Web Scraping Project-1 (Beautiful-Soup):

In [1]:

```
!pip install bs4
!pip install requests
```

```
Collecting bs4
  Downloading bs4-0.0.1.tar.gz (1.1 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: beautifulsoup4 in c:\users\user\anaconda3\lib
\site-packages (from bs4) (4.9.3)
Requirement already satisfied: soupsieve>1.2 in c:\users\user\anaconda3\lib
\site-packages (from beautifulsoup4->bs4) (2.2.1)
Building wheels for collected packages: bs4
  Building wheel for bs4 (setup.py): started
  Building wheel for bs4 (setup.py): finished with status 'done'
  Created wheel for bs4: filename=bs4-0.0.1-py3-none-any.whl size=1273 sha25
6=575ac41de6ab61e7e2c930e9a3a07bbba55b8fe13170e9ecc181b63d20913d38
  Stored in directory: c:\users\user\appdata\local\pip\cache\wheels\75\78\21
\68b124549c9bdc94f822c02fb9aa3578a669843f9767776bca
Successfully built bs4
Installing collected packages: bs4
Successfully installed bs4-0.0.1
Requirement already satisfied: requests in c:\users\user\anaconda3\lib\site-
packages (2.25.1)
Requirement already satisfied: idna<3,>=2.5 in c:\users\user\anaconda3\lib\s
ite-packages (from requests) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\user\anaconda3
\lib\site-packages (from requests) (2020.12.5)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\user\anacon
da3\lib\site-packages (from requests) (1.26.4)
Requirement already satisfied: chardet<5,>=3.0.2 in c:\users\user\anaconda3
\lib\site-packages (from requests) (4.0.0)
```

Importing required libraries

In [1]:

```
import bs4
from bs4 import BeautifulSoup
import requests
```

1) Write a python program to display all the header tags from wikipedia.org.

In [32]:

```
from urllib.request import urlopen
```

In [33]:

```
html = urlopen('https://en.wikipedia.org/wiki/Main_Page')

soup = BeautifulSoup(html, "html.parser")

titles = soup.find_all(['h1', 'h2','h3','h4','h5','h6'])

print('List all the header tags of wikipedia:', *titles, sep='\n\n')
```

```
List all the header tags of wikipedia:

<h1 class="firstHeading mw-first-heading" id="firstHeading" style="display:
none"><span class="mw-page-title-main">Main Page</span></h1>

<h1><span class="mw-headline" id="Welcome_to_Wikipedia">Welcome to <a href
="/wiki/Wikipedia" title="Wikipedia">Wikipedia</a></span></h1>

<h2 class="mp-h2" id="mp-tfa-h2"><span id="From_today.27s_featured_article">
</span><span class="mw-headline" id="From_today's_featured_article">From tod
ay's featured article</span></h2>

<h2 class="mp-h2" id="mp-dyk-h2"><span class="mw-headline" id="Did_you_know
_...">Did you know ...</span></h2>

<h2 class="mp-h2" id="mp-itn-h2"><span class="mw-headline" id="In_the_news">
In the news</span></h2>

<h2 class="mp-h2" id="mp-otd-h2"><span class="mw-headline" id="On_this_day">
On this day</span></h2>

<h2 class="mp-h2" id="mp-tfp-h2"><span id="Today.27s_featured_picture"></spa
n><span class="mw-headline" id="Today's_featured_picture">Today's featured p
icture</span></h2>

<h2 class="mp-h2" id="mp-other"><span class="mw-headline" id="Other_areas_of
_Wikipedia">Other areas of Wikipedia</span></h2>

<h2 class="mp-h2" id="mp-sister"><span id="Wikipedia.27s_sister_projects"></
span><span class="mw-headline" id="Wikipedia's_sister_projects">Wikipedia's
sister projects</span></h2>

<h2 class="mp-h2" id="mp-lang"><span class="mw-headline" id="Wikipedia_langu
ages">Wikipedia languages</span></h2>

<h2>Navigation menu</h2>

<h3 class="vector-menu-heading" id="p-personal-label">
<span class="vector-menu-heading-label">Personal tools</span>
</h3>

<h3 class="vector-menu-heading" id="p-namespaces-label">
<span class="vector-menu-heading-label">Namespaces</span>
</h3>

<h3 class="vector-menu-heading" id="p-views-label">
<span class="vector-menu-heading-label">Views</span>
</h3>

<h3>
```

```
<label for="searchInput">Search</label>
</h3>

<h3 class="vector-menu-heading" id="p-navigation-label">
<span class="vector-menu-heading-label">Navigation</span>
</h3>

<h3 class="vector-menu-heading" id="p-interaction-label">
<span class="vector-menu-heading-label">Contribute</span>
</h3>

<h3 class="vector-menu-heading" id="p-tb-label">
<span class="vector-menu-heading-label">Tools</span>
</h3>

<h3 class="vector-menu-heading" id="p-coll-print_export-label">
<span class="vector-menu-heading-label">Print/export</span>
</h3>

<h3 class="vector-menu-heading" id="p-wikibase-otherprojects-label">
<span class="vector-menu-heading-label">In other projects</span>
</h3>

<h3 class="vector-menu-heading" id="p-lang-label">
<span class="vector-menu-heading-label">Languages</span>
</h3>
```

2) Write a python program to display IMDB's Top rated 100 movies' data (i.e. name, rating, year of release) and make data frame.

In [63]:

```
url="https://www.imdb.com/search/title/?count=100&groups=top_1000&sort=user_rating"
```

In [141]:

```
page1=requests.get(url)
page1
```

Out[141]:

```
<Response [200]>
```

In [65]:

```
soup=BeautifulSoup(page1.content,"html.parser")
soup
```

Out[65]:

```
<!DOCTYPE html>

<html xmlns:fb="http://www.facebook.com/2008/fbml" xmlns:og="http://ogp.m
e/ns#">
<head>
<meta charset="utf-8"/>
<script type="text/javascript">var IMDbTimer={starttime: new Date().getTim
e(),pt:'java'};</script>
<script>
    if (typeof uet == 'function') {
      uet("bb", "LoadTitle", {wb: 1});
    }
</script>
<script>(function(t){ (t.events = t.events || {})["csm_head_pre_title"] =
new Date().getTime(); })(IMDbTimer);</script>
<title>IMDb "Top 1000"
(Sorted by IMDb Rating Descending) - IMDb</title>
```

In [97]:

```
scraped_movies=soup.find_all('h3',class_='lister-item-header')
scraped_movies
```

Out[97]:

```
[<h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">1.</span>
 <a href="/title/tt0111161/">The Shawshank Redemption</a>
 <span class="lister-item-year text-muted unbold">(1994)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">2.</span>
 <a href="/title/tt0068646/">The Godfather</a>
 <span class="lister-item-year text-muted unbold">(1972)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">3.</span>
 <a href="/title/tt9263550/">Rocketry: The Nambi Effect</a>
 <span class="lister-item-year text-muted unbold">(2022)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">4.</span>
 <a href="/title/tt0468569/">The Dark Knight</a>
```

In [100]:

```
movies=[]

for movie in scraped_movies:
    movie=movie.a.text
    movies.append(movie)
```

In [101]:

```
movies
```

Out[101]:

```
['The Shawshank Redemption',
 'The Godfather',
 'Rocketry: The Nambi Effect',
 'The Dark Knight',
 'The Lord of the Rings: The Return of the King',
 "Schindler's List",
 'The Godfather Part II',
 '12 Angry Men',
 'Jai Bhim',
 'Pulp Fiction',
 'Inception',
 'The Lord of the Rings: The Two Towers',
 'Fight Club',
 'The Lord of the Rings: The Fellowship of the Ring',
 'Forrest Gump',
 'Il buono, il brutto, il cattivo',
 'Soorarai Pottru',
 'The Matrix',
```

In [108]:

```
rating_of_movies=soup.find_all('div',class_='inline-block ratings-imdb-rating')
rating_of_movies
```

Out[108]:

```
[<div class="inline-block ratings-imdb-rating" data-value="9.3" name="ir">
 <span class="global-sprite rating-star imdb-rating"></span>
 <strong>9.3</strong>
 </div>,
 <div class="inline-block ratings-imdb-rating" data-value="9.2" name="ir">
 <span class="global-sprite rating-star imdb-rating"></span>
 <strong>9.2</strong>
 </div>,
 <div class="inline-block ratings-imdb-rating" data-value="9" name="ir">
 <span class="global-sprite rating-star imdb-rating"></span>
 <strong>9.0</strong>
 </div>,
 <div class="inline-block ratings-imdb-rating" data-value="9" name="ir">
 <span class="global-sprite rating-star imdb-rating"></span>
 <strong>9.0</strong>
 </div>,
 <div class="inline-block ratings-imdb-rating" data-value="9" name="ir">
 <span class="global-sprite rating-star imdb-rating"></span>
```

In [113]:

```
rating=[]
for rt in rating_of_movies:
    rt=rt.strong.text
    rating.append(rt)
```

In [118]:

```
rating
```

Out[118]:

```
['9.3',
 '9.2',
 '9.0',
 '9.0',
 '9.0',
 '9.0',
 '9.0',
 '9.0',
 '8.9',
 '8.9',
 '8.8',
 '8.8',
 '8.8',
 '8.8',
 '8.8',
 '8.8',
 '8.7',
 '8.7',
```

In [122]:

```
year=soup.find_all('span',class_='lister-item-year text-muted unbold')
year
```

Out[122]:

```
[<span class="lister-item-year text-muted unbold">(1994)</span>,
 <span class="lister-item-year text-muted unbold">(1972)</span>,
 <span class="lister-item-year text-muted unbold">(2022)</span>,
 <span class="lister-item-year text-muted unbold">(2008)</span>,
 <span class="lister-item-year text-muted unbold">(2003)</span>,
 <span class="lister-item-year text-muted unbold">(1993)</span>,
 <span class="lister-item-year text-muted unbold">(1974)</span>,
 <span class="lister-item-year text-muted unbold">(1957)</span>,
 <span class="lister-item-year text-muted unbold">(2021)</span>,
 <span class="lister-item-year text-muted unbold">(1994)</span>,
 <span class="lister-item-year text-muted unbold">(2010)</span>,
 <span class="lister-item-year text-muted unbold">(2002)</span>,
 <span class="lister-item-year text-muted unbold">(1999)</span>,
 <span class="lister-item-year text-muted unbold">(2001)</span>,
 <span class="lister-item-year text-muted unbold">(1994)</span>,
 <span class="lister-item-year text-muted unbold">(1966)</span>,
 <span class="lister-item-year text-muted unbold">(2020)</span>,
 <span class="lister-item-year text-muted unbold">(1999)</span>,
```

In [123]:

```
year_of_release=[]
for yr in year:
    yr=yr.text
    year_of_release.append(yr)
```

In [124]:

```
year_of_release
```

Out[124]:

```
['(1994)',
 '(1972)',
 '(2022)',
 '(2008)',
 '(2003)',
 '(1993)',
 '(1974)',
 '(1957)',
 '(2021)',
 '(1994)',
 '(2010)',
 '(2002)',
 '(1999)',
 '(2001)',
 '(1994)',
 '(1966)',
 '(2020)',
 '(1999)',
```

In [2]:

```
import numpy as np
import pandas as pd
```

In [139]:

```
IMDB_top_rated_100_movies=pd.DataFrame({'Movie_Name':movies,'Ratings':rating,'Year':year_of
IMDB_top_rated_100_movies
```

Out[139]:

	Movie_Name	Ratings	Year
0	The Shawshank Redemption	9.3	(1994)
1	The Godfather	9.2	(1972)
2	Rocketry: The Nambi Effect	9.0	(2022)
3	The Dark Knight	9.0	(2008)
4	The Lord of the Rings: The Return of the King	9.0	(2003)
...	...	...	...
95	The Great Dictator	8.4	(1940)
96	Chhichhore	8.3	(2019)
97	Ratsasan	8.3	(2018)
98	Ayla: The Daughter of War	8.3	(2017)
99	Vikram Vedha	8.3	(2017)

100 rows × 3 columns

3) Write a python program to display IMDB's Top rated 100 Indian movies' data (i.e. name, rating, year of release) and make data frame.

