Scraping. 2017. [online]. [cit. 21. 2. 2021]. Dostupné na: https://towardsdatascience.com/ethics-in-web-scraping-b96b18136f01
[5] KOMARA, S. – PÁLEŠ, M.: Web scraping v súčasnom konkurenčnom prostredí. In: Sborník příspěvků z 15. ročníku Mezinárodní vědecké konference KONKURENCE, Jihlava, Česká republika, 2023.
[6] MILEV, P.: Conceptual Approach for Development of Web Scraping Application for Tracking Information. In: Economic Alternatives, 2017, (3), s. 475 – 485.
[7] MITCHELL, R.: Web Scraping with Python. O'Reilly, 2018.
[8] PÁLEŠ, M.: Jazyk Python pre aktuárov. Bratislava: Letra Edu, 2022.
[9] PAPCO, L.: Efekt krátkodobého pronajímání nemovitostí na cenu nemovitostí v Praze. Diplomová práce: FPH, VŠE v Praze, 2021.
[10] SIRISURIYA, D. S.: A comparative study on web scraping. 2015. [online]. [cit. 19. 2. 2021]. Dostupné na: http://ir.kdu.ac.lk/bitstream/handle/345/1051/com-059.pdf?sequence=1&isAllowed=y
[11] STEFANOVA, K.: Factors and Main Directions for Business Intelligence Systems Design and Development. 2008. [online]. [cit. 14. 3. 2022]. Dostupné na: http://unweyearbook.org/uploads/Yearbook/Yearbook_2008_No6_K%20Stefanova.pdf
[12] TSAKUNOV, I.: Využití data miningu pro analýzu českého realitního trhu. Bakalářská práce: FIS, VŠE v Praze, 2022.
[13] ZANIKOV, M.: Analýza a vizualizace dat z webových portálů nabídek práce. Diplomová práce: FIS, VŠE v Praze, 2020.
[14] ZHU, A.: How to Scrape and Extract Data from PDFs Using Python and tabula-py. 2021. [online]. [cit. 19. 4. 2023]. Dostupné na: https://towardsdatascience. com/scrape-data-from-pdf-files-using-python-fe2dc96b1e68
[15] https://en.wikipedia.org/wiki/Unicode [cit. 19. 5. 2022].
[16] https://en.wikipedia.org/wiki/Web_scraping [cit. 19. 5. 2022]

RESUMÉ

V tomto príspevku sa zameriavame na špecifickú problematiku scrapingu, ktorou je extrakcia údajov z PDF dokumentov s využitím modulu tabula-py jazyka Python. Prezentujeme aj transformáciu neštruktúrovaných údajov na štruktúrované, čo môže byť využité aj pri klasickom web scrapingu.
Web scraping, web harvesting alebo extrakcia údajov z webu zahŕňa metódy na extrakciu údajov z webových stránok. Softvér web scrapingu môže priamo pristupovať na webovú stránku pomocou HTP protokolu alebo pomocou webového prehliadača. V súčasnosti tento pojem zahŕňa automatizované procesy implementované pomocou robota alebo webového prehľadávača na zhromažďovanie a kopírovanie špecifických údajov z webu, zvyčajne do centrálnej lokálnej databázy alebo tabuľky, na neskoršie vyhľadávanie alebo analýzu. Scraping webovej stránky zahŕňa jej načítanie a extrakciu údajov z nej. Obsah stránky možno analyzovať, prehľadávať a preformátovať a jej údaje skopírovať do tabuľky alebo načítať do databázy. Príkladom môže byť vyhľadanie a skopírovanie mien a telefónnych čísel, názov spoločnosti a ich URL alebo e-mailových adries do zoznamu (scrapingových kontaktov). Okrem toho sa web scraping používa ako súčasť aplikácií na indexovanie webu, ťažbu z webu a dolovanie údajov, online sledovanie zmeny cien a porovnávanie cien, scrapovanie recenzií produktov (na sledovanie konkurencie), zhromažďovanie

zoznamov nehnuteľností, údajov o počasí, detekciu zmien webových stránok a pod. Webové stránky sú vytvorené pomocou značkovacích jazykov založených na texte (HTML a XHTML) a často obsahujú množstvo užitočných údajov v textovej forme. Väčšina webových stránok je však navrhnutá pre koncových používateľov a nie na automatizované používanie. V dôsledku toho boli vyvinuté špecializované nástroje a softvér na uľahčenie scrapovania webových stránok.
Existuje mnoho dostupných softvérových nástrojov, ktoré možno použiť na web scraping. Takýto softvér sa môže pokúsiť automaticky rozpoznať dátovú štruktúru stránky alebo poskytnúť nahrávacie rozhranie, ktoré odstraňuje nutnosť manuálneho zápisu kódu na scrapovanie webu, alebo niektoré skriptovacie funkcie, ktoré možno použiť na extrahovanie a transformáciu obsahu, a databázové rozhrania, ktoré môžu ukladať skopírované údaje v lokálnych databázach. Niektorý softvér možno použiť aj na priamu extrakciu údajov z rozhrania API (Application Programming Interface) [16]. Jedným často využívaným softvérom je jazyk Python (napríklad s využitím modulu BeautifulSoup). Strojové učenie môže byť využité na vytvorenie robustného algoritmu web scrapingu, ktorý je navrhnutý tak, aby neustále scrapoval konkrétnu webovú stránku, aj keď je HTML kód zmenený. Algoritmus je určený na použitie na webových stránkach, ktoré majú opakujúcu sa štruktúru HTML obsahujúcu údaje, pre ktoré je možné použiť web scraping. Štandardný algoritmus scrapovania teda využíva statické cesty na navigáciu programu cez HTML k hľadaným údajom. Použitie statických ciest v nestatickom prostredí má napríklad za následok pokazenie scrapera pri aktualizácii HTML. Techniky strojového učenia sa využívajú vtedy, keď údaje musia byť pre model strojového učenia preformátované do číselnej formy. Predspracovanie údajov sa realizuje pomocou extrakcie textových prvkov a využíva *bag-of-words model* s ďalšími úpravami na vytvorenie optimálneho výkonu. Predspracované údaje sa vložia do stroja Support Vector Machine (SVM, podporný vektorový stroj) na klasifikáciu údajov a extrahovanie údajov, ktoré boli pôvodne požadované.

RESUME
In this paper, we focus on the specific issue of scraping, which is the extraction of data from PDF documents using the tabula-py module of the Python language. We also present the transformation of unstructured data into structured data, which can also be used in classic web scraping.
Web scraping, web harvesting, or web data extraction involves methods for extracting data from web pages. Web scraping software can directly access a website using the HTP protocol or using a web browser. Currently, the term includes auto-mated processes implemented using a robot or web crawler to collect and copy specific data from the web, usually into a central local database or table, for later retrieval or analysis. Scraping a website involves its loading and data extracttion data from it. The content of the page can be analyzed, searched and reformatted, and its data can be copied into a table or loaded into a database. An example could be searching and copying names and phone numbers, names of companies and their URLs or email addresses into a list (scraping contacts). In addition, web scraping is used as part of applications used for web indexing, web mining and data mining, online price change tracking and price comparison, scraping product reviews (to track competitors), collecting real estate listings, weather data, website change detection etc. Web pages are created using text-based markup languages (HTML and XHTML) and often contain a lot of useful data in text form. However, the most websites are designed for end users and not for an automated use. As a result, specialized tools and software have been developed to facilitate website scraping.

There are many software tools available that can be used for web scraping. Such software may attempt to automatically recognize the data structure of a page or provide an upload interface that eliminates the need to manually write web scraping code, or some scripting functions that can be used to extract and transform content, and database interfaces that can store scraped data in local databases. Some software can also be used to directly extract data from an Application Programming Interface (API) [16]. One frequently used software is the Python language (for example, using the BeautifulSoup module). Machine learning can be used to create a robust web scraping algorithm that is designed to continuously scrape a specific web page even if the HTML code is changed. The algorithm is intended for use on web pages that have a repeating HTML structure containing data that can be web scraped. Thus, the standard scraping algorithm uses static paths to navigate the program through HTML to the searched data. For example, using static paths in a non-static environment results in the scraper breaking when updating the HTML. Machine learning techniques are used when data needs to be reformatted into numerical form for a machine learning model. Data pre-processing is performed using text feature extraction and uses a "bag-of-words model" with additional adjustments to create optimal performance. The pre-processed data is fed into a Support Vector Machine (SVM) to classify the data and extract the data that was originally requested.

PROFESIJNÝ ŽIVOTOPIS

Ing. Silvia Komara, PhD., *pôsobí na Katedre štatistiky Fakulty hospodárskej informatiky Ekonomickej univerzity v Bratislave. Jej vedecká činnosť je orientovaná v prvom rade na modelovanie a analýzu ekonomických časových radov a finančných časových radov so zameraním na kvalitu krátkodobej prognózy. Okrem toho sa venuje metódam strojového učenia (machine learning). Vyučuje predmety analýza časových radov, Machine Learning na 2. stupni štúdia v študijnom programe Data Science v ekonómii a predmety štatistika a štatistika v anglickom jazyku na 1. stupni štúdia. Absolvovala viaceré zahraničné pobyty na univerzitách v Madride, Sydney, Ostrave, Girone a i.*

Doc. Ing. Michal Páleš, PhD., *pôsobí ako vedúci Katedry matematiky a aktuárstva Fakulty hospodárskej informatiky Ekonomickej univerzity v Bratislave. V rámci pedagogickej činnosti vyučuje predmety matematika, matematika pre ekonómov, teória pravdepodobnosti, softvérové aplikácie pre aktuárov, teória rizika v poistení, úvod do aktuárstva a vybrané kapitoly z matematiky pre ekonómov. Vo svojej vedeckej práci sa orientuje na aktuársku vedu, využitie kvantitatívnych metód v ekonómii a softvérovú podporu riadenia rizík (najmä jazyk R). Je členom Slovenskej spoločnosti aktuárov a autorom viacerých vedeckých monografií, medzinárodne ocenených vysokoškolských učebníc a článkov z oblasti aktuárstva.*

KONTAKT

silvia.komara@euba.sk
michal.pales@euba.sk