Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería Informática

Trabajo Fin de Grado

Análisis y comparacion de paquetes para el desarrollo de web scraping en R

Autor: David Márquez Mínguez

Tutor: Juan José Cuadrado Gallego

2021

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

Trabajo Fin de Grado

Análisis y comparacion de paquetes para el desarrollo de web scraping en R

Autor: David Márquez Mínguez

Tutor: Juan José Cuadrado Gallego

Tribunal:

Presidente: . . . . . .

Vocal 1º: . . . . . .

Vocal 2º: . . . . . .

Fecha de depósito: . . . . de . . . . . de . . . . .

Resumen

Resumen..... correo de contacto: David Márquez Mínguez <david.marquez@edu.uah.es>.

Palabras clave: Trabajo fin de /grado, LaTeX, soporte de español e inglés, hasta cinco....

Abstract

Abstract..... contact email: David Márquez Mínguez <david.marquez@edu.uah.es>.

Keywords: Bachelor final project , LaTeX, English/Spanish support, maximum of five....

Resumen extendido

Con un máximo de cuatro o cinco páginas. Se supone que sólo está definido como obligatorio para los TFGs y PFCs de UAH.

Índice general

Índice de figuras

Índice de tablas

Índice de listados de código fuente

Capítulo 1

Introducción y objetivos

1.1 Contexto

La *World Wide Web* o lo que comúnmente se conoce como la web, es la estructura de datos más grande en la actualidad, y continúa creciendo de forma exponencial. Este gran crecimiento se debe a que el proceso de publicación de dicha información se ha ido facilitando con el tiempo.

Tradicionalmente el proceso de inserción y extracción de la información se realizaba a través del copy-paste. Aunque este método en ocasiones pueda ser la única opción, es una técnica muy ineficiente y poco productiva, pues provoca que el conjunto final de datos no esté bien estructurado. El web scraping o minado web trata precisamente de eso, de automatizar la extracción y almacenamiento de información extraída de un sitio web [1].

La forma en la que se extraen datos de internet puede ser muy diversa, aunque comúnmente se emplea el protocolo HTTP, existen otras formas de extraer datos de una web de forma automática [2]. Este proyecto, se centra en la metodología existente de obtención de información, de como se tratan los datos y la forma en la que se almacenan. Durante los siguientes apartados se realizará una especificación mas concreta del objetivo del proyecto, así como de la estructura y limitaciones del mismo.

1.2 Motivación

El proceso de extracción y recopilación de datos no estructurados en la web es un área interesante en muchos contextos, ya sea para uso científico o personal. En ciencia por ejemplo, los conjuntos de datos se comparten y utilizan por múltiples investigadores, y a menudo también son compartidos públicamente. Dichos conjuntos de datos se proporcionan a través de una API [1] estructurada, pero puede suceder que solo sea posible acceder a ellos a través de formularios de búsqueda y documentos HTML. En el uso personal también ha crecido a medida que han comenzado a surgir servicios que proporcionan a los usuarios herramientas para combinar información de diferentes sitios web en su propias colección de páginas.

Además de ser un ambito interesante, el minado web también es un area muy requerida, algunos de las campos de mayor demanda tienen relacion con la venta minorista, mercado de valores, análisis de las redes sociales, investigaciónes biomédicas, psicológia...

[1] Conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción. [3]

1.3 Objetivo y limitaciones

Existen muchos tipos de técnicas y herramientas para realizar web scraping, desde programas con interfaz gráfica, hasta bibliotecas software de desarrollo. El objetivo de esta tesis es realizar un análisis cuantitativo de las diferentes técnicas y paquetes software para el desarrollo de web scraping.

¿Cuál es la solución más rentable para el minado web? ¿Cuál de las soluciones tiene un mejor rendimiento? Para responder a esta pregunta, se realizará un estudio comparando las diferentes características de los paquetes software, con el objetivo finalmente de poder determinar cuál es el más óptimo en términos de memoria, rendimiento...

En cuanto a las restricciones para el funcionamiento de un scraper, estas pueden ser varias, ya sean legales o por la incapacidad de acceder a una gran parte del contenido no indexado en internet. Aunque el uso de los scrapers está generalmente permitido, en algunos países como en Estados Unidos, las cortes en múltiples ocasiones han reconocido que ciertos usos no deberían estar autorizados [1]. El desarrollo de este proyecto no se verá perjudicado por este tipo de cuestiones, pues solo se limitará al estudio y análisis de los mismos.

1.4 Estructura del documento

Para poder facilitar la composición de la memoria se detalla a continuación la estructura de la misma:

1. **Bloque I: Introducción.**
   - **Capítulo 1: Introducción.**

     En la introducción se especifica tanto el contexto como la motivación a realizar el proyecto, así como las limitaciones esperadas durante la realización del mismo.
2. **Bloque II: Marco teórico.**
   - **Capítulo 2: Web scraping, extracción de datos en la web.**

     Durante este capítulo se explica en que consiste el web scraping, sus posibilidades prácticas y aspectos más generales.
   - **Capítulo 3: Introducción a los paquetes seleccionados y proceso de búsqueda.**

     Se realiza la selección de paquetes y se dictamina cuál ha sido la razón por la que los paquetes han sido seleccionados. Además, se especifican las características principales de cada uno, así como una visión general de sus funcionalidades.
3. **Bloque III: Marco práctico.**
   - **Capítulo 4: Selección de variables de análisis y proceso de estudio.**

     Durante este capítulo se especifica el proceso de análisis a realizar, cuáles son los test a los que los paquetes serán sometidos y que variables se tomarán a estudio para los mismos.
   - **Capítulo 5: Análisis y comparativa de paquetes.**

     Una vez introducidos todos los paquetes y el estudio al que van a ser sometidos, se realizá la comparativa de los mismos. Inicialmente, los paquetes serán analizados uno por uno y finalmente se hará una comparativa con los datos obtenidos.
4. **Bloque IV: Conclusiones y futuras líneas de trabajo.**

Capítulo 2

Web scraping, extracción de datos en la web

2.1 En que consiste realmente el web scraping

En la actualidad el web scraping se puede definir como una *"solución tecnológica para extraer información de sitios web, de forma rápida, eficiente y automática, ofreciendo datos en un formato más estructurado y más fácil de usar [4]"*. Sin embargo, esta definición no ha sido siempre así, los métodos de minado web han evolucionado desde procedimientos más pequeños con ayuda humana, hasta sistemas automatizados capaces de convertir sitios web completos en conjuntos de datos bien organizados.

Existen diferentes herramientas de minado, no solo capaces de analizar los lenguajes de marcado o archivos JSON, también capaces de realizar un procesamiento del lenguaje natural para simular cómo los usuarios navegan por el contenido web.

2.1.1 Extracción de los datos

En realidad el minado web es una práctica muy sencilla, se extraen datos de la web y se almacenan en una estructura de datos para su posterior análisis o recuperación. En este proceso, un agente de software, también conocido como robot, imita la navegación humana convencional y paso a paso accede a tantos sitios web como sea necesario [5]. Las fases por las que pasa el agente software en cuestión se determinan a continuación:

1. **Fase de búsqueda:** Esta primera fase consiste en el acceso al sitio web del que se quiere obtener la información. El acceso se proporciona realizando una solicitud de comunicación HTTP. Una vez establecida la comunicación, la información se gestiona a partir de los métodos GET y POST usuales.

2. **Fase de extracción:** Una vez que el acceso ha sido permitido, es posible obtener la información de interés. Se suelen emplear para este propósito expresiones regulares o librerías de análisis HTML. Los diferentes softwares empleados para este propósito se especifican en la sección 2.1.2.

3. **Fase de transformación:** El objetivo de esta fase es transformar toda la información extraída en un conjunto estructurado, para una posible extracción o análisis posterior. Los tipos de estructuras más comunes en este caso son soluciones basadas en cadenas de texto o archivos CSV y XML.

Una vez que el contenido ha sido extraído y transformado en un conjunto ordenado, es posible realizar un análisis de la información de una forma más eficaz y sencilla que aplicando el método tradicional. El proceso descrito se resume en la ilustración 2.1.

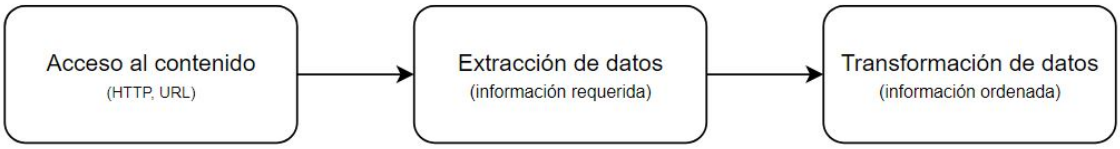

Figura 2.1: Fases del web scraping

Por otro lado, a lo largo del apéndice A, se ilustra el funcionamiento del agente software en cada una de las fases descritas anteriormente, así como de su comportamiento con el servidor web al que se desea acceder.

2.1.2 Herramientas software disponibles

El software disponible que emplean los web scrapers puede dividirse en varios enfoques, ya sean bibliotecas de programación de propósito general, frameworks, o entornos de escritorio.

Por lo general tanto los frameworks como los entornos de escritorio presentan una solución más sencilla e integradora con respecto a bibliotecas de programación. Esto es debido a que ambos, no se ven afectados a los posibles cambios HTML de los recursos a los que accede. Además, estas necesitan la integración de otras múltiples bibliotecas adicionales para el acceso, análisis y extracción del contenido.

Este trabajo se desarrolla sobre bibliotecas de programación, las cuales se implementan como un programa software convencional utilizando estructuras de control y de datos del propio lenguaje. Por lo general, bibliotecas como *curl [6]* conceden acceso al sitio web deseado haciendo uso del protocolo HTTP, mientras que los contenidos extraídos se analizan a través de funciones como la coincidencia de expresiones regulares y la tokenización.

Comprender como las bibliotecas obtienen los datos de los sitios web, pasa por conocer las diferentes formas en las que los documentos HTML se organizan. Existen dos técnicas, dependiendo si se realiza un renderizado previo o no [7]. La primera técnica consiste simplemente en parsear, es decir, realizar un análisis léxico-sintáctico sobre estructuras XML o HTML. Se suele emplear XPath o selectores CSS para su realización. Por otro, lado si es necesario que parte de la lógica del sitio web pase al lado del cliente, este deberá pasar por un proceso de renderizado previo.

Con el paso del tiempo, cada vez se extiende más el uso de bibliotecas de desarrollo como JQuery, encargadas de pasan parte de la lógica del lado del servidor al lado del cliente con el objetivo de favorecer la interactividad. Estas páginas no podrán ser analizadas si no se renderizan antes.

2.1.2.1 XPath

XPath es un lenguaje que permite construir expresiones que recorren y procesan un documento XML [8]. Puede utilizarse para navegar por documentos HTML, ya que este es un lenguaje similar en cuanto a estructura a XML. *XPath* es comúnmente utilizado en el web scraping para extraer datos de documentos HTML, además utiliza la misma notación empleada en las URL para navegar por la estructura del sitio web en cuestión.

2.1.2.2 Selectores CSS

El segundo método de extracción de datos en documento HTML se realiza a través de lo que se conoce como selectores CSS [9]. CSS es el lenguaje utilizado para dar estilo a los documentos HTML, por otro lado, los selectores son patrones que se utilizan para hacer coincidir y extraer elementos HTML basados en sus propiedades CSS.

Hay múltiples sintaxis de selector diferentes, estas se corresponden con la forma en la que el documento CSS está estructurado. En el fragmento de código A.3 se hace uso de los selectores *'.text-primary'* y *'.lister-item-header a'* para acceder al contenido web deseado.

2.1.3 Tipologias

Dependiendo de como se acceda y extraiga la información, existen dos técnicas de web scraping. Se mencionan los siguientes supuestos a continuación:

- Si la información que se almacena no procede de sitios web concretos, sino que durante el análisis de páginas web se encuentran enlaces que retroalimentan el análisis de otras nuevas, el método se conoce como web crawling [10].
- Si la información se extrae de sitios web concretos, donde ya se conoce como extraer y generar un valor por la información, la técnica se conoce como web scraping genérico. Mientras que en el web crawling el resultado de ejecución es la obtención de nuevas páginas, en el web scraping el resultado es la propia información.

Es decir, la principal diferencia entre ambas, es que mientras los web scrapers extraen información de páginas webs concretas, los web crawlers almacenan y acceden a las páginas a través de los enlaces contenidos en las mismas. En la figura 2.1 se mostraba la arquitectura en fases de un web scraper, veamos a continuación como es la arquitectura de un web crawler.

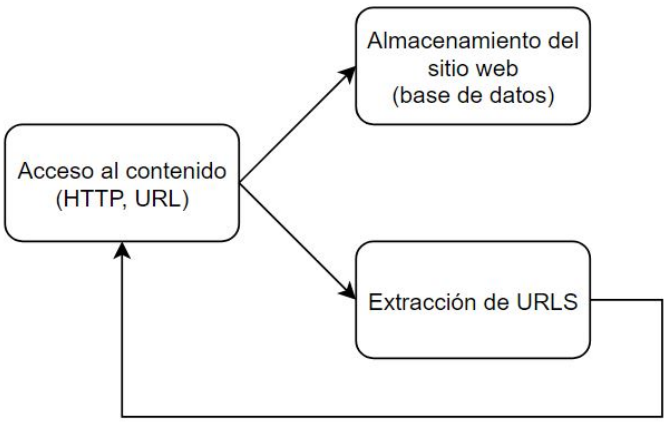

Figura 2.2: Fases del web crawling

Ya sea empleando cualquiera de estas dos tipologías, existen páginas que no pueden ser analizadas o rastreadas. Esto es debido a que algunos sitios web solo están disponibles con una autorización previa, o necesitan información especial para su acceso.

2.2 Posibilidades prácticas del web scraping

Son muchas las aplicaciones prácticas de la minería web, la mayoría de estas entran en el ámbito de la ciencia de los datos. En la siguiente lista se exponen algunos casos de su uso en la vida real [11]:

- Bancos y otras instituciones financieras utilizan minería web para analizar a la competencia. Inversores también utilizan web scraping, para hacer un seguimiento de artículos de prensa relacionados con los activos de su cartera.
- En las redes sociales se emplea minería de datos para conocer la opinión de la gente a cerca de un determinado tema.
- Existen aplicaciones capaces de analizar diferentes sitios web y encontrar los mismos productos a precio reducido. Incluso capaces de detectar ofertas de artículos a tiempo récord.
- etcétera

El minado web contiene infinidad de aplicaciones, muy diversas e interesante capaces de automatizar el trabajo y conseguir la información de forma ordenada.

No obstante, muchos sitios web ofrecen alternativas como el uso de APIs o ficheros estructuras para el acceso a dichos datos. En general el web scraping es una técnica que consume bastantes recursos, por lo que el desarrollador debe limitar su uso si existen otras alternativas, como las APIs, que proporcionan los mismos resultados.

2.3 Retos del web scraping

La forma en la que se crean los sistemas de extracción web se ha discutido desde diferentes perspectivas a lo largo del tiempo. Se emplean métodos científicos como el aprendizaje automático, la lógica o el procesamiento del lenguaje natural para lograr su implementación.

Uno de los principales retos a afrontar tiene relación con las fuentes cambiantes de información. A menudo, una herramienta de extracción tiene que obtener datos de forma rutinaria de una determinada página o sitio web que puede evolucionar con el tiempo. Estos cambios se producen sin previo aviso, son imprevisibles, por lo que es bastante probable que los raspadores web se vean sometidos a cambios. Por ello, surge la necesidad crear herramientas flexibles capaces de detectar y afrontar modificaciones estructurales de las fuentes web relacionadas.

Otros problemas recaen sobre la información extraída. En primer lugar, uno de los aspectos que se deben tener en cuenta al obtener información trata sobre la fiabilidad de la misma. Aunque la información exista y se pueda ser analizada, esta puede que no sea correcta. La gramática y la ortografía pueden ser un problema en la fase de análisis, ya que información puede perderse o ser falsamente recogida. Por otro lado, tanto aplicaciones que tratan con datos personales, como software de minado deben ofrecer garantías de privacidad. Por lo tanto, los posibles intentos de violar la privacidad del usuario deben ser identificados y contrarrestados a tiempo y de forma adecuada.

Puesto que multitud de técnicas de minería web requieren ayuda humana, un primer reto consiste en proporcionar un alto grado de automatización, reduciendo así al máximo el esfuerzo humano. Sin embargo, la ayuda humana puede desempeñar un papel importante a la hora de elevar el nivel de precisión alcanzado por un sistema de extracción de datos web, por ello la clave está en encontrar un equilibrio entre automatización e intervención humana.

Por último, a pesar de que las herramientas de web scraping han evolucionado con el tiempo, los aspectos legales están algo inexplorados pues dependen de los términos y condiciones de cada sitio web en cuestión.

2.4 Aspectos ético-legales del web scraping

Para comprender los aspectos legales del web scraping, debemos recordar el robot o agente software definido en el apartado 2.1.1. Este agente software, previamente examinado por el servidor, es el que se encarga de acceder y realizar un recorrido por el contenido web.

Durante el acceso al contenido, se espera que este agente se ajuste a los términos de uso del sitio en cuestión, así como el cumplimiento del archivo *'robots.txt'* [1], con el objetivo de evitar accesos no deseados y sobrecargas en el servidor.

Puesto que el documento *'robots.txt'* no es de obligado cumplimiento, a lo largo de los años la reputación del web scraping ha decrecido de forma significativa. Muchos agentes software no siguen las indicaciones determinadas, por lo que definir la cantidad de accesos y archivos a los que se accede dependerá de la ética de cada desarrollador.

Con el objetivo de tener una cierta garantía de que nuestro agente software cumple con los aspectos ético-legales, se deben tener en consideración las siguientes cuestiones [13]:

- Leer los términos de uso de la página web en al que se vaya a realizar el minado.
- Inspeccionar y cumplir con el documento robots.txt, para ser capaces de identificar los accesos del servidor.
- Realizar peticiones al servidor de forma controlada. Puede que el índice de solicitudes al servidor no este especificado en el documento, si esto sucede debemos determinar un número de solicitudes razonable, por ejemplo, una solicitud por segundo.

[1] Archivo alojado en el servidor web, que gestiona el tráfico del mismo e indica los documentos a los que no se debe acceder de forma automática [12].

Capítulo 3

Paquetes orientados al minado web en Python

Se realiza tanto un proceso de búsqueda, como un proceso de selección de herramientas capaces de realizar web scraping. Como ya se mencionó en la sección 2.1.2 este trabajo se centra sobre bibliotecas de programación, por lo que tanto frameworks como entornos de escritorio quedaran exentos análisis y posteriores test. Con el fin de encontrar la mayor cantidad de paquetes posibles se opta por emplear los siguientes repositorios:

- GitHub.
- PyPI.
- Libreria estandar de Python.

A lo largo de las siguientes secciones se mostrarán las librerías de código abierto más interesantes para entre propósito, además se dará una breve explicación del funcionamiento y peculiaridades de las mismas.

3.1 Proceso de búsqueda y selección de paquetes orientados al web scraping

3.2 Paquetes encontrados durante el proceso de búsqueda

3.3 Paquetes seleccionados para el proceso de análisis

Capítulo 4

Selección de variables de análisis y proceso de estudio

Capítulo 5

Presupuesto

Blah, blah, blah.

Bibliografía

[1] Wikipedia, "Web scraping," https://es.wikipedia.org/wiki/Web_scraping. [Ultimo acceso 18/octubre/2021].

[2] B. Zhao, *Web Scraping*, 05 2017, pp. 1–3.

[3] Wikipedia, "Interfaz de programación de aplicaciones," https://en.wikipedia.org/wiki/API. [Ultimo acceso 18/octubre/2021].

[4] O. Castrillo-Fernández, *Web Scraping: Applications and Tools*, ser. Report No. 2015 / 10. European Public Sector Information Platform, 2015.

[5] D. Glez-Peña, A. Lourenco, H. López-Fernández, M. Reboiro-Jato, and F. Fdez-Riverola, "Web scraping technologies in an API world," *Briefings in Bioinformatics*, vol. 15, no. 5, pp. 788–797, April 2013.

[6] CRAN, "curl: A modern and flexible web client for r," https://cran.r-project.org/web/packages/curl/index.html. [Ultimo acceso 26/octubre/2021].

[7] D. Francisco López, "Revisión de los paquetes para realizar web scraping en r: Análisis cualitativo y cuantitativo," Master's thesis, Universidad de Alcalá Escuela Politécnica Superior, 2018.

[8] Wikipedia, "Xpath," https://es.wikipedia.org/wiki/XPath. [Ultimo acceso 04/noviembre/2021].

[9] W3Schools, "Css selector reference," https://www.w3schools.com/cssref/css_selectors.asp. [Ultimo acceso 04/noviembre/2021].

[10] Wikipedia, "Web crawler," https://en.wikipedia.org/wiki/Web_crawler. [Ultimo acceso 02/noviembre/2021].

[11] S. Broucke Vande and B. Baesens, *Web Scraping for Data Science with Python*, 11 2017, pp. 14–16.

[12] Google, "Introducción a los archivos robots.txt," https://developers.google.com/search/docs/advanced/robots/intro?hl=es. [Ultimo acceso 27/octubre/2021].

[13] M. Beckman, S. Guerrier, J. Lee, R. Molinari, S. Orso, and I. Rudnytskyi, "An introduction to statistical programming methods with r, chapter 10," https://smac-group.github.io/ds/index.html. [Ultimo acceso 27/octubre/2021].

[14] CRAN, "robotstxt: A 'robots.txt' parser and 'webbot'/'spider'/'crawler' permissions checker," https://cran.r-project.org/web/packages/robotstxt/index.html. [Ultimo acceso 28/octubre/2021].

[15] ——, "xml2: Parse xml," https://cran.r-project.org/web/packages/xml2/index.html. [Ultimo acceso 29/octubre/2021].

[16] ——, “rvest: Easily harvest (scrape) web pages,” https://cran.r-project.org/web/packages/rvest/index.html. [Ultimo acceso 29/octubre/2021].

[17] ——, “magrittr: A forward-pipe operator for r,” https://cran.r-project.org/web/packages/magrittr/index.html. [Ultimo acceso 29/octubre/2021].

Apéndice A

Funcionamiento básico de un web scraper

Siguiendo las directrices determinadas en la sección 2.1.1, se muestra el funcionamiento de un web scraper durante las tres fases definidas. Para ello se realizará un pequeño ejemplo mostrando el comportamiento del mismo y de como interactúa con el servidor web al que se desea acceder.

Para el desarrollo de este ejemplo se han empleado bibliotecas software basadas en el lenguaje de programación R, capaces de extraer datos de cualquier web. En este caso la web sujeta al análisis será, *imdb* encargada de asignar un ranking entre películas y series.

A.1 Fase de búsqueda

Antes de comenzar con la primera de las tres etapas, debemos asegurarnos que nuestro agente software cumple con todos los aspectos ético-legales descritos en la sección 2.4. Los términos y servicios de la página deberán ser leídos, al igual que el documento *'robots.txt'* con el objetivo de conocer cuáles son los accesos disponibles y el índice de solicitudes a realizar.

En el fragmento de código A.1 se muestra la solicitud al servidor realizada. A través de la biblioteca *robotstxt* [14] y haciendo uso de la función *get_robotstxt* se obtiene el documento deseado.

Es posible que la solicitud del documento no se realice correctamente, pues o bien la página no dispone del documento en ese instante, o la función ha fallado durante su solicitud. En cualquiera de estos dos escenarios, es posible realizar una doble comprobación accediendo al mismo a través de la propia URL, *https://www.imdb.com/robots.txt*.

Listado A.1: Solicitud del documento *robots.txt*

```
install.packages("robotstxt")
library("robotstxt")

robots <- get_robotstxt(domain = "https://www.imdb.com/")

head(robots)
```

```
# robots.txt for https://www.imdb.com properties
User-agent: *
Disallow: /OnThisDay
Disallow: /ads/
Disallow: /ap/
Disallow: /mymovies/
Disallow: /r/
Disallow: /register
Disallow: /registration/
...
```

Una vez se ha accedido al documento *'robots.txt'*, conociendo los posibles accesos al servidor, y determinando el número de solicitudes máximas por segundo, es posible acceder y descargar el fichero HTML de forma segura. Para este propósito se empleará la función *read_html* de la biblioteca *xml2* [15] la cual se detalla a continuación.

Listado A.2: Acceso y descarga del archivo HTML

```
install.packages("xml2")

library("xml2")

url <- "imdb.com/search/title/?count=100&release_date=2016,2016&title_type=feature"
html_doc <- read_html(url)
```

El valor que retorna la función *read_html*, se trata del archivo HTML integro, donde se incluyen cabecera, cuerpo y demás etiquetas del mismo. Una vez que se dispone de la página, es posible comenzar con la extracción de datos de interés de la misma.

A.2 Fase de extracción

Para el minado se empleará *rvest* [16], una de las bibliotecas software más comunes en este aspecto, diseñada para trabajar con *magrittr* [17] y facilitar tareas de la extracción. Además, será necesario el uso de funciones como *html_nodes()* y *html_text()*.

Durante esta fase de extracción se obtendrán tanto los títulos como el ranking asignado a cada película o serie. Para realizar la extracción de forma correcta, se deberá conocer la etiqueta HTML que envuelve dicha información.

Listado A.3: Extracción de datos de interés del documento

```
install.packages("rvest")
install.packages("magrittr")

library("rvest")
library("magrittr")

rank_data <- html_nodes(html_doc, '.text-primary') %>%
             html_text()

title_data <- html_nodes(html_doc, '.lister-item-header a') %>%
              html_text()

head(rank_data)
head(title_data)
```

```
[1] "1." "2." "3." "4." "5." "6."
```

```
[1] "Animales nocturnos"   "Train to Busan"       "La llegada (Arrival)"
[4] "Escuadrón suicida"    "Deadpool"             "Hush (Silencio)"
```

A.3 Fase de transformación

Una vez los datos han sido extraídos, la última fase consiste en la transformación de los mismos. Los datos desordenados deberán ser convertidos en estructuras de datos ordenadas y coherentes con el fin su posible almacenamiento en una base de datos.

Listado A.4: Transformación de datos en un data frame

```
movies_df <- data.frame(Rank = rank_data, Title = title_data)

head(movies_df)
```

```
  Rank                Title
1   1.   Animales nocturnos
2   2.       Train to Busan
3   3. La llegada (Arrival)
4   4.    Escuadrón suicida
5   5.             Deadpool
6   6.      Hush (Silencio)
```

Una vez los datos han sido ordenados en una estructura de datos propia, en este caso un data frame, es posible trabajar con ellos de forma más cómoda y sencilla.

Universidad de Alcalá
Escuela Politécnica Superior

Universidad
de Alcalá