LAB 3: Data Collection and Creation

- Create/make your own data-set
- Using Web Scraping- Static and Dynamic Webpages

3.1: Web Scraping using Beautiful Soup

- Create your own data set from already Available data but not directely downloadable

Web Scraping using Beautiful Soup

Web Scraping (also termed Screen Scraping, Web Data Extraction, Web Harvesting etc.) is a technique employed to extract large amounts of data from websites whereby the data is extracted and saved to a local file in your computer or to a database in table (spreadsheet) format.

Task: Extract all relevant information about the movies from the webpage

IMDB_scrapper:

This is a script written to extract all relevant information related to top 100 movies from the website

1. Following features were considered:
    1. Date of scraping
    2. title (movie name)
    3. certification
    4. duration of the movie (time)
    5. genre
    6. release date
    7. release country
    8. rating
    9. users
    10. critic
    11. summary of the movie
    12. director
    13. writer
    14. primary-actor
    15. meta score
    16. primary image
    17. primary video
    18. other images link
    19. other video link

20. all actors and their characters in the movie
21. plot
22. plot keywords
23. languages
24. filming_location
25. 1 budget
26. opening_weekend
27. gross_amount
28. cumilative_gross
29. production_company
30. sound_mix
31. aspect_ratio

2. All the above features are extracted for top 100 movies listed on the webpage.
3. After extraction, all the information about each movie is stored in a json format.

Step 1: Legal and imports

```
!pip install BeautifulSoup4 tqdm
```

```
Requirement already satisfied: BeautifulSoup4 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (4.41.1)
```

```
import json
import pandas as pd #pip install pandas
from datetime import date
from datetime import datetime
from bs4 import BeautifulSoup #pip install BeautifulSoup4
from tqdm import tqdm_notebook as tqdm #pip install tqdm
from urllib.request import urlopen, Request
from pprint import pprint
from IPython.display import display
from ipywidgets import Checkbox


def str_formatter(string, pp, cycle):
    pp.text(string)

plain = get_ipython().display_formatter.formatters['text/plain']
plain.for_type(str, str_formatter)
```

```
<function IPython.lib.pretty._repr_pprint>
```

```
webpage='https://www.imdb.com/search/title/?count=100&groups=top_1000&sort=user_rating%27'


box = Checkbox(False, indent=False)
display(box, f"I, {input('Enter your name: ')}, agree to the above Legal and Ethical concerns
```

```
Enter your name: Rohit Byas Sherwan
```

☐

```
I, Rohit Byas Sherwan, agree to the above Legal and Ethical concerns. If I do anything,
```

```
hdr = {
  'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
  'From': 'nitdelhi.ac.in',
  'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8',
  'Accept-Charset': 'ISO-8859-1,utf-8;q=0.7,*;q=0.3',
  'Accept-Encoding': 'none',
  'Accept-Language': 'en-US,en;q=0.8',
}


def extract_soup_object(link):
  request=Request(link, headers=hdr)
  html = urlopen(request).read().decode()
  soup = BeautifulSoup(html,'html.parser')
  return soup


extract_soup_object(webpage).prettify()


soup=extract_soup_object(webpage)
data=soup.find_all('h3',{'class':"lister-item-header"})
data[0]
```

```
<h3 class="lister-item-header">
<span class="lister-item-index unbold text-primary">1.</span>
<a href="/title/tt2948372/?ref_=adv_li_tt">Soul</a>
<span class="lister-item-year text-muted unbold">(2020)</span>
</h3>
```

```
for item in data:
  print(item.a.get('href'))
count = len(data)
count
```

```
/title/tt2948372/?ref_=adv_li_tt
/title/tt10288566/?ref_=adv_li_tt
/title/tt4154796/?ref_=adv_li_tt
/title/tt8367814/?ref_=adv_li_tt
/title/tt6751668/?ref_=adv_li_tt
/title/tt0068646/?ref_=adv_li_tt
/title/tt8946378/?ref_=adv_li_tt
```

```
        /title/tt8579674/?ref_=adv_li_tt
        /title/tt0241527/?ref_=adv_li_tt
        /title/tt1070874/?ref_=adv_li_tt
        /title/tt0111161/?ref_=adv_li_tt
        /title/tt0120737/?ref_=adv_li_tt
        /title/tt0993846/?ref_=adv_li_tt
        /title/tt7131622/?ref_=adv_li_tt
        /title/tt0816692/?ref_=adv_li_tt
        /title/tt1392190/?ref_=adv_li_tt
        /title/tt7286456/?ref_=adv_li_tt
        /title/tt4154756/?ref_=adv_li_tt
        /title/tt0468569/?ref_=adv_li_tt
        /title/tt1375666/?ref_=adv_li_tt
        /title/tt3501632/?ref_=adv_li_tt
        /title/tt3281548/?ref_=adv_li_tt
        /title/tt0099685/?ref_=adv_li_tt
        /title/tt0110912/?ref_=adv_li_tt
        /title/tt0414387/?ref_=adv_li_tt
        /title/tt0264464/?ref_=adv_li_tt
        /title/tt8503618/?ref_=adv_li_tt
        /title/tt2584384/?ref_=adv_li_tt
        /title/tt0109830/?ref_=adv_li_tt
        /title/tt0361748/?ref_=adv_li_tt
        /title/tt0133093/?ref_=adv_li_tt
        /title/tt6966692/?ref_=adv_li_tt
        /title/tt0407887/?ref_=adv_li_tt
        /title/tt0162222/?ref_=adv_li_tt
        /title/tt1856101/?ref_=adv_li_tt
        /title/tt3748528/?ref_=adv_li_tt
        /title/tt0076759/?ref_=adv_li_tt
        /title/tt2015381/?ref_=adv_li_tt
        /title/tt0167260/?ref_=adv_li_tt
        /title/tt0120338/?ref_=adv_li_tt
        /title/tt1201607/?ref_=adv_li_tt
        /title/tt0114369/?ref_=adv_li_tt
        /title/tt0099785/?ref_=adv_li_tt
        /title/tt9071322/?ref_=adv_li_tt
        /title/tt0088763/?ref_=adv_li_tt
        /title/tt0482571/?ref_=adv_li_tt
        /title/tt1568346/?ref_=adv_li_tt
        /title/tt1950186/?ref_=adv_li_tt
        /title/tt0137523/?ref_=adv_li_tt
        /title/tt0330373/?ref_=adv_li_tt
        /title/tt0059742/?ref_=adv_li_tt
        /title/tt0477348/?ref_=adv_li_tt
        /title/tt1392214/?ref_=adv_li_tt
        /title/tt0167261/?ref_=adv_li_tt
        /title/tt0903624/?ref_=adv_li_tt
        /title/tt0120815/?ref_=adv_li_tt
        /title/tt1302006/?ref_=adv_li_tt
        /title/tt1853728/?ref_=adv_li_tt
        /title/tt0454848/?ref =adv li tt
```

```
def extract_links(page):
  soup=extract_soup_object(page)
  data=soup.find_all('h3',{'class':"lister-item-header"})
  return ['https://www.imdb.com'+item.a.get('href') for item in data]
```

```
movie_links=extract_links(webpage)
movie_links[0:5]
```

```
[https://www.imdb.com/title/tt2948372/?ref_=adv_li_tt,
 https://www.imdb.com/title/tt10288566/?ref_=adv_li_tt,
 https://www.imdb.com/title/tt4154796/?ref_=adv_li_tt,
 https://www.imdb.com/title/tt8367814/?ref_=adv_li_tt,
 https://www.imdb.com/title/tt6751668/?ref_=adv_li_tt]
```

Step 2: Making Functions of all the attributes

```
def get_scraping_date():
  now = datetime.now()
  date=now.strftime("%d-%m-%Y %H:%M:%S")
  return date


def movie_name(soup):
  return soup.find('div',{'class':'title_wrapper'}).find('h1').text.split('\xa0')[0]


def other_attributes_part1(soup):
  temp=soup.find('div',{'class':'title_wrapper'}).find('div',{'class': 'subtext'}).text.repla
  certification=temp[0].split('\n')[1]
  time=temp[1].split('\n')[1]
  genre=temp[2].replace('\n','')
  release_date=temp[3].split('\n')[1].split('(')[0]
  release_country=temp[3].split('\n')[1].split('(')[1].split(')')[0]
  return certification, time, genre, release_date, release_country


def movie_rating(soup):
    return soup.find('div',{'class':'ratingValue'}).text.replace('\n','')

def user_critic(soup):
    q=soup.find('div',{'class':'imdbRating'}).find('div',{'class': 'hiddenImportant'}).find_a
    temp=[item.text for item in q]
    user,critic=temp[0].split(' ')[0],temp[1].split(' ')[0]
    return user, critic

def summary(soup):
    return soup.find('div',{'class':'plot_summary'}).find('div',{'class': 'summary_text'}).te

def writer_director_stars(soup):
    t=soup.find_all('div',{'class':'credit_summary_item'})
    temp=[item.text for item in t]
    director=temp[0].split(':')[1].replace('\n','')
    writer=temp[1].split(':')[1].replace('\n','')
    primary_actor=temp[2].split(':')[1].replace('\n','').split('|')[0]
    return director, writer, primary_actor
```

```
        return director, writer, primary_actor

    def meta_score(soup):
        return soup.find('div',{'class':'metacriticScore score_favorable: titleReviewBarSubItem'}

    def primary_image(soup):
        return soup.find('div',{'class':'poster'}).find('img')['src']

    def primary_video(soup):
        return 'https://www.imdb.com'+soup.find('div',{'class': 'videoPreview__videoContainer'}).

    def other_image_video(soup):
        temp=soup.find_all('div',{'class':'combined-see-more see-more'})
        temp_1=[item.find('a')['href'] for item in temp]
        image=temp_1[0]
        video=temp_1[1]
        return image, video

    def all_actors(soup, value):
        temp=soup.find_all('tr', {'class':value})
        return [item.text for item in temp]

    def beautify(temp_list):
        actor=[]
        character=[]
        for item in temp_list:
            actor.append(item.split('...')[0])
            character.append(item.split('...')[1])
        actors=[item.replace('\n','').replace(' ','') for item in actor]
        characters=[item.replace('\n','').replace(' ','') for item in character]
        return actors, characters

    def plot(soup):
        return soup.find('div',{'class':'inline canwrap'}).find('span').text. replace(' ','')

    def plot_keywords(soup):
        keywords=[]
        temp=soup.find('div',{'class':'see-more inline canwrap'}).text. replace('\n','')
        keywords.append(temp.split('|')[0].split(':')[1])
        keywords.extend(temp.split('|')[1:-1])
        return keywords



    def other_attributes_part2(soup):
      h=soup.find('div',{'id':'titleDetails'}).find_all('div',{'class': 'txt-block'})
      details=[]
      for item in h:
        try:
          details.append(item.text.split(':')[1])
        except:
          continue
      languages_details[2].replace('\n','').split('|')
```

```
    languages=details[2].replace('\n','').split('|')
    filming_location=details[5].replace('\n','').replace('\xa0»','')
    budget=details[6].replace('\n','').split('(')[0]
    opening_weekend=details[7].replace('\n','')
    gross_amount=details[8].replace(' ','')
    cumilative_gross=details[9].replace(' ','')
    production_company=details[10].replace('\n','').replace('\xa0»','')
    sound_mix=details[12].replace('\n','').split('|')
    aspect_ratio=details[-1]
    return languages, filming_location, budget, opening_weekend, gross_amount, cumilative_gross



def imdb_scrapper(link):
  data={}
  data['website']='IMDB'
  data['link']=link
  scrape_date=get_scraping_date()
  data['scrape_date']=scrape_date
  name=link.split('/')[4]
  data['unique_id']=name
  soup=extract_soup_object(link)
  try:
    data['movie_name']=movie_name(soup)
  except:
    data['movie_name']=None
  try:
    certification, time, genre, release_date, release_country=other_attributes_part1(soup)
    data['certification']=certification
    data['time']=time
    data['genre']=genre
    data['release_date']=release_date
    data['release_country']=release_country
  except:
    data['certification']=None
    data['time']=None
    data['genre']=None
    data['release_date']=None
    data['release_country']=None
  try:
    data['movie_rating']=movie_rating(soup)
  except:
    data['movie_rating']=None
  try:
    user, critic=user_critic(soup)
    data['user']=user
    data['critic']=critic
  except:
    data['user']=None
    data['critic']=None
  try:
    data['summary']=summary(soup)
```

```
    except:
      data['summary']=None
    try:
      director, writer, primary_actor=writer_director_stars(soup)
      data['director']=director
      data['writer']=writer
      data['primary_actor']=primary_actor
    except:
      data['director']=None
      data['writer']=None
      data['primary_actor']=None
    try:
      data['meta_score']=meta_score(soup)
    except:
      data['meta_score']=None
    try:
      data['primary_image']=primary_image(soup)
    except:
      data['primary_image']=None
    try:
      data['primary_video']=primary_video(soup)
    except:
      data['primary_video']=None
    try:
      image_links,video_links=other_image_video(soup)
      data['other_image_links']=image_links
      data['other_video_links']=video_links
    except:
      data['other_image_links']=None
      data['other_video_links']=None
    try:
      odd_actors=all_actors(soup, 'odd')
      even_actors=all_actors(soup, 'even')
      actors=odd_actors+even_actors
      actors, characters=beautify(actors)
      data['all_actors']=actors
      data['all_characters']=characters
    except:
      data['all_actors']=None
      data['all_characters']=None
    try:
      data['plot']=plot(soup)
    except:
      data['plot']=None
    try:
      data['plot_keywords']=plot_keywords(soup)
    except:
      data['plot_keywords']=None
    try:
      languages, filming_location, budget, opening_weekend, gross_amount, cumilative_gross, pro
      data['languages']=languages
      data['filming_location']=filming_location
```

```
    data['budget']=budget
    data['opening_weekend']=opening_weekend
    data['gross_amount']=gross_amount
    data['cumilative_gross']=cumilative_gross
    data['production_company']=production_company
    data['sound_mix']=sound_mix
    data['aspect_ratio']=aspect_ratio
  except:
    data['languages']=None
    data['filming_location']=None
    data['budget']=None
    data['opening_weekend']=None
    data['gross_amount']=None
    data['cumilative_gross']=None
    data['production_company']=None
    data['sound_mix']=None
    data['aspect_ratio']=None
  return data



imdb_scrapper('https://www.imdb.com/title/tt0107290/?ref_=adv_li_tt')
```

```
    {all_actors: [SamNeill,
      JeffGoldblum,
      BobPeck,
      BDWong,
      ArianaRichards,
      WayneKnight,
      MiguelSandoval,
      ChristopherJohnFields,
      LauraDern,
      RichardAttenborough,
      MartinFerrero,
      JosephMazzello,
      SamuelL.Jackson,
      GeraldR.Molen,
      CameronThor],
     all_characters: [Grant,
      Malcolm,
      Muldoon,
      Wu(asB.D.Wong),
      Lex,
      Nedry,
      Rostagno,
      Volunteer#1,
      Ellie,
      Hammond,
      Gennaro,
      Tim,
      Arnold,
      Harding(asJerryMolen),
      Dodgson],
     aspect_ratio:  1.37 ,
```

```
budget: $63,000,000            ,
certification: PG-13,
critic: 358,
cumilative_gross: $1,033,927,022,
director: Steven Spielberg ,
filming_location: Jurassic Kahili Ranch, Kilauea, Kauai, Hawaii, USASee more      ,
genre: Action,Adventure,Sci-Fi,
gross_amount: $404,214,720,
languages: [English, Spanish],
link: https://www.imdb.com/title/tt0107290/?ref_=adv_li_tt,
meta_score: None,
movie_name: Jurassic Park,
movie_rating: 8.1/10 ,
opening_weekend:  $47,026,828,13 June 1993 ,
other_image_links: /title/tt0107290/videogallery?ref_=tt_pv_vi_sm,
other_video_links: /title/tt0107290/mediaindex?ref_=tt_pv_mi_sm,
plot: Hugeadvancementsinscientifictechnologyhaveenabledamogultocreateanislandfullofl
plot_keywords: [ dinosaur,
  tyrannosaurus rex,
  sneeze,
  severed arm,
  bipedal dinosaur],
primary_actor: Sam Neill, Laura Dern, Jeff Goldblum ,
primary_image: https://m.media-amazon.com/images/M/MV5BMjM2MDgxMDg0Nl5BMl5BanBnXkFtZ
primary_video: None,
production_company:  Universal Pictures, Amblin Entertainment See more      ,
release_country: USA,
```

Part 2 : Web Scraping from dynamic page

1. Create your own data set from already Available data but not directely downloadable
2. Case Study 1: Visit this website : https://timesofindia.indiatimes.com/news
3. Check how we can download the data from here

```
rid = 2
page = urlopen(Request("https://timesofindia.indiatimes.com/news/"+str(rid), headers=hdr)).re
soup = BeautifulSoup(page, 'html.parser')
print(soup.prettify())
```

```
          <a href="#">
           Times Syndication Service
          </a>
         </p>
        </div>
        <!--/toifooter.cms?s=&version=4&newlogin=potime:16-->
       </footer>
       <!--Server: 140-->
       <link href="/slick_css/version-1,minify-1.cms" media="screen" rel="stylesheet" typ
       <script src="/slick_min_js_v18/version-1,minify-1.cms" type="text/javascript">
       </script>
       <script src="https://image.timespoints.iimg.in/static/tpsdk/tp-sdk.js" type="text/
       </script>
       <script src="/jsrender/version-1.cms" type="text/javascript">
```

```
    </script>
    <script src="https://jssocdn.indiatimes.com/crosswalk/jsso_crosswalk_legacy_0.5.3.
    </script>
    <script async="true" defer="true" src="https://www.google.com/recaptcha/api.js">
    </script>
    <link href="/usermanagementcssv2/version-21,minify-1.cms" media="screen" rel="styl
    <script src="/toi_js/version-598,minify-1.cms" type="text/javascript">
    </script>
    <script src="/pn/version-39,minify-1.cms" type="text/javascript">
    </script>
    <script>
     (function(){
           if (TimesGDPR && TimesGDPR.common.consentModule.gdprCallback){
                 TimesGDPR.common.consentModule.gdprCallback(function(dataObj){ //Bloc
                     var showJs = !dataObj.isEUuser || (dataObj && dataObj.isEUuser &&

                     if(showJs){
                         var pbjsEl = document.createElement("script");
                         pbjsEl.type = "text/javascript";
                         pbjsEl.async = true;
                         pbjsEl.src = "https://static.clmbtech.com/ase/2658/3/aa.js";
                         pbjsEl.onload = function(){

                         }
                         var pbjsTargetEl = document.getElementsByTagName("head")[0];
                         pbjsTargetEl.insertBefore(pbjsEl, pbjsTargetEl.firstChild);
                     }
                 });
             }
         })();
    </script>
    <script id="weather" type="text/x-jquery-tmpl">

     <span class="temp_h">~district~</span><span class="temp_g"><span style="backgroun
    </script>
    <script>
     var forecastObj = new Object();
                     forecastObj.type = "weather";
                     forecastObj.cityname = "";
    </script>
    <!--/forecast.cms?type=weather&version=2potime:7-->
   </div>
  </body>
 </html>
```

Case Study 2: PiB

```
rid = 214647
page = urlopen(Request("http://pibarchive.nic.in/archive2/erelcontent.aspx?relid="+str(rid),
soup = BeautifulSoup(page, 'html.parser')
print(soup.prettify())
```

```
            <br/>
            Inspired by it, Sikhs globally have done pioneering service in several sphere
```

```
          <br/>
          <br/>
          May the Sri Guru Granth Sahib Ji keep guiding humanity forever.
          <a href="https://t.co/UFcgXPjXRz">
           pic.twitter.com/UFcgXPjXRz
          </a>

         </p>
         — Narendra Modi (@narendramodi)
         <a href="https://twitter.com/narendramodi/status/1296066158483783680?ref_src=t
          August 19, 2020
         </a>
        </blockquote>
        <script async="" charset="utf-8" src="https://platform.twitter.com/widgets.js">
        </script>
        <p style="text-align:center">
         <span style="font-family:Times New Roman,Times,serif">
          <span style="font-size:16px">
           ***
          </span>
         </span>
        </p>
        <p>
         <span style="font-family:Times New Roman,Times,serif">
          <span style="font-size:16px">
           <strong>
            VRRK/SH
           </strong>
          </span>
         </span>
        </p>
        <br>
         <span style="color:ThreeDDarkShadow;">
          (Release ID :214647)
         </span>
         <br>
         </br>
        </br>
       </div>
       <div style="float: right; padding-right: 100px; margin-top: -20px;">
        <div class="fb-share-button" data-href="http://pib.nic.in/newsite/PrintRelease.
        </div>
       </div>
       <div style="float: right; padding-right: 30px; margin-top: -20px;">
        <a class="twitter-share-button" data-text="PM greets People on Parkash Purab of
         Tweet
        </a>
       </div>
      </div>
      <br>
      </br>
     </div>
    </form>
   </body>
  </html>
```

```
a = soup.find_all('p')
for x in a:
  print(x.text)
```

```
The Prime Minister, Shri Narendra Modi has greeted the people on the  auspicious occasic
The Prime Minister said, "The Sri Guru Granth Sahib Ji teaches us service, compassion ar
The Sri Guru Granth Sahib Ji illuminates the entire world with its pure teachings.
Inspired by it, Sikhs globally have done pioneering service in several spheres. Their cc
May the Sri Guru Granth Sahib Ji keep guiding humanity forever."
The Sri Guru Granth Sahib Ji teaches us service, compassion and furthers harmony. It lay
The Sri Guru Granth Sahib Ji illuminates the entire world with its pure teachings.Inspir
***
VRRK/SH
```

Double-click (or enter) to edit

Exploring Selenium

```
!pip install selenium
!apt-get update # to update ubuntu to correctly run apt install
!apt install chromium-chromedriver
!cp /usr/lib/chromium-browser/chromedriver /usr/bin
```

```
Get:12 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:13 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu180
Hit:14 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Get:15 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease [21.3 kB
Get:16 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:17 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [1,881 k
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [2,140
Get:19 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main Sources [1,71
Get:20 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1,3
Get:21 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [2
Get:22 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [2,307 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [45.
Get:24 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [309
Get:25 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main amd64 Package
Get:26 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64 Package
Fetched 11.4 MB in 7s (1,695 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  chromium-browser chromium-browser-l10n chromium-codecs-ffmpeg-extra
Suggested packages:
  webaccounts-chromium-extension unity-chromium-extension adobe-flashplugin
The following NEW packages will be installed:
  chromium-browser chromium-browser-l10n chromium-chromedriver
  chromium-codecs-ffmpeg-extra
0 upgraded, 4 newly installed, 0 to remove and 53 not upgraded.
```

```
0 upgraded, 4 newly installed, 0 to remove and 55 not upgraded.
Need to get 81.0 MB of archives.
After this operation, 273 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 chromium-codecs-
Get:2 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 chromium-browser
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 chromium-browser
Get:4 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 chromium-chromed
Fetched 81.0 MB in 9s (8,861 kB/s)
Selecting previously unselected package chromium-codecs-ffmpeg-extra.
(Reading database ... 145483 files and directories currently installed.)
Preparing to unpack .../chromium-codecs-ffmpeg-extra_87.0.4280.66-0ubuntu0.18.04.1_am
Unpacking chromium-codecs-ffmpeg-extra (87.0.4280.66-0ubuntu0.18.04.1) ...
Selecting previously unselected package chromium-browser.
Preparing to unpack .../chromium-browser_87.0.4280.66-0ubuntu0.18.04.1_amd64.deb ...
Unpacking chromium-browser (87.0.4280.66-0ubuntu0.18.04.1) ...
Selecting previously unselected package chromium-browser-l10n.
Preparing to unpack .../chromium-browser-l10n_87.0.4280.66-0ubuntu0.18.04.1_all.deb .
Unpacking chromium-browser-l10n (87.0.4280.66-0ubuntu0.18.04.1) ...
Selecting previously unselected package chromium-chromedriver.
Preparing to unpack .../chromium-chromedriver_87.0.4280.66-0ubuntu0.18.04.1_amd64.deb
Unpacking chromium-chromedriver (87.0.4280.66-0ubuntu0.18.04.1) ...
Setting up chromium-codecs-ffmpeg-extra (87.0.4280.66-0ubuntu0.18.04.1) ...
Setting up chromium-browser (87.0.4280.66-0ubuntu0.18.04.1) ...
update-alternatives: using /usr/bin/chromium-browser to provide /usr/bin/x-www-browse
update-alternatives: using /usr/bin/chromium-browser to provide /usr/bin/gnome-www-br
Setting up chromium-chromedriver (87.0.4280.66-0ubuntu0.18.04.1) ...
Setting up chromium-browser-l10n (87.0.4280.66-0ubuntu0.18.04.1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
cp: '/usr/lib/chromium-browser/chromedriver' and '/usr/bin/chromedriver' are the same
```

```
import sys
sys.path.insert(0,'/usr/lib/chromium-browser/chromedriver')
from selenium import webdriver
chrome_options = webdriver.ChromeOptions()
chrome_options.add_argument('--headless')
chrome_options.add_argument('--no-sandbox')
chrome_options.add_argument('--disable-dev-shm-usage')
driver = webdriver.Chrome('chromedriver', options=chrome_options)


url = "https://inshorts.com/en/read/"
driver.get(url)


html = driver.page_source.encode('utf-8')


import time
from urllib.parse import quote_plus
from selenium.webdriver.common.action_chains import ActionChains


page_num = 0
```

```
ids = []
visited = []
while driver.find_elements_by_css_selector('.load-more'):
  driver.implicitly_wait(5)
  driver.find_element_by_css_selector('.load-more').click()
  page_num += 1
  print("getting page number "+str(page_num))
  soup = BeautifulSoup(driver.page_source, features="html.parser")
  headlines = []
  summaries = []
  news = []
  for i in soup.findAll("span", {"itemprop" : "mainEntityOfPage"}):
    if i['itemid'].split('-')[-1] not in ids:
      ids.append(i['itemid'].split('-')[-1])
      visited.append(0)
  for i in soup.findAll("span", {"itemprop" : "headline"}):
    headlines.append(i.text)
  for i in soup.findAll("div", {"itemprop" : "articleBody"}):
    summaries.append(i.text)
  if page_num == 1:
    break
print(ids[0])
print(headlines[0])
print(summaries[0])
```

```
getting page number 1
1610957002045
OPPO launches Reno5 Pro 5G smartphone & Enco X wireless earphones
OPPO has launched Reno5 Pro 5G smartphone and OPPO Enco X True Wireless earphones in a l
```