PRACTICA 1 – TIPOLOGIA Y CICLO DE VIDAD DE LOS DATOS

Angel Bustamante Maldonado

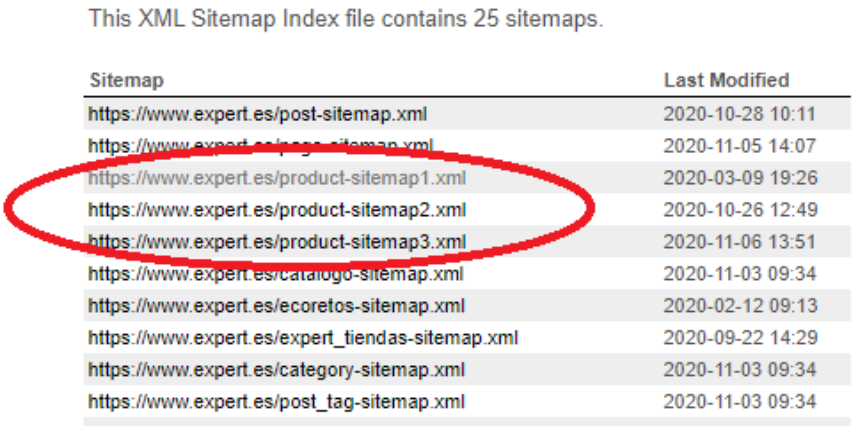
Eduardo Diaz Villanueva

1. Contexto. Explicar en qué contexto se ha recolectado la información. Explique por qué el sitio web elegido proporciona dicha información.

Queremos recolectar datos de internet para hacer un estudio. La mayoría de webs no proporcionan APIs para descargar esos datos por lo que utilizaremos la técnica del webscraping. Hemos elegido el sector de electrodomésticos para realizar el citado estudio y concretamente la cadena Expert ya que al analizar el archivo robots.txt hemos comprobado que no estaba prohibido el uso de robots.

```
User-Agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php
Sitemap: https://www.expert.es/sitemap_index.xml
```

Nos interesa tener los datos de los productos que se ofertan en esta cadena y después de acceder al sitemap y de hacer un análisis manual vimos de donde podíamos sacar la información que nos interesaba.



This XML Sitemap Index file contains 25 sitemaps.

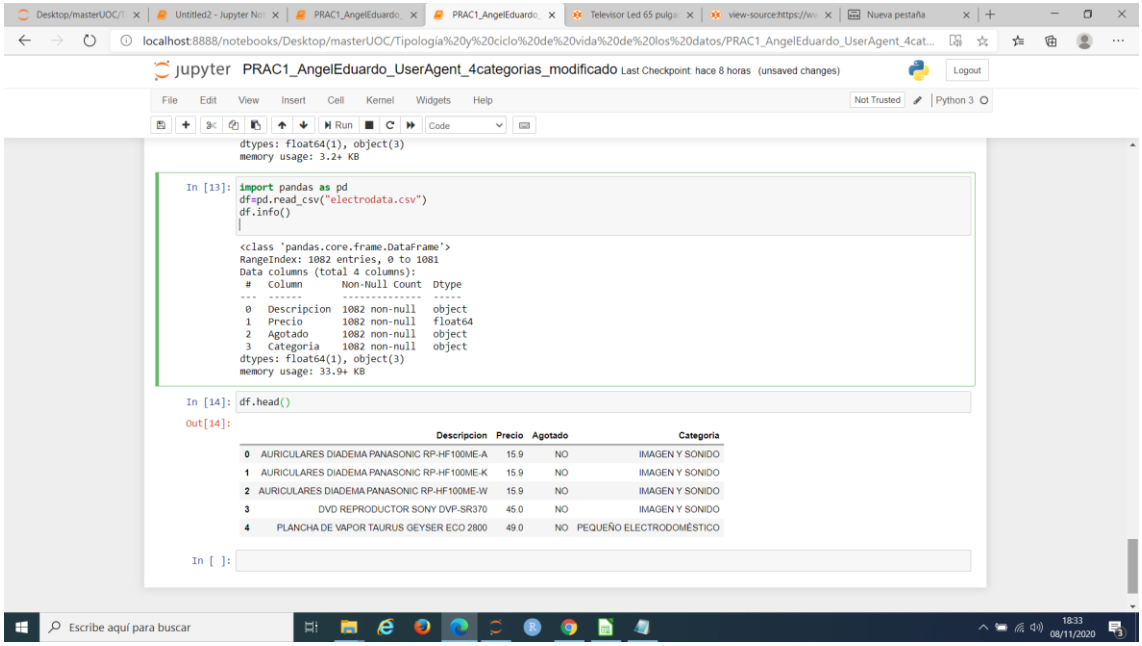
Sitemap	Last Modified
https://www.expert.es/post-sitemap.xml	2020-10-28 10:11
https://www.expert.es/page-sitemap.xml	2020-11-05 14:07
https://www.expert.es/product-sitemap1.xml	2020-03-09 19:26
https://www.expert.es/product-sitemap2.xml	2020-10-26 12:49
https://www.expert.es/product-sitemap3.xml	2020-11-06 13:51
https://www.expert.es/catalogo-sitemap.xml	2020-11-03 09:34
https://www.expert.es/ecoretos-sitemap.xml	2020-02-12 09:13
https://www.expert.es/expert_tiendas-sitemap.xml	2020-09-22 14:29
https://www.expert.es/category-sitemap.xml	2020-11-03 09:34
https://www.expert.es/post_tag-sitemap.xml	2020-11-03 09:34