In [140]:

```
url="https://www.imdb.com/list/ls009997493/?sort=user_rating,desc&st_dt=&mode=detail&page=1
```

In [142]:

```
page2=requests.get(url)
page2
```

Out[142]:

```
<Response [200]>
```

In [151]:

```
soup2=BeautifulSoup(page2.content,"html.parser")
soup2
```

Out[151]:

```
<!DOCTYPE html>

<html xmlns:fb="http://www.facebook.com/2008/fbml" xmlns:og="http://ogp.m
e/ns#">
<head>
<meta charset="utf-8"/>
<script type="text/javascript">var IMDbTimer={starttime: new Date().getTim
e(),pt:'java'};</script>
<script>
    if (typeof uet == 'function') {
      uet("bb", "LoadTitle", {wb: 1});
    }
</script>
<script>(function(t){ (t.events = t.events || {})["csm_head_pre_title"] =
new Date().getTime(); })(IMDbTimer);</script>
<title>IMDB Top 100 Hindi Movies - IMDb</title>
<script>(function(t){ (t.events = t.events || {})["csm_head_post_title"] =
```

In [152]:

```
indian_movies=soup2.find_all('h3',class_='lister-item-header')
indian_movies
```

Out[152]:

```
[<h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">1.</span>
 <a href="/title/tt0079221/">Golmaal</a>
 <span class="lister-item-year text-muted unbold">(1979)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">2.</span>
 <a href="/title/tt1187043/">3 Idiots</a>
 <span class="lister-item-year text-muted unbold">(2009)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">3.</span>
 <a href="/title/tt0400234/">Black Friday</a>
 <span class="lister-item-year text-muted unbold">(2004)</span>
 </h3>,
 <h3 class="lister-item-header">
 <span class="lister-item-index unbold text-primary">4.</span>
 <a href="/title/tt0085913/">Masoom</a>
```

In [153]:

```
movies=[]

for i in indian_movies:
    i=i.a.text
    movies.append(i)
```

In [154]:

```
movies
```

Out[154]:

```
['Golmaal',
 '3 Idiots',
 'Black Friday',
 'Masoom',
 'Guide',
 'Taare Zameen Par',
 'Satya',
 'Pyaasa',
 'Khosla Ka Ghosla!',
 'Angoor',
 'Jaane Bhi Do Yaaro',
 'Chhoti Si Baat',
 'Chupke Chupke',
 'Gangs of Wasseypur',
 'Swades: We, the People',
 'Zindagi Na Milegi Dobara',
 'Paan Singh Tomar',
 'Bhaag Milkha Bhaag',
```

In [159]:

```
ratings=soup2.find_all('div',class_='ipl-rating-star small')
ratings
```

Out[159]:

```
[<div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox="0
0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.63 2
9.24l5.46 4.73L5.82 21z"></path>
 <path d="M0 0h24v24H0z" fill="none"></path>
 </svg>
 </span>
 <span class="ipl-rating-star__rating">8.5</span>
 </div>,
 <div class="ipl-rating-star small">
 <span class="ipl-rating-star__star">
 <svg class="ipl-icon ipl-star-icon" fill="#000000" height="24" viewbox="0
0 24 24" width="24" xmlns="http://www.w3.org/2000/svg">
 <path d="M0 0h24v24H0z" fill="none"></path>
 <path d="M12 17.27L18.18 21l-1.64-7.03L22 9.24l-7.19-.61L12 2 9.19 8.63 2
```

In [167]:

```
rating=[]
for r in ratings:
    r=r.text.replace('\n',"")
    rating.append(r)
```

In [168]:

```
rating
```

Out[168]:

```
['8.5',
 '8.4',
 '8.4',
 '8.4',
 '8.4',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.3',
 '8.2',
 '8.2',
 '8.2',
 '8.2',
 '8.2',
```

In [169]:

```
year=soup2.find_all('span',class_='lister-item-year text-muted unbold')
year
```

Out[169]:

```
[<span class="lister-item-year text-muted unbold">(1979)</span>,
 <span class="lister-item-year text-muted unbold">(2009)</span>,
 <span class="lister-item-year text-muted unbold">(2004)</span>,
 <span class="lister-item-year text-muted unbold">(1983)</span>,
 <span class="lister-item-year text-muted unbold">(1965)</span>,
 <span class="lister-item-year text-muted unbold">(2007)</span>,
 <span class="lister-item-year text-muted unbold">(1998)</span>,
 <span class="lister-item-year text-muted unbold">(1957)</span>,
 <span class="lister-item-year text-muted unbold">(2006)</span>,
 <span class="lister-item-year text-muted unbold">(1982)</span>,
 <span class="lister-item-year text-muted unbold">(1983)</span>,
 <span class="lister-item-year text-muted unbold">(1976)</span>,
 <span class="lister-item-year text-muted unbold">(1975)</span>,
 <span class="lister-item-year text-muted unbold">(2012)</span>,
 <span class="lister-item-year text-muted unbold">(2004)</span>,
 <span class="lister-item-year text-muted unbold">(2011)</span>,
 <span class="lister-item-year text-muted unbold">(2012)</span>,
 <span class="lister-item-year text-muted unbold">(2013)</span>,
```

In [171]:

```
year_of_release=[]
for yr1 in year:
    yr1=yr1.text
    year_of_release.append(yr1)
```

In [172]:

```
year_of_release
```

Out[172]:

```
['(1979)',
 '(2009)',
 '(2004)',
 '(1983)',
 '(1965)',
 '(2007)',
 '(1998)',
 '(1957)',
 '(2006)',
 '(1982)',
 '(1983)',
 '(1976)',
 '(1975)',
 '(2012)',
 '(2004)',
 '(2011)',
 '(2012)',
 '(2013)',
```

In [173]:

```
IMDB_top_rated_100_indianmovies=pd.DataFrame({'Movie_Name':movies,'Ratings':rating,'Year':y
IMDB_top_rated_100_indianmovies
```

Out[173]:

	Movie_Name	Ratings	Year
0	Golmaal	8.5	(1979)
1	3 Idiots	8.4	(2009)
2	Black Friday	8.4	(2004)
3	Masoom	8.4	(1983)
4	Guide	8.4	(1965)
...	...	...	...
95	Jaane Tu... Ya Jaane Na	7.4	(2008)
96	Rangeela	7.4	(1995)
97	Socha Na Tha	7.4	(2005)
98	Qayamat Se Qayamat Tak	7.4	(1988)
99	Gunda	7.3	(1998)

100 rows × 3 columns

4) Write s python program to display list of respected former presidents of India(i.e. Name , Term of office) from https://presidentofindia.nic.in/former-presidents.htm (https://presidentofindia.nic.in/former-presidents.htm)

In [306]:

```
url="https://presidentofindia.nic.in/former-presidents.htm"
```

In [307]:

```
page3=requests.get(url)
page3
```

Out[307]:

```
<Response [200]>
```

In [308]:

```
soup3=BeautifulSoup(page3.content,"html.parser")
soup3
```

Out[308]:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://ww
w.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html lang="en" xml:lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1"><title>
        Former Presidents - The President of India
</title><meta content="text/html; charset=utf-8" http-equiv="Content-Typ
e"/><meta content="no-cache" http-equiv="pragma"/>
<!--<meta http-equiv="Content-Style-Type" content="text/css" /><meta http-
equiv="Content-Script-Type" content="type" />-->
<meta content="telephone=no" name="format-detection"/><meta content="IE=Em
ulateIE10" http-equiv="X-UA-Compatible"/>
<!-- Start Favicon -->
<link href="favicon.ico" rel="shortcut icon" type="image/x-icon"/><link hr
ef="js/panorama_viewer.css" rel="stylesheet" type="text/css"/>
<!-- Start Viewport -->
<!--<meta name="viewport" content="width=device-width, initial-scale=1, ma
```

In [309]:

```
names=soup3.find_all('div',class_='presidentListing')
names
```

Out[309]:

```
[<div class="presidentListing">
 <h3>Shri Ram Nath Kovind (birth - 1945)</h3>
 <p><span class="terms">Term of Office:</span> 25 July, 2017 to 25 July, 2
022 </p>
 <p><a href="https://ramnathkovind.nic.in" target="_blank">https://ramnath
kovind.nic.in</a></p>
 </div>,
 <div class="presidentListing">
 <h3>Shri Pranab Mukherjee (1935-2020)</h3>
 <p><span class="terms">Term of Office:</span> 25 July, 2012 to 25 July, 2
017 </p>
 <p><a href="http://pranabmukherjee.nic.in" target="_blank">http://pranabm
ukherjee.nic.in</a></p>
 </div>,
 <div class="presidentListing">
 <h3>Smt Pratibha Devisingh Patil (birth - 1934)</h3>
 <p><span class="terms">Term of Office:</span> 25 July, 2007 to 25 July, 2
012 </p>
```

In [310]:

```
name=[]
for i in names:
    i=i.h3.text
    name.append(i)
```

In [311]:

```
name
```

Out[311]:

```
['Shri Ram Nath Kovind (birth - 1945)',
 'Shri Pranab Mukherjee (1935-2020)',
 'Smt Pratibha Devisingh Patil (birth - 1934)',
 'DR. A.P.J. Abdul Kalam (1931-2015)',
 'Shri K. R. Narayanan (1920 - 2005)',
 'Dr Shankar Dayal Sharma (1918-1999)',
 'Shri R Venkataraman (1910-2009)',
 'Giani Zail Singh (1916-1994)',
 'Shri Neelam Sanjiva Reddy (1913-1996)',
 'Dr. Fakhruddin Ali Ahmed (1905-1977)',
 'Shri Varahagiri Venkata Giri (1894-1980)',
 'Dr. Zakir Husain (1897-1969)',
 'Dr. Sarvepalli Radhakrishnan (1888-1975)',
 'Dr. Rajendra Prasad (1884-1963) ']
```

In [312]:

```
time_period=soup3.find_all('p',)
print(time_period)

term_office=[]
for tm in time_period:
    tm=tm.text
    term_office.append(tm)
```

```
[<p><span class="terms">Term of Office:</span> 25 July, 2017 to 25 July, 202
2 </p>, <p><a href="https://ramnathkovind.nic.in" target="_blank">https://ra
mnathkovind.nic.in</a></p>, <p><span class="terms">Term of Office:</span> 25
July, 2012 to 25 July, 2017 </p>, <p><a href="http://pranabmukherjee.nic.in"
target="_blank">http://pranabmukherjee.nic.in</a></p>, <p><span class="term
s">Term of Office:</span> 25 July, 2007 to 25 July, 2012 </p>, <p><a href="h
ttp://pratibhapatil.nic.in" target="_blank">http://pratibhapatil.nic.in</a>
</p>, <p><span class="terms">Term of Office:</span> 25 July, 2002 to 25 Jul
y, 2007 </p>, <p><a href="http://abdulkalam.nic.in" target="_blank">http://a
bdulkalam.nic.in</a></p>, <p><span class="terms">Term of Office:</span> 25 J
uly, 1997 to 25 July, 2002 </p>, <p><span class="terms">Term of Office:</spa
n> 25 July, 1992 to 25 July, 1997 </p>, <p><span class="terms">Term of Offic
e:</span> 25 July, 1987 to 25 July, 1992 </p>, <p><span class="terms">Term o
f Office:</span> 25 July, 1982 to 25 July, 1987 </p>, <p><span class="term
s">Term of Office:</span> 25 July, 1977 to 25 July, 1982 </p>, <p><span clas
s="terms">Term of Office:</span> 24 August, 1974 to 11 February, 1977</p>, <
p><span class="terms">Term of Office:</span> 3 May, 1969 to 20 July, 1969 an
d 24 August, 1969 to 24 August, 1974</p>, <p><span class="terms">Term of Off
ice:</span> 13 May, 1967 to 3 May, 1969</p>, <p><span class="terms">Term of
Office:</span> 13 May, 1962 to 13 May, 1967</p>, <p><span class="terms">Term
of Office:</span> 26 January, 1950 to 13 May, 1962</p>, <p class="copyrigh
t">Copyright © 2022 The Rashtrapati Bhavan.</p>, <p class="lastUpdated"><spa
n id="ContentPlaceHolder1_UserPageLastUpdateDate1_lblPageLastUpdateDate">Pag
e last updated on: <strong>22-July-2022 15:58 PM</strong></span>
</p>, <p class="govNic"><a href="http://www.nic.in/" target="_blank"><img al
t="National Informatics Centre - www.nic.in - External website that opens in
a new window" height="28" src="Images/nic-logo.gif" title="National Informat
ics Centre - www.nic.in - External website that opens in a new window" width
="28"/></a>
 Website hosted by National Informatics Centre. Website Content provided by
the President's Secretariat.</p>]
```

In [313]:

```
sz=len(term_office)
for i in range(0,sz):
    if( "Term of Office" in term_office[i]):
        term_of_office=term_office[i].replace("Term of Office: ","")
        print(term_of_office)
```

```
25 July, 2017 to 25 July, 2022
25 July, 2012 to 25 July, 2017
25 July, 2007 to 25 July, 2012
25 July, 2002 to 25 July, 2007
25 July, 1997 to 25 July, 2002
25 July, 1992 to 25 July, 1997
25 July, 1987 to 25 July, 1992
25 July, 1982 to 25 July, 1987
25 July, 1977 to 25 July, 1982
24 August, 1974 to 11 February, 1977
3 May, 1969 to 20 July, 1969 and 24 August, 1969 to 24 August, 1974
13 May, 1967 to 3 May, 1969
13 May, 1962 to 13 May, 1967
26 January, 1950 to 13 May, 1962
```

5) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape: a) Top 10 ODI teams in men's cricket along with the records for matches, points and rating. b) Top 10 ODI Batsmen along with the records of their team and rating. c) Top 10 ODI bowlers along with the records of their team and rating.

In [3]:

```
page4=requests.get("https://www.icc-cricket.com/rankings/mens/team-rankings/odi")
page4
```

Out[3]:

```
<Response [200]>
```

In [4]:

```
soup4=BeautifulSoup(page4.content,"html.parser")
soup4
```

Out[4]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Men's ODI Team Rankings | ICC" name="twitter:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council ranking for One Day
International (ODI) cricket teams. Discover latest ICC rankings table, pre
dict upcoming matches, see points and ratings for all teams." name="descri
ption"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council ranking for One Day
International (ODI) cricket teams. Discover latest ICC rankings table, pre
dict upcoming matches, see points and ratings for all teams." name="twitte
r:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
```

In [5]:

```
men_team=soup4.find_all('span',class_='u-hide-phablet')
men_team
```

Out[5]:

```
[<span class="u-hide-phablet">England</span>,
 <span class="u-hide-phablet">New Zealand</span>,
 <span class="u-hide-phablet">India</span>,
 <span class="u-hide-phablet">Pakistan</span>,
 <span class="u-hide-phablet">Australia</span>,
 <span class="u-hide-phablet">South Africa</span>,
 <span class="u-hide-phablet">Bangladesh</span>,
 <span class="u-hide-phablet">Sri Lanka</span>,
 <span class="u-hide-phablet">West Indies</span>,
 <span class="u-hide-phablet">Afghanistan</span>,
 <span class="u-hide-phablet">Ireland</span>,
 <span class="u-hide-phablet">Scotland</span>,
 <span class="u-hide-phablet">Zimbabwe</span>,
 <span class="u-hide-phablet">Namibia</span>,
 <span class="u-hide-phablet">Netherlands</span>,
 <span class="u-hide-phablet">UAE</span>,
 <span class="u-hide-phablet">Oman</span>,
 <span class="u-hide-phablet">United States</span>,
 <span class="u-hide-phablet">Nepal</span>,
 <span class="u-hide-phablet">Papua New Guinea</span>]
```

