Fakulteta za računalništvo in informatiko, Univerza v Ljubljani

Aljaž Srša 63120233 in Gregor Sušnik 63120102

Zajemanje vsebine spletnih strani

Poročilo druge seminarske naloge pri predmetu Iskanje in ekstrakcija podatkov s spleta
Asistent: asist. prof. dr. Slavko Žitnik

Povzetek

V seminarski nalogi implementirava zajemanje podatkov na tri različne načine . Pri vsakem načinu opiševa uporabljene izraze, ki zajamejo zahtevano vsebino na spletni strani. Implementirane metode preizkusiva na določenih parih spletnih strani – overstock, rtvslo.si in newegg.com. Dobljene rezultate primerjava v poglavju rezultati.

• Uvod

Ročno zajemanje podatkov iz spletnih strani je precej zahtevno in ne praktično. Še posebej, če imamo opravka z veliko količino podatkov. Zajemanje večje količine podatkov poskušamo avtomatizirati, saj s tem prihranimo veliko časa in dela, ki bi ga opravljali ročno.

V drugi seminarski nalogi smo se lotili procesa zajemanja podatkov. Ideja je bila implementirati na tri različne načine za pridobivanje podatkov, in sicer: uporaba regularnih izrazov, uporaba XPath izrazov in uporaba RoadRunner pristopa. Vsak način smo na koncu preizkusili na treh različnih tipih strani. Vsak tip strani je vseboval dve strani. Nad vsemi tremi pari stranmi – skupaj torej šest strani, so se preizkusili vsi trije načini za pridobivanje podatkov in med posameznima stranema v paru se je primerjalo pridobljene podatke.

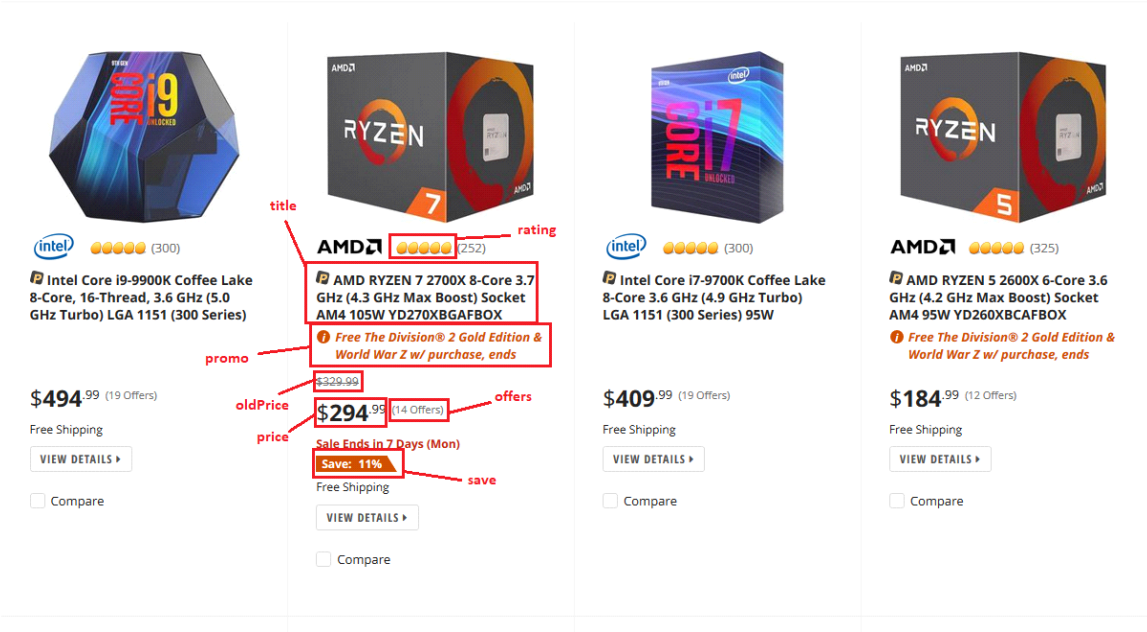
• Metode pridobivanja podatkov

Seminarsko nalogo sva izdelala v jeziku Python. Program se začne izvajati v glavni datoteki main.py, v kateri se prebere HTML datoteka s pomočjo funkcije read(). Prebrana datoteka se s pomočjo knjižnice BeautifulSoup olepšala v smislu optimizacije, zaključka značk, itp. Tako pripravljen HTML je vhod za vse tri funkcije pridobivanja podatkov. Vsak način je implementiran kot funkcija v ločeni datoteki. Kateri tip spletne strani pa ureja spremenljivka pageType. Tip je lahko nastavljen na 0 (Overstock), 1 (rtvslo.si) ali 2 (newegg.com)

• Opis strani newegg

Dve dodatni spletni strani sva izbrala iz domene ***newegg***, ki je spletna trgovina za nakupovanje računalniških komponent in ostalih elektronskih izdelkov. Tu sva na voljo imela dva tipa strani: seznam izdelkov ali podroben opis posameznega izdelka. Odločila sva se za uporabo strani s seznamom izdelkov (seznam procesorjev in seznam grafičnih kartic) in iz

njih izluščila pomembne podatke. Slika 1 prikazuje tiste elemente na strani, ki sva jih želela izluščiti.



Slika 1: Podatki, ki nas zanimajo

Podatki, označeni na zgornji sliki so naslednji:

- **title** - ime in kratek opis izdelka
- **promo** - opcijske promocijske ponudbe z nakupom izdelka
- **rating** - ocena izdelka
- **oldPrice** - stara cena pred popustom
- **price** - trenutna cena izdelka
- **offers** - število ponudb
- **save** - vrednost popusta v odstotkih

- **Uporaba regularnih izrazov**

Funkcija sprejme obdelan HTML string in tip spletne strani. Na podlagi podatka o tipu spletne strani se izvede koda. Za zajem podatkov poskrbijo spodnji regularni izrazi. Pridobljeni podatki se shranijo v slovar, ta pa se s pomočjo funkcije json_dump() izvozi v berljivo obliko.

Overstock

regex = r'<b>([0-9].+)<\/b>.\n.*\n.*<s>([\$0-9,.]+).*\n.*<b>([\$0-9.]+).*\n.*>([\$0-9.,]+)\s.([0-9]+.).*\n.*\n.*<span class=\"normal\">(.*/.*\n.*\n.*/.*\n.*\n.*\n.*)
<a'*

Razlaga regularnega izraza:

- <b>([0-9].+)<\/b> poišče naziv izdelka, ki pa se vedno prične z neko številko
- .*\n.*\n.* Izpusti nekaj vsebine, ki je med nazivom izdelka in cenami
- ([\$0-9,.]+) Zajame vsebino 'listPrice'
- .*\n.* Izpusti nekaj vsebine

- <b>([\$0-9.]+) Zajame vsebino 'price'
- .*\n.* Izpusti nekaj vsebine
- >([\$0-9.,]+)\s.([0-9]+.).*\n.*\n.* Zajame 'saving' in 'savingPercent'
- <span class=\"normal\">(.*|.*\n.*\n.*|.*\n.*\n.*\n.*)
<a' Zajame vsebino opisa izdela. Pri čemer ima opis lahko več vrstic opisa.

Rtvslo.si

regex = r'<h1>(.)<\/h1>[\s\S]+<div class=\"subtitle\">(.*)<\/div>[\s\S]+<p class=\"lead\">(.*)<\/p>[\s\S]+<div class=\"author-name\">(.*)<\/div>[\s\S]+\"publish-meta\">[\n\W]+(.*)<br\/>[\s\S]+<\/div>[\n]*<\/figure>[\n]*<p([\s\S]*.*)<div class=\"gallery\">'*

Razlaga regularnega izraza:

- <h1>(.*)</h1> poišče naslov članka
- >[\s\S]+ izpusti nekaj nezanimive vsebine
- <div class=\"subtitle\">(.*) poišče podnaslov
- [\s\S]+ ponovno izpusti nekaj vsebine
- <p class=\"lead\">(.*)</p> poišče nagovorno besedilo
- [\s\S]+ izpusti nekaj vsebine
- <div class=\"author-name\">(.*) poišče avtorja članka
- [\s\S]+ izpusti nekaj vsebine
- \"publish-meta\">[\n\W]+(.*)
' poišče datum objave članka
- *<\/div>[\n]*<\/figure>[\n]*<p([\s\S]*.*)<div class=\"gallery\">* zajame celotno vsebino članka

Vsebina (content) članka vsebuje poleg besedila tudi preostale značke in znake. Te s pomočjo funkcije lxml.html.fromstring(content).text_content() odstraniva. Tako se iz podanega besedila izlušči le tekst.

newegg.com

regex = r'<div class=\"item-container\">(?:\n.*){,15}(?:rating-(\d).*(?:\n.*){,4})?Details\">(?:<i.*<\/i>)?((?:.*\n){,5})<!--p.*\n.*>(?:.*<\/i>)?(.*)<\/p>(?:\n.*){,13}s\">(\n.*)?(?:\n.*){,9}(\$).*>([0-9]?\,?[0-9]{2,}).*>(\.\d\d)(?:[^()]*\(([\d]{1,}.*)\).*)?(?:(?:\n.*){,9}\n)?(?:.*>([0-9]{,2}\%))?'

