Universidad
Internacional
Menéndez Pelayo

Máster Universitario en Investigación en Inteligencia Artificial

Curso 2020-2021

Recuperación y extracción de información, grafos y redes sociales

Práctica Bloque II: Recuperación de información y minería de texto

11 de abril de 2021

Laura Rodríguez Navas
DNI: 43630508Z
e-mail: rodrigueznavas@posgrado.uimp.es

Índice

1. Resumen

En esta práctica se ha implementado un rastreador web (crawler) en Python [1] (ver sección 2), que se complementa con un proceso de agrupamiento, también implementado en Python, de la información extraída de las páginas web que ha recopilado (ver sección 3).

2. Rastreador web (crawler)

En esta sección se describe como se ha implementado el rastreador web (crawler) en Python usando la librería Scrapy. Para empezar con la implementación se debe ejecutar el siguiente comando:

```
$ scrapy startproject books
```

Este comando crea un proyecto Scrapy en el directorio books, siguiendo la estructura por defecto común para todos los proyectos Scrapy, y el fichero *scrapy.cfg* que contiene el nombre del módulo de Python que define la configuración del proyecto books (*books.settings*). El proyecto lo he nombrado books, porqué se rastreará el catálogo de libros que se encuentra en la página web: http://books.toscrape.com.

Una vez se ha creado el proyecto, se definen los ítems de cada libro que se quieren extraer del catálogo. En este caso los ítems que se van a extraer son: el título, la categoría, la descripción, el precio y la valoración de cada libro. Para ello, se tiene que modificar el fichero *books/items.py*, para incluir los cinco ítems que se quieren extraer. Vemos el contenido de *items.py* a continuación:

```
import scrapy


class BooksItem(scrapy.Item):
  # define the fields for your item here like:
  # name = scrapy.Field()
  title = scrapy.Field()
  category = scrapy.Field()
  description = scrapy.Field()
  price = scrapy.Field()
  rating = scrapy.Field()
```

El siguiente paso es describir la manera de extraer la información definida en el fichero *items.py*. Para ello, se utilizarán reglas de expresión XPath y CSS. Por ejemplo, si nos fijamos en el código HTML de uno de los libros que se van rastrear (ver Figura 1), veremos que el título del libro es fácil de extraer con la siguiente regla de expresión CSS: **"h1 ::text"**. Cuando la extracción de información se complica un poco más, se usan reglas de expresión XPath. Por ejemplo, para extraer las descripciones de todos los libros se usará la regla de expresión: **"//div[@id='product_description']/following-sibling::p/text()"**. Una vez, definidas todas las reglas de expresión para cada ítem que se va a rastrear, se crea la araña *books/spiders/books_toscrape.py*.

Las arañas son clases que definen cómo se rastrea una página web determinada (o un grupo de páginas web), incluido cómo realizar el rastreo y cómo extraer la información deseada. En otras palabras, las arañas son el lugar donde se define el comportamiento personalizado para rastrear y analizar las páginas web. En el caso de la práctica, en la araña *books.toscrape* será el lugar donde se definen las reglas de expresión. En las arañas también se tienen que especificar las solicitudes iniciales para rastrear las URLs y una función de devolución de llamada (*parse*) a la que se llamará para generar los ítems de respuesta de esas solicitudes. Por último, los ítems devueltos por las arañas normalmente se conservan en una base de datos o se escriben en un archivo. En el caso de la práctica, en la araña *books.toscrape*, los ítems (título, categoría, descripción, precio y valoración de cada libro) serán guardados en el fichero *books.json*. Esta araña que procesa todas las URLs descubiertas de la

práctica utilizando la función *parse*, que a su vez llama a la función *parse_book_page* donde son definidas todas las reglas de expresión de cómo extraer la información deseada, se muestra a continuación:

```
import scrapy


class BooksToscrapeSpider(scrapy.Spider):
  name = 'books.toscrape'
  allowed_domains = ['books.toscrape.com']
  start_urls = ['http://books.toscrape.com/']

  def parse(self, response):
    for book_url in response.css("article.product_pod > h3 > a ::attr(href)").extract():
      yield scrapy.Request(response.urljoin(book_url), callback=self.parse_book_page)
    next_page = response.css("li.next > a ::attr(href)").extract_first()
    if next_page:
      yield scrapy.Request(response.urljoin(next_page), callback=self.parse)

  @staticmethod
  def parse_book_page(response):
    item = {}
    product = response.css("div.product_main")
    item["title"] = product.css("h1 ::text").extract_first()
    item['category'] = response.xpath("//ul[@class='breadcrumb']/li[@class='active']/
    preceding-sibling::li[1]/a/text()").extract_first()
    item['description'] = response.xpath("//div[@id='product_description']/following-sibling
    ::p/text()").extract_first()
    price = response.xpath('//th[text()="Price (incl. tax)"]/following-sibling::td/text()').
    extract_first()
    item['price'] = price.replace('£', '')
    rating = response.xpath('//*[contains(@class, "star-rating")]/@class').extract_first()
    item['rating'] = rating.replace('star-rating ', '')
    yield item
```

En este momento ya se puede iniciar la araña, pero primero es recomendable modificar el fichero *books/settings.py* para limitar el acceso de la araña al catálogo web, ya que podemos generar un ataque DDoS. Para ello, debemos descomentar la variable DOWNLOAD_DELAY y darle un valor en segundos (p.ej. DOWNLOAD_DELAY = 3).

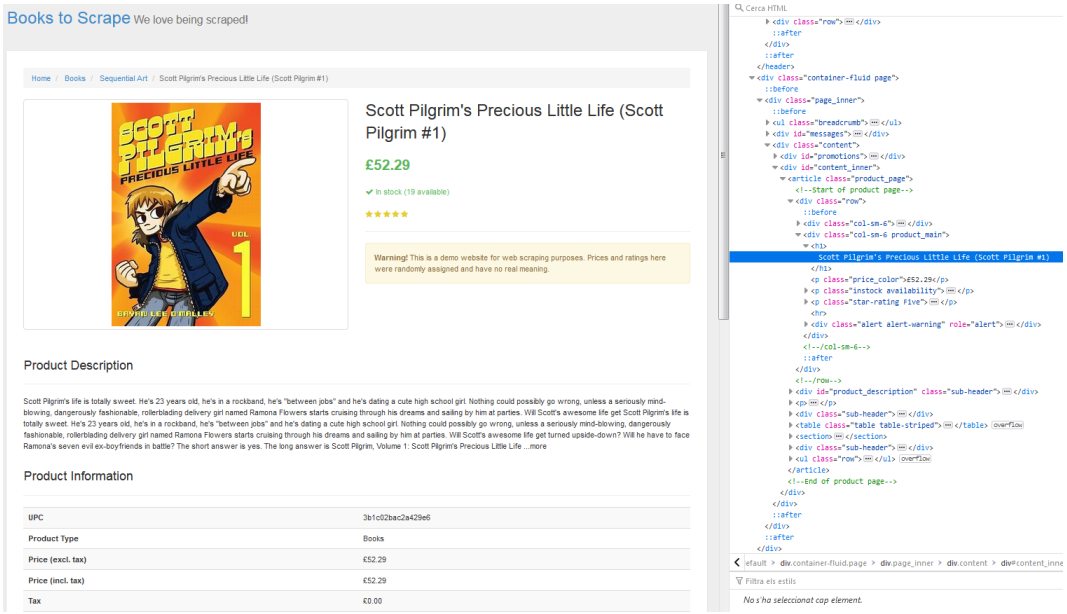

Figura 1: Ejemplo de libro a rastrear.

Finalmente, ya podemos iniciar la araña para que recupere la información del catálogo y la guarde en el fichero *books.json*:

```
$ cd books
$ scrapy crawl books.toscrape -o books.json
```

3. K-means

K-Means es un algoritmo no supervisado de Clustering. Se utiliza cuando tenemos un montón de datos sin etiquetar. El objetivo de este algoritmo es el de encontrar "K" grupos (clústers) entre los datos.

El algoritmo trabaja iterativamente para asignar a cada "punto" (las filas de nuestro conjunto de entrada forman una coordenada) uno de los "K" grupos basado en sus características. Son agrupados en base a la similitud de sus features (las columnas). Como resultado de ejecutar el algoritmo tendremos:

- Los "centroids" de cada grupo que serán unas "coordenadas" de cada uno de los K conjuntos que se utilizarán para poder etiquetar nuevas muestras.
- Etiquetas para el conjunto de datos de entrenamiento. Cada etiqueta perteneciente a uno de los K grupos formados.

Los grupos se van definiendo de manera "orgánica", es decir que se va ajustando su posición en cada iteración del proceso, hasta que converge el algoritmo. Una vez hallados los "centroids" deberemos analizarlos para ver cuales son sus características únicas, frente a la de los otros grupos. Estos grupos son las etiquetas que genera el algoritmo.

Casos de Uso de K-Means

El algoritmo de Clustering K-means es uno de los más usados para encontrar grupos ocultos, o sospechados en teoría sobre un conjunto de datos no etiquetado. Esto puede servir para confirmar -o desterrar- alguna teoría que teníamos asumida de nuestros datos. Y también puede ayudarnos a descubrir relaciones asombrosas entre conjuntos de datos, que de manera manual, no hubiéramos reconocido. Una vez que el algoritmo ha ejecutado y obtenido las etiquetas, será fácil clasificar nuevos valores o muestras entre los grupos obtenidos.

Nuestro caso de uso es -> Categorización de Inventario: agrupar los libros por categorías.

3.1. Datos de Entrada para K-Means

Las "features" o características que utilizaremos como entradas para aplicar el algoritmo k-means deberán ser de valores numéricos, continuos en lo posible. En caso de valores categóricos (por ej. Ciencia Ficción, Terror,etc) se puede intentar pasarlo a valor numérico.

Categorización de Ciencia Ficción, Terror...

4. HOLA

El conjunto de datos contiene diferenciadas 50 categorías - temáticas de libros. Pero alguna de las categorías solo aparece una vez en el conjunto de datos. Así pues, no se considera para el uso del clustering ya que no se podrán formar un grupo de más de un libro. Se eliminan del conjunto de datos.

Las temáticas a eliminar son: ["Academic", "Adult Fiction", "Crime", "Cultural", "Erotica", "Novels", "Paranormal", "Parenting", "Short Stories", "Suspense"]

Una vez eliminadas tenemos

Bibliografía

[1] Laura Rodríguez-Navas. Recuperación de información y minería de texto. https://github.com/lrodrin/masterAI/tree/master/A14, 2021.