In [8]:

```
men=[]
for i in men_team:
    i=i.text
    men.append(i)
```

In [70]:

```
men.append('Papua New Guinea')
```

In [71]:

```
men
```

Out[71]:

```
['England',
 'New Zealand',
 'India',
 'Pakistan',
 'Australia',
 'South Africa',
 'Bangladesh',
 'Sri Lanka',
 'West Indies',
 'Afghanistan',
 'Ireland',
 'Scotland',
 'Zimbabwe',
 'Namibia',
 'Netherlands',
 'UAE',
 'Oman',
 'United States',
 'Nepal',
 'Papua New Guinea']
```

In [24]:

```
match1=soup4.find('td',class_='rankings-block__banner--matches')
match1.text
```

Out[24]:

```
'27'
```

In [85]:

```
match_f=list(match1)
match_f
```

Out[85]:

```
['27']
```

In [37]:

```
point1=soup4.find('td',class_='rankings-block__banner--points')
point1.text
```

Out[37]:

```
'3,226'
```

In [86]:

```
point_f=list(point1)
point_f
```

Out[86]:

```
['3,226']
```

In [25]:

```
match2=soup4.find_all('td',class_='table-body__cell u-center-text')
match2
```

Out[25]:

```
[<td class="table-body__cell u-center-text">22</td>,
 <td class="table-body__cell u-center-text">2,508</td>,
 <td class="table-body__cell u-center-text">31</td>,
 <td class="table-body__cell u-center-text">3,447</td>,
 <td class="table-body__cell u-center-text">22</td>,
 <td class="table-body__cell u-center-text">2,354</td>,
 <td class="table-body__cell u-center-text">29</td>,
 <td class="table-body__cell u-center-text">3,071</td>,
 <td class="table-body__cell u-center-text">21</td>,
 <td class="table-body__cell u-center-text">2,111</td>,
 <td class="table-body__cell u-center-text">30</td>,
 <td class="table-body__cell u-center-text">2,753</td>,
 <td class="table-body__cell u-center-text">29</td>,
 <td class="table-body__cell u-center-text">2,658</td>,
 <td class="table-body__cell u-center-text">41</td>,
 <td class="table-body__cell u-center-text">2,902</td>,
 <td class="table-body__cell u-center-text">18</td>,
 <td class="table-body__cell u-center-text">1,238</td>,
```

In [27]:

```
match=[]
for m in match2:
    m=m.text
    match.append(m)
```

In [28]:

```
match
```

Out[28]:

```
['22',
 '2,508',
 '31',
 '3,447',
 '22',
 '2,354',
 '29',
 '3,071',
 '21',
 '2,111',
 '30',
 '2,753',
 '29',
 '2,658',
 '41',
 '2,902',
 '18',
 '1.238'.
```

In [69]:

```
a=str(match)
points=[match[i] for i in range(len(match)) if i%2!=0]
points
```

Out[69]:

```
['2,508',
 '3,447',
 '2,354',
 '3,071',
 '2,111',
 '2,753',
 '2,658',
 '2,902',
 '1,238',
 '1,214',
 '1,254',
 '1,098',
 '642',
 '673',
 '697',
 '919',
 '641',
 '331',
 '209']
```

In [102]:

```
points.insert(0,point1.text)
```

In [103]:

```
points
```

Out[103]:

```
['3,226',
 '2,508',
 '3,447',
 '2,354',
 '3,071',
 '2,111',
 '2,753',
 '2,658',
 '2,902',
 '1,238',
 '1,214',
 '1,254',
 '1,098',
 '642',
 '673',
 '697',
 '919',
 '641',
 '331',
 '209']
```

In [106]:

```
match_c=[match[i] for i in range(len(match)) if i%2==0]
match_c
```

Out[106]:

```
['22',
 '31',
 '22',
 '29',
 '21',
 '30',
 '29',
 '41',
 '18',
 '23',
 '27',
 '26',
 '19',
 '21',
 '22',
 '30',
 '27',
 '22',
 '26']
```

In [107]:

```
match_c.insert(0,match1.text)
```

In [108]:

```
match_c
```

Out[108]:

```
['27',
 '22',
 '31',
 '22',
 '29',
 '21',
 '30',
 '29',
 '41',
 '18',
 '23',
 '27',
 '26',
 '19',
 '21',
 '22',
 '30',
 '27',
 '22',
 '26']
```

In [51]:

```
rating1=soup4.find('td',class_='rankings-block__banner--rating u-text-right')
rating1.text.strip()
```

Out[51]:

```
'119'
```

In [52]:

```
rating=soup4.find_all('td',class_='table-body__cell u-text-right rating')
rating
```

Out[52]:

```
[<td class="table-body__cell u-text-right rating">114</td>,
 <td class="table-body__cell u-text-right rating">111</td>,
 <td class="table-body__cell u-text-right rating">107</td>,
 <td class="table-body__cell u-text-right rating">106</td>,
 <td class="table-body__cell u-text-right rating">101</td>,
 <td class="table-body__cell u-text-right rating">92</td>,
 <td class="table-body__cell u-text-right rating">92</td>,
 <td class="table-body__cell u-text-right rating">71</td>,
 <td class="table-body__cell u-text-right rating">69</td>,
 <td class="table-body__cell u-text-right rating">53</td>,
 <td class="table-body__cell u-text-right rating">46</td>,
 <td class="table-body__cell u-text-right rating">42</td>,
 <td class="table-body__cell u-text-right rating">34</td>,
 <td class="table-body__cell u-text-right rating">32</td>,
 <td class="table-body__cell u-text-right rating">32</td>,
 <td class="table-body__cell u-text-right rating">31</td>,
 <td class="table-body__cell u-text-right rating">24</td>,
 <td class="table-body__cell u-text-right rating">15</td>,
 <td class="table-body__cell u-text-right rating">8</td>]
```

In [77]:

```
ratings=[]
for r in rating:
    r=r.text
    ratings.append(r)
```

In [78]:

```
ratings
```

Out[78]:

```
['114',
 '111',
 '107',
 '106',
 '101',
 '92',
 '92',
 '71',
 '69',
 '53',
 '46',
 '42',
 '34',
 '32',
 '32',
 '31',
 '24',
 '15',
 '8']
```

In [109]:

```
ratings.insert(0,rating1.text.strip())
```

In [110]:

```
ratings
```

Out[110]:

```
['119',
 '114',
 '111',
 '107',
 '106',
 '101',
 '92',
 '92',
 '71',
 '69',
 '53',
 '46',
 '42',
 '34',
 '32',
 '32',
 '31',
 '24',
 '15',
 '8']
```

In [112]:

```
len(points),len(men),len(match_c),len(ratings)
```

Out[112]:

```
(20, 20, 20, 20)
```

In [114]:

```
#(a) Top 10 ODI teams in men's cricket along with the records for matches, points and ratin

Top_10_ODI_MenTeam=pd.DataFrame({'team':men,'matches':match_c,'points':points,'rating':rati
Top_10_ODI_MenTeam.head(10)
```

Out[114]:

	team	matches	points	rating
0	England	27	3,226	119
1	New Zealand	22	2,508	114
2	India	31	3,447	111
3	Pakistan	22	2,354	107
4	Australia	29	3,071	106
5	South Africa	21	2,111	101
6	Bangladesh	30	2,753	92
7	Sri Lanka	29	2,658	92
8	West Indies	41	2,902	71
9	Afghanistan	18	1,238	69

In [128]:

```
page5=requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi/batting')
page5
```

Out[128]:

```
<Response [200]>
```

In [129]:

```
soup5=BeautifulSoup(page5.content,"html.parser")
soup5
```

Out[129]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Men's ODI Batting | Player Rankings | ICC" name="twitte
r:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Men's ODI Batting | Player Rankings | ICC" property="o
```

In [131]:

```
player=soup5.find_all('td',class_='table-body__cell rankings-table__name name')
player
```

Out[131]:

```
[<td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/1277">Rassie van der Dussen</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/834">Quinton de Kock</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/1568">Imam-ul-Haq</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/164">Virat Kohli</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/107">Rohit Sharma</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/506">Jonny Bairstow</a>
 </td>.
```

In [134]:

```
name=[]
for n in player:
    n=n.text.strip()
    name.append(n)
```

In [135]:

```
name
```

Out[135]:

```
['Rassie van der Dussen',
 'Quinton de Kock',
 'Imam-ul-Haq',
 'Virat Kohli',
 'Rohit Sharma',
 'Jonny Bairstow',
 'David Warner',
 'Ross Taylor',
 'Steve Smith',
 'Shikhar Dhawan',
 'Joe Root',
 'Fakhar Zaman',
 'Kane Williamson',
 'Shai Hope',
 'Tamim Iqbal',
 'Jason Roy',
 'Paul Stirling',
 'Mushfiqur Rahim',
```

In [136]:

```
name.insert(0,'Babar Azam')
```

In [137]:

```
name
```

Out[137]:

```
['Babar Azam',
 'Rassie van der Dussen',
 'Quinton de Kock',
 'Imam-ul-Haq',
 'Virat Kohli',
 'Rohit Sharma',
 'Jonny Bairstow',
 'David Warner',
 'Ross Taylor',
 'Steve Smith',
 'Shikhar Dhawan',
 'Joe Root',
 'Fakhar Zaman',
 'Kane Williamson',
 'Shai Hope',
 'Tamim Iqbal',
 'Jason Roy',
 'Paul Stirling',
```

In [138]:

```
Team=soup5.find_all('span',class_='table-body__logo-text')
Team
```

Out[138]:

```
[<span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">PAK</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">PAK</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">IRE</span>,
 <span class="table-body__logo-text">BAN</span>,
```

In [139]:

```
team=[]
for t in Team:
    t=t.text
    team.append(t)
```

In [140]:

```
team
```

Out[140]:

```
['SA',
 'SA',
 'PAK',
 'IND',
 'IND',
 'ENG',
 'AUS',
 'NZ',
 'AUS',
 'IND',
 'ENG',
 'PAK',
 'NZ',
 'WI',
 'BAN',
 'ENG',
 'IRE',
 'BAN'.
```

In [141]:

```
team.insert(0,'PAK')
```

In [142]:

```
team
```

Out[142]:

```
['PAK',
 'SA',
 'SA',
 'PAK',
 'IND',
 'IND',
 'ENG',
 'AUS',
 'NZ',
 'AUS',
 'IND',
 'ENG',
 'PAK',
 'NZ',
 'WI',
 'BAN',
 'ENG',
 'IRE'.
```

In [143]:

```
Rating=soup5.find_all('td',class_='table-body__cell rating')
Rating
```

Out[143]:

```
[<td class="table-body__cell rating">789</td>,
 <td class="table-body__cell rating">784</td>,
 <td class="table-body__cell rating">779</td>,
 <td class="table-body__cell rating">744</td>,
 <td class="table-body__cell rating">740</td>,
 <td class="table-body__cell rating">732</td>,
 <td class="table-body__cell rating">725</td>,
 <td class="table-body__cell rating">701</td>,
 <td class="table-body__cell rating">697</td>,
 <td class="table-body__cell rating">696</td>,
 <td class="table-body__cell rating">691</td>,
 <td class="table-body__cell rating">690</td>,
 <td class="table-body__cell rating">685</td>,
 <td class="table-body__cell rating">679</td>,
 <td class="table-body__cell rating">675</td>,
 <td class="table-body__cell rating">672</td>,
 <td class="table-body__cell rating">664</td>,
 <td class="table-body__cell rating">657</td>,
```

In [148]:

```
rating=[]
for rt in Rating:
    rt=rt.text
    rating.append(rt)
```

In [149]:

```
rating
```

Out[149]:

```
['789',
 '784',
 '779',
 '744',
 '740',
 '732',
 '725',
 '701',
 '697',
 '696',
 '691',
 '690',
 '685',
 '679',
 '675',
 '672',
 '664',
 '657',
```

In [150]:

```
rating.insert(0,'890')
rating
```

Out[150]:

```
['890',
 '789',
 '784',
 '779',
 '744',
 '740',
 '732',
 '725',
 '701',
 '697',
 '696',
 '691',
 '690',
 '685',
 '679',
 '675',
 '672',
 '664',
```

In [152]:

```
#(b) Top 10 ODI Batsmen along with the records of their team and rating.

Top_10_ODI_Batsman=pd.DataFrame({'Name':name,'Team':team,'Rating':rating})
Top_10_ODI_Batsman.head(10)
```

Out[152]:

	Name	Team	Rating
0	Babar Azam	PAK	890
1	Rassie van der Dussen	SA	789
2	Quinton de Kock	SA	784
3	Imam-ul-Haq	PAK	779
4	Virat Kohli	IND	744
5	Rohit Sharma	IND	740
6	Jonny Bairstow	ENG	732
7	David Warner	AUS	725
8	Ross Taylor	NZ	701
9	Steve Smith	AUS	697

In [153]:

```
page6=requests.get('https://www.icc-cricket.com/rankings/mens/player-rankings/odi/bowling')
page6
```

Out[153]:

```
<Response [200]>
```

In [155]:

```
soup6=BeautifulSoup(page6.content,"html.parser")
soup6
```

Out[155]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Men's ODI Bowling | Player Rankings | ICC" name="twitte
r:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Men's ODI Bowling | Player Rankings | ICC" property="o
```

In [156]:

```
player=soup6.find_all('td',class_='table-body__cell rankings-table__name name')
player
```

Out[156]:

```
[<td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/857">Josh Hazlewood</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/4572">Mujeeb Ur Rahman</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/1124">Jasprit Bumrah</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/4530">Shaheen Afridi</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/618">Mohammad Nabi</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/mens/player-rankings/1597">Mehedi Hasan</a>
 </td>.
```

In [159]:

```
name=[]
for n in player:
    n=n.text.strip()
    name.append(n)
```

In [161]:

```
name.insert(0,'Trent Boult')
name
```

Out[161]:

```
['Trent Boult',
 'Josh Hazlewood',
 'Mujeeb Ur Rahman',
 'Jasprit Bumrah',
 'Shaheen Afridi',
 'Mohammad Nabi',
 'Mehedi Hasan',
 'Matt Henry',
 'Mitchell Starc',
 'Rashid Khan',
 'Mustafizur Rahman',
 'Chris Woakes',
 'Andy McBrine',
 'Kagiso Rabada',
 'Adam Zampa',
 'Shakib Al Hasan',
 'Alzarri Joseph',
 'Yuzvendra Chahal',
```

In [162]:

```
team=soup6.find_all('span',class_='table-body__logo-text')
team
```

Out[162]:

```
[<span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">AFG</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">PAK</span>,
 <span class="table-body__logo-text">AFG</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">AFG</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">IRE</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">AUS</span>,
```

In [163]:

```
team_name=[]
for t in team:
    t=t.text
    team_name.append(t)
```

In [165]:

```
team_name.insert(0,'NZ')
team_name
```

Out[165]:

```
['NZ',
 'AUS',
 'AFG',
 'IND',
 'PAK',
 'AFG',
 'BAN',
 'NZ',
 'AUS',
 'AFG',
 'BAN',
 'ENG',
 'IRE',
 'SA',
 'AUS',
 'BAN',
 'WI',
 'IND',
```

In [166]:

```
Rating=soup6.find_all('td',class_='table-body__cell rating')
Rating
```

Out[166]:

```
[<td class="table-body__cell rating">718</td>,
 <td class="table-body__cell rating">676</td>,
 <td class="table-body__cell rating">662</td>,
 <td class="table-body__cell rating">661</td>,
 <td class="table-body__cell rating">657</td>,
 <td class="table-body__cell rating">655</td>,
 <td class="table-body__cell rating">654</td>,
 <td class="table-body__cell rating">653</td>,
 <td class="table-body__cell rating">651</td>,
 <td class="table-body__cell rating">640</td>,
 <td class="table-body__cell rating">640</td>,
 <td class="table-body__cell rating">630</td>,
 <td class="table-body__cell rating">625</td>,
 <td class="table-body__cell rating">622</td>,
 <td class="table-body__cell rating">619</td>,
 <td class="table-body__cell rating">609</td>,
 <td class="table-body__cell rating">604</td>,
 <td class="table-body__cell rating">600</td>,
```

In [167]:

```
rating=[]
for ra in Rating:
    ra=ra.text
    rating.append(ra)
```

In [169]:

```
rating.insert(0,'775')
rating
```

Out[169]:

```
['775',
 '718',
 '676',
 '662',
 '661',
 '657',
 '655',
 '654',
 '653',
 '651',
 '640',
 '640',
 '630',
 '625',
 '622',
 '619',
 '609',
 '604'.
```

In [171]:

```
#(c) Top 10 ODI bowlers along with the records of their team and rating.

Top_10_ODI_Bowlers=pd.DataFrame({'Name':name,'Team':team_name,'Rating':rating})
Top_10_ODI_Bowlers.head(10)
```

Out[171]:

	Name	Team	Rating
0	Trent Boult	NZ	775
1	Josh Hazlewood	AUS	718
2	Mujeeb Ur Rahman	AFG	676
3	Jasprit Bumrah	IND	662
4	Shaheen Afridi	PAK	661
5	Mohammad Nabi	AFG	657
6	Mehedi Hasan	BAN	655
7	Matt Henry	NZ	654
8	Mitchell Starc	AUS	653
9	Rashid Khan	AFG	651

6) Write a python program to scrape cricket rankings from icc-cricket.com. You have to scrape: a) Top 10 ODI teams in women's cricket along with the records for matches, points and rating. b) Top 10 women's ODI Batting players along with the records of their team and rating. c) Top 10 women's ODI all-rounder along with the records of their team and rating.

In [173]:

```
page7=requests.get("https://www.icc-cricket.com/rankings/womens/team-rankings/odi")
page7
```

Out[173]:

```
<Response [200]>
```

In [174]:

```
soup7=BeautifulSoup(page7.content,"html.parser")
soup7
```

Out[174]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Women's ODI Team Rankings | ICC" name="twitter:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official International Cricket Council rankings for test ma
tch cricket teams. Discover latest ICC rankings table, predict upcoming ma
tches, see points and ratings for all teams." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official International Cricket Council rankings for test ma
tch cricket teams. Discover latest ICC rankings table, predict upcoming ma
tches, see points and ratings for all teams." name="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Team Rankings | ICC" property="og:title"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
```

In [175]:

```
women_team=soup7.find_all('span',class_='u-hide-phablet')
women_team
```

Out[175]:

```
[<span class="u-hide-phablet">Australia</span>,
 <span class="u-hide-phablet">England</span>,
 <span class="u-hide-phablet">South Africa</span>,
 <span class="u-hide-phablet">India</span>,
 <span class="u-hide-phablet">New Zealand</span>,
 <span class="u-hide-phablet">West Indies</span>,
 <span class="u-hide-phablet">Bangladesh</span>,
 <span class="u-hide-phablet">Pakistan</span>,
 <span class="u-hide-phablet">Ireland</span>,
 <span class="u-hide-phablet">Sri Lanka</span>,
 <span class="u-hide-phablet">Zimbabwe</span>]
```

In [177]:

```
women=[]
for i in women_team:
    i=i.text
    women.append(i)
```

In [178]:

```
women
```

Out[178]:

```
['Australia',
 'England',
 'South Africa',
 'India',
 'New Zealand',
 'West Indies',
 'Bangladesh',
 'Pakistan',
 'Ireland',
 'Sri Lanka',
 'Zimbabwe']
```

In [180]:

```
Match=soup7.find_all('td',class_='table-body__cell u-center-text')
Match
```

Out[180]:

```
[<td class="table-body__cell u-center-text">34</td>,
 <td class="table-body__cell u-center-text">4,097</td>,
 <td class="table-body__cell u-center-text">35</td>,
 <td class="table-body__cell u-center-text">4,157</td>,
 <td class="table-body__cell u-center-text">33</td>,
 <td class="table-body__cell u-center-text">3,392</td>,
 <td class="table-body__cell u-center-text">32</td>,
 <td class="table-body__cell u-center-text">3,161</td>,
 <td class="table-body__cell u-center-text">31</td>,
 <td class="table-body__cell u-center-text">2,815</td>,
 <td class="table-body__cell u-center-text">12</td>,
 <td class="table-body__cell u-center-text">930</td>,
 <td class="table-body__cell u-center-text">30</td>,
 <td class="table-body__cell u-center-text">1,962</td>,
 <td class="table-body__cell u-center-text">11</td>,
 <td class="table-body__cell u-center-text">516</td>,
 <td class="table-body__cell u-center-text">11</td>,
 <td class="table-body__cell u-center-text">495</td>,
 <td class="table-body__cell u-center-text">8</td>,
 <td class="table-body__cell u-center-text">0</td>]
```

In [182]:

```
match=[]
for m in Match:
    m=m.text
    match.append(m)
```

In [183]:

```
match
```

Out[183]:

```
['34',
 '4,097',
 '35',
 '4,157',
 '33',
 '3,392',
 '32',
 '3,161',
 '31',
 '2,815',
 '12',
 '930',
 '30',
 '1,962',
 '11',
 '516',
 '11',
 '495',
 '8',
 '0']
```

In [209]:

```
a=str(match)
match_fn=[match[i] for i in range(len(match)) if i%2==0]
match_fn.insert(0,'29')
print(match_fn)
```

```
['29', '34', '35', '33', '32', '31', '12', '30', '11', '11', '8']
```

In [211]:

```
match_fn
```

Out[211]:

```
['29', '34', '35', '33', '32', '31', '12', '30', '11', '11', '8']
```

In [210]:

```
points=[match[i] for i in range(len(match)) if i%2!=0]
points.insert(0,'4,837')
points
```

Out[210]:

```
['4,837',
 '4,097',
 '4,157',
 '3,392',
 '3,161',
 '2,815',
 '930',
 '1,962',
 '516',
 '495',
 '0']
```

In [212]:

```
rating=soup7.find_all('td',class_='table-body__cell u-text-right rating')
rating
```

Out[212]:

```
[<td class="table-body__cell u-text-right rating">121</td>,
 <td class="table-body__cell u-text-right rating">119</td>,
 <td class="table-body__cell u-text-right rating">103</td>,
 <td class="table-body__cell u-text-right rating">99</td>,
 <td class="table-body__cell u-text-right rating">91</td>,
 <td class="table-body__cell u-text-right rating">78</td>,
 <td class="table-body__cell u-text-right rating">65</td>,
 <td class="table-body__cell u-text-right rating">47</td>,
 <td class="table-body__cell u-text-right rating">45</td>,
 <td class="table-body__cell u-text-right rating">0</td>]
```

In [213]:

```
rating_fn=[]
for r in rating:
    r=r.text
    rating_fn.append(r)
```

In [214]:

```
rating_fn.insert(0,'167')
rating_fn
```

Out[214]:

```
['167', '121', '119', '103', '99', '91', '78', '65', '47', '45', '0']
```

In [215]:

```
#(a) Top 10 ODI teams in women's cricket along with the records for matches, points and rat

Top_10_ODI_WomenTeam=pd.DataFrame({'team':women,'matches':match_fn,'points':points,'rating'
Top_10_ODI_WomenTeam.head(10)
```

Out[215]:

	team	matches	points	rating
0	Australia	29	4,837	167
1	England	34	4,097	121
2	South Africa	35	4,157	119
3	India	33	3,392	103
4	New Zealand	32	3,161	99
5	West Indies	31	2,815	91
6	Bangladesh	12	930	78
7	Pakistan	30	1,962	65
8	Ireland	11	516	47
9	Sri Lanka	11	495	45

In [217]:

```
page8=requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi/batting
page8
```

Out[217]:

```
<Response [200]>
```

In [218]:

```
soup8=BeautifulSoup(page8.content,'html.parser')
soup8
```

Out[218]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Women's ODI Batting | Player Rankings | ICC" name="twit
ter:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI Batting | Player Rankings | ICC" property
```

In [219]:

```
player=soup8.find_all('td',class_='table-body__cell rankings-table__name name')
player
```

Out[219]:

```
[<td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/1817">Beth Mooney</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/1800">Natalie Sciver</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/3176">Laura Wolvaardt</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/469">Meg Lanning</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/465">Rachael Haynes</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/1809">Smriti Mandhana</a>
 </td>.
```

In [222]:

```
name=[]
for i in player:
    i=i.text.strip()
    name.append(i)
```

In [223]:

```
name.insert(0,'Alyssa Healy')
name
```

Out[223]:

```
['Alyssa Healy',
 'Beth Mooney',
 'Natalie Sciver',
 'Laura Wolvaardt',
 'Meg Lanning',
 'Rachael Haynes',
 'Smriti Mandhana',
 'Amy Satterthwaite',
 'Harmanpreet Kaur',
 'Chamari Athapaththu',
 'Tammy Beaumont',
 'Ellyse Perry',
 'Deandra Dottin',
 'Stafanie Taylor',
 'Amelia Kerr',
 'Sophie Devine',
 'Heather Knight',
 'Marizanne Kapp',
```

In [224]:

```
team=soup8.find_all('span',class_='table-body__logo-text')
team
```

Out[224]:

```
[<span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">SL</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">NZ</span>,
```

In [225]:

```
team_name=[]
for t in team:
    t=t.text
    team_name.append(t)
```

In [227]:

```
team_name.insert(0,'AUS')
team_name
```

Out[227]:

```
['AUS',
 'AUS',
 'ENG',
 'SA',
 'AUS',
 'AUS',
 'IND',
 'NZ',
 'IND',
 'SL',
 'ENG',
 'AUS',
 'WI',
 'WI',
 'NZ',
 'NZ',
 'ENG',
 'SA',
```

In [228]:

```
rating=soup8.find_all('td',class_='table-body__cell rating')
rating
```

Out[228]:

```
[<td class="table-body__cell rating">749</td>,
 <td class="table-body__cell rating">740</td>,
 <td class="table-body__cell rating">732</td>,
 <td class="table-body__cell rating">710</td>,
 <td class="table-body__cell rating">701</td>,
 <td class="table-body__cell rating">698</td>,
 <td class="table-body__cell rating">681</td>,
 <td class="table-body__cell rating">662</td>,
 <td class="table-body__cell rating">655</td>,
 <td class="table-body__cell rating">645</td>,
 <td class="table-body__cell rating">642</td>,
 <td class="table-body__cell rating">626</td>,
 <td class="table-body__cell rating">618</td>,
 <td class="table-body__cell rating">598</td>,
 <td class="table-body__cell rating">591</td>,
 <td class="table-body__cell rating">585</td>,
 <td class="table-body__cell rating">585</td>,
 <td class="table-body__cell rating">569</td>,
```

In [229]:

```
rating_r=[]
for j in rating:
    j=j.text
    rating_r.append(j)
```

In [230]:

```
rating_r.insert(0,'785')
rating_r
```

Out[230]:

```
['785',
 '749',
 '740',
 '732',
 '710',
 '701',
 '698',
 '681',
 '662',
 '655',
 '645',
 '642',
 '626',
 '618',
 '598',
 '591',
 '585',
 '585'.
```

