In [55]:

```
from bs4 import BeautifulSoup
import requests,openpyxl
```

In [3]:

```
source=requests.get("https://www.imdb.com/chart/top/?sort=rk,asc&mode=simple&page=1")
```

In [4]:

```
soup=BeautifulSoup(source.text,"html.parser")
```

In [5]:

```
soup
```

```
ef="/feature/genre/?ref_=nv_ch_gr" role="menuitem" tabindex="-1"><span
class="ipc-list-item__text" role="presentation">Browse Movies by Genre
</span></a><a aria-disabled="false" class="ipc-list__item nav-link sc-e
vZas nMWbL ipc-list__item--indent-one" href="/chart/boxoffice/?ref_=nv_
ch_cht" role="menuitem" tabindex="-1"><span class="ipc-list-item__text"
role="presentation">Top Box Office</span></a><a aria-disabled="false" c
lass="ipc-list__item nav-link sc-evZas nMWbL ipc-list__item--indent-on
e" href="/showtimes/?ref_=nv_mv_sh" role="menuitem" tabindex="-1"><span
class="ipc-list-item__text" role="presentation">Showtimes &amp;amp; Tickets
</span></a><a aria-disabled="false" class="ipc-list__item nav-link sc-e
vZas nMWbL ipc-list__item--indent-one" href="/news/movie/?ref_=nv_nw_m
v" role="menuitem" tabindex="-1"><span class="ipc-list-item__text" role
="presentation">Movie News</span></a><a aria-disabled="false" class="ip
c-list__item nav-link sc-evZas nMWbL ipc-list__item--indent-one" href
="/india/toprated/?ref_=nv_mv_in" role="menuitem" tabindex="-1"><span c
lass="ipc-list-item__text" role="presentation">India Movie Spotlight</s
pan></a></ul></div></div></span></div><div class="sc-ksZaOG gSOLIo" dat
a-testid="grouped-link-category"><div class="sc-breuTD gIvE navlinkcat
sc-fnykZs SyRwP" data-testid="nav-link-category" role="presentation"><i
```

In [57]:

```
movies=soup.find("tbody",class_="lister-list").find_all("tr")

for movie in movies:
    name=movie.find("td",class_="titleColumn").a.text
    rn=movie.find("td",class_="titleColumn").get_text(strip=True).split(".")[0]
    rate=movie.find("td",class_="ratingColumn").strong.text
    yr=movie.find("td",class_="titleColumn").span.text.strip("()")
    print(rn,name,yr,rate)
```

```
43 Casablanca 1942 8.5
44 Seppuku 1962 8.5
45 Hotaru no haka 1988 8.5
46 The Intouchables 2011 8.5
47 Modern Times 1936 8.4
48 Once Upon a Time in the West 1968 8.4
49 Rear Window 1954 8.4
50 Nuovo Cinema Paradiso 1988 8.4
51 Alien 1979 8.4
52 City Lights 1931 8.4
53 Apocalypse Now 1979 8.4
54 Memento 2000 8.4
55 Django Unchained 2012 8.4
56 Raiders of the Lost Ark 1981 8.4
57 WALL·E 2008 8.4
58 The Lives of Others 2006 8.4
59 Sunset Blvd. 1950 8.4
60 Paths of Glory 1957 8.4
61 The Shining 1980 8.4
62 The Great Dictator 1940 8.4
```

In [45]:

```
name=[]
for movie in movies:
    name.append(movie.find("td",class_="titleColumn").a.text)
print(name)
```

['The Shawshank Redemption', 'The Godfather', 'The Dark Knight', 'The Godf
ather Part II', '12 Angry Men', "Schindler's List", 'The Lord of the Ring
s: The Return of the King', 'Pulp Fiction', 'The Lord of the Rings: The Fe
llowship of the Ring', 'Il buono, il brutto, il cattivo', 'Forrest Gump',
'Fight Club', 'The Lord of the Rings: The Two Towers', 'Inception', 'The E
mpire Strikes Back', 'The Matrix', 'GoodFellas', "One Flew Over the Cucko
o's Nest", 'Se7en', 'Shichinin no samurai', "It's a Wonderful Life", 'The
Silence of the Lambs', 'Saving Private Ryan', 'Cidade de Deus', 'Interstel
lar', 'La vita è bella', 'The Green Mile', 'Star Wars', 'Terminator 2: Jud
gment Day', 'Back to the Future', 'Sen to Chihiro no kamikakushi', 'The Pi
anist', 'Psycho', 'Gisaengchung', 'Léon', 'The Lion King', 'Gladiator', 'A
merican History X', 'The Departed', 'The Usual Suspects', 'The Prestige',
'Whiplash', 'Casablanca', 'Seppuku', 'Hotaru no haka', 'The Intouchables',
'Modern Times', 'Once Upon a Time in the West', 'Rear Window', 'Nuovo Cine
ma Paradiso', 'Alien', 'City Lights', 'Apocalypse Now', 'Memento', 'Django
Unchained', 'Raiders of the Lost Ark', 'WALL·E', 'The Lives of Others', 'S
unset Blvd.', 'Paths of Glory', 'The Shining', 'The Great Dictator', 'Aven
gers: Infinity War', 'Witness for the Prosecution', 'Aliens', 'Spider-Man:
Into the Spider-Verse', 'American Beauty', 'Dr. Strangelove or: How I Lear
ned to Stop Worrying and Love the Bomb', 'The Dark Knight Rises', 'Oldeubo
i', 'Inglourious Basterds', 'Amadeus', 'Coco', 'Toy Story', 'Joker', 'Brav
eheart', 'Das Boot', 'Avengers: Endgame', 'Mononoke-hime', 'Once Upon a Ti
me in America', 'Good Will Hunting', 'Kimi no na wa.', "Singin' in the Rai
n", '3 Idiots', 'Requiem for a Dream', 'Tengoku to jigoku', 'Toy Story 3',
'Capharnaüm', 'Star Wars: Episode VI - Return of the Jedi', 'Eternal Sunsh
ine of the Spotless Mind', '2001: A Space Odyssey', 'Reservoir Dogs', 'Idi
i smotri', 'Jagten', 'Citizen Kane', 'M - Eine Stadt sucht einen Mörder',
'Lawrence of Arabia', 'North by Northwest', 'Ikiru', 'Vertigo', 'The Apart
ment', "Le fabuleux destin d'Amélie Poulain", 'A Clockwork Orange', 'Doubl
e Indemnity', 'Full Metal Jacket', 'Scarface', 'Hamilton', 'Incendies', 'T
op Gun: Maverick', 'To Kill a Mockingbird', 'Heat', 'The Sting', 'Up', 'Jo
daeiye Nader az Simin', 'Metropolis', 'Taxi Driver', 'L.A. Confidential',
'Die Hard', 'Snatch', 'Indiana Jones and the Last Crusade', 'Ladri di bici
clette', 'Taare Zameen Par', '1917', 'Der Untergang', 'Dangal', 'Per qualc
he dollaro in più', 'Batman Begins', 'The Kid', 'Some Like It Hot', 'The F
ather', 'All About Eve', 'The Wolf of Wall Street', 'Green Book', 'Judgmen
t at Nuremberg', 'Ran', 'Casino', 'The Truman Show', "Pan's Labyrinth", 'T
here Will Be Blood', 'Unforgiven', 'The Sixth Sense', 'Shutter Island', 'A
Beautiful Mind', 'Jurassic Park', 'Yôjinbô', 'The Treasure of the Sierra M
adre', 'Monty Python and the Holy Grail', 'The Great Escape', 'No Country
for Old Men', 'Kill Bill: Vol. 1', 'Rashômon', 'Spider-Man: No Way Home',
'The Thing', 'Finding Nemo', 'The Elephant Man', 'Chinatown', 'Raging Bul
l', 'V for Vendetta', 'Gone with the Wind', 'Lock, Stock and Two Smoking B
arrels', 'Inside Out', 'Dial M for Murder', 'El secreto de sus ojos', 'Hau
ru no ugoku shiro', 'Three Billboards Outside Ebbing, Missouri', 'The Brid
ge on the River Kwai', 'Trainspotting', 'Prisoners', 'Warrior', 'Fargo',
'Gran Torino', 'Tonari no Totoro', 'Catch Me If You Can', 'Million Dollar
Baby', 'Bacheha-Ye aseman', 'Blade Runner', 'The Gold Rush', 'Before Sunri
se', 'Klaus', '12 Years a Slave', 'Harry Potter and the Deathly Hallows: P
art 2', 'On the Waterfront', 'Ben-Hur', 'The Grand Budapest Hotel', 'Gone
Girl', 'Smultronstället', 'The General', 'The Third Man', 'In the Name of
the Father', 'Barry Lyndon', 'The Deer Hunter', 'Hacksaw Ridge', 'Le salai
re de la peur', 'Salinui chueok', 'Sherlock Jr.', 'Relatos salvajes', 'Mr.
Smith Goes to Washington', 'Mad Max: Fury Road', 'Det sjunde inseglet', 'M
ary and Max.', 'How to Train Your Dragon', 'Room', 'Monsters, Inc.', 'Jaw
s', 'Dead Poets Society', 'The Big Lebowski', 'Tôkyô monogatari', "La pass
ion de Jeanne d'Arc", 'Ford v Ferrari', 'Hotel Rwanda', 'Rocky', 'Platoo
n', 'Ratatouille', 'Spotlight', 'The Terminator', 'Logan', 'Stand by Me',
'Rush', 'Network', 'Before Sunset', 'Into the Wild', 'The Wizard of Oz',
'Pather Panchali', 'Groundhog Day', 'The Best Years of Our Lives', 'The Ex
orcist', 'To Be or Not to Be', 'The Incredibles', 'La haine', 'Pirates of

```
the Caribbean: The Curse of the Black Pearl', 'La battaglia di Algeri', 'J
ai Bhim', "Hachi: A Dog's Tale", 'The Grