PRA 01: Asignatura:M2.851-Tipología y ciclo de vida de los datos aula 2

Almudena Caballero - Angel A. Urbina [MASTER DATA SCIENCE]

Presentación

En esta práctica se elabora un caso práctico orientado a aprender a identificar los datos relevantes para un proyecto analítico y usar herramientas de extracción de datos. Para hacer esta práctica tendréis que trabajar en grupos de 2 personas. Tendréis que entregar en el REC un solo fichero con el enlace al repositorio Git donde haya las soluciones, incluyendo los nombres de los componentes del grupo. Podéis utilizar la Wiki o README.md del repositorio para describir vuestro grupo y los diferentes archivos de vuestra entrega. Cada miembro del grupo tendrá que contribuir con su usuario del repositorio. Podéis mirar estos ejemplos como guía (recuerden que se trata de ejemplos y no de respuestas perfectas para la práctica):

- Ejemplo: https://github.com/rafoelhonrado/foodPriceScraper
- Ejemplo complejo: https://github.com/tteguayco/Web-scraping

Además, deben entregar un vídeo explicativo de la práctica en donde cada uno de los integrantes del grupo explique con sus propias palabras tanto las respuestas del proyecto como el código utilizado para llevar a cabo la extracción. El vídeo debe ser enviado a través de un enlace a Google Drive que deben proporcionar, junto con el enlace al repositorio Git, al momento de entregar la práctica.

Descripción de la práctica a realizar

El objetivo de esta actividad será la creación de un dataset a partir de los datos contenidos en un sitio web. El idioma del sitio web elegido deberá ser español, inglés o catalán. Se deberán resolver los siguientes apartados:

- 1. Contexto. Explicar en qué contexto se ha recolectado la información. Explicar por qué el sitio web elegido proporciona dicha información.
- 2. Título. Definir un título que sea descriptivo para el dataset.
- 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.
- 4. Representación gráfica. Dibujar un esquema o diagrama que identifique el dataset visualmente y el proyecto elegido.
- 5. Contenido. Explicar los campos que incluye el dataset, el periodo de tiempo de los datos y cómo se han recogido.
- 6. Agradecimientos. Presentar al propietario del conjunto de datos. Es necesario incluir citas de análisis anteriores o, en caso de no haberlas, justificar esta búsqueda con análisis

similares. Justificar qué pasos se han seguido para actuar de acuerdo a los principios éticos y legales en el contexto del proyecto.

- 7. Inspiración. Explicar por qué es interesante este conjunto de datos y qué preguntas se pretenden responder. Es necesario comparar con los análisis anteriores presentados en el apartado 6.
- 8. Licencia. Seleccionar una de estas licencias para el dataset resultante y justificar 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.
- 9. Código. Adjuntar en el repositorio Git el código con el que se ha generado el dataset, preferiblemente en Python o, alternativamente, en R.
- 10. Dataset. Publicar el dataset obtenido(*) en formato CSV en Zenodo con una breve descripción. Obtener y adjuntar el enlace del DOI.
- 11. Vídeo. Se debe hacer entrega de un vídeo explicativo de la práctica en donde cada uno de los integrantes del grupo explique con sus propias palabras tanto las respuestas del proyecto como el código utilizado para llevar a cabo la extracción. El vídeo debe ser enviado a través de un enlace a Google Drive que deben proporcionar, junto con el enlace al repositorio Git, al momento de entregar la práctica.

1.-Contexto.(2,5 % puntuación)

PREGUNTA 1.1

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

Nos planteamos recopilar y almacenar, para posteriormente analizar, la información relativa a las posibles ofertas de estudios de máster. El objetivo de obtener esta información es el disponer de un único fichero, sin necesidad de visitar cada una de las webs, con la información que consideramos más relevante para elegir el máster considerado. Elegimos Emagister (https://www.emagister.com), web buscadora de másteres, y tratamos de obtener toda la información que nos ayude a la elección del máster deseado. Consideramos que la información relevante para desarrollar esta decisión es:

- Descripción del máster: tipología, duración, metodología, ...
- Precio
- Requisitos
- A quién va dirigido

- Posible financiación
- Opiniones

2.-Título.(2,5 % puntuación)

Pregunta 2.1

Definir un título que sea descriptivo para el dataset.

Empleamos como título del dataset InfoMaster.csv

3.-Descripción del dataset. (2,5 % puntuación)

Pregunta 3.1

Desarrollar una descripción breve del conjunto de datos que se ha extraído. Es necesario que esta descripción tenga sentido con el título elegido.

El dataset recoge la información relativa a:

- Descripción general: descripción relativa a las características del máster impartido por cada entidad como metodología, duración, tipología, fecha de inicio, ...
- Precio, así como si existe o no la posibilidad de financiación
- Información general: información relativa al máster como público objetivo del máster, requisitos, objetivos, ...
- Opiniones: tanto del centro que lo imparte como del propio máster

Nótese que no todas las entidades ofrecen la misma cantidad de información por lo que puede haber algunos valores que, para determinadas entidades, no podamos obtener.

4.-Representación gráfica.(5 % puntuación)

Pregunta 4.1

Dibujar un esquema o diagrama que identifique el dataset visualmente y el proyecto elegido.

Hacemos la búsqueda inicial en Emagister, esta búsqueda nos proporciona nuevas webs con la información de cada entidad que imparte el máster buscado. Es la información contenida en cada una de esas webs la que recoge nuestro dataset final. Podemos obtener una imagen más clara de cómo hemos obtenido los datos que contiene nuestro dataset en el siguiente esquema:

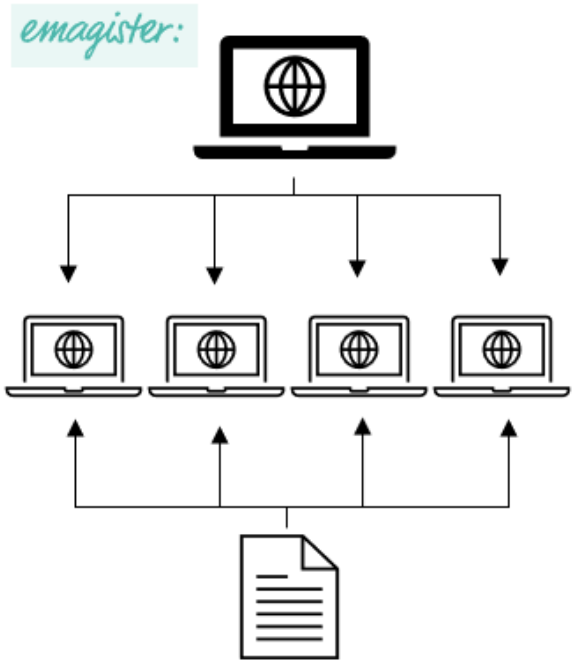

Figura 1: Esquema del proceso

5.-Contenido. (10 % puntuación)

PREGUNTA 5.1

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

El contenido del dataset ha sido obtenido mediante técnicas de web scraping, empleando Python, con a fecha 02/04/2022. El dataset se compone de las siguientes variables:

1. Nombre: nombre del máster
2. Entidad: centro educativo que imparte el máster
3. Precio
4. Descripción: descripción detallada de la información relevante del máster
5. Tipología: online o presencial
6. Programa
7. Web: enlace a la web con la información recopilada

6.-Agradecimientos. (15 % puntuación)

PREGUNTA 6.1

Presentar al propietario del conjunto de datos. Es necesario incluir citas de análisis anteriores o, en caso de no haberlas, justificar esta búsqueda con análisis similares. Justificar qué pasos se han seguido para actuar de acuerdo a los principios éticos y legales en el contexto del proyecto.

Emagister, tal y como ellos se definen, es el punto de encuentro entre los que buscan y ofrecen información. Tras más de una década trabajando para hacer de su directorio el más completo del mundo, tanto en volumen como en profundidad de información, cuentan con más de 100.000 centros de formación. Los principios por los que se rige Emagister son:

1. Hacer accesible la formación a todo el mundo apostando por la creación de un directorio de formación que dé cabida a toda la formación existente (Grados, postgrados, másteres, cursos de especialización, ...) y facilite el poder compartir el conocimiento
2. Lifelong Learning": consideran que el aprendizaje es un proceso continuo a lo largo de la vida y en todos los ámbitos. Por tanto, debe ser modular, on-demand y hecho a la medida de cada persona

Respecto a los principios éticos y legales, hemos actuado de acuerdo con lo marcado según Emagister ya que en sus términos de titularidad y propiedad intelectual e industrial se recoge: "El Usuario se compromete a respetar los derechos de Propiedad Intelectual e Industrial de titularidad de EMAGISTER. Podrá visualizar los elementos de las diferentes websites e incluso imprimirlos, copiarlos y almacenarlos en el disco duro de su ordenador o en cualquier otro soporte físico siempre y cuando sea, única y exclusivamente, para su uso personal y privado. El Usuario deberá abstenerse de suprimir, alterar, eludir o manipular cualquier dispositivo de protección o sistema de seguridad que estuviera instalado en las páginas de EMAGISTER" Es por ello por lo que asumimos que el uso moderado de web scraping es adecuado.

7.-Inspiración.(12,5 % puntuación)

Pregunta 7.1

Explicar por qué es interesante este conjunto de datos y qué preguntas se pretenden responder. Es necesario comparar con los análisis anteriores presentados en el apartado 6.

Motivados por la reciente tarea de elección de un máster, nos planteamos una forma más cómoda y sencilla de recolectar la información de las distintas webs. Muchas veces, las tareas de búsqueda se convierten en un proceso tedioso en el que, finalmente, acabas con numerosas ventanas abiertas en las que pierdes la visión de toda la información. Así, pretendemos obtener, de forma rápida y centralizada, una herramienta que nos permita disponer de toda la información útil sobre un máster de interés. Este objetivo lo conseguimos con Emagister, ya que se trata de una página de búsqueda de formaciones, mediante la cual podemos recopilar la información. Somos, además, partidarios de los principios sobre los que se rigen: información accesible para todos y aprendizaje continuo a lo largo de todos los ámbitos de la vida.

8.-Licencia.(5 % puntuación)

Pregunta 8.1

Seleccionar una de estas licencias para el dataset resultante y justificar 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.

Para no incurrir en los términos de titularidad y propiedad intelectual e industrial marcados por Emagister, le asignamos a nuestro dataset la licencia CC BY-NC-ND 4.0 ya que ésta permite usar una obra mientras cites al autor, sea para proyectos no comerciales y no se modifique de ninguna manera

9.-Código. (20 % puntuación)

PREGUNTA 9.1

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

Repositorio GitHub

10.-Dataset. (20 % puntuación)

PREGUNTA 10.1

Publicar el dataset obtenido(*) en formato CSV en Zenodo con una breve descripción. Obtener y adjuntar el enlace del DOI.

11.-Vídeo. (5 % puntuación)

PREGUNTA 11.1

Se debe hacer entrega de un vídeo explicativo de la práctica en donde cada uno de los integrantes del grupo explique con sus propias palabras tanto las respuestas del proyecto como el código utilizado para llevar a cabo la extracción. El vídeo debe ser enviado a través de un enlace a Google Drive que deben proporcionar, junto con el enlace al repositorio Git, al momento de entregar la práctica.

```
#!/usr/bin/env python
# coding: utf-8

# PRAC01 Tipologia UOC Curso 2022
#
# Scrapper de WEB http://www.emagister.com
#
# Equipo: Angel A. Urbina & Almudena Caballero
#
# Fecha: 02 Abril 2022
#
# Versi n: 1.1
#
# Notas: Incluye an lisis Secuencial y Multiproceso
#

# In[1]:


# Construcci n de Loggings para informar

# Objetivo: Disponer tanto en pantalla como en file (log) de el procesamiento
    que se realiza

import logging
import logging.config

# https://coderzcolumn.com/tutorials/python/logging-config-simple-guide-to-
    configure-loggers-from-dictionary-and-config-files-in-python

# Carga de Archivo de configurac n del login
logging.config.fileConfig('loggingUOCPRA01.conf')

# Creacci n del logger
logger = logging.getLogger('UOCPRA01')

# 'application' code
#logger.debug('debug message')
#logger.info('info message')
#logger.warning('warn message')
#logger.error('error message')
#logger.critical('critical message')


# In[2]:


# Librerias Necesarias

from urllib.request import urlopen
from urllib.error import HTTPError
from urllib.error import URLError
from bs4 import BeautifulSoup

import pandas as pd
import numpy as np
import re

import os
```

```
58
59 import requests
60 import random
61 import time
62 from time import sleep
63
64 # Normalizaci n textos (quitar codigos con acentos)
65 from unicodedata import normalize
66
67
68 # # 00.- Funciones Auxiliares
69
70 # In[3]:
71
72
73 # Extraccion informacion del nombre archivo de un URL
74 # Funci n Auxiliar
75
76 # https://stackoverflow.com/questions/66876071/extracting-a-complex-substring-
    using-regex-with-data-from-a-string-in-python
77 # https://stackoverflow.com/questions/14473180/regex-to-get-a-filename-from-a-
    url
78
79 def ExtraeName(cadena):
80     """
81     Funci n extraccion informacion del nombre archivo de un URL
82     ExtraeName(cadena)
83     entrada:
84         cadena --> Url
85     return:
86         string final del url
87     """
88
89     regex=r"(?<=\/)[^\/\?#]+(?=[^\/]*$)"
90     matches = re.findall(regex, cadena)
91     return str(matches)
92
93
94 # In[4]:
95
96
97 # Generaci n Directorio y Devluci n Path Archivo al Directorio
98 # Funcion Auxiliar
99
100 def GenDirecFile(path, filename):
101     """
102     GenDirecFile(path, filename, r)
103     Entrada:
104         path -> Path del directorio a generar
105         filename -> Nombre Archivo a guardar
106     Devuelve:
107         path y filename
108     """
109     # Generaci n directorio si no existe
110     if not os.path.exists(path):
111         os.makedirs(path)
112     # Grabaci n en el directorio en formato binario (imagenes)
113     return os.path.join(path, filename)
114
```

```
115
116 # In[5]:
117
118
119 # Descarga Imagenes
120 # Funcion Auxiliar
121
122 def DescargaImagenes(path, link, institucion):
123     """
124     DescargaImagenes(path, link, institucion)
125     Entrada:
126         path -> Path del directorio a generar
127         link -> Link a el archivo a descargar
128         institucion -> Nombre de la institucion cuyo logo es la imagen
129     Devuelve:
130         Genera Archivo en directorio path
131     """
132     # Delays para evitar problemas
133     time.sleep(random.randint(1,2))
134     # Obtener Imagen
135     #r = requests.get(s, allow_redirects=False)
136     r = requests.get(link, allow_redirects=True)
137     # Generar Nombre Archivo
138     nombre = str(institucion)
139     # Directorio donde se generara salida
140     file01=GenDirecFile(str(path), nombre)
141     # Grabacion resultados
142     with open(file01, 'wb') as file:
143         file.write(r.content)
144
145
146 # In[6]:
147
148
149 # Grabaci n Archivo en Directorio
150 # Funcion Auxiliar
151
152 # path = IMAGENES
153 # filename = img_alt + '.jpg'
154 # buffer = r.content
155 # open(nombre, 'wb').write(r.content)
156
157 def GraFileDirec(path, filename, r):
158     """
159     GraFileDirec(path, filename, r)
160     path -> Path del directorio a generar
161     filename -> Nombre Archivo a guardar
162     r --> Objeto BeatifulSoup
163         r.content -> Contenido Stream del Objeto
164     """
165     # Generaci n directorio si no existe
166     if not os.path.exists(path):
167         os.makedirs(path)
168     # Grabaci n en el directorio en formato binario (imagenes)
169     with open(os.path.join(path, filename), 'wb') as temp_file:
170         temp_file.write(r.content)
171
172
173 # # 01.- An lisis de Links asociados a la WEB
```

```
174
175 # In[6]:
176
177
178 # 01.01- El primer paso es la identificaci n de los links asociados a cada
    Master
179
180 # Pagina WEB a analizar
181
182 URLWebPageAnalizar = 'https://www.emagister.com/master/web/sitemap'
183
184 logger.info('Analisis WEB %s', URLWebPageAnalizar)
185
186 # Lista donde se almacenaran los links encontrados
187 ListaWebs=[]
188
189 # Abrir Web
190 html = urlopen(URLWebPageAnalizar)
191
192 # Generar objeto BeatifulSoup
193 bs = BeautifulSoup(html, 'html.parser')
194 for link in bs.find_all('a'):
195     # Busca todos los elementos que esten en 'href'
196     try:
197         StringWeb = link.attrs['href']
198     except:
199         continue
200     # Comprueba que lo obtenido se corresponda con el formato de una pagina WEB
201     # match (True si es un link / False en resto)
202     match = re.search(r'https?://[^\s<>"]+|www\.[^\s<>"]+', str(StringWeb))
203     if match:
204         ListaWebs.append(link.attrs['href'])
205         logger.info('Encontrado link %s', link.attrs['href'])
206
207 # Guardar links obtenidos en Archivo Auxiliar FILE_Links
208 # - Nombre Archivo
209 FILE_Links = '01_ListaWebs.xlsx'
210 # - Genera DataFrame apartir de la lista de WEBs generadas
211 df = pd.DataFrame(ListaWebs)
212 # - Escribe Archivo
213 df.to_excel(FILE_Links)
214 logger.info('Generado Archivo %s', FILE_Links)
215
216
217 # In[7]:
218
219
220 # Nos informa del N mero Total de Links encontrados
221 logger.info('Encontrado %s links', len(ListaWebs))
222
223
224 # In[8]:
225
226
227 # 01.02.- Funci n para Construcci n del Diccionario de Informaci n de los
    links encontrados.
228
229 diccionario={}
230
```

```
def RecorrerWebs(ListaWebs):
    """
    Construye Diccionario Valores de una lista de Webs

    Funcion:
       RecorrerWebs( ListaWeb)
    Entrada:
        ListaWeb --> Lista de Direcciones de Web
    Salida:
        Diccionario con informaci n:
            Keys:
                Indice correlativo
            Values:
                - Pagina donde buscamos
                - Titulo Master
                - Link al master concreto
    """

    # Leer Webs obtenidas paso previo
    # - indice
    indice = 0
    # - Delay 1 seg para no ser banneado
    sleep(1)

    for indi, ListaWeb in enumerate(ListaWebs):
        # Delay 1 seg para no ser banneado
        sleep(1)
        # Captura de posibles errores Webs
        try:
            html = urlopen(ListaWeb)
        except HTTPError as e:
            # Lanza un aviso con indice de ListaWebs y error
            logger.warning('%s warning %s',indi, e)
            continue
        except URLError as e:
            # Lanza aviso si no encuentra la Web
            logger.warning('%s ',ListaWeb)
            logger.warning('%s ',e)
            continue
        # Captura posibles errores Beatifulsoup
        try:
            bs = BeautifulSoup(html, 'html.parser')
        except AttributeError as e:
            logger.warning('%s warning %s',indi, e)
            continue

        # Captura de informaci n b sica
        divs = bs.find_all(class_ ="course-box-item title-box")

        for s in divs:
            inks = s.find_all('h3')
            for t in inks:
                lista = t.find_all('a')
                # Obtenci n indice y valor de lista
                #for indice, x in enumerate(lista):
                for x in lista:
                    indice = indice+1
                    # Titulo
                    x2 = x.attrs['title']
```

```
                    # Referencia al titulo
                    x1 = x.attrs['href']
                    logger.info('%s: %s',indice, x2)

                    # Construccion diccionario salida
                    diccionario[indice] = [ListaWeb,x2,x1]
    return diccionario


# In[9]:


# 01.03.- Construcci n Efectiva del Diccionario

# Nombre Diccionario Construido:
#    a --> diccionario

a = RecorrerWebs(ListaWebs)

logger.info('Construido Diccionario')


# 01.04.- Guardamos la informaci n obtenida en un Archivo para ello:
#
#         01.04.01.- Generamos Dataframe
#         01.04.02.- Generamos File a partir del Dataframe
#
#  Pasos:

# In[10]:


# 01.04.01.- Generaci n de Dataframe a partir del Diccionario

# https://stackoverflow.com/questions/54416620/dictionary-with-multiple-key-
    values-to-dataframe

TablaResult = (pd.DataFrame.from_dict(a, orient='index')
                   .rename(columns=lambda x: x+1)
                   .add_prefix('n')
                   .rename_axis('indice')
                   .reset_index())

logger.info('Construida Tabla TablaResult')

# Poner nombres columnas
TablaResult.columns = ['key','Pagina_Origen', 'Titulo', 'Pag_Referencia']

# Poner de indice a key
TablaResult=TablaResult.set_index('key')

# Verificaci n Construcci n Ok( No necesario )
# TablaResult.head(3)

logger.info('Tama o tabla  %s', TablaResult.shape)
logger.info('Columnas  %s', TablaResult.columns)


# In[11]:
```

```
348
349
350 # 01.04.02.- Generaci n Archivo parcial de resultados
351
352 # Generaci n Nombres Archivos a generar
353 from datetime import datetime
354
355 nombre=datetime.today().strftime('%d-%m-%y')
356
357 FILE_01 = "LINK-"+nombre+".xlsx"
358
359 TablaResult.to_excel(FILE_01)
360 logger.info('Generado Archivo %s', FILE_01)
361
362
363 # # 02.- Analisis de Links Identificados
364 #
365 #     02.1- Para ello leeremos Archivo del paso previo
366 #     02.2- Haremos el analisis
367
368 # In[12]:
369
370
371 # 02.01.- Identificaci n Archivos en nuestra Maquina
372
373 # Directorio donde estamos
374 cwd=os.getcwd()
375 print("-------------------------------------------------------------------------------")
376 print("A.- Directorio en maquina donde estamos")
377 print (cwd)
378
379 # Lista de Archivos disponibles en directorio 'Directorio'
380 Directorio = cwd
381 print("-------------------------------------------------------------------------------")
382 print("C.- Listado de Archivos Disponibles")
383 ListaFile=os.listdir(Directorio)
384
385 # Selecci n Archivos que hayan sido generados en lanzamientos previos del pto
    01.
386 # - Uso Regex
387 cadena=r"^LINK-"
388 prog1 = re.compile(cadena)
389
390 # Extraer lista de archivos que cumplan criterio
391
392 Archivos_FILT=[]
393 for s in ListaFile:
394     if prog1.search(s):
395         Archivos_FILT.append(s)
396 # Listado de Nombres Archivos validos
397 Archivos_FILT
398
399
400 # In[13]:
401
402
403 # Escogeremos el archivo que nos interese de los identificados en el paso previo
404
405 # Leer Archivo
```

```
406 FILE_02 = 'LINK-27-03-22.xlsx'
407
408 df = pd.read_excel(FILE_02, index_col=None)
409
410 logger.info('Leido Archivo %s', FILE_02)
411
412
413 # In[14]:
414
415
416 # Funci n Lectura Informaci n Webs secundarias de Masters
417
418 def WebSecundaria(Link):
419     """
420     Funcion:
421         WebSecundaria(link, indice)
422         Version 2.0
423     Entrada:
424         link -> Link a analizar
425
426     Return:
427         Lista valores obtenidos
428         - Titulo -> Titulo Master
429         - Entidad -> Nombre entidad que hace master
430         - urlImagen -> Link Imagen
431         - nombreImagen -> Nombre Imagen
432         - Telefono -> Telefono contacto
433         - Precio -> Precio Master
434         - Metodologia -> Metodologia Master
435         - Lugar -> Lugar Imparticion
436         - Duracion -> Duracion Master
437         - Tipologia -> Tipologia Master
438         - BolsaEmp -> Bolsa Empleo
439         - x3 -> Precio Master
440         - x4 -> Texto Descriptivo Master
441         - x5 -> Tipologia Master
442         - x6 -> Programa Master
443     """
444     # Abrir Web
445     # Captura de posibles errores Webs
446
447     # Delays para evitar problemas
448     time.sleep(random.randint(1,2))
449
450     try:
451         html = urlopen(Link)
452     except HTTPError as e:
453         # Lanza un aviso con indice de ListaWebs y error
454         logger.warning('warning %s', e)
455         return
456     except URLError as e:
457         # Lanza aviso si no encuentra la Web
458         logger.warning('%s ',e)
459         return
460
461     # Generar objeto BeatifulSoup
462     bs = BeautifulSoup(html, 'html.parser')
463
464     # Titulo Master
```

```
    for t1 in bs.select(".title-box__name"):
        #print(t1.text)
        x1 = t1.text
        Titulo = x1.strip()

    # Nombre entidad que hace el Master
    for t1 in bs.select(".course-box__link"):
        #print(t1.text)
        Entidad = t1.text.strip()

    # Imagen en pagina Master
    a=bs.select(".boxes-untrack__logo > img:nth-child(1)")
    # Si hay imagen
    if a:
        # Link Imagen
        urlImagen=a[0]['src'].strip()
        # Extraer nombre Archivo Imagen
        nombreImagen= ExtraeName(urlImagen)
    else:
        urlImagen=""
        nombreImagen="Sin Imagen"

    # Telefono
    a=bs.select(".app_contactCenterSectionDesktop:nth-child(2) > div:nth-child
    (1) > div:nth-child(1) > div:nth-child(2) > a:nth-child(1) > span:nth-child
    (1)")
    if a:
        # Extracci n Telefono
        Telefono=a[0].text
        # - Elimina caracteres especiales
        Telefono=re.sub(r"[^a-zA-Z0-9]","",Telefono)
        # - Elimina espacios blanco principio y final
        Telefono=Telefono.strip()
    else:
        Telefono="Sin Datos"

     # Precio
    a=bs.select(".app_course_price_box > div:nth-child(2) > div:nth-child(1) >
    span:nth-child(1)")
    if a:
        # Extracci n Precio
        Precio=a[0].text
        # - Elimina caracteres especiales
        Precio=re.sub(r"[^a-zA-Z0-9]","",Precio)
        # - Elimina espacios blanco principio y final
        Precio=Precio.strip()
        # - Si todos los caracteres son numericoa convierte a float
        if Precio.isnumeric():
            # Convertir en float
            Precio=float(Precio)

    else:
        Precio="Precio a consultar"

    # Metodologia
    a=bs.select("ul.course-detail__list:nth-child(1) > li:nth-child(2) > p:nth-
    child(2) > span:nth-child(1)")
    if a:
        # Metodologia
```

```
520             Metodologia=a[0].text.strip()
521         else:
522             Metodologia="Sin Datos"
523 
524         # Lugar impartici n Master
525         a=bs.select("ul.course-detail__list:nth-child(1) > li:nth-child(3) > p:nth-
            child(2) > span:nth-child(1)")
526         if a:
527             # Lugar
528             Lugar=a[0].text.strip()
529         else:
530             Lugar="Sin Datos"
531 
532         # TipoCurso
533         a=bs.select("ul.course-detail__list:nth-child(1) > li:nth-child(4) > p:nth-
            child(2) > span:nth-child(1)")
534         if a:
535             # TipoCurso
536             TipoCurso=a[0].text.strip()
537         else:
538             TipoCurso="Sin Datos"
539 
540         # Tipologia
541         a=bs.select("ul.course-detail__list:nth-child(1) > li:nth-child(1) > p:nth-
            child(2) > span:nth-child(1)")
542         if a:
543             # Tipologia
544             Tipologia=a[0].text.strip()
545         else:
546             Tipologia="Sin Datos"
547 
548          # Bolsa Empleo
549         a=bs.select("ul.course-detail__list:nth-child(2) > li:nth-child(4) > p:nth-
            child(2) > span:nth-child(1)")
550         if a:
551             # Bolsa Empleo (Si/No)
552             BolsaEmpl=a[0].text.strip()
553         else:
554             BolsaEmpl="Sin Datos"
555 
556         # Precio del Master
557         # - Si no tiene valor asigna precio Sin Datos
558         if not bs.select("div.price-box:nth-child(1) > div:nth-child(1) > div:nth-
            child(1) > span:nth-child(1)"):
559             x3= "SinDatos"
560         # - En el resto de casos
561         for t1 in bs.select("div.price-box:nth-child(1) > div:nth-child(1) > div:nth
            -child(1) > span:nth-child(1)"):
562             # - Obtenci n del valor
563             x31 = t1.text
564             # - Elimina caracteres especiales
565             x3=re.sub(r"[^a-zA-Z0-9]","",x31)
566             # - Elimina espacios blanco principio y final
567             x3=x3.strip()
568             # - Si todos los caracteres son numericoa convierte a float
569             if x3.isnumeric():
570                 # Convertir en float
571                 x3=float(x3)
572                 # Definirlo como precio ( La informaci n esta en dos sitios )
```

```
573             Precio=x3
574
575     # Texto Descriptivo
576     # - Si no tiene valor asigna cadena vacia
577     if not bs.select(".course-box__text"):
578         x4="SinDatos"
579     # - En el resto de casos
580     for t1 in bs.select(".course-box__text"):
581         # - Obtenci n del valor
582         x41 = t1.text
583         # - Eliminamos acentos. Basado en:
584 # https://es.stackoverflow.com/questions/135707/c%C3%B3mo-puedo-reemplazar-las-
    letras-con-tildes-por-las-mismas-sin-tilde-pero-no-l
585         # -> NFD y eliminar diacr ticos
586         x4 = re.sub(r"([^n\u0300-\u036f]|n(?!\u0303(?![\u0300-\u036f])))[\u0300
    -\u036f]+", r"\1",
587         normalize("NFD", x41), 0, re.I)
588         # -> NFC
589         x42 = normalize( 'NFC', x4)
590
591         # - Eliminamos caracteres especiales
592         x4=re.sub(r"[^a-zA-Z0-9]"," ",x42)
593         # - Elimina espacios blanco principio y final
594         x4=x4.strip()
595
596
597     # Tipologia del Master
598     # - Si no tiene valor asigna cadena vacia
599     if not bs.select(".course-venues__address > span:nth-child(1)"):
600         x5=""
601     # - En el resto de los casos
602     for t1 in bs.select(".course-venues__address > span:nth-child(1)"):
603         # Si no hay valores
604         if not t1:
605             x5="Sin Datos"
606         #print(t1.text)
607         x5 = t1.text.strip()
608
609     # Programa Master
610     texto=[]
611     for table in bs.select(".lessons-box.app_lessons-box"):
612         for i, tr in enumerate(table.findAll('strong')):
613             texto.append(tr.text)
614             for td in tr.findAll('li'):
615                 texto.append(td.text)
616     x6 = texto
617
618     # Construcci n Lista resultado
619 #         - Titulo -> Titulo Master -- 'MASTER'
620 #         - Entidad -> Nombre entidad que hace master -- 'ENTIDAD'
621 #         - Precio -> Precio Master -- 'PRECIO'
622 #         - TipoCurso -> Tipo Curso -- 'TIPO_CURSO'
623 #         - Tipologia -> Tipologia Master -- 'TIPOLOGIA'
624 #         - Metodologia -> Metodologia Master -- 'METODOLOGIA'
625 #         - Lugar -> Lugar Imparticion -- LUGAR
626 #         - BolsaEmp -> Bolsa Empleo -- 'BOLSAEMP'
627 #         - Telefono -> Telefono contacto --  'TELEFONO'
628 #         - x4 -> Texto Descriptivo Master
629 #         - x6 -> Programa Master
```

```
#         - urlImagen -> Link Imagen
#         - nombreImagen -> Nombre Imagen
#         'WEB'

#         - x3 -> Precio Master (NO LO USAREMOS)
#         - x5 -> Tipologia Master (NO LO USAREMOS)


    logger.info('Procesada %s', x1)
    return (Titulo,
            Entidad,
            Precio,
            Tipologia,
            Metodologia,
            Lugar,
            BolsaEmpl,
            Telefono,
            x4,
            x6,
            urlImagen,
            nombreImagen,
            Link)


## 03.- ANALISIS
#
# Tenemos dos opciones:
#
#     03.01- SECUENCIAL
#     03.02.-MULTIPROCESO

# In[ ]:


# 03.01.-PROCESO SECUENCIAL de generaci n Datos Master

# Datos --> DataFrame df
#  - Columnas:
#       'key'-> Indice
#       'Pagina_Origen' -> Web Origen Busqueda
#     , 'Titulo' -> Nombre Master
#     , 'Pag_Referencia' --> Web de detalle Master
#

# Construcci n Lista Webs a analizar
listadoWebs = df['Pag_Referencia'].tolist()
# Tama o Datos
#tamano = len(listadoWebs)
tamano = 5

# Control Tiempo
start = time.time()

logger.info('*********************************')
logger.info('*   INICIO PROCESO SECUENCIAL   *')
logger.info('*********************************')

# Para cada valor del indice
for indice in range(tamano):
```

```
    # Paro para evitar bloqueo
    sleep(1)
    # Construcci n Diccionario
    #s = WebSecundaria(listadoWebs[indice], indice)
    resultado_seq = WebSecundaria(listadoWebs[indice])
    #logger.info('%s: Informacion de %s', indice, s[indice][2])

logger.info('*********************************')
logger.info('*     FIN PROCESO SECUENCIAL    *')
logger.info('*********************************')

end = time.time()
duracion = (end-start)
logger.info('El proceso secuencial duro %s segundos', duracion)


# In[15]:


# 03.02.-PROCESO USANDO MULTIPROCESO de generaci n Datos Master

from multiprocessing import Pool


# In[16]:


# Lista de web a analizar

# Construcci n Lista Webs a analizar
listadoWebs = df['Pag_Referencia'].tolist()


# Para hacer pruebas
url_list=listadoWebs[0:1000]


# In[17]:


url_list


# In[18]:


# https://medium.com/@kunal.rustagi/boost-your-web-crawler-using-multiple-
    processes-in-python-3cc3ff519226

start = time.time()

logger.info('*********************************')
logger.info('*    INICIO MULTIPROCESO        *')
logger.info('*********************************')

p = Pool(10)
resultados_mul = p.map(WebSecundaria, url_list)
p.terminate()
p.join()
```

```
747
748
749
750 logger.info('*******************************')
751 logger.info('*      FIN MULTIPROCESO         *')
752 logger.info('*******************************')
753
754 end = time.time()
755 duracion = (end-start)
756 logger.info('La secuencia Multiproceso duro %s segundos', duracion)
757
758
759 # In[19]:
760
761
762 # 03.03.- Creaci n Dataframe resultado
763
764 DfResult = pd.DataFrame(resultados_mul, columns = ['MASTER',
765                                                    'ENTIDAD',
766                                                    'PRECIO',
767                                                    'TIPOLOGIA',
768                                                    'METODOLOGIA',
769                                                    'DURACION',
770                                                    'BOLSAEMP',
771                                                    'TELEFONO',
772                                                    'X4',
773                                                    'X6',
774                                                    'URLIMAGEN',
775                                                    'NOMBREIMAGEN',
776                                                    'WEB'])
777
778
779 # In[20]:
780
781
782 DfResult
783
784
785 # In[ ]:
786
787
788 DfResult.columns
789
790
791 # In[21]:
792
793
794 # 03.04.- Generaci n Archivo parcial de resultados
795
796 # Generaci n Nombres Archivos a generar
797 from datetime import datetime
798
799 nombre=datetime.today().strftime('%d-%m-%y')
800
801 FILE_02 = "RESULT-"+nombre+".xlsx"
802
803 # Directorio donde se generara salida
804 path='RESULTADOS'
805 file=GenDirecFile(path, FILE_02)
```

```
806
807 # Escritura
808 DfResult.to_excel(file)
809 logger.info('Generado Archivo %s', FILE_02)
810
811
812 # In[22]:
813
814
815 # Extracci n Imagenes de las WEBs
816
817 # Construccion diccionario imagenes de las entidades
818
819 # Filtrado Entidades con Logo
820 #DfResultLOGO=DfResult[DfResult['URLIMAGEN']!='']
821
822 # Lista entidades unicas
823 listaOrganizaciones = DfResult['ENTIDAD'].unique().tolist()
824
825 # Construccion diccionario imagenes de las entidades
826 DiccioImagenes = {}
827 # Para cada entidad
828 for s in listaOrganizaciones:
829     # Hacer entrada por entidad con lista de imagenes de la entidad
830     DiccioImagenes[s] = DfResult[DfResult['ENTIDAD'] == s]['URLIMAGEN'].unique()
    .tolist()
831
832 logger.info('Generado Diccionario de tama o %s', len(DiccioImagenes))
833
834
835 # In[23]:
836
837
838 # Prueba
839
840 DiccioImagenes['Instituto Espa ol de Formaci n Social']
841
842
843 # In[24]:
844
845
846 # Directorio Imagenes
847 Path = 'IMAGENES'
848
849 logger.info('Inicio Descarga Imagenes')
850
851 # Itera en el diccionario Previo
852 for k, v in DiccioImagenes.items():
853     # Descarga Archivo en Path
854     # v --> Lista de Links de imagenes
855     # k --> Institucion
856     # Archivo resultante con nombre
857     logger.info('Descargando Logo de %s', k)
858     # Pasar lista a String
859     v = v[0]
860     logger.info('Link %s', v)
861     # - Eliminamos caracteres especiales
862     k=re.sub(r"[^a-zA-Z0-9]"," ",k)
863     # - Elimina espacios blanco principio y final
```

```
    k=k.strip()
    DescargaImagenes(Path, v, k)

logger.info('Fin Descarga Imagenes')


# In[ ]:
```