In [231]:

```
#(b) Top 10 women's ODI Batting players along with the records of their team and rating.

Top_10_ODI_WomenBatsman=pd.DataFrame({'Name':name,'Team':team_name,'Rating':rating_r})
Top_10_ODI_WomenBatsman.head(10)
```

Out[231]:

	Name	Team	Rating
0	Alyssa Healy	AUS	785
1	Beth Mooney	AUS	749
2	Natalie Sciver	ENG	740
3	Laura Wolvaardt	SA	732
4	Meg Lanning	AUS	710
5	Rachael Haynes	AUS	701
6	Smriti Mandhana	IND	698
7	Amy Satterthwaite	NZ	681
8	Harmanpreet Kaur	IND	662
9	Chamari Athapaththu	SL	655

In [232]:

```
page9=requests.get('https://www.icc-cricket.com/rankings/womens/player-rankings/odi/all-rou
page9
```

Out[232]:

```
<Response [200]>
```

In [233]:

```
soup9=BeautifulSoup(page9.content,"html.parser")
soup9
```

Out[233]:

```
<!DOCTYPE html>

<html lang="en">
<head>
<meta content="ICC Women's ODI All Rounder | Player Rankings | ICC" name
="twitter:title"/>
<meta content="website" property="og:type"/>
<meta content="summary_large_image" property="twitter:card"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="description"/>
<meta content="@icc" property="twitter:site"/>
<meta content="Official ICC Cricket website - live matches, scores, news,
highlights, commentary, rankings, videos and fixtures from the Internation
al Cricket Council." name="twitter:description"/>
<meta content="https://www.icc-cricket.com/resources/ver/i/elements/defaul
t-thumbnail.jpg" name="twitter:image"/>
<meta content="ICC Women's ODI All Rounder | Player Rankings | ICC" proper
```

In [234]:

```
player=soup9.find_all('td',class_='table-body__cell rankings-table__name name')
player
```

Out[234]:

```
[<td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/1800">Natalie Sciver</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/424">Marizanne Kapp</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/3126">Hayley Matthews</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/3756">Amelia Kerr</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/3192">Deepti Sharma</a>
 </td>,
 <td class="table-body__cell rankings-table__name name">
 <a href="/rankings/womens/player-rankings/4128">Ashleigh Gardner</a>
 </td>.
```

In [238]:

```
name=[]
for w in player:
    w=w.text.strip()
    name.append(w)
```

In [239]:

```
name.insert(0,'Ellyse Perry')
name
```

Out[239]:

```
['Ellyse Perry',
 'Natalie Sciver',
 'Marizanne Kapp',
 'Hayley Matthews',
 'Amelia Kerr',
 'Deepti Sharma',
 'Ashleigh Gardner',
 'Jess Jonassen',
 'Jhulan Goswami',
 'Sophie Ecclestone',
 'Katherine Brunt',
 'Stafanie Taylor',
 'Sophie Devine',
 'Rumana Ahmed',
 'Nida Dar',
 'Chloe-Lesleigh Tryon',
 'Salma Khatun',
 'Chamari Athapaththu',
 'Sune Luus',
 'Harmanpreet Kaur']
```

In [240]:

```
team=soup9.find_all('span',class_='table-body__logo-text')
team
```

Out[240]:

```
[<span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">AUS</span>,
 <span class="table-body__logo-text">IND</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">ENG</span>,
 <span class="table-body__logo-text">WI</span>,
 <span class="table-body__logo-text">NZ</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">PAK</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">BAN</span>,
 <span class="table-body__logo-text">SL</span>,
 <span class="table-body__logo-text">SA</span>,
 <span class="table-body__logo-text">IND</span>]
```

In [241]:

```
Team=[]
for h in team:
    h=h.text
    Team.append(h)
```

In [242]:

```
Team.insert(0,'AUS')
Team
```

Out[242]:

```
['AUS',
 'ENG',
 'SA',
 'WI',
 'NZ',
 'IND',
 'AUS',
 'AUS',
 'IND',
 'ENG',
 'ENG',
 'WI',
 'NZ',
 'BAN',
 'PAK',
 'SA',
 'BAN',
 'SL',
 'SA',
 'IND']
```

In [243]:

```
rating=soup9.find_all('td',class_='table-body__cell rating')
rating
```

Out[243]:

```
[<td class="table-body__cell rating">372</td>,
 <td class="table-body__cell rating">349</td>,
 <td class="table-body__cell rating">339</td>,
 <td class="table-body__cell rating">336</td>,
 <td class="table-body__cell rating">271</td>,
 <td class="table-body__cell rating">270</td>,
 <td class="table-body__cell rating">246</td>,
 <td class="table-body__cell rating">219</td>,
 <td class="table-body__cell rating">217</td>,
 <td class="table-body__cell rating">215</td>,
 <td class="table-body__cell rating">207</td>,
 <td class="table-body__cell rating">202</td>,
 <td class="table-body__cell rating">201</td>,
 <td class="table-body__cell rating">200</td>,
 <td class="table-body__cell rating">197</td>,
 <td class="table-body__cell rating">178</td>,
 <td class="table-body__cell rating">178</td>,
 <td class="table-body__cell rating">171</td>,
 <td class="table-body__cell rating">160</td>]
```

In [244]:

```
Ratings=[]
for r in rating:
    r=r.text
    Ratings.append(r)
```

In [245]:

```
Ratings.insert(0,'374')
Ratings
```

Out[245]:

```
['374',
 '372',
 '349',
 '339',
 '336',
 '271',
 '270',
 '246',
 '219',
 '217',
 '215',
 '207',
 '202',
 '201',
 '200',
 '197',
 '178',
 '178',
 '171',
 '160']
```

In [246]:

```
#(c) Top 10 women's ODI all-rounder along with the records of their team and rating.

Top_10_ODI_Women_all_rounder=pd.DataFrame({'Name':name,'Team':Team,'Rating':Ratings})
Top_10_ODI_Women_all_rounder.head(10)
```

Out[246]:

	Name	Team	Rating
0	Ellyse Perry	AUS	374
1	Natalie Sciver	ENG	372
2	Marizanne Kapp	SA	349
3	Hayley Matthews	WI	339
4	Amelia Kerr	NZ	336
5	Deepti Sharma	IND	271
6	Ashleigh Gardner	AUS	270
7	Jess Jonassen	AUS	246
8	Jhulan Goswami	IND	219
9	Sophie Ecclestone	ENG	217

7) Write a python program to scrape mentioned news details from https://www.cnbc.com/world/?region=world (https://www.cnbc.com/world/?region=world) : i) Headline ii) Time iii) News Link

In [ ]:

```
url=" https://www.cnbc.com/world/?region=world "
```

In [247]:

```
page10=requests.get( "https://www.cnbc.com/world/?region=world" )
page10
```

Out[247]:

```
<Response [200]>
```

In [249]:

```
soup10=BeautifulSoup(page10.content,"html.parser")
print(soup10.prettify())
```

```
<!DOCTYPE html>
<html itemscope="" itemtype="https://schema.org/WebPage" lang="en" prefix
="og=https://ogp.me/ns#">
 <head>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/icomoon.ttf" rel="preload" type="font/ttf"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/351C86_0_0.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/351C86_1_0.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/351C86_2_0.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/351C86_3_0.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/351C86_4_0.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
bcfm.com/dist/LyonText-Bold-Web.woff2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="anonymous" href="https://static-redesign.cn
```

In [250]:

```
Headline=soup10.find_all('a',class_='LatestNews-headline')
Headline
```

Out[250]:

```
[<a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/over
-60percent-of-fortune-1000-corporate-boards-lack-latino-representation.htm
l" title="Latinos are seeing the least amount of growth in corporate board
representation">Latinos are seeing the least amount of growth in corporate
board representation</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/mode
rna-asks-fda-to-authorize-omicron-covid-boosters-for-children-as-young-as-
6-years-old.html" title="Moderna asks FDA to authorize omicron Covid boost
ers for kids 6- to 11-year-olds">Moderna asks FDA to authorize omicron Cov
id boosters for kids 6- to 11-year-olds</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/inve
sting-club-were-making-a-small-buy-in-fridays-terrible-market-adding-high-
quality-on-sale.html" title="We're making a small buy in Friday's terrible
market, adding high-quality on sale">We're making a small buy in Friday's
terrible market, adding high-quality on sale</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/ford
s-supply-chain-problems-include-blue-oval-badges-for-f-series-pickups.htm
l" title="Ford's supply-chain problems include blue oval badges for F-Seri
```

In [252]:

```
headline=[]
for t in Headline:
    t=t.text
    headline.append(t)
```

In [253]:

```
headline
```

Out[253]:

```
['Latinos are seeing the least amount of growth in corporate board represe
ntation',
 'Moderna asks FDA to authorize omicron Covid boosters for kids 6- to 11-y
ear-olds',
 "We're making a small buy in Friday's terrible market, adding high-qualit
y on sale",
 "Ford's supply-chain problems include blue oval badges for F-Series picku
ps",
 "Inflation is eating up small business owners' profits, says new survey",
 'Bond market plunge means the low for stocks is not in yet, Bank of Ameri
ca says',
 'Inventory is piling up as consumer demand slows. Bad news for these stoc
ks',
 "What's behind this latest selloff in financial markets",
 'Prosecutors recommend against charging Gaetz in sex trafficking probe: R
eport',
 'Citi says sterling-dollar parity is possible as UK risks currency crisi
s'.
```

In [254]:

```
timing=soup10.find_all('time',class_='LatestNews-timestamp')
timing
```

Out[254]:

```
[<time class="LatestNews-timestamp">8 Min Ago</time>,
 <time class="LatestNews-timestamp">11 Min Ago</time>,
 <time class="LatestNews-timestamp">15 Min Ago</time>,
 <time class="LatestNews-timestamp">17 Min Ago</time>,
 <time class="LatestNews-timestamp">23 Min Ago</time>,
 <time class="LatestNews-timestamp">26 Min Ago</time>,
 <time class="LatestNews-timestamp">30 Min Ago</time>,
 <time class="LatestNews-timestamp">38 Min Ago</time>,
 <time class="LatestNews-timestamp">47 Min Ago</time>,
 <time class="LatestNews-timestamp">56 Min Ago</time>,
 <time class="LatestNews-timestamp">1 Hour Ago</time>,
 <time class="LatestNews-timestamp">1 Hour Ago</time>,
 <time class="LatestNews-timestamp">2 Hours Ago</time>,
 <time class="LatestNews-timestamp">2 Hours Ago</time>,
 <time class="LatestNews-timestamp">2 Hours Ago</time>,
 <time class="LatestNews-timestamp">2 Hours Ago</time>,
 <time class="LatestNews-timestamp">3 Hours Ago</time>,
 <time class="LatestNews-timestamp">3 Hours Ago</time>,
```

In [255]:

```
Time=[]
for s in timing:
    s=s.text
    Time.append(s)
```

In [256]:

```
Time
```

Out[256]:

```
['8 Min Ago',
 '11 Min Ago',
 '15 Min Ago',
 '17 Min Ago',
 '23 Min Ago',
 '26 Min Ago',
 '30 Min Ago',
 '38 Min Ago',
 '47 Min Ago',
 '56 Min Ago',
 '1 Hour Ago',
 '1 Hour Ago',
 '2 Hours Ago',
 '2 Hours Ago',
 '2 Hours Ago',
 '2 Hours Ago',
 '3 Hours Ago',
 '3 Hours Ago',
```

In [258]:

```
link=soup10.find_all('a',class_='LatestNews-headline')
link
```

Out[258]:

```
[<a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/over
-60percent-of-fortune-1000-corporate-boards-lack-latino-representation.htm
l" title="Latinos are seeing the least amount of growth in corporate board
representation">Latinos are seeing the least amount of growth in corporate
board representation</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/mode
rna-asks-fda-to-authorize-omicron-covid-boosters-for-children-as-young-as-
6-years-old.html" title="Moderna asks FDA to authorize omicron Covid boost
ers for kids 6- to 11-year-olds">Moderna asks FDA to authorize omicron Cov
id boosters for kids 6- to 11-year-olds</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/inve
sting-club-were-making-a-small-buy-in-fridays-terrible-market-adding-high-
quality-on-sale.html" title="We're making a small buy in Friday's terrible
market, adding high-quality on sale">We're making a small buy in Friday's
terrible market, adding high-quality on sale</a>,
 <a class="LatestNews-headline" href="https://www.cnbc.com/2022/09/23/ford
s-supply-chain-problems-include-blue-oval-badges-for-f-series-pickups.htm
l" title="Ford's supply-chain problems include blue oval badges for F-Seri
```

In [264]:

```
URL=[]
for j in link:
    URL.append(j.get('href'))
```

In [265]:

```
URL
```

Out[265]:

```
['https://www.cnbc.com/2022/09/23/over-60percent-of-fortune-1000-corporate
-boards-lack-latino-representation.html',
 'https://www.cnbc.com/2022/09/23/moderna-asks-fda-to-authorize-omicron-co
vid-boosters-for-children-as-young-as-6-years-old.html',
 'https://www.cnbc.com/2022/09/23/investing-club-were-making-a-small-buy-i
n-fridays-terrible-market-adding-high-quality-on-sale.html',
 'https://www.cnbc.com/2022/09/23/fords-supply-chain-problems-include-blue
-oval-badges-for-f-series-pickups.html',
 'https://www.cnbc.com/2022/09/23/small-businesses-say-inflation-is-cuttin
g-into-profits-new-survey.html',
 'https://www.cnbc.com/2022/09/23/bond-market-plunge-means-the-low-for-sto
cks-is-not-in-yet-bank-of-americas-hartnett-says.html',
 'https://www.cnbc.com/2022/09/23/inventory-is-piling-up-as-consumer-deman
d-slows-thats-bad-news-for-these-stocks.html',
 'https://www.cnbc.com/2022/09/23/from-the-fed-to-europes-currency-crisis-
heres-whats-behind-this-selloff-in-financial-markets.html',
 'https://www.cnbc.com/2022/09/23/matt-gaetz-sex-trafficking-probe-prosecu
tors-recommend-against-charges.html',
```