- Razlaga regularnega izraza:
  <div class=\"item-container\">(?:\n.*){,15} zajame le elemente v tabeli – izpusti »featured item«, ki je prikazan nad tabelo izdelkov, nato pa preskoči kar nekaj vrstic vsebine
- (?:rating-(\d).*(?:\n.*){,4})? Zajame oceno artikla in spusti nekaj vrstic, če te obstajajo
- Details\">(?:<i.*<\/i>)?((?:.*\n){,5}) zajame opis izdelka, ki lahko vsebuje dodatne classe v znački <i> in je lahko dolg več vrstic
- <!--p.*\n.*>(?:.*<\/i>)?(.*)<\/p>(?:\n.*){,13} zajame promocijski napis, če ta obstaja. Značka <i> je lahko opcijska. Nato preskoči nekaj vrsti vsebine
- s\">(\n.*)? Zajame staro ceno, če ta obstaja – je napisana

- (?:\n.*){,9}(\$) preskoči nekaj vsebine do elementa s ceno. Najprej zajame znak za dolar
- .*>([0-9]?\,?[0-9]{2,}).*>(\.\d\d) nato izpusti nekaj znakov in zajame tisočico, vključno z vejico, če ta obstaja. Po tisočici zajame še vsaj dvomestno številko. Izpusti nekaj vsebine in zajame še dve decimalni mesti. Vse skupaj predstavlja ceno
- (?:[^(]*\(([\d]{1,}.*)\).*)? Poišče prvi znak (in zajame številko on besedo 'offers', ki predstavlja število ponudb. Element ne obstaja pri vseh artiklih
- (?:(?:\n.*){,9}\n)?(?:.*>([0-9]{,2}\%))? Preskoči nekaj vsebine, če je to potrebno in zajame element, ki predstavlja procent prihranka, v primeru, da je akcijska cena. Element ne obstaja pri vseh artiklih

- **Uporaba XPath**

Funkcija sprejme obdelan HTML string in tip spletne strani. Na podlagi podatka o tipu spletne strani se izvede koda. Za zajem podatkov poskrbijo spodnji XPath izrazi. Pridobljeni podatki se shranijo v slovar, ta pa se s pomočjo funkcije json_dump() izvozi v berljivo obliko.

Overstock

Najprej poiščemo del v HTML, kjer se nahajajo željeni podatki. Iščejo se vsi takšni <TR> nodi, ki imajo atribut bgcolor nastavljen na določeno vrednost pri tem pa mora omenjen node imeti še točno dva otroka tipa <TD>

objects = tree.xpath('//tbody/tr[(contains(@bgcolor, "#ffffff") or contains(@bgcolor, "#dddddd")) and count(td[@valign="top"]) = 2]

Nato se nad trenutno lokacijo pridobivajo podatki o vsebini, ki nas zanimakar v for zanki.

```
for obj in objects:
    title = obj.xpath('string(a/b/text())')
    listPrice = obj.xpath('string(table/tbody/tr/td[1]/table/tbody/tr[1]/td[2]/s/text())')
    price = obj.xpath('string(table/tbody/tr/td[1]/table/tbody/tr[2]/td[2]/span/b/text())')
    saving = obj.xpath('substring-before(table/tbody/tr/td[1]/table/tbody/tr[3]/td[2]/span/text(),
    " ")')
    savingPercent = obj.xpath('substring(substring-
    after(table/tbody/tr/td[1]/table/tbody/tr[3]/td[2]/span/text(), " "),2 ,3)')
    content = obj.xpath('string(table/tbody/tr/td[2]/span/text())')
```

Tudi tukaj se izvede json dump, ki se vrne v main kot rezultat funkcije.