2. Definir un título para el dataset. Elegir un título que sea descriptivo.

electrodata.csv ya que refiere a electrodomésticos y datos.

3. Descripción del dataset. Desarrollar una descripción breve del conjunto de datos que se ha extraído (es necesario que esta descripción tenga sentido con el título elegido).

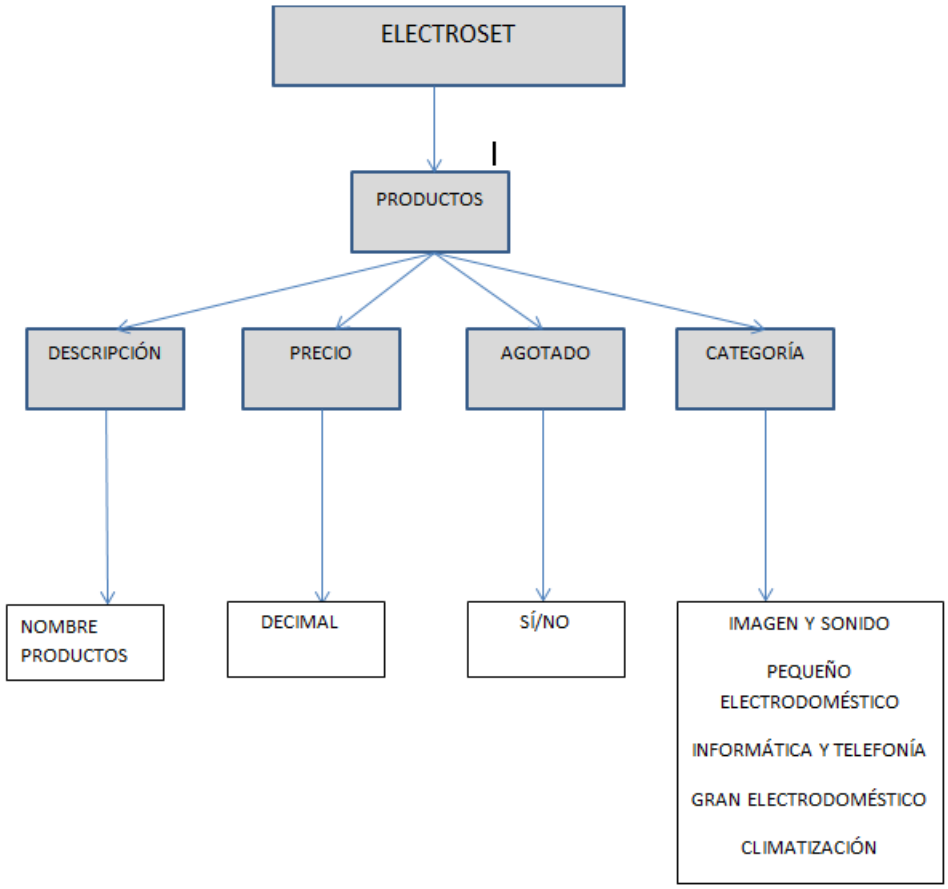
Se ha extraído un dataset de la web https://www.expert.es en la que constan el nombre del producto, su precio, si está agotado o no, y su categoría. Volcamos el csv en un dataframe para comprobar los tipos de datos.



Vemos algunas características del dataset en la figura

```
In [9]: 100 * df['Categoria'].value_counts() / len(df['Categoria'])

Out[9]: PEQUEÑO ELECTRODOMÉSTICO      31.423290
        IMAGEN Y SONIDO                20.887246
        GRAN ELECTRODOMÉSTICO          18.391867
        INFORMÁTICA Y TELEFONÍA        14.695009
        CLIMATIZACIÓN                  14.602588
        Name: Categoria, dtype: float64

In [10]: 100 * df['Agotado'].value_counts() / len(df['Agotado'])

Out[10]: NO    93.900185
         SI     6.099815
         Name: Agotado, dtype: float64

In [11]: df['Precio'].describe()

Out[11]: count    1082.000000
         mean      125.670939
         std       165.375629
         min         1.099000
         25%        27.900000
         50%        55.000000
         75%       149.000000
         max       999.000000
         Name: Precio, dtype: float64
```

4. Representación gráfica. Presentar una imagen o esquema que identifique el dataset visualmente.



5. Contenido. Explicar los campos que incluye el dataset, el periodo de tiempo de los datos y cómo se ha recogido.

El dataset consta de 1082 registros y 4 campos:

- La descripción o nombre del producto que es un objeto string.
- El precio que es un float.
- Agotado que es un objeto string que puede ser “SI” o “NO” significando que el producto está agotado o no.
- Y categoría, un string con la categoría en la que está encuadrado el producto, que a partir del dataframe como se puede ver en la figura son IMAGEN Y SONIDO, PEQUEÑO ELECTRODOMÉSTICO, INFORMÁTICA Y TELEFONÍA, GRAN ELECTRODOMÉSTICO,CLIMATIZACIÓN

```
In [7]: df['Categoria'].unique()

Out[7]: array(['IMAGEN Y SONIDO  ', 'PEQUEÑO ELECTRODOMÉSTICO  ',
       'INFORMÁTICA Y TELEFONÍA  ', 'GRAN ELECTRODOMÉSTICO  ',
       'CLIMATIZACIÓN  '], dtype=object)
```

Los datos se han recogido en un instante concreto (es una foto fija), esta fecha es muy importante, por ejemplo en este caso se observa que la mitad de los productos agotados .pertenecen a CLIMATIZACIÓN cosa lógica teniendo en cuenta en la época del año en que estamos.

En la figura podemos ver la parte de código que recoge los datos de cada página html

```
try:
    # Nombre/Titulo del producto
    print ("...Obtenemos los nombres")
    tags = soup.find('h1', class_='product_title entry-title')
    descripcion = tags.getText()

    # Precio del producto
    print ("...Obtenemos los precios")
    tags = soup.find('span', class_='product-price')
    precio = tags.getText()
    precio = precio.replace(",",".")
    precio=precio.replace('€','')


    # Categoria del producto
    print ("...Obtenemos la categoria")
    ##tags=soup.find('div', class_='col-wrapper title-wrapper')
    tags = soup.find('nav', class_= 'like-h1-style woocommerce-breadcrumb')
    categoria = tags.getText()
    categoria = categoria[1:categoria.find("|")]

    # Existencia del producto
    print("....Miramos si está agotado")
    name=None
    agotado = "NO"
    name=soup.find('div', class_='out-of-stock-flag')
    if (name):
        agotado="SI"
```

Este script sólo tiene sentido lanzarlo cada cierto tiempo, pongamos cada trimestre para ver la evolución de los precios, y por eso no es muy importante que tarde una hora y media en rellenar el dataset. Este tiempo se debe al retardo que hemos tenido que aplicar para evitar los bloqueos.

6. Agradecimientos. Presentar al propietario del conjunto de datos. Es necesario incluir citas de investigación o análisis anteriores (si los hay).

Agradecemos a EXPERT ESPAÑA el uso de sus datos con fines educativos.

7. Inspiración. Explique por qué es interesante este conjunto de datos y qué preguntas se pretenden responder.

Aunque esta práctica tiene un fin educativo hay algún uso para el que podría ser interesante. Por ejemplo, este conjunto de datos puede ser interesante como primer paso para alimentar un comparador de precios en el ámbito de los electrodomésticos. Evidentemente sería necesario normalizar el campo de descripción del producto para poder comparar productos iguales.

8. Licencia. Seleccione una de estas licencias para su dataset y explique el motivo de su selección: ○ Released Under CC0: Public Domain License ○ Released Under CC BY-NC-SA 4.0 License ○ Released Under CC BY-SA 4.0 License ○ Database released under Open Database License, individual contents under Database Contents License ○ Other (specified above) ○ Unknown License

Nosotros nos decantamos por Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0) que según la web oficial creative.commons.org permite compartir y adaptar el dataset siempre que mantenga estos términos en posibles adaptaciones que publique. El punto clave de esta licencia es que no

permite utilizar el dataset con fines comerciales, lógico para que no lo puedan aprovechar los competidores de Expert.

9. Código. Adjuntar el código con el que se ha generado el dataset, preferiblemente en Python o, alternativamente, en R.

https://github.com/ediazvi/electrodomesticoswebscraping

```
import requests
from bs4 import BeautifulSoup
import time
import csv
import random
import unicodedata

# Preparamos variables
url_list = list()
productos = list()

#
# Comenzaremos descargando el mapa de la pagina desde  https://www.expert.es/sitemap_index.xml
url = "https://www.expert.es/sitemap_index.xml"

# Para evitar los bloqueos simularemos ser un navegador e incluso cambiaremos de user_agent
para
# evitar los bloqueos en las paginas de los productos.
# Informacion de los user agent obtenida de
https://developers.whatismybrowser.com/useragents/explore/
# User-Agent: Mozilla/<version> (<system-information>) <platform> (<platform-details>)
<extensions>
user_agent_list = [
    'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/605.1.15 (KHTML, like Gecko)
Version/13.1.1 Safari/605.1.15',
    'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0',
    'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/83.0.4103.97 Safari/537.36',
    'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:77.0) Gecko/20100101 Firefox/77.0',
    'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/83.0.4103.97 Safari/537.36',
]

print("Iniciamos web scraping...")

# Generamos la consulta
user_agent = random.choice(user_agent_list)
headers = {'User-Agent': user_agent}
page = requests.get(url,headers=headers)
```

```
if (page.status_code == 200):

    # Esperamos un tiempo para simular el usuario
    time.sleep(1)
    # Extraemos las etiquetas loc de los enlaces a las paginas
    soup = BeautifulSoup(page.content, 'html.parser')
    url_sitemap = soup.findAll("loc")
    #print(url_sitemap)

    # Recorremos cada una de las paginas buscando enlaces a paginas de productos.
    print("...Recorremos las paginas")
    for each_site in url_sitemap:

        # Generamos la consulta
        user_agent = random.choice(user_agent_list)
        headers = {'User-Agent': user_agent}
        page = requests.get(each_site.text,headers=headers)
        print ("%s , %i" %(each_site.text, page.status_code))

        if (page.status_code == 200):
            print("...Buscamos productos")
            soup = BeautifulSoup(page.content, 'html.parser')
            #print(soup)
            full_tag = soup.findAll("loc")
            #print(full_tag)

            for each_tag in full_tag:
                # Filtramos para quedarnos solo con las del dominio /producto
                if each_tag.text.find('https://www.expert.es/producto/') != -1:
                    print(each_tag.text)
                    url_list.append(each_tag.text)

        # Esperamos un tiempo aleatorio entre peticiones del bucle
        time.sleep(1 + random.randrange(5))

    # Comprobamos
    print ("...Se han encontrado %i enlaces" %len(url_list))
    #print(url_list)

else:
    print("La pagina solicitada no responde!")
    print("......Respuesta de la pagina : ", page.status_code)

# ------------------------------------------------------------------------------------------
if len(url_list) == 0:

    # Como hemos sido "Baneados" cargo un fichero de texto obtenido de las paginas
```