In [266]:

```
cnbc_news_details=pd.DataFrame({'Headline':headline,'Time':Time,'News_Link':URL})
cnbc_news_details
```

Out[266]:

	Headline	Time	News_Link
0	Latinos are seeing the least amount of growth ...	8 Min Ago	https://www.cnbc.com/2022/09/23/over-60percent...
1	Moderna asks FDA to authorize omicron Covid bo...	11 Min Ago	https://www.cnbc.com/2022/09/23/moderna-asks-f...
2	We're making a small buy in Friday's terrible ...	15 Min Ago	https://www.cnbc.com/2022/09/23/investing-club...
3	Ford's supply-chain problems include blue oval...	17 Min Ago	https://www.cnbc.com/2022/09/23/fords-supply-c...
4	Inflation is eating up small business owners' ...	23 Min Ago	https://www.cnbc.com/2022/09/23/small-business...
5	Bond market plunge means the low for stocks is...	26 Min Ago	https://www.cnbc.com/2022/09/23/bond-market-pl...
6	Inventory is piling up as consumer demand slow...	30 Min Ago	https://www.cnbc.com/2022/09/23/inventory-is-p...
7	What's behind this latest selloff in financial...	38 Min Ago	https://www.cnbc.com/2022/09/23/from-the-fed-t...
8	Prosecutors recommend against charging Gaetz i...	47 Min Ago	https://www.cnbc.com/2022/09/23/matt-gaetz-sex...
9	Citi says sterling-dollar parity is possible a...	56 Min Ago	https://www.cnbc.com/2022/09/23/citi-says-ster...
10	Here's what experts say to do with your money ...	1 Hour Ago	https://www.cnbc.com/2022/09/23/what-to-do-wit...
11	Goldman says buy these kinds of stocks as firm...	1 Hour Ago	https://www.cnbc.com/2022/09/23/goldman-says-b...
12	Costco isn't raising membership fees after ear...	2 Hours Ago	https://www.cnbc.com/2022/09/23/costco-maintai...
13	Mass protests in Iran, sparked by woman's deat...	2 Hours Ago	https://www.cnbc.com/2022/09/23/mass-protests-...
14	One thing this millennial business owner says ...	2 Hours Ago	https://www.cnbc.com/2022/09/23/millennial-bus...
15	JetBlue ground operations workers seek union v...	2 Hours Ago	https://www.cnbc.com/2022/09/23/jetblue-fleet-...
16	Buy these 'free cash flow favorites' as the S&...	3 Hours Ago	https://www.cnbc.com/2022/09/23/buy-these-free...
17	Google CEO Pichai tells employees not to 'equa...	3 Hours Ago	https://www.cnbc.com/2022/09/23/google-ceo-pic...
18	What Cramer is watching — 2-year yield unstopp...	3 Hours Ago	https://www.cnbc.com/2022/09/23/what-cramer-is...
19	Markets don't like U.K. stimulus plan because ...	4 Hours Ago	https://www.cnbc.com/2022/09/23/-financial-mar...
20	Here are Friday's biggest analyst calls: Apple...	4 Hours Ago	https://www.cnbc.com/2022/09/23/here-are-frida...
21	Inflation Reduction Act's expanded biofuel inc...	4 Hours Ago	https://www.cnbc.com/2022/09/23/inflation-redu...

	Headline	Time	News_Link
22	Getting married? How to know when to combine y...	4 Hours Ago	https://www.cnbc.com/2022/09/23/how-to-know-wh...
23	Stocks making the biggest moves premarket: Fed...	5 Hours Ago	https://www.cnbc.com/2022/09/23/stocks-making-...
24	Wedbush upgrades fuboTV to outperform citing '...	5 Hours Ago	https://www.cnbc.com/2022/09/23/wedbush-upgrad...
25	5 things to know before the stock market opens...	5 Hours Ago	https://www.cnbc.com/2022/09/23/5-things-to-kn...
26	Finding yield in health care stocks during the...	6 Hours Ago	https://www.cnbc.com/2022/09/23/finding-yield-...
27	British pound plunges, bonds sink after govern...	6 Hours Ago	https://www.cnbc.com/2022/09/23/british-pound-...
28	Goldman cuts S&P target to 3,600, sees it fall...	6 Hours Ago	https://www.cnbc.com/2022/09/23/goldman-cuts-y...
29	BMO upgrades Domino's Pizza, predicts 35% rebo...	6 Hours Ago	https://www.cnbc.com/2022/09/23/bmo-upgrades-d...

8) Write a python program to scrape the details of most downloaded articles from AI in last 90 days. https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles (https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles) Scrape below mentioned details : i) Paper Title ii) Authors iii) Published Date iv) Paper URL

In [268]:

```
url='https://www.journals.elsevier.com/artificial-intelligence/most-downloaded-articles'
```

In [269]:

```
page11=requests.get(url)
page11
```

Out[269]:

```
<Response [200]>
```

In [271]:

```
soup11=BeautifulSoup(page11.content,"html.parser")
print(soup11.prettify())
```

```
<!DOCTYPE html>
<html>
 <head>
  <link as="font" crossorigin="" href="/fonts/NexusSansWebPro-Regular.woff
2" rel="preload" type="font/woff2"/>
  <link as="font" crossorigin="" href="/fonts/ElsevierGulliver-Regular.wof
f2" rel="preload" type="font/woff2"/>
  <script src="//assets.adobedtm.com/4a848ae9611a/4da1513b0562/launch-a536
f9fbceee.min.js">
  </script>
  <meta name="next-font-preconnect"/>
  <meta content="width=device-width" name="viewport"/>
  <meta charset="utf-8"/>
  <meta content="en_US" name="og:locale"/>
  <meta content="Most Downloaded Articles - Artificial Intelligence - Jour
nal - Elsevier" property="og:title"/>
  <meta content="The journal of Artificial Intelligence (AIJ)  welcomes pa
pers on broad aspects of AI that constitute advances in the overall field
including, but not limited …" property="og:description"/>
```

In [272]:

```
title=soup11.find_all('h2',class_='sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA
title
```

Out[272]:

```
[<h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">R
eward is enough</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">M
aking sense of raw input</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">L
aw and logic: A review from an argumentation perspective</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">C
reativity and artificial intelligence</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">A
rtificial cognition for social human-robot interaction: An implementation
</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">E
xplanation in artificial intelligence: Insights from the social sciences</
h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">M
aking sense of sensory input</h2>,
 <h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0 sc-1nmom32-1 hqhUYH ebTA-dR">C
onflict-based search for optimal multi-agent pathfinding</h2>,
```

In [273]:

```
Title=[]
for t in title:
    t=t.text
    Title.append(t)
```

In [274]:

```
Title
```

Out[274]:

```
['Reward is enough',
 'Making sense of raw input',
 'Law and logic: A review from an argumentation perspective',
 'Creativity and artificial intelligence',
 'Artificial cognition for social human-robot interaction: An implementati
on',
 'Explanation in artificial intelligence: Insights from the social science
s',
 'Making sense of sensory input',
 'Conflict-based search for optimal multi-agent pathfinding',
 'Between MDPs and semi-MDPs: A framework for temporal abstraction in rein
forcement learning',
 'The Hanabi challenge: A new frontier for AI research',
 'Evaluating XAI: A comparison of rule-based and example-based explanation
s',
 'Argumentation in artificial intelligence',
 'Algorithms for computing strategies in two-player simultaneous move game
s'.
```

In [275]:

```
authors=soup11.find_all('span',class_='sc-1w3fpd7-0 pgLAT')
authors
```

Out[275]:

```
[<span class="sc-1w3fpd7-0 pgLAT">Silver, David, Singh, Satinder, Precup,
Doina, Sutton, Richard S. </span>,
 <span class="sc-1w3fpd7-0 pgLAT">Evans, Richard, Bošnjak, Matko and 5 mor
e</span>,
 <span class="sc-1w3fpd7-0 pgLAT">Prakken, Henry, Sartor, Giovanni </span
>,
 <span class="sc-1w3fpd7-0 pgLAT">Boden, Margaret A. </span>,
 <span class="sc-1w3fpd7-0 pgLAT">Lemaignan, Séverin, Warnier, Mathieu and
3 more</span>,
 <span class="sc-1w3fpd7-0 pgLAT">Miller, Tim </span>,
 <span class="sc-1w3fpd7-0 pgLAT">Evans, Richard, Hernández-Orallo, José a
nd 3 more</span>,
 <span class="sc-1w3fpd7-0 pgLAT">Sharon, Guni, Stern, Roni, Felner, Arie
l, Sturtevant, Nathan R. </span>,
 <span class="sc-1w3fpd7-0 pgLAT">Sutton, Richard S., Precup, Doina, Sing
h, Satinder </span>,
 <span class="sc-1w3fpd7-0 pgLAT">Bard, Nolan, Foerster, Jakob N. and 13 m
ore</span>.
```

In [276]:

```
Authors=[]
for a in authors:
    a=a.text
    Authors.append(a)
```

In [277]:

```
Authors
```

Out[277]:

```
['Silver, David, Singh, Satinder, Precup, Doina, Sutton, Richard S. ',
 'Evans, Richard, Bošnjak, Matko and 5 more',
 'Prakken, Henry, Sartor, Giovanni ',
 'Boden, Margaret A. ',
 'Lemaignan, Séverin, Warnier, Mathieu and 3 more',
 'Miller, Tim ',
 'Evans, Richard, Hernández-Orallo, José and 3 more',
 'Sharon, Guni, Stern, Roni, Felner, Ariel, Sturtevant, Nathan R. ',
 'Sutton, Richard S., Precup, Doina, Singh, Satinder ',
 'Bard, Nolan, Foerster, Jakob N. and 13 more',
 'van der Waa, Jasper, Nieuwburg, Elisabeth, Cremers, Anita, Neerincx, Mar
k ',
 'Bench-Capon, T.J.M., Dunne, Paul E. ',
 'Bošanský, Branislav, Lisý, Viliam and 3 more',
 'Luo, Wenhan, Xing, Junliang and 4 more',
 'Blum, Avrim L., Langley, Pat ',
 'Arora, Saurabh, Doshi, Prashant ',
 'Aas, Kjersti, Jullum, Martin, Løland, Anders '.
```

In [278]:

```
date=soup11.find_all('span',class_='sc-1thf9ly-2 bKddwo')
date
```

Out[278]:

```
[<span class="sc-1thf9ly-2 bKddwo"><span>October 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>October 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>October 2015</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>August 1998</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>June 2017</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>February 2019</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>April 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>February 2015</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>August 1999</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>March 2020</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>February 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>October 2007</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>August 2016</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>April 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>December 1997</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>August 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>September 2021</span></span>,
 <span class="sc-1thf9ly-2 bKddwo"><span>June 2021</span></span>,
```

In [279]:

```
Published_date=[]
for p in date:
    p=p.text
    Published_date.append(p)
```

In [280]:

```
Published_date
```

Out[280]:

```
['October 2021',
 'October 2021',
 'October 2015',
 'August 1998',
 'June 2017',
 'February 2019',
 'April 2021',
 'February 2015',
 'August 1999',
 'March 2020',
 'February 2021',
 'October 2007',
 'August 2016',
 'April 2021',
 'December 1997',
 'August 2021',
 'September 2021',
 'June 2021',
```

In [282]:

```
link=soup11.find_all('a',class_='sc-5smygv-0 nrDZj')
link
```

Out[282]:

```
[<a class="sc-5smygv-0 nrDZj" href="https://www.sciencedirect.com/science/
article/pii/S0004370221000862"><h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0
sc-1nmom32-1 hqhUYH ebTA-dR">Reward is enough</h2></a>,
 <a class="sc-5smygv-0 nrDZj" href="https://www.sciencedirect.com/science/
article/pii/S0004370221000722"><h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0
sc-1nmom32-1 hqhUYH ebTA-dR">Making sense of raw input</h2></a>,
 <a class="sc-5smygv-0 nrDZj" href="https://www.sciencedirect.com/science/
article/pii/S0004370215000910"><h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0
sc-1nmom32-1 hqhUYH ebTA-dR">Law and logic: A review from an argumentation
perspective</h2></a>,
 <a class="sc-5smygv-0 nrDZj" href="https://www.sciencedirect.com/science/
article/pii/S0004370298000551"><h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0
sc-1nmom32-1 hqhUYH ebTA-dR">Creativity and artificial intelligence</h2></
a>,
 <a class="sc-5smygv-0 nrDZj" href="https://www.sciencedirect.com/science/
article/pii/S0004370216300790"><h2 class="sc-1qrq3sd-1 MKjKb sc-1nmom32-0
sc-1nmom32-1 hqhUYH ebTA-dR">Artificial cognition for social human-robot i
nteraction: An implementation</h2></a>,
```

In [283]:

```
paper_link=[]
for lk in link:
    paper_link.append(lk.get('href'))
```

In [284]:

```
paper_link
```

Out[284]:

```
['https://www.sciencedirect.com/science/article/pii/S0004370221000862',
 'https://www.sciencedirect.com/science/article/pii/S0004370221000722',
 'https://www.sciencedirect.com/science/article/pii/S0004370215000910',
 'https://www.sciencedirect.com/science/article/pii/S0004370298000551',
 'https://www.sciencedirect.com/science/article/pii/S0004370216300790',
 'https://www.sciencedirect.com/science/article/pii/S0004370218305988',
 'https://www.sciencedirect.com/science/article/pii/S0004370220301855',
 'https://www.sciencedirect.com/science/article/pii/S0004370214001386',
 'https://www.sciencedirect.com/science/article/pii/S0004370299000521',
 'https://www.sciencedirect.com/science/article/pii/S0004370219300116',
 'https://www.sciencedirect.com/science/article/pii/S0004370220301533',
 'https://www.sciencedirect.com/science/article/pii/S0004370207000793',
 'https://www.sciencedirect.com/science/article/pii/S0004370216300285',
 'https://www.sciencedirect.com/science/article/pii/S0004370220301958',
 'https://www.sciencedirect.com/science/article/pii/S0004370297000635',
 'https://www.sciencedirect.com/science/article/pii/S0004370221000515',
 'https://www.sciencedirect.com/science/article/pii/S0004370221000539',
 'https://www.sciencedirect.com/science/article/pii/S0004370221000096',
```

In [285]:

```
AI_Journals=pd.DataFrame({'Paper_Title':Title,'Authors':Authors,'Published_Date':Published_
AI_Journals
```

Out[285]:

	Paper_Title	Authors	Published_Date	Paper_URL
0	Reward is enough	Silver, David, Singh, Satinder, Precup, Doina,...	October 2021	https://www.sciencedirect.com/science/article/...
1	Making sense of raw input	Evans, Richard, Bošnjak, Matko and 5 more	October 2021	https://www.sciencedirect.com/science/article/...
2	Law and logic: A review from an argumentation ...	Prakken, Henry, Sartor, Giovanni	October 2015	https://www.sciencedirect.com/science/article/...
3	Creativity and artificial intelligence	Boden, Margaret A.	August 1998	https://www.sciencedirect.com/science/article/...
4	Artificial cognition for social human–robot in...	Lemaignan, Séverin, Warnier, Mathieu and 3 more	June 2017	https://www.sciencedirect.com/science/article/...
5	Explanation in artificial intelligence: Insigh...	Miller, Tim	February 2019	https://www.sciencedirect.com/science/article/...
6	Making sense of sensory input	Evans, Richard, Hernández-Orallo, José and 3 more	April 2021	https://www.sciencedirect.com/science/article/...
7	Conflict-based search for optimal multi-agent ...	Sharon, Guni, Stern, Roni, Felner, Ariel, Stur...	February 2015	https://www.sciencedirect.com/science/article/...
8	Between MDPs and semi-MDPs: A framework for te...	Sutton, Richard S., Precup, Doina, Singh, Sati...	August 1999	https://www.sciencedirect.com/science/article/...
9	The Hanabi challenge: A new frontier for AI re...	Bard, Nolan, Foerster, Jakob N. and 13 more	March 2020	https://www.sciencedirect.com/science/article/...
10	Evaluating XAI: A comparison of rule-based and...	van der Waa, Jasper, Nieuwburg, Elisabeth, Cre...	February 2021	https://www.sciencedirect.com/science/article/...

	Paper_Title	Authors	Published_Date	Paper_URL
11	Argumentation in artificial intelligence	Bench-Capon, T.J.M., Dunne, Paul E.	October 2007	https://www.sciencedirect.com/science/article/...
12	Algorithms for computing strategies in two-pla...	Bošanský, Branislav, Lisý, Viliam and 3 more	August 2016	https://www.sciencedirect.com/science/article/...
13	Multiple object tracking: A literature review	Luo, Wenhan, Xing, Junliang and 4 more	April 2021	https://www.sciencedirect.com/science/article/...
14	Selection of relevant features and examples in...	Blum, Avrim L., Langley, Pat	December 1997	https://www.sciencedirect.com/science/article/...
15	A survey of inverse reinforcement learning: Ch...	Arora, Saurabh, Doshi, Prashant	August 2021	https://www.sciencedirect.com/science/article/...
16	Explaining individual predictions when feature...	Aas, Kjersti, Jullum, Martin, Løland, Anders	September 2021	https://www.sciencedirect.com/science/article/...
17	A review of possible effects of cognitive bias...	Kliegr, Tomáš, Bahník, Štěpán, Fürnkranz, Joha...	June 2021	https://www.sciencedirect.com/science/article/...
18	Integrating social power into the decision-mak...	Pereira, Gonçalo, Prada, Rui, Santos, Pedro A.	December 2016	https://www.sciencedirect.com/science/article/...
19	“That's (not) the output I expected!” On the r...	Riveiro, Maria, Thill, Serge	September 2021	https://www.sciencedirect.com/science/article/...
20	Explaining black-box classifiers using post-ho...	Kenny, Eoin M., Ford, Courtney, Quinn, Molly, ...	May 2021	https://www.sciencedirect.com/science/article/...
21	Algorithm runtime prediction: Methods & evalua...	Hutter, Frank, Xu, Lin, Hoos, Holger H., Leyto...	January 2014	https://www.sciencedirect.com/science/article/...
22	Wrappers for feature subset selection	Kohavi, Ron, John, George H.	December 1997	https://www.sciencedirect.com/science/article/...
23	Commonsense visual sensemaking for autonomous ...	Suchan, Jakob, Bhatt, Mehul, Varadarajan, Srik...	October 2021	https://www.sciencedirect.com/science/article/...

	Paper_Title	Authors	Published_Date	Paper_URL
24	Quantum computation, quantum	Ying, Mingsheng	February 2010	https://www.sciencedirect.com/science/article/...

9) Write a python program to scrape mentioned details from dineout.co.in : i) Restaurant name ii) Cuisine iii) Location iv) Ratings v) Image URL

In [286]:

```
page12=requests.get('https://www.dineout.co.in/delhi-restaurants/buffet-special')
page12
```

Out[286]:

```
<Response [200]>
```

In [290]:

```
soup12=BeautifulSoup(page12.content,"html.parser")
soup12
```

Out[290]:

```
<!DOCTYPE html>
<html lang="en"><head><meta charset="utf-8"/><meta content="IE=edge" http-
equiv="X-UA-Compatible"/><meta content="width=device-width, initial-scale=
1.0, maximum-scale=1.0, user-scalable=no" name="viewport"/><link href="/ma
nifest.json" rel="manifest"/><style type="text/css">
            @font-face {
                font-family: 'dineicon';
                src:  url('/fonts/dineicon.eot');
                src:  url('/fonts/dineicon.eot#iefix') format('embedded-op
entype'),
                url('/fonts/dineicon.ttf') format('truetype'),
                url('/fonts/dineicon.woff') format('woff'),
                url('/fonts/dineicon.svg#dineicon') format('svg');
                font-weight: normal;
                                font-style: normal;
                                font-display: swap;
            }
            .hide {
```

In [291]:

```
restaurant=soup12.find_all('a',class_='restnt-name ellipsis')
restaurant
```

Out[291]:

```
[<a analytics-action="RestaurantCardClick" analytics-label="86792_Castle Bar
beque" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-r
estarant" href="/delhi/castle-barbeque-connaught-place-central-delhi-86792">
Castle Barbeque</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="59633_Jungle Jam
boree" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-r
estarant" href="/delhi/jungle-jamboree-lajpat-nagar-3-south-delhi-59633">Jun
gle Jamboree</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="38113_Castle Bar
beque" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-r
estarant" href="/delhi/castle-barbeque-tagore-garden-west-delhi-38113">Castl
e Barbeque</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="406_Cafe Knosh"
class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-restaran
t" href="/delhi/cafe-knosh-shahdara-east-delhi-406">Cafe Knosh</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="79307_The Barbeq
ue Company" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon
|w1-restarant" href="/delhi/the-barbeque-company-sector-38a-noida-79307">The
Barbeque Company</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="2687_India Gril
l" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-resta
rant" href="/delhi/india-grill-saket-south-delhi-2687">India Grill</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="52501_Delhi Barb
eque" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-re
starant" href="/delhi/delhi-barbeque-mahipalpur-south-delhi-52501">Delhi Bar
beque</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="34822_The Monarc
h - Bar Be Que Village" class="restnt-name ellipsis" data-w-onclick="sendAna
lyticsCommon|w1-restarant" href="/delhi/the-monarch-bar-be-que-village-indir
apuram-ghaziabad-34822">The Monarch - Bar Be Que Village</a>,
 <a analytics-action="RestaurantCardClick" analytics-label="549_Indian Grill
Room" class="restnt-name ellipsis" data-w-onclick="sendAnalyticsCommon|w1-re
starant" href="/delhi/indian-grill-room-golf-course-road-gurgaon-549">Indian
Grill Room</a>]
```

In [292]:

```
rest_name=[]
for y in restaurant:
    y=y.text
    rest_name.append(y)
```

In [293]:

```
rest_name
```

Out[293]:

```
['Castle Barbeque',
 'Jungle Jamboree',
 'Castle Barbeque',
 'Cafe Knosh',
 'The Barbeque Company',
 'India Grill',
 'Delhi Barbeque',
 'The Monarch - Bar Be Que Village',
 'Indian Grill Room']
```

In [294]:

```
cuisine=soup12.find_all('div',class_='detail-info')
cuisine
```

Out[294]:

```
[<div class="detail-info"><ul><li><span class="double-line-ellipsis"><span
>₹ 2,000 for 2 (approx)</span><span> | </span><a data-w-onclick="stopClick
Propagation|w1-restarant" href="/delhi-restaurants/chinese-cuisine">Chines
e</a><span>, </span><a data-w-onclick="stopClickPropagation|w1-restarant"
href="/delhi-restaurants/north-indian-cuisine">North Indian</a></span></li
><li class="ellipsis"></li></ul></div>,
 <div class="detail-info"><ul><li><span class="double-line-ellipsis"><span
>₹ 1,680 for 2 (approx)</span><span> | </span><a data-w-onclick="stopClick
Propagation|w1-restarant" href="/delhi-restaurants/north-indian-cuisine">N
orth Indian</a><span>, </span><a data-w-onclick="stopClickPropagation|w1-r
estarant" href="/delhi-restaurants/asian-cuisine">Asian</a><span>, </span>
<a data-w-onclick="stopClickPropagation|w1-restarant" href="/delhi-restaur
ants/italian-cuisine">Italian</a></span></li><li class="ellipsis"></li></u
l></div>,
 <div class="detail-info"><ul><li><span class="double-line-ellipsis"><span
>₹ 2,000 for 2 (approx)</span><span> | </span><a data-w-onclick="stopClick
Propagation|w1-restarant" href="/delhi-restaurants/chinese-cuisine">Chines
e</a><span>, </span><a data-w-onclick="stopClickPropagation|w1-restarant"
```

In [303]:

```
Cuisine=[]
for c in cuisine:
    c=c.text.split('|')[1]
    Cuisine.append(c)
```

In [304]:

```
Cuisine
```

Out[304]:

```
[' Chinese, North Indian',
 ' North Indian, Asian, Italian',
 ' Chinese, North Indian',
 ' Italian, Continental',
 ' North Indian, Chinese4 deals available',
 ' North Indian, Italian',
 ' North Indian',
 ' North Indian',
 ' North Indian, Mughlai']
```

In [305]:

```
place=soup12.find_all('div',class_='restnt-loc ellipsis')
place
```

Out[305]:

```
[<div class="restnt-loc ellipsis" data-w-onclick="stopClickPropagation|w1-
restarant"><a data-name="Connaught Place" data-type="LocalityClick" href
="/delhi-restaurants/central-delhi/connaught-place">Connaught Place</a>, <
a data-name="Central Delhi" data-type="AreaClick" href="/delhi-restaurant
s/central-delhi">Central Delhi</a></div>,
 <div class="restnt-loc ellipsis" data-w-onclick="stopClickPropagation|w1-
restarant"><a href="/delhi-restaurants/3cs-mall-landmark">3CS Mall,</a><a
data-name="Lajpat Nagar - 3" data-type="LocalityClick" href="/delhi-restau
rants/south-delhi/lajpat-nagar-3">Lajpat Nagar - 3</a>, <a data-name="Sout
h Delhi" data-type="AreaClick" href="/delhi-restaurants/south-delhi">South
Delhi</a></div>,
 <div class="restnt-loc ellipsis" data-w-onclick="stopClickPropagation|w1-
restarant"><a href="/delhi-restaurants/pacific-mall-landmark">Pacific Mal
l,</a><a data-name="Tagore Garden" data-type="LocalityClick" href="/delhi-
restaurants/west-delhi/tagore-garden">Tagore Garden</a>, <a data-name="Wes
t Delhi" data-type="AreaClick" href="/delhi-restaurants/west-delhi">West D
elhi</a></div>,
 <div class="restnt-loc ellipsis" data-w-onclick="stopClickPropagation|w1-
```

In [307]:

```
Location=[]
for l in place:
    l=l.text
    Location.append(l)
```

In [308]:

```
Location
```

Out[308]:

```
['Connaught Place, Central Delhi',
 '3CS Mall,Lajpat Nagar - 3, South Delhi',
 'Pacific Mall,Tagore Garden, West Delhi',
 'The Leela Ambience Convention Hotel,Shahdara, East Delhi',
 'Gardens Galleria,Sector 38A, Noida',
 'Hilton Garden Inn,Saket, South Delhi',
 'Taurus Sarovar Portico,Mahipalpur, South Delhi',
 'Indirapuram Habitat Centre,Indirapuram, Ghaziabad',
 'Suncity Business Tower,Golf Course Road, Gurgaon']
```

In [309]:

```
rating=soup12.find_all('div',class_='restnt-rating rating-4')
rating
```

