Práctica 1:

Web Scraping de *Infojobs.net*

por

Ander Elkoroaristizabal

Resumen:

En esta práctica he decidido desarrollar un módulo Python con la capacidad de

1. realizar una búsqueda por palabras clave en el portal de empleo `www.Infojobs.net`,
2. permitir al usuario añadir los filtros deseados a la búsqueda en Infojobs,
3. recopilar las ofertas resultado,
4. obtener la información considerada más relevante de las ofertas recopiladas,
5. y devolver la información como *dataset* y como archivo `csv`.

Este *scraper* nos permite por lo tanto a los analistas de datos obtener un *dataset* con las ofertas de cierto tipo de empleo en Infojobs que después podemos analizar.

1. Contexto

Tal y como dicen en este artículo de El País,

> ***www.Infojobs.net*** *es el portal de búsqueda de empleo líder en España y en sus 20 años de experiencia puede presumir de contar con ofertas de trabajo de las empresas más destacadas de España y la mayor bolsa de ofertas.*

Es por lo tanto la página web de referencia sobre ofertas de empleo a nivel estatal, y la opción idónea si queremos obtener esta información.

Cabe decir que Infojobs tiene su propia API, `https://developer.infojobs.net`. A pesar de ello he considerado interesante desarrollar este *scraper*, por dos motivos principales. Por una parte los términos de uso de la API pueden llegar a ser restrictivos dependiendo del uso comercial que quiera dársele a los datos. Por otra parte tener un *scraper* nos permite obtener información que la API no devuelve, como la descripción completa. Es necesario mencionar que haría falta desarrollar más este *scraper* en caso de querer utilizarlo de manera satisfactoria para estos dos casos más ambiciosos.

También es importante comentar que la política respecto a *scraper*s de Infojobs (reflejada en el archivo `robots.txt`[1]) no restringe el acceso ni al buscador ni a las ofertas, los dos directorios web que nosotros accedemos.

2. Título

Como el *scraper* desarrollado obtiene de Infojobs su producto característico, las ofertas de trabajo, he decidido nominarlo *Infojobs Scraper*. Como este *scraper* acepta palabras clave como parámetros y el *dataset* obtenido depende de la fecha y hora en los cuales se ha obtenido, el título que damos a cada *dataset* sigue el siguiente patrón: `{keywords}_{fecha_y_hora}.csv`.

Si bien el *scraper* permite también añadir filtros a la búsqueda he considerado que añadirlos al título era excesivo, por lo que corresponde al usuario llevar la cuenta de los filtros aplicados en cada *dataset* resultado.

[1] Podemos ver una copia del archivo `robots.txt` en el repositorio.

3. Descripción del *dataset*

El *dataset* consiste en, dadas unas palabras clave a buscar y ciertos filtros que se aplican, todas las ofertas del portal Infojobs satisfaciendo estos criterios. Esto es, la *url* de cada oferta junto con las propiedades más importantes.

4. Representación gráfica

He creado la siguiente imagen para representar el *scraper*. En ella tenemos marcadas diferentes ofertas laborales de *Data Scientist*[2] sobre un mapa de España, y una espátula (un *scraper*) rascando el logo de Infojobs:

5. Contenido

A continuación detallamos un poco más el proceso de obtención del *dataset*, cuyo esquema ya hemos explicado en el resumen, y el resultado obtenido.

Primero el programa pide al usuario que introduzca unas palabras clave que describan posiciones laborales, como pueden ser *Data Scientist* o *Big Data Engineer*. Entonces realiza una búsqueda en Infojobs y comienza la recopilación de urls de entre las distintas páginas resultado de la búsqueda. Este el punto más complejo del *scraper*: las páginas resultado del buscador se cargan según te desplazas hacía abajo, por lo que es necesario emular o automatizar este comportamiento. La solución que utilizamos en el *scraper* es usar `Selenium` junto con el *driver* `chromedriver`, que nos permiten controlar el navegador *Chrome* desde Python. Esto genera otro problema: Infojobs reconoce `chromedriver` como robot y nos obliga a resolver un CAPTCHA. En esta versión del *scraper* (no adaptada a fines comerciales) optamos por resolverlo manualmente, tras lo cual no nos dará más problemas. También aprovechamos este parón para aplicar los filtros que queramos de entre lo que Infojobs permite en la ventana de *Chrome* abierta por `Selenium`, que se aplicarán al resultado.

Una vez tenemos la lista de urls la obtención de las características de cada oferta es más simple: basta con utilizar el módulo `urllib` y un `user-agent` que nos identifique como navegador y no como robot para obtener el html de cada oferta y `BeautifulSoup` para obtener las características en si. Cada registro del *dataset* contiene la siguiente información:

- `position`: título del puesto.
- `company`: empresa.
- `company_valuation`: valoración de la empresa según Infojobs.

