Scraper und Datenextraktion

Scraper

Für jede Stellenbörse gibt es ein eigenes Script, das den Namen der Stellenbörse trägt. Alle Web-Scraper basieren auf der Bibliothek „Selenium“. Es wird der Browser Google Chrome verwendet um die Daten von der Webseite zu scrapen. An allen Scraper wird der Jobtitel und der Suchort als Argument übergeben. Der Standardwert für den Suchort ist Deutschland.

Die Scraper arbeiten alle nach dem gleichen Grundprinzip:

1. Webseite aufrufen
2. Jobtitel und Suchort angeben / einfügen
3. Wenn möglich nach Datum filtern
4. Etwa zehn Seiten bearbeiten
    - Von der Übersichtsseite die direkte URL zur Stellenanzeige scrapen
    - Teilweise gibt es eine Übersichtsseite, die durch Scrollen verlängert wird, oder weitere Übersichtsseite aufrufen
5. Überprüfen, ob die URL der Stellenanzeige schon in der Datenbank vorhanden ist
    - Duplikate entfernen
    - Spalte URL der Tabelle Rohdaten einlesen und abgleichen, dass nur die behalten werden, die nicht in der Datenbank enthalten sind
6. Von der Stellenanzeige den gesamten HTML-Code scrapen und im HTML- und Text-Format speichern
7. Weitere Daten hinzufügen
    - Name er Stellenbörse
    - URL der Stellenanzeige
    - Datum wann gescrapt wurde
    - Wert für die Existenz der Stellenanzeige
8. Daten zur Datenbank hinzufügen
    - ➢ *ACHTUNG:* Die *Spalte ID* wird von der SQL-Datenbank *Automatisch ergänzt*

Zu jedem wichtigen Ereignis werden Einträge in eine LOG-Datei erstellt. Zudem wird auch parallel bei kritischen Ereignissen eine E-Mail versendet.

Für jeden Scraper gibt es „Sonderfunktionen“. Diese sind:

- LOG-Datei schreiben
- E-Mail schreiben
- Wartezeit erstellen

Es ist empfehlenswert die Scripte für die Scraper und Datenextraktion täglich auf ihre Funktion zu Prüfen. Bei Bedarf ist eine Anpassung vorzunehmen, wenn

- bei den Scrapern weitere Stufen der Bot-Erkennung greifen
- bei der Datenextraktion die Klassennamen nicht mehr stimmen

Datenextraktion

Im Script „Aufarbeitung“ werden alle Daten der jeweiligen Stellenbörsen aus der View „aktuell“ eingelesen, nach den Stellenbörsen gefiltert. Danach wird für jede Stellenbörse die Datenextraktion durchführt und in einem DataFrame abspeichert.

Die Datenextraktion funktioniert für alle Stellenbörsen nach dem gleichen Grundprinzip:

1. Daten aus Datenbank laden
2. Daten nach Stellenbörse aufteilen
3. Extraktion der Daten mit BeautifulSoup für
    - Ort
    - Jobtitel (Beruf)
    - Firma
    - Anstellungsart (Teilzeit, Vollzeit, etc.)
    - Beschreibung (Beschreibung von Stelle, Firma, etc.)

    Wenn möglich sonst numpy.nan
4. ID aus den bearbeiteten Daten übernehmen – ***zwingend!!!***
    - Wird für die Erstellung einer View benötigt, die Spalten aus zwei Tabellen zusammenfügt
5. Die erstellten Listen auf gleiche länge bringen mit numpy.nan
6. DataFrame für jede Stellenbörse erstellen

Nachdem für alle Stellenbörsen die DataFrames erstellt sind, werden diese zu einem DataFrame zusammengefügt. Da beim scrapen auch Stellenanzeigen mitgenommen werden, die nichts mit unserem Ziel gemeinsam haben, wird noch nach den Jobtiteln gefiltert.

Da die Trennung von Jobtitel und Firma bei Stepstone nicht immer funktioniert, wird für die Filterung der neue DataFrame wieder aufgeteilt. Der erste DataFrame enthält alles, wo in der Spalte „beruf“ NaN-Werte enthalten sind, und der zweite DataFrame enthält alles, wo in der Spalte „beruf“ **keine** NaN-Werte enthalten sind. Die beiden DataFrames werden nun nach den Jobtiteln gefiltert und anschließend wieder zusammengefügt.

Danach wird dann die Tabelle geleert und die neuen Daten in die Tabelle geschrieben. Das Leeren der Tabelle erfolgt, weil in der Tabelle nur die aktuellen Stellenanzeigen zur weiteren Verarbeitung enthalten sein sollen. Die weitere Verarbeitung kann beispielsweise ein „Slack-Bot“ sein, der den Studenten dann die Stellenanzeigen postet.

Da der View sich täglich ändert, werden vorher alle Einträge in der Tabelle „daten_aufgearbeitet“ gelöscht und neu befüllt / geschrieben. Mit der View „aufgearbeitete_daten“ ist es dann möglich die Daten für weitere Zwecke zu nutzen. Die View „aufgearbeitete_daten“ enthält Daten aus den Tabellen „Rohdaten“ und „daten_aufgearbeitet“.

Besonderheiten

Allgemein

Einige Angaben sind nicht immer zu extrahieren, da diese

- nicht angegeben sind.
- als iFrame vorhanden sind.
    - Das Extrahieren der Daten aus dem iFrame wurde aus zeitlichen Gründen nicht gemacht.

Bei allen Jobtitel ist die Geschlechterkennzeichnung versucht worden zu entfernen.

Der Scraper bricht ab, wenn nach oder vor der Eingabe von Jobtitel und Suchort eine Benutzeranmeldung gefordert ist.

StepStone

Im oberen Teil der Stellenanzeige stehen die Angaben über Ort, Beschäftigungsverhältnis, etc. Beim Extrahieren der Daten werden alle gefundenen Angaben in einem String ohne ein Trennzeichen hintereinander gesetzt. Dadurch ist eine Aufteilung von Jobtitel und Firma nur möglich, sofern es sich um die Gängigsten unternehmerischen Rechtsformen oder um einschlägige Firmen handelt. Deshalb kommt es vor, dass beim Extrahieren die Spalte „beruf“ mit NaN-Werten aufgefüllt wird. Dann stehen die Berufe zusammen mit dem Firmennahmen in der Spalte „firma“.

Monster

Als Besonderheit bietet Monster eine zuverlässige Erkennung als Bot. Sobald ein Bot erkannt wurde, besteht die Möglichkeit, dass eine Änderung der Klassennamen vorgenommen wird. Deswegen ist es auch erforderlich lange Wartezeiten mit einer großen Range einzubauen.

Bei der Datenextraktion wird in der Anstellungsart und in der auch NaN-Werten eingetragen. Dies geschieht in der Regel dann, wenn die Anstellungsart nicht angegeben ist, oder während des scrapens die Bot-Erkennung gegriffen hat.

Um nicht als Bot erkannt zu werden, ist es notwendig eine Wartezeit mit einer Range von zwei bis zehn Sekunden einzubauen. Dies bietet aber **keine absolute Sicherheit** das die **Bot-Erkennung umgangen** wird.

LinkedIn

Die Bot-Erkennung arbeitet bei LinkedIn zuverlässig. Häufig aber vollkommen zufällig wird der Suchort abgeändert. Deswegen ist es auch erforderlich lange Wartezeiten mit einer großen Range einzubauen. Auch hier gilt, dass es **keine absolute Sicherheit** gibt das die **Bot-Erkennung umgangen** wird.

Indeed

Ebenfalls hat Indeed ein Bot-Erkennung. Bisher ist nur die Eingabe von Jobtitel und Suchort betroffen.