Out[309]:

```
[<div class="restnt-rating rating-4">4.1</div>,
 <div class="restnt-rating rating-4">3.9</div>,
 <div class="restnt-rating rating-4">3.9</div>,
 <div class="restnt-rating rating-4">4.3</div>,
 <div class="restnt-rating rating-4">4</div>,
 <div class="restnt-rating rating-4">3.9</div>,
 <div class="restnt-rating rating-4">3.7</div>,
 <div class="restnt-rating rating-4">3.8</div>,
 <div class="restnt-rating rating-4">4.3</div>]
```

In [311]:

```
Ratings=[]
for g in rating:
    g=g.text
    Ratings.append(g)
```

In [312]:

```
Ratings
```

Out[312]:

```
['4.1', '3.9', '3.9', '4.3', '4', '3.9', '3.7', '3.8', '4.3']
```

In [313]:

```
link=soup12.find_all('img',class_='no-img')
link
```

Out[313]:

```
[<img alt="Castle Barbeque" class="lazy-load-img no-img" data-gatype="Rest
aurantImageClick" data-src="https://im1.dineout.co.in/images/uploads/resta
urant/sharpen/8/k/b/p86792-16062953735fbe1f4d3fb7e.jpg?tr=tr:n-medium" dat
a-url="/delhi/castle-barbeque-connaught-place-central-delhi-86792" data-w-
onclick="cardClickHandler"/>,
 <img alt="Jungle Jamboree" class="lazy-load-img no-img" data-gatype="Rest
aurantImageClick" data-src="https://im1.dineout.co.in/images/uploads/resta
urant/sharpen/5/p/m/p59633-166088382462ff137009010.jpg?tr=tr:n-medium" dat
a-url="/delhi/jungle-jamboree-lajpat-nagar-3-south-delhi-59633" data-w-onc
lick="cardClickHandler"/>,
 <img alt="Castle Barbeque" class="lazy-load-img no-img" data-gatype="Rest
aurantImageClick" data-src="https://im1.dineout.co.in/images/uploads/resta
urant/sharpen/3/j/o/p38113-15959192065f1fcb666130c.jpg?tr=tr:n-medium" dat
a-url="/delhi/castle-barbeque-tagore-garden-west-delhi-38113" data-w-oncli
ck="cardClickHandler"/>,
 <img alt="Cafe Knosh" class="lazy-load-img no-img" data-gatype="Restauran
tImageClick" data-src="https://im1.dineout.co.in/images/uploads/restauran
t/sharpen/4/p/m/p406-15438184745c04ccea491bc.jpg?tr=tr:n-medium" data-url
```

In [316]:

```
image_link=[]
for i in link:
    image_link.append(i.get('data-src'))
```

In [317]:

```
image_link
```

Out[317]:

```
['https://im1.dineout.co.in/images/uploads/restaurant/sharpen/8/k/b/p86792-1
6062953735fbe1f4d3fb7e.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/p/m/p59633-1
66088382462ff137009010.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/j/o/p38113-1
5959192065f1fcb666130c.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/4/p/m/p406-154
38184745c04ccea491bc.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/7/p/k/p79307-1
6051787755fad1597f2bf9.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/2/v/t/p2687-14
82477169585cce712b90f.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/d/i/p52501-1
661855212630de5eceb6d2.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/3/n/o/p34822-1
5599107305cfa594a13c24.jpg?tr=tr:n-medium',
 'https://im1.dineout.co.in/images/uploads/restaurant/sharpen/5/y/f/p549-165
000147262590640c0afc.jpg?tr=tr:n-medium']
```

In [318]:

```
Dineout=pd.DataFrame({'Restaurant_name':rest_name,'Cuisine':Cuisine,'Location':Location,'Ra
                      'Image_URL':image_link})
Dineout
```

Out[318]:

	Restaurant_name	Cuisine	Location	Ratings	Ima
0	Castle Barbeque	Chinese, North Indian	Connaught Place, Central Delhi	4.1	https://im1.dineout.co.in/images/upload
1	Jungle Jamboree	North Indian, Asian, Italian	3CS Mall,Lajpat Nagar - 3, South Delhi	3.9	https://im1.dineout.co.in/images/upload
2	Castle Barbeque	Chinese, North Indian	Pacific Mall,Tagore Garden, West Delhi	3.9	https://im1.dineout.co.in/images/upload
3	Cafe Knosh	Italian, Continental	The Leela Ambience Convention Hotel,Shahdara, ...	4.3	https://im1.dineout.co.in/images/upload
4	The Barbeque Company	North Indian, Chinese4 deals available	Gardens Galleria,Sector 38A, Noida	4	https://im1.dineout.co.in/images/upload
5	India Grill	North Indian, Italian	Hilton Garden Inn,Saket, South Delhi	3.9	https://im1.dineout.co.in/images/upload
6	Delhi Barbeque	North Indian	Taurus Sarovar Portico,Mahipalpur, South Delhi	3.7	https://im1.dineout.co.in/images/upload
7	The Monarch - Bar Be Que Village	North Indian	Indirapuram Habitat Centre,Indirapuram, Ghaziabad	3.8	https://im1.dineout.co.in/images/upload
8	Indian Grill Room	North Indian, Mughlai	Suncity Business Tower,Golf Course Road, Gurgaon	4.3	https://im1.dineout.co.in/images/upload

10) Write a python program to scrape the details of top publications from Google Scholar from https://scholar.google.com/citations?view_op=top_venues&hl=en (https://scholar.google.com/citations?view_op=top_venues&hl=en) i) Rank ii) Publication iii) h5-index iv)

In [7]:

```
page13=requests.get('https://scholar.google.com/citations?view_op=top_venues&hl=en')
page13
```

Out[7]:

```
<Response [200]>
```

In [8]:

```
soup13=BeautifulSoup(page13.content,"html.parser")
print(soup13.prettify())
```

```
<!DOCTYPE html>
<html>
 <head>
  <title>
   English - Google Scholar Metrics
  </title>
  <meta content="text/html;charset=utf-8" http-equiv="Content-Type"/>
  <meta content="IE=Edge" http-equiv="X-UA-Compatible"/>
  <meta content="always" name="referrer"/>
  <meta content="width=device-width,initial-scale=1,minimum-scale=1,maximu
m-scale=2" name="viewport"/>
  <meta content="telephone=no" name="format-detection"/>
  <link href="/favicon.ico" rel="shortcut icon"/>
  <style>
   html,body,form,table,div,h1,h2,h3,h4,h5,h6,img,ol,ul,li,button{margin:
0;padding:0;border:0;}table{border-collapse:collapse;border-width:0;empty-
cells:show;}html,body{height:100%}#gs_top{position:relative;box-sizing:bor
der-box;min-height:100%;min-width:964px;-webkit-tap-highlight-color:rgba
(0,0,0,0);}#gs_top>*:not(#x){-webkit-tap-highlight-color:rgba(204,204,20
```

In [9]:

```
rank=soup13.find_all('td',class_='gsc_mvt_p')
rank
```

Out[9]:

```
[<td class="gsc_mvt_p">1.</td>,
 <td class="gsc_mvt_p">2.</td>,
 <td class="gsc_mvt_p">3.</td>,
 <td class="gsc_mvt_p">4.</td>,
 <td class="gsc_mvt_p">5.</td>,
 <td class="gsc_mvt_p">6.</td>,
 <td class="gsc_mvt_p">7.</td>,
 <td class="gsc_mvt_p">8.</td>,
 <td class="gsc_mvt_p">9.</td>,
 <td class="gsc_mvt_p">10.</td>,
 <td class="gsc_mvt_p">11.</td>,
 <td class="gsc_mvt_p">12.</td>,
 <td class="gsc_mvt_p">13.</td>,
 <td class="gsc_mvt_p">14.</td>,
 <td class="gsc_mvt_p">15.</td>,
 <td class="gsc_mvt_p">16.</td>,
 <td class="gsc_mvt_p">17.</td>,
 <td class="gsc_mvt_p">18.</td>,
```

In [27]:

```
Rank=[]
for r in rank:
    r=r.text.strip('.')
    Rank.append(r)
```

In [28]:

```
Rank
```

Out[28]:

```
['1',
 '2',
 '3',
 '4',
 '5',
 '6',
 '7',
 '8',
 '9',
 '10',
 '11',
 '12',
 '13',
 '14',
 '15',
 '16',
 '17',
 '18'.
```

In [14]:

```
Publish=soup13.find_all('td',class_='gsc_mvt_t')
Publish
```

Out[14]:

```
[<td class="gsc_mvt_t">Nature</td>,
 <td class="gsc_mvt_t">The New England Journal of Medicine</td>,
 <td class="gsc_mvt_t">Science</td>,
 <td class="gsc_mvt_t">IEEE/CVF Conference on Computer Vision and Pattern
Recognition</td>,
 <td class="gsc_mvt_t">The Lancet</td>,
 <td class="gsc_mvt_t">Advanced Materials</td>,
 <td class="gsc_mvt_t">Nature Communications</td>,
 <td class="gsc_mvt_t">Cell</td>,
 <td class="gsc_mvt_t">International Conference on Learning Representation
s</td>,
 <td class="gsc_mvt_t">Neural Information Processing Systems</td>,
 <td class="gsc_mvt_t">JAMA</td>,
 <td class="gsc_mvt_t">Chemical Reviews</td>,
 <td class="gsc_mvt_t">Proceedings of the National Academy of Sciences</td
>,
 <td class="gsc_mvt_t">Angewandte Chemie</td>,
 <td class="gsc_mvt_t">Chemical Society Reviews</td>,
```

In [16]:

```
Publication=[]
for u in Publish:
    u=u.text
    Publication.append(u)
```

In [17]:

```
Publication
```

Out[17]:

```
['Nature',
 'The New England Journal of Medicine',
 'Science',
 'IEEE/CVF Conference on Computer Vision and Pattern Recognition',
 'The Lancet',
 'Advanced Materials',
 'Nature Communications',
 'Cell',
 'International Conference on Learning Representations',
 'Neural Information Processing Systems',
 'JAMA',
 'Chemical Reviews',
 'Proceedings of the National Academy of Sciences',
 'Angewandte Chemie',
 'Chemical Society Reviews',
 'Journal of the American Chemical Society',
 'IEEE/CVF International Conference on Computer Vision',
 'Nucleic Acids Research',
```

In [18]:

```
h5=soup13.find_all('a',class_='gs_ibl gsc_mp_anchor')
h5
```

Out[18]:

```
[<a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=H--JoiVp8x8J.2022">444</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=NR7xTEnpNgsJ.2022">432</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=oY2eER5-jTUJ.2022">401</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=FXe-a9w0eycJ.2022">389</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=dj7TIF9zE7gJ.2022">354</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=iGi98NXoUDsJ.2022">312</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=m_DUlvLcyY0J.2022">307</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=VlyKmNJzCgkJ.2022">300</a>,
 <a class="gs_ibl gsc_mp_anchor" href="/citations?hl=en&amp;oe=ASCII&amp;v
q=en&amp;view_op=list_hcore&amp;venue=0032SoU2xY4J.2022">286</a>,
```

In [19]:

```
h5_index=[]
for h in h5:
    h=h.text
    h5_index.append(h)
```

In [20]:

```
h5_index
```

Out[20]:

```
['444',
 '432',
 '401',
 '389',
 '354',
 '312',
 '307',
 '300',
 '286',
 '278',
 '267',
 '265',
 '256',
 '245',
 '244',
 '242',
 '239',
 '238',
```

In [21]:

```
H5=soup13.find_all('span',class_='gs_ibl gsc_mp_anchor')
H5
```

Out[21]:

```
[<span class="gs_ibl gsc_mp_anchor">667</span>,
 <span class="gs_ibl gsc_mp_anchor">780</span>,
 <span class="gs_ibl gsc_mp_anchor">614</span>,
 <span class="gs_ibl gsc_mp_anchor">627</span>,
 <span class="gs_ibl gsc_mp_anchor">635</span>,
 <span class="gs_ibl gsc_mp_anchor">418</span>,
 <span class="gs_ibl gsc_mp_anchor">428</span>,
 <span class="gs_ibl gsc_mp_anchor">505</span>,
 <span class="gs_ibl gsc_mp_anchor">533</span>,
 <span class="gs_ibl gsc_mp_anchor">436</span>,
 <span class="gs_ibl gsc_mp_anchor">425</span>,
 <span class="gs_ibl gsc_mp_anchor">444</span>,
 <span class="gs_ibl gsc_mp_anchor">364</span>,
 <span class="gs_ibl gsc_mp_anchor">332</span>,
 <span class="gs_ibl gsc_mp_anchor">386</span>,
 <span class="gs_ibl gsc_mp_anchor">344</span>,
 <span class="gs_ibl gsc_mp_anchor">415</span>,
 <span class="gs_ibl gsc_mp_anchor">550</span>,
```

In [22]:

```
h5_median=[]
for t in H5:
    t=t.text
    h5_median.append(t)
```

In [23]:

```
h5_median
```

Out[23]:

```
['667',
 '780',
 '614',
 '627',
 '635',
 '418',
 '428',
 '505',
 '533',
 '436',
 '425',
 '444',
 '364',
 '332',
 '386',
 '344',
 '415',
 '550',
```

In [29]:

```
Google_Scholar=pd.DataFrame({'Rank':Rank,'Publication':Publication,'h5-index':h5_index,'h5-
Google_Scholar
```

Out[29]:

	Rank	Publication	h5-index	h5-median
0	1	Nature	444	667
1	2	The New England Journal of Medicine	432	780
2	3	Science	401	614
3	4	IEEE/CVF Conference on Computer Vision and Pat...	389	627
4	5	The Lancet	354	635
...	...	...	...	...
95	96	Journal of Business Research	145	233
96	97	Molecular Cancer	145	209
97	98	Sensors	145	201
98	99	Nature Climate Change	144	228
99	100	IEEE Internet of Things Journal	144	212

100 rows × 4 columns

In [ ]: