Tipología y ciclo de vida de los datos

PRÁCTICA 1 (25 % nota final)

Aleksandar Rasevic Lukic

Adrian Läufer Nicolás

Webscraping

1. **Contexto.** Explicar en qué contexto se ha recolectado la información. Explicar por qué el sitio web elegido proporciona dicha información. Indicar la dirección del sitio web.

A la hora de comprar una casa o un piso como inversión inmobiliaria el proceso de búsqueda se convierte en un proceso repetitivo donde se accede a los portales inmobiliarios con tal de encontrar la opción más oportuna. Durante este proceso el posible comprador debe de comparar factores como el precio por m^2 o la posible rentabilidad de la vivienda en función del precio de compra entre otros datos que no están directamente proporcionados por los portales inmobiliarios y que en cualquier caso deberían de comprobarse a la hora de comprar una vivienda como inversión. Además, resulta indispensable realizar comparaciones entre diferentes portales con tal de detectar posibles cambios en las ofertas de los inmuebles.

Por otra parte, si uno accede a un portal inmobiliario con otro objetivo al de comprar una vivienda, como podría ser para estimar cuánto vale su propio inmueble, la obtención de información puede ser aún más laboriosa ya que para ello tan sólo se pueden emplear los filtros de búsqueda y así seleccionar viviendas similares para realizar comparaciones. En cambio, si los datos se pudieran obtener directamente de la web, éstos podrían ser utilizados para aplicar algoritmos de *machine learning,* como por ejemplo métodos supervisados con el objetivo de realizar una predicción del valor de nuestro inmueble.

Por estos motivos, nos ha resultado interesante aplicar la técnica del *web scraping* a algún portal inmobiliario con el objetivo de obtener la mayor información disponible de las viviendas en cierta localidad y así poder realizar más fácilmente comparaciones.

En España existe una gran cantidad de portales inmobiliarios que publican viviendas en alquiler o en venta. La mayoría de ellos publican los mismos anuncios que son proporcionados por las inmobiliarias, pero pueden existir ligeras diferencias entre los portales ya que no todas las inmobiliarias publican sus anuncios en todos los portales posibles. En la siguiente tabla se muestran los principales portales inmobiliarios ordenados según las visitas obtenidas durante un año.

Portal	Número de visitas
Idealista	50,2 millones
Fotocasa	13,5 millones
Habitaclia	9 millones
Pisos.com	6,7 millones
Yaencontre	4,1 millones
Tucasa.com	1,8 millones
ThinkSpain	1,1 millones
Trovimap	445.000
Globaliza	282.300

Tabla 1: Número de visitas webs en cada uno de los portales inmobiliarios más conocidos en España (Fuente: Similarweb.com)

Tal y como se puede ver de la tabla, los principales portales en cuanto a visitas son el idealista o y fotocasa. El principal motivo es que éstos portales son los que publican más anuncios de viviendas en alquiler o en venta. Mientras que los portales el idealista o fotocasa tienen alrededor de 1,5 Mio de anuncios, otros portales menos famosos como pisos.com o yaencontré.com tienen alrededor de entre 800.000 y 500.000 anuncios.

Al ser los portales de mayor importancia, el web scraping se realizó a las webs www.elidealista.com y www.fotocasa.com.

El dataset creado a partir del web scraping se centra en una localidad. Con el objetivo de testear el éxito del web scraping realizado y obtener un dataset con cierto número de registros, se consideró realizar el web scraping para los inmuebles en venta en la localidad de Vilanova i la Geltrú. Por lo tanto, las webs scrapedas han sido las siguientes:

- https://www.fotocasa.es/es/comprar/viviendas/vilanova-i-la-geltru/todas-las-zonas/l
- https://www.idealista.com/venta-viviendas/vilanova-i-la-geltru-barcelona/

Esta ciudad tiene una población de cerca de 70.000 habitantes y alrededor de 1.000 inmuebles en venta según los dos portales donde se ha realizado el scrapeado.

2. **Título.** Definir un título conciso y que sea descriptivo para el dataset.

"Inmuebles en venta en Vilanova i la Geltrú"

3. **Descripción del dataset.** Desarrollar una breve descripción del conjunto de datos que se ha extraído. Es necesario que esta descripción sea coherente con el título elegido.

El dataset obtenido contiene las principales características, es decir número de habitaciones, metros cuadrados, plaza de parking,... de las viviendas actualmente en venta en la localidad de Vilanova i la Geltrú y publicadas en los portales inmobiliarios de elidealista.com y fotocasa.es.

4. **Representación gráfica.** Dibujar un esquema o diagrama que refleje visualmente el dataset y el proyecto elegido.

El siguiente diagrama representa gráficamente el proceso seguido para realizar el proyecto. Con el objetivo final de obtener el dataset se siguieron los siguientes pasos:

1. **Web Crawling:** Se inspeccionaron varias páginas webs de donde se podía obtener los datos deseados y descritos en el apartado 1.

2. **Web Scrapping:** Una vez seleccionadas las páginas webs de donde extraer la información, se realizó un web scrapping de donde se extrajo la información deseada a partir de identificar la estructura de cada web. Para realizar esta tarea se aplicó el lenguaje de programación python y utilizamos el entorno de desarrollo selenium. Mediante él se realizaron diferentes peticiones html para poder extraer los datos de diferentes páginas webs.

3. **Data set:** Por último, una vez extraída la información necesaria de cada web, se procedió a guardar los datos en un archivo csv.

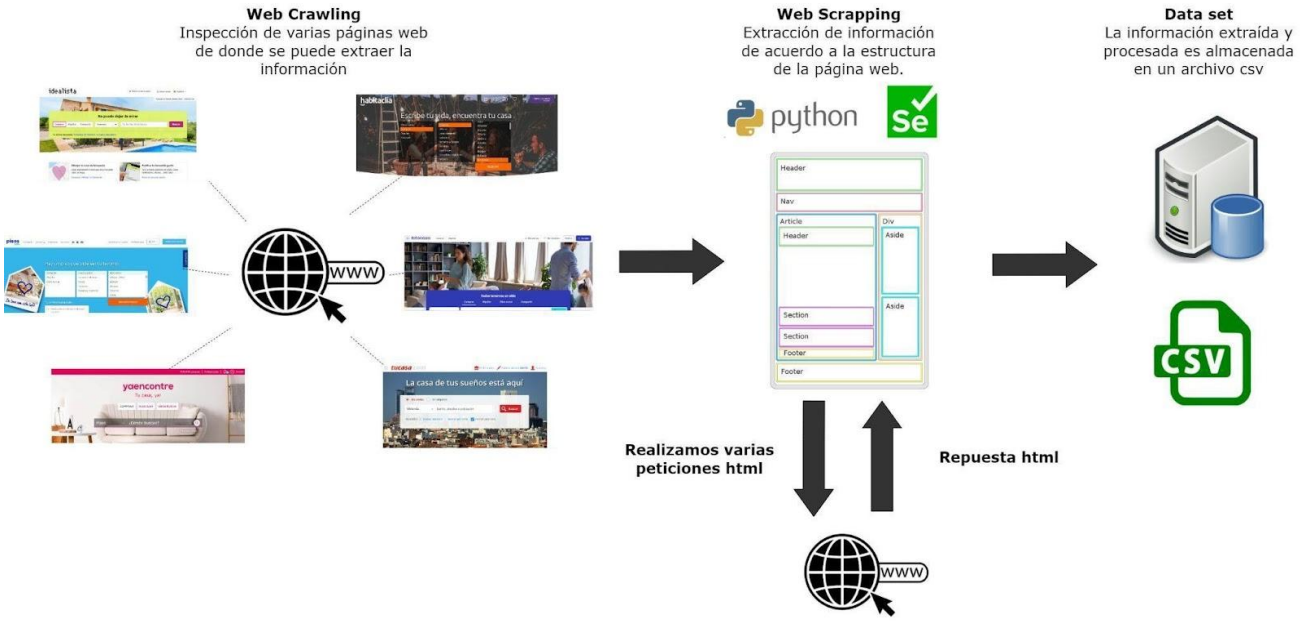

Ilustración 1: Diagrama que representa el proceso seguido en el proyecto para obtener los datos

A continuación, se muestra una representación gráfica del dataset proporcionada por los portales inmobiliarios.

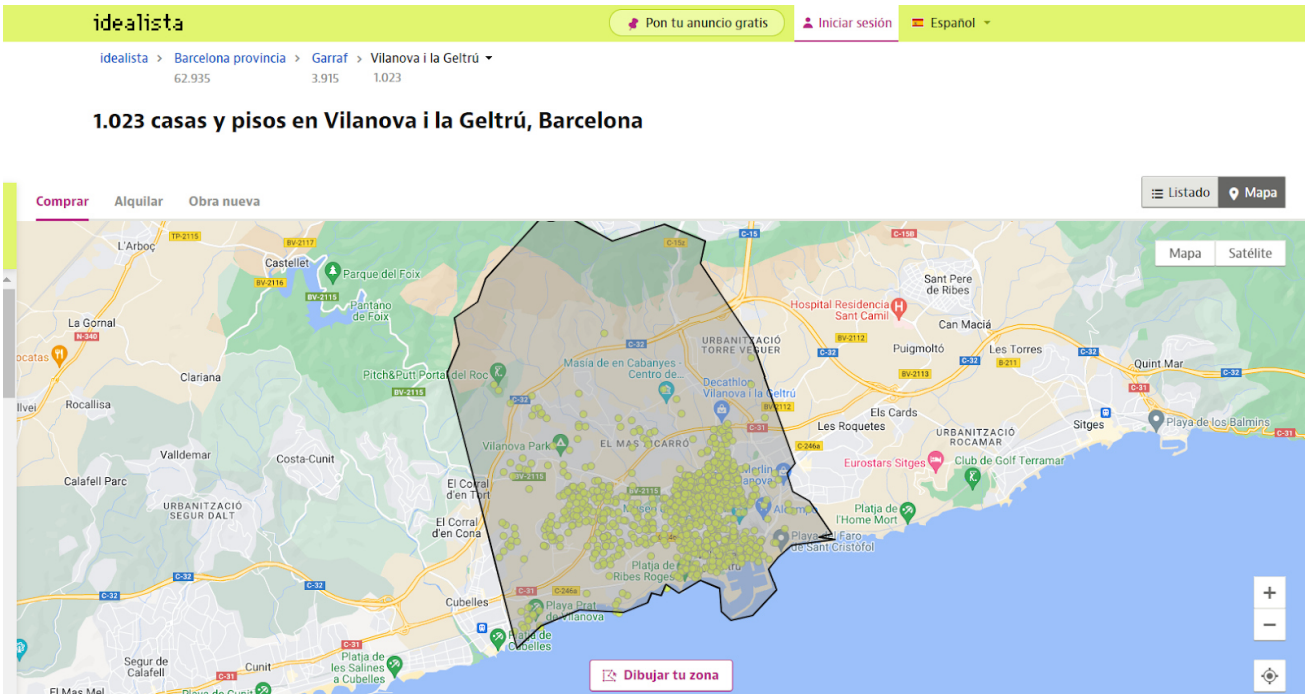

Ilustración 2: Representación gráfica de los pisos en venta en Vilanova i la Geltrú según el portal elidealista.es

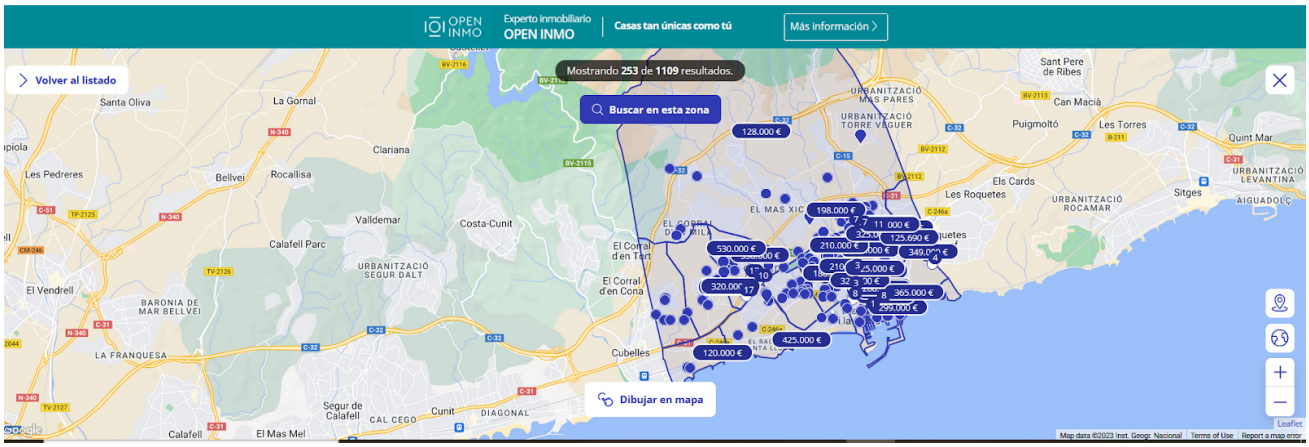

Ilustración 3: Representación gráfica de los pisos en venta en Vilanova i la Geltrú según el portal Fotocasa.es

5. **Contenido.** Explicar los campos que se incluyen en el dataset y el período de tiempo al que pertenecen los datos.

El dataset publicado consiste en la concatenación de los resultados del web scraping de dos portales de compraventa de propiedades diferentes: Fotocasa e Idealista. A continuación se listan los campos en el orden en el que aparecen en el fichero csv subido al sitio web Zenodo, y de cada uno de ellos se describe el tipo de datos esperado y su interpretación.

Es importante destacar que de acuerdo a las instrucciones del enunciado las tareas de limpieza de datos han sido pospuestas hasta la PRA2; por tanto, el tipo de datos indicado será el deseado como resultado de las operaciones de preprocesado, mientras que en el dataset el formato predominante son las cadenas de texto.

- Name: cadena de texto. Representa una breve descripción de la propiedad en venta.
- Price: integer. Representa el precio en euros de venta de la propiedad.
- Rooms: integer. Representa el número de habitaciones de la propiedad.
- M2: integer. Representa los metros cuadrados de la propiedad en venta.
- Parking: booleano. Verdadero si la propiedad en venta dispone de plaza de párking; falso en caso contrario o la información no ha podido ser localizada.
- Link: cadena de texto. Hiperenlace hacia la URL de la propiedad específica en el portal en cuestión.
- Source: cadena de texto. Representa la fuente original de la información. Puede tomar los valores 'idealista' o 'fotocasa'.

Es oportuno mencionar que una de las restricciones encontradas para la compilación de este conjunto de datos ha sido la de encontrar variables que se encontrasen simultáneamente en los dos conjuntos de datos para que pudieran ser concatenadas. Así, no consideramos otras variables que sólo hemos podido extraer de uno de los dos portales, como la existencia de aire acondicionado,

ascensor, el tipo de calefacción o el número de extras, todo ello información exclusiva de Fotocasa. Finalmente, respecto al periodo de recolección de datos los datos de Idealista se obtuvieron el 19/04/23, mientras que la versión final de los datos de Fotocasa fue obtenida el 25/04/23.

6. **Propietario.** Presentar al propietario del conjunto de datos. Es necesario incluir citas de análisis anteriores o, en su defecto, justificar esta búsqueda con análisis similares. Indicar qué pasos se han seguido para actuar de acuerdo con los principios éticos y legales en el contexto del proyecto elegido.

Para localizar a los propietarios de los datos presentados en esta memoria y publicados en Zenodo, podemos utilizar la librería python-whois de Python, tal y como se indica en el punto 1.2.1. "Evaluación inicial" del documento "Web scraping" publicado en la asignatura. Dado que en nuestro caso scrapeamos dos sitios web diferentes, tendremos dos respuestas propietarios diferentes.

Para los sitios web "https://www.fotocasa.es" y "https://www.fotocasa.es/es/" o variaciones de estas dos cadenas de texto sólo se obtienen respuestas nulas procedentes de la librería whois. Sin embargo, la URL "https://www.fotocasa.com" sí que devuelve resultados, indicando que las otras dos son alias de esta última. Mostramos a continuación el fichero json obtenido:


```
{
 "domain_name": [
   "FOTOCASA.COM",
   "fotocasa.com"
 ],
 "registrar": "Amazon Registrar, Inc.",
 "whois_server": "whois.registrar.amazon.com",
 "referral_url": null,
 "updated_date": "2023-03-12 01:28:43",
 "creation_date": "1999-04-15 04:00:00",
 "expiration_date": "2024-04-15 04:00:00",
 "name_servers": [
   "NS-1436.AWSDNS-51.ORG",
   "NS-1605.AWSDNS-08.CO.UK",
   "NS-224.AWSDNS-28.COM",
   "NS-888.AWSDNS-47.NET"
 ],
 "status": [
   "clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited",
   "clientTransferProhibited https://icann.org/epp#clientTransferProhibited",
   "clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited"
 ],
 "emails": [
   "abuse@amazonaws.com",
   "64896376-acfc-4d29-849e-96e31f45b32a@identity-protect.org"
```

```
  ],
  "dnssec": "unsigned",
  "name": "On behalf of fotocasa.com owner",
  "org": "Identity Protection Service",
  "address": "PO Box 786",
  "city": "Hayes",
  "state": "Middlesex",
  "registrant_postal_code": "UB3 9TR",
  "country": "GB"
}
```


Destacamos algunos datos interesantes: por ejemplo, que la creación del sitio aparece indicada como 1999-04-15, o que el país de registro es Gran Bretaña, condado de Middlesex, ciudad de Hayes y que se encuentra hosteado en AWS. A partir de esta respuesta, podríamos decir que el nombre con el que se identifica al propietario del sitio web es "On behalf of fotocasa.com owner", aunque esto sea probablemente una línea generada por algún servicio web de plataforma.

Respecto a la segunda URL bajo investigación, "https://www.idealista.com", whois obtiene la repsuesta siguiente:


```
{
  "domain_name": "IDEALISTA.COM",
  "registrar": "DonDominio (SCIP)",
  "whois_server": "whois.scip.es",
  "referral_url": null,
  "updated_date": [
    "2021-07-20 08:19:18",
    "2021-07-20 10:20:31"
  ],
  "creation_date": "1999-08-21 21:28:24",
  "expiration_date": "2023-07-15 15:55:10",
  "name_servers": [
    "ANYCAST1.IRONDNS.NET",
    "ANYCAST2.IRONDNS.NET",
    "SEC1.RCODE0.NET",
    "SEC2.RCODE0.NET"
  ],
  "status": [
    "clientTransferProhibited https://icann.org/epp#clientTransferProhibited",
    "ok http://www.icann.org/epp#ok",
    "clientTransferProhibited http://www.icann.org/epp#clientTransferProhibited"
  ],
  "emails": [
    "abuse@scip.es",
```

```
    "idealista.com@whoisprivacycontact.com"
  ],
  "dnssec": [
    "unsigned",
    "Unsigned"
  ],
  "name": "Whois Privacy Service Protects this domain",
  "org": "Soluciones Corporativas IP, c/o Whois Proxy",
  "address": "C/ Menestrals 14",
  "city": "Manacor",
  "state": "Illes Balears",
  "registrant_postal_code": "07500",
  "country": "ES"
}
```


De nuevo, algunos datos que llaman la atención: sitio web creado en 1999-08-21, posterior a Fotocasa. La dirección de registro aparece como la ciudad de Manacor, en la isla de Mallorca de la província de las Islas Baleares. Llama la atención que los dominios aparecen como .NET sin que aparezcan términos como GCP/Azure/AWS, por lo que es posible que el sitio web esté hosteado en un servidor físico propiedad de la empresa. Finalmente, el nombre que encontramos arroja poca información sobre el propietario, indicando que "Whois Privacy Service Protects this domain", indicando que el propietario prefiere permanecer anónimo.

Pasamos ahora al subapartado de análisis anteriores. Para empezar, es importante explicar que al tratarse ambas páginas web de datos de bienes raíces, existe un gran interés por parte de los usuarios para la extracción de datos. Tanto es así que podemos encontrar desde vídeos en YouTube que utilizan estos portales para ejemplificar el uso de las librerías de web scraping más populares, https://www.youtube.com/watch?v=1gQyRuIpqUU&t=575s, hasta empresas privadas que comercializan la creación de informes de comparativa y predicción de precios de bienes raíces bajo demanda, https://datstrats.com/bienes-inmuebles/, pasando por repositorios públicos de estudiantes del mismo máster de años anteriores, https://github.com/EdelBlau/PEC_TPC, o de otros miembros de la comunida de GitHub, https://github.com/David-Carrasco/Scrapy-Idealista.

La relevancia de estos análisis es dispar: las soluciones comerciales trabajan con código cerrado y por tanto su interés es meramente anecdótico. Las soluciones de años anteriores que hemos encontrado, si bien obtienen conjuntos de datos similares al que hemos presentado en apartados anteriores, utilizan librerías de Python diferentes. Si bien a primera vista esto puede parecer una diferencia puramente técnica, en la práctica nos encontramos con que hoy en día el sitio web Idealista bloquea por defecto los intentos de scraping utilizando la librería de Python scrapy, como se discute en el siguiente enlace:
https://stackoverflow.com/questions/71800678/getting-around-a-403-error-when-using-scrapy.
Además, aunque en la discusión de los repositorios anteriores se proponen soluciones como el uso de proxys, después de dos años la mayor parte de ellos han sido inhabilitados. Finalmente, a partir del tutorial de YouTube presentado se puede encontrar una gran variedad de vídeos de temática

similar, pero la mayor parte de ellos adolecen de que sólo extraen datos muy genéricos y que a los pocos meses de ser publicados ya pueden considerarse obsoletos; por tanto, son más útiles como guías de programación que como ejemplos profundos de análisis. En el punto siete comparamos nuestro trabajo con los precedentes, resaltando las diferencias relevantes.

Finalmente, una breve relación sobre las consideraciones ético-legales de nuestro trabajo. De acuerdo a consideraciones de privacidad, el dataset final generado no contiene información personal de los propietarios o agencias que han publicado el anuncio: hemos evitado extraer información como el nombre o el número de teléfono de contacto, por lo que los datos publicados son relativamente anónimos. En todo caso, la información que estamos tratando difícilmente podría considerarse como sensible o delicada, dado que se trata de anuncios de compraventa publicados de forma voluntaria por los propietarios de los bienes o sus gestores en un sitio especializado. La información que publicamos es, además, accesible a cualquier usuario de estos sitios web sin necesidad de registros, por lo que las actividades desarrolladas en el código son legítimas.

7. **Inspiración.** Explicar por qué puede ser interesante este conjunto de datos y qué preguntas se pretenden responder con ellos. Es necesario comparar con los análisis anteriores o análisis similares presentados en el apartado 6.

Destacamos dos proyectos de ciencia de datos que pueden llevarse a cabo con los datos extraídos y un proyecto para el que necesitaríamos extender el código y técnicas presentadas en esta memoria.

El primer proyecto se trataría de un análisis estadístico que buscase responder preguntas comparativas tanto entre los dos portales web como dentro de un mismo portal. ¿Tienen precios comparables los anuncios listados en Fotocasa e Idealista? Para las mismas características, ¿aparecen variaciones de precio entre las páginas web? ¿Encontramos patrones para los precios más altos, como por ejemplo el uso de la palabra 'chalet' en la descripción de la vivienda? Planteadas y respondidas estas preguntas para el dataset presentado, podríamos seleccionar otros puntos al azar de la geografía española y repetir las tareas llevadas a cabo para ver si los patrones encontrados son consistentes.

El segundo proyecto se trataría de la creación de un modelo de regresión que tratase de ajustar el precio de venta en función del resto de variables extraídas. Aunque un modelo de predicción del precio de bienes raíces real es mucho más complejo que el subconjunto que estamos tratando, incluso un modelo relativamente tosco nos ayudaría a detectar outliers que podrían indicar oportunidades de compra.

El tercer proyecto, que repetimos no podría ser llevado a cabo con la información presentada, es la de un sistema de recomendación. Ambos portales permiten a sus usuarios publicar descripciones de los inmuebles anunciados; estos textos podrían analizarse con un modelo de embedding tipo Doc2Vec para generar recomendaciones personalizadas. Estas descripciones no han sido incluidas en el dataset por la problemática que suponía su extracción, la cual discutimos en el punto 9.

Finalmente, si relacionamos nuestro proyecto con los enlazados en el punto anterior, podemos destacar lo siguiente: respecto a los repositorios de GitHub encontrados que han realizado scraping del mismo sitio web, que hemos utilizado una tecnología diferente. Mientras que estos han optado por la librería scrapy, nosotros hemos utilizado una combinación de Selenium y BeautifulSoup; por tanto nuestro trabajo ofrece una solución alternativa para situaciones complejas, como el bloqueo automático de scrapy en Idealista mencionado en el punto anterior y un punto de partida para otros desarrolladores interesados en el web scraping. Respecto a los tutoriales de YouTube enlazados, nuestro trabajo va mucho más allá en cuanto a exploración de elementos, control de errores y modularidad del código presentado. Mientras que estos tutoriales suelen limitarse a obtener el nombre de un elemento o el enlace de la página siguiente, nuestro análisis ha concretado los objetos y posibilidades que existen en las páginas adaptándose a la versión actual de los sitios web estudiados. La comparación con las soluciones comerciales queda fuera de nuestro alcance porque tanto el código como los informes son de uso privado.

8. **Licencia.** Seleccionar una licencia adecuada para el dataset resultante y justificar el motivo de su elección.

De todas las licencias disponibles, la más apropiada para este trabajo sería probablemente la licencia MIT, posiblemente la licencia más permisiva. El proceso de selección ha sido exploratorio: elegida como base MIT, ¿qué motivos pueden llevarnos a imponer nuevas restricciones? Generalmente, dos motivos: que la comunidad a la que pertenece el proyecto tenga una licencia de preferencia o motivos comerciales. Dado que nuestro proyecto no se encuentra vinculado a ninguna organización de software particular, como podrían ser Apache o Rust, no hay necesidad de imponer nuevas restricciones por esta vía. En cuanto a los motivos comerciales, como hemos indicado en el punto seis ya existen soluciones comerciales mucho más sofisticadas y de escala muy superior a la tratada en este proyecto, por lo que también parece innecesario limitar los forks privados del proyecto. Por tanto, MIT satisface todas nuestras necesidades.

9. **Código.** Código implementado para la obtención del dataset, preferiblemente en Python o, alternativamente, en R.

El código implementado para la obtención del dataset está almacenado en el github siguiente:
https://github.com/arasevicuoc/TCVD_PRA1

Dado que el código se encuentra debidamente modularizado y comentado en la carpeta /source/ del repositorio enlazado en esta sección, aprovecharemos esta sección para discutir algunas de las decisiones de diseño llevadas a cabo.

El proyecto presentado extrae información de dos sitios web diferentes; por tanto, para dotar de independencia a las extracciones de datos se ha decidido que cada uno de estos procesos estará almacenado en una librería distinta con un único método público, que es el que debe ser llamado para ser ejecutado desde una función externa. Aunque este método externo dispone de cierta flexibilidad, permitiéndose opciones como por ejemplo cambiar la zona de venta o el número de páginas máximas a ser investigado, sin una investigación exhaustiva de la generación de URLs de

los sitios web estudiados no podemos asegurar que se cubran todas las posibilidades; tampoco se hace ningún control de errores específico si el usuario ingresa índices negativos en el número máximo de páginas o ubicaciones fuera del estado español. Por tanto, debería ser concebido más bien como un wrapper sobre una función estática más que como una verdadera función.

Esta consideración se ve reforzada cuando entramos en las funciones privadas de cualquiera de los ficheros: un gran número de elementos html y xpaths que se encuentran hardcodeados sin métodos de búsqueda dinámica. En este proyecto, la evolución del código refleja la metodología iterativa empleada: creación de un script mínimo que localice la información de interés, investigación del html extraído y uso de las herramientas para desarrolladores del navegador para investigar otros elementos a incluir. Una vez hemos conseguido una cantidad de información aceptable, reescritura del código en forma de funciones y librerías, aglutinando código según su propósito o sus características en común.
Dejando de lado las funciones cuyo objetivo es envolver instrucciones de inicialización de drivers, click sobre cookies, scroll downs, cargas en dataframes o ficheros csv, las funciones que merecen más atención por su estructura y relevancia son las siguientes: en el archivo de fotocasa_scraping.py, _card_type_is_minimal_info y _card_type_is_not_minimal_info, que extraen información sobre las características de la propiedad de forma diferente según el tipo de CardPack que estemos tratando, mientras que en el archivo de idealista_scraping.py destacamos las funciones _get_generic_information y _get_non_generic_information, que extraen información de diferentes puntos del div que las envuelve.

En el portal web Fotocasa, cada anuncio de compra aparece dentro de un elemento div que contiene una serie de datos generales sobre la propiedad, como pueden ser el precio, el título del anuncio, su ubicación geográfica, si tiene ascensor, etc. Al investigar con el inspector de objetos se puede ver que estos elementos contienen en su interior el string ‘CardPack’, que es la forma a la que nos referimos a ellos dentro del código. Algunos experimentos al principio del desarrollo revelaron tres tipos diferentes de CardPacks, “Basic”, “Advanced” y “Premium”, muy similares entre ellos y para los que no era necesario indicar de forma explícita la característica a ser extraída porque esta información viene incluida en la subclase del div externo, por lo que hemos podido utilizar expresiones regular para extraer la información de forma flexible. Sin embargo, al avanzar por las páginas del portal, el número de anuncios patrocinados o que incluyen información considerada ‘premium’ empieza a disminuir, apareciendo en su lugar un nuevo tipo de CardPack llamado “Minimal”, que a diferencia de los casos anteriores no contiene indicación del nombre de la característica, sino que podemos inferir el tipo de característica por el texto que contiene. Por ejemplo, si el span contiene el string ‘habs’, podemos inferir que se trata de la característica de habitaciones. Debido a estas diferencias en los CardPacks, decidimos crear dos funciones diferentes dependiendo de si el CardPack es de tipo “Minimal” o no.

En el portal web Idealista, los anuncios y las características de las propiedades representadas tienen una estructura mucho más rígida que en el caso de Fotocasa. Podemos encontrar toda la información de la misma manera en los div ‘item-info-container’, llamado info_container a lo largo del código. Distinguimos dos partes de estos info_containers: una cabecera con información genérica, como el enlace hacia la página web específica de la propiedad, su precio o su título, y a

continuación una lista de longitud variable llamada ‘item-detail’, que según su longitud puede contener unas características u otras. Así, si la longitud de la lista es 1, la información que contiene corresponde al número de habitaciones; si la longitud de la lista es 2, la información que contiene corresponde al número de habitaciones y al número de metros cuadrados. También es posible encontrar información fuera de esta lista, como por ejemplo el parking, que se encuentra en un span separado llamado ‘item-parking’ que existirá o no según si la propiedad tiene o no plaza de aparcamiento. En este caso, la distinción entre las funciones _get_generic_information y _get_non_generic_information depende de si la información está presente en todos los elementos o es posible que no aparezca según las características del anuncio.

Finalmente, el código de main.py sirve como punto común de ejecución de ambas librerías y su función es la creación del dataset único enlazado en el repositorio, output_concat.csv. Dado que este archivo desde el punto de vista funciona tiene un único propósito la modularización observada en otros apartados del código no se ha aplicado por diseño.

Como dificultades del proyecto, destacamos tres: la dificultad para obtener descripciones completas de los inmuebles en Fotocasa y los bloqueos por IP de Idealista tanto por el uso de la librería scrapy como por la extracción masiva de datos.

Sobre las descripciones completas, en el listado de anuncios suelen cargarse sólo dos o tres descripciones completas, mientras que el resto se oculta detrás de elementos del tipo 'Leer más' que no contienen elementos 'href', sino que ejecutan código JS que carga la información bajo demanda. Aunque es posible seguir los enlaces de cada una de las referencias de forma individual, el tiempo de ejecución total y la cantidad de peticiones enviadas hacia el servidor dificulta la ejecución local de un código de este estilo, por lo que ha sido omitido para esta práctica.

Sobre el uso de la librería scrapy se da una situación particular al tratar de extraer información de Idealista: una única petición resulta en un bloqueo indefinido de la IP emisora. Dado que la librería de scrapy contiene entre sus parámetros por defecto el cumplimiento forzado del archivo robots.txt, parece improbable que el bloqueo pueda deberse a que se hayan visitado URLs desaconsejadas para un User-Agent particular. Además, utilizando Selenium sólo hemos podido realizar una extracción completa de datos antes de ser bloqueados de nuevo. Aunque existen formas de sortear estos problemas como pueden ser reiniciar el router para obtener una IP nueva o utilizar servidores proxy, parece que la situación está relacionada con que Idealista está actualmente comercializando un servicio de inteligencia de negocio para la creación de reports de mercado a partir de los datos que publican en su portal web (https://www.idealista.com/en/news/tags/property-market-statistics/) por lo que existe la posibilidad de que en estos momentos estén siendo más agresivos con los intentos de web scraping.

10. **Dataset.** Publicar el dataset obtenido en formato CSV en Zenodo, incluyendo una breve descripción de la misma. Obtener y adjuntar el enlace del DOI del dataset (https://doi.org/…). El dataset también deberá incluirse en la carpeta /dataset del repositorio.

https://doi.org/10.5281/zenodo.7861754

11. **Vídeo.** Realizar un breve vídeo explicativo de la práctica (máximo 10 minutos), que deberá contar con la participación de los dos integrantes del grupo. En el vídeo se deberá realizar una presentación del proyecto, destacando los puntos más relevantes, tanto de las respuestas a los apartados como del código utilizado para extraer los datos. Indicar el enlace del vídeo (https://drive.google.com/…), que deberá ubicarse en el Google Drive de la UOC.

https://drive.google.com/drive/folders/1YBXcvQTC1XqJb9DHZqB_t6xYgvrPPaln?usp=sharing

Contribuciones	Firma
Investigación previa	ARL / ALN
Redacción de las respuestas	ARL / ALN
Desarrollo del código	ARL / ALN
Participación en el video	ARL / ALN