Rtvslo.si

Najprej se poišče lokacijo v HTML kodi, kjer se nahajajo željeni podatki. Nato pa se od omenjene lokacije na podlagi relativne poti zajemajo zahtevani podatki.

```
rootObject = tree.xpath('//div[contains(@class, "news-container")]/div')[0]
author = rootObject.xpath('string(div[@class="article-
meta"]/div[@class="author"]/div/text())')
publishTime = rootObject.xpath('string(div[@class="article-meta"]/div[@class="publish-
                                    meta"]/text())')
```

```
title = rootObject.xpath('string(header/h1/text())')
subTitle = rootObject.xpath('string(header/div[@class="subtitle"]/text())')
lead = rootObject.xpath('string(header/p/text())')
```

Vsebina članka pa se zajeme s spodnjim XPath izrazom:

```
contentList = rootObject.xpath('div[@class="article-body"]/article[@class="article"]//p | '
                               'div[@class="article-body"]/article[@class="article"]//strong')
content = ""
for p in contentList:
    if p.text is not None:
        content += '\n' + p.text
        if p.tail is not None:
            content += '\n' + p.tail
```

newegg.com

```
    objects = tree.xpath('//div[@class="item-container"]')
    for obj in objects:
        dataInfo = obj.xpath('div[@class="item-info"]')

        item = {}
        title = dataInfo[0].xpath('string(a[@class="item-title"]/text())')
        rating = dataInfo[0].xpath('string(div[@class="item-branding"]/a[@class="item-
rating"]/@title)')
        promo = dataInfo[0].xpath('string(p[@class="item-promo"]/text())')
        priceCurrent = dataInfo[0].xpath('string(div[@class="item-action"]/ul/li[@class="price-
current"])')
        offers = dataInfo[0].xpath('string(div[@class="item-action"]/ul/li[@class="price-
current"]/a/text())')
        priceWas = dataInfo[0].xpath('string(div[@class="item-action"]/ul/li[@class="price-
was"]/text())')
        priceSave = dataInfo[0].xpath('string(div[@class="item-action"]/ul/li[@class="price-
save"]/span[@class="price-save-percent"]/text())')
```

- **Uporaba RoadRunner algoritma**

Pri algoritmu RoadRunner sva najprej začela z razdelitvijo HTML datoteke glede na nove vrstice (\n) ali pa HTML značke. Te naj bi nato uporabila pri ujemanju nizov in ujemanju značk, vendar pa nama je to povzročilo nekaj težav, zaradi česar nama ni uspelo razviti tega algoritma.

- **Rezultati**

Ker so celotni izpisi dolgi in bi pri vključevanju celotnega izpisa za vse domene bilo kar nekaj strani izpisa, sva se odločila, da jih shraniva posebaj. Vsi rezultati so zato shranjeni v datotekah tipa JSON, ki se nahajajo v mapi **implementation/results**. Na spodnjih slikah Slika 2, Slika 3 ter Slika 4 so prikazani le delčki izsekov izluščenih podatkov za vsako domeno. Pri uporabi XPath izrazov in regularnih izrazov sva dobila enako število izluščenih elementov za vse tri tipe strani. Besedilo se med uporabljenima metodama popolnoma ujema.

Overstock


```
"2": {
    "Title": "10-Kt. Diamond Ring (.25 TW)",
    "content": "Nineteen round diamonds accent this 10-karat yellow gold ring with filigree accents.",
    "listPrice": "$250.00",
    "price": "$74.90",
    "saving": "$175.10",
    "savingPercent": "70%"
},
```


Slika 2: Rezultat izluščenega besedila za en element strani tipa Overstock

RTVSLO.si


```
"1": {
    "Author": "Miha Merljak",
    "Lead": "To je novi audi A6. V razred najdražjih in najbolj premijskih žrebcev je vnesel nemir,
    "PublishTime": "28. december 2018 ob 08:51",
    "SubTitle": "Test nove generacije",
    "Title": "Audi A6 50 TDI quattro: nemir v premijskem razredu",
    "content": "Samo poglejte njegovo masko – to ogromno satovje z radarji na takem položaju, da se
}
```


Slika 3: Rezultat izluščenega besedila za en element strani tipa Rtvslo.si

Newegg


```
"8": {
    "Offers": "12",
    "OldPrice": "$199.99",
    "Price": "$164.99",
    "Promo": "Extra savings w/ promo code AMDRYZEN26, limited offer",
    "Rating": "5",
    "Save": "18%",
    "Title": "AMD RYZEN 5 2600 6-Core 3.4 GHz (3.9 GHz Max Boost) Socket AM4 65W YD2600BBAFBOX Desktop Processor"
},
```


Slika 4:Rezultat izluščenega besedila za en element strani tipa Newegg