[2]Provenientes del *dataset* de ejemplo `Data_Scientist_28-03-2021_20_49_31.csv` presente en el repositorio.

- `city`: ciudad donde se ofrece el trabajo.
- `country`: pais donde esta la ciudad, no siempre España.
- `contract_type`: tipo de contrato.
- `salary`: rango salarial.
- `min_exp`: experiencia mínima esperada.
- `url`: url a la oferta.

Cada registro se construye como sigue: primero se obtiene la url de la oferta de alguna de las páginas resultado de la búsqueda y después se analiza el panel titular de la oferta, en el cual la mayoría de propiedades buscadas tienen su propio identificador (`id`) html. En el caso del salario y la experiencia mínima, que no tienen identificador propio, buscamos los términos "Salario" y "Experiencia mínima", siempre presentes en el resultado.

Respecto al periodo de validez de los *datasets* obtenidos, varía según el registro la (oferta): puede ir desde días hasta meses. Por lo tanto el *dataset* en su conjunto no podemos esperar que sea valido por más de unos días.

6. Agradecimientos

Los datos han sido recopilados del dominio web `www.Infojobs.net`, el cual pertenece a la empresa tecnológica Adevinta, especializada en páginas web de búsqueda.

Este *scraper* no está basado en ningún análisis previo de *Infojobs*, si bien sí que ha utilizado los siguientes recursos web:

1. La web `https://builtwith.com/infojobs.net`, puesto que la librería Python `buildwith` daba error `405`.
2. Esta página de StackOverflow, donde explican como hacer *scroll* de una página utilizando `Selenium`.
3. Esta página de StackOverflow, donde dan otra manera de hacer *scrolling* usando el submódulo `Keys` cuando la manera de la página anterior no funciona.
4. Esta página de StackOverflow que he utilizado para convertir las *keywords* dadas en *link* a Infojobs.
5. Esta página de StackOverflow donde explican como obtener la url actual del *driver* de `Selenium`.

A posteriori he encontrado dos repositorios que realizan un *scraping* de Inojobs:

- Este *scraper* hecho por Albert Solà en 2014, escrito PHP, y
- Este módulo desarrollado por Páll Hilmarsson en 2013 y dependiente de un *scraper* también desarrollado por él mismo. Si bien el código sigue colgado en Github, parece que la página web que lo utilizaba ha sido desmantelada.

Podemos ver que ambos *scrapers* son relativamente antiguos, por lo que cabe dudar de que sigan funcionando. Además el *scraper* desarrollado para esta práctica se trata del único disponible públicamente escrito en Python y sin dependencia de otros módulos no comunes.

7. Inspiración

Como profesional del ámbito de los datos me parece especialmente útil disponer de una manera de obtener datos de empleo dentro del sector de forma analizable, que es precisamente lo que este *scraper* facilita. En particular considero útil tener estos datos además guardados de forma local, resumida y reanalizable.

Esta última propiedad es la que hace este *scraper* idóneo para proyectos de minería de datos cuando la API de Infojobs no facilite la información buscada: como Infojobs contiene información de todos los sectores laborales, y el *scraper* devuelve tanto información genérica como la url de cada oferta (útil en caso de querer obtener información más específica de su descripción[3], el *dataset* obtenido es el punto de inicio perfecto para hacer análisis más detallados de cualquier sector o posición particular que tengamos en mente. Por ejemplo se podría restringir la búsqueda del *scraper* al sector *Data*, y estudiar la incidencia de las distintas tecnologías.

Pero dependiendo del nivel de detalle deseado podemos no necesitar siquiera este análisis extra: con un *dataset* resultado podríamos comenzar de inmediato a estudiar por ejemplo la distribución de las ofertas según la experiencia mínima esperada, que resulta especialmente interesante en un país con un 40% de paro juvenil (cifra impulsada por la Covid-19).

Ninguno de los dos *scrapers* mencionados en el apartado anterior explican posibles análisis que quisiesen hacer, por lo que no podemos compararlos.

8. Licencia

Tanto el *scraper* como los *datasets* resultados se distribuyen según la CC BY-NC-SA 4.0 License, puesto que esta **permite tanto compartir el material como adaptarlo**, pero **exige atribución**, y **no permite el uso comercial**.

9. Código

El código fuente de este *scraper* está íntegramente reproducido en este mismo repositorio:

`https://github.com/ander-elkoroaristizabal/InfojobsScraper`

10. Dataset

Algunos datasets obtenidos pueden encontrarse en la carpeta `examples` de este repositorio. Además se ha subido el *dataset* considerado más característico a Zenodo: el correspondiente a las ofertas de *Data Scientist* en Infojobs.net el 28 de marzo de 2021 a las 9. El DOI obtenido es el **10.5281/zenodo.4662621**, utilizando el cual podemos acceder al dataset:

[3]La descripción es un ejemplo de información que la API no devuelve al completo.