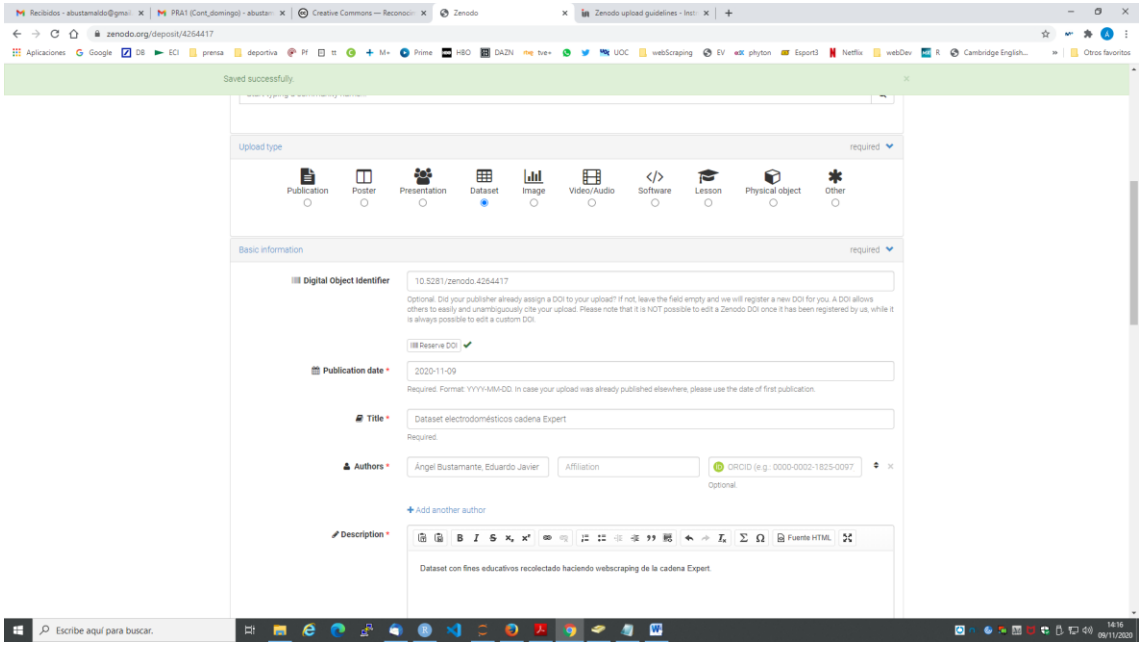
```
    # https://www.expert.es/product-sitemap1.xml
    # https://www.expert.es/product-sitemap1.xml
    # https://www.expert.es/product-sitemap3.xml
    # antes del bloqueo, y continuamos el script como si lo hubieramos obtenido de la web.

    print("Leemos el fichero de texto...")
    url_list = list()
    url_file = open('sites.txt')
    for linea in url_file:
        url_list.append(linea.replace("\n",""))
    print ("...Se han encontrado %i enlaces" %len(url_list))
# ------------------------------------------------------------------------------------------

# Recorremos cada una de las paginas del producto obteniendo la descripcion y el precio.
print("Recorremos los sites...")

f =  open('electrodata.csv', 'w', newline='',encoding="utf-8")
f.write("Descripcion,Precio,Agotado,Categoria\n")

descripcion = ""
precio = ""
agotado = "NO"
categoria = ""

for url in url_list:

     # Generamos la consulta
    user_agent = random.choice(user_agent_list)
    headers = {'User-Agent': user_agent}
    page = requests.get(url,headers=headers)
    print ("%s , %i" %(url, page.status_code))

    if (page.status_code == 200):

        #time.sleep(1)

        print("...Buscamos los datos de los productos")
        soup = BeautifulSoup(page.content, 'html.parser')
        #print(soup)

        try:
            # Nombre/Titulo del producto
            # Ejemplo:
            # <h1 class="product_title entry-title" itemprop="name">ENCIMERA GAS ELECTROLUX
EGT6633NOK</h1>
            #print ("...Obtenemos los nombres")
            tags = soup.find('h1', class_='product_title entry-title')
            descripcion = tags.getText()
```

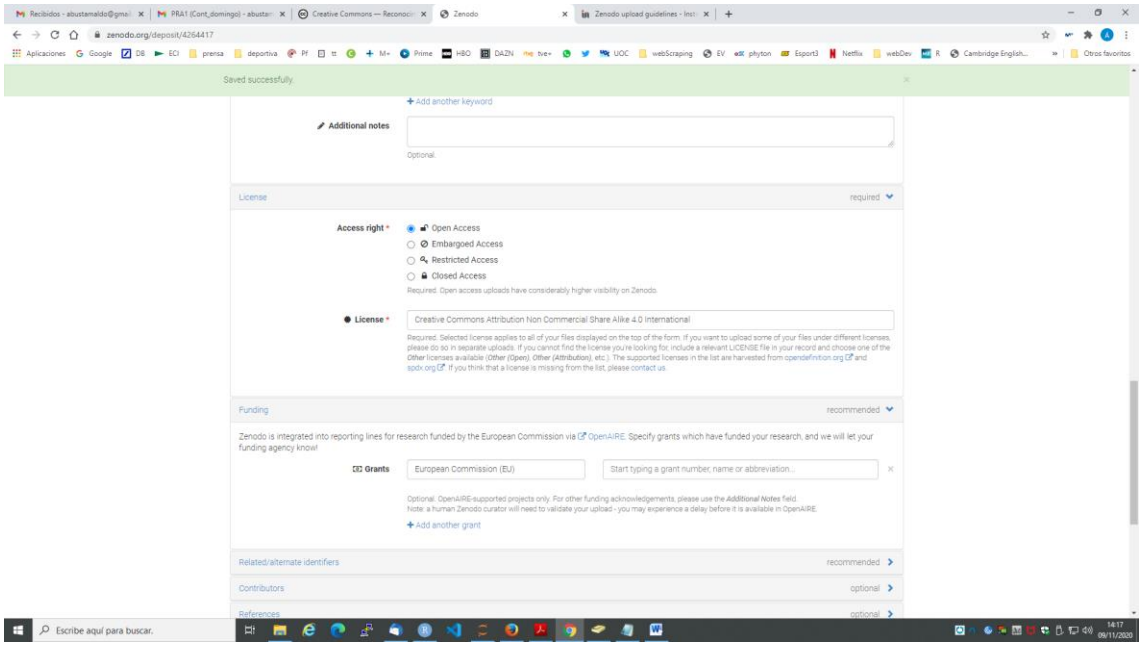
```
            # Forzamos la normalización
            descripcion=unicodedata.normalize("NFKD",descripcion)

            # Precio del producto
            # Ejemplo:
            # #<span class="product-price"><span class="woocommerce-Price-amount
amount">550<span class="woocommerce-Price-currencySymbol">€</span></span></span>
            #print ("...Obtenemos los precios")
            # Reemplazamos la coma por el punto y eliminamos el caracter €
            tags = soup.find('span', class_='product-price')
            precio = tags.getText()
            precio = precio.replace(",",".")
            precio = precio.replace('€','')
             # Forzamos la normalización
            precio=unicodedata.normalize("NFKD",precio)

            # Categoria del producto
            #print ("...Obtenemos la categoria")
            ##tags=soup.find('div', class_='col-wrapper title-wrapper')
            tags = soup.find('nav', class_= 'like-h1-style woocommerce-breadcrumb')
            categoria = tags.getText()
            # Recortamos la cadena
            categoria = categoria[1:categoria.find("|")]
            # Forzamos la normalización
            categoria=unicodedata.normalize("NFKD",categoria)

            # Existencia del producto
            # En algunos productos hemos encontrado este flag
            # que queremos registar tambien.
            #print("....Miramos si está agotado")
            tags=None
            agotado = "NO"
            tags=soup.find('div', class_='out-of-stock-flag')
            if (tags):
                agotado="SI"
             # Forzamos la normalización
            agotado=unicodedata.normalize("NFKD",agotado)

            print ("%s,%s,%s,%s" %(descripcion, precio, agotado, categoria))
            f.write("%s,%s,%s,%s\n" %(descripcion, precio, agotado, categoria))

        except:
            print("...No se pudo encontrar alguno de los datos.")

    time.sleep(1)
    #time.sleep(1 + random.randrange(5))

f.close()
```

10. Dataset. Publicación del dataset en formato CSV en Zenodo (obtención del DOI) con una breve descripción.

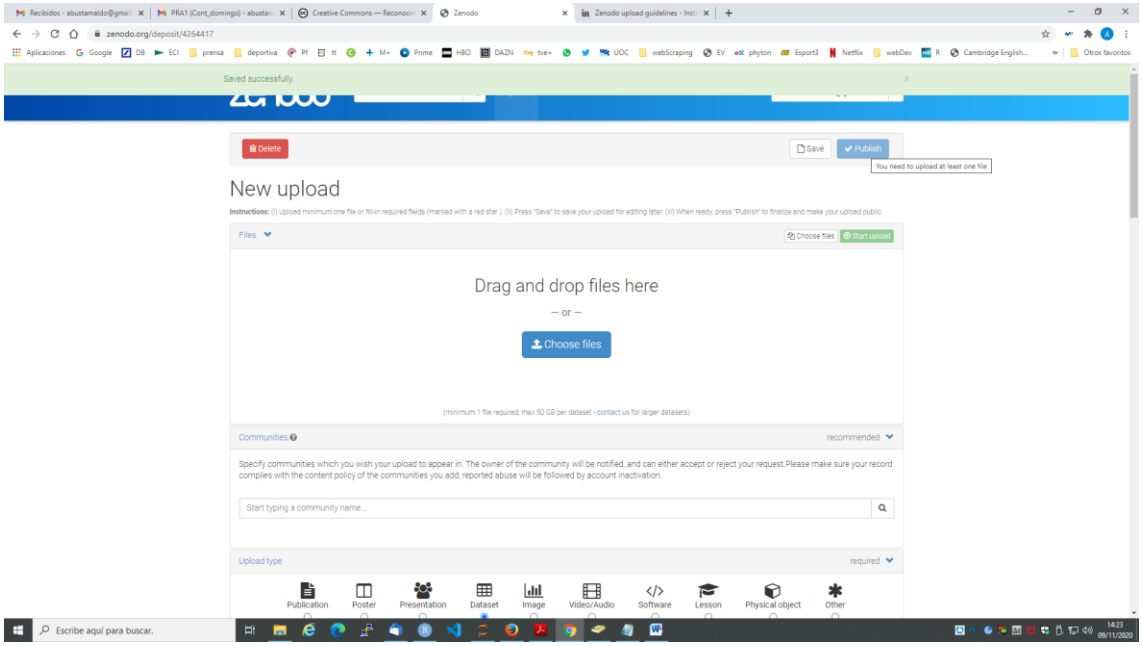
El código que nos han dado es 10.5281/zenodo.4264417

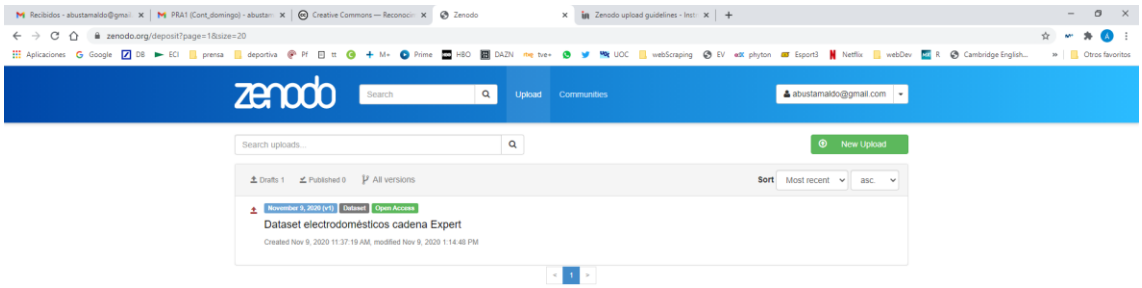
Adjunto pantallazos obtenidos en el proceso de alta

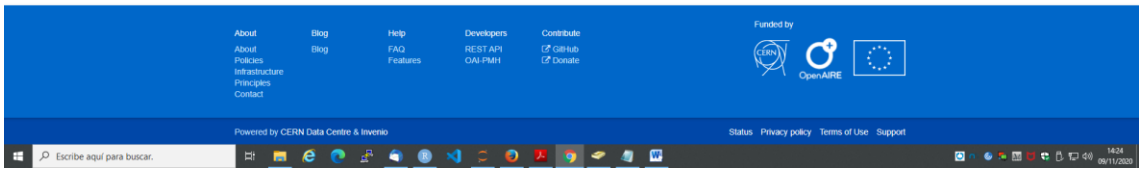




Como vemos en las siguientes pantallas nos deja grabar, pero no publicar.







Contribuciones

Ambos hemos colaborado desde el inicio de la practica en todos los aspectos de la elaboración de la misma. Hemos estado en contacto en todo momento usando la plataforma Github como almacenamiento, así como el email, WhatApp y llamadas de teléfono para resolver dudas y acordar siguientes pasos.

Contribuciones	Firma
Investigación previa	AB, ED
Redacción de las repuestas	AB, ED
Desarrollo código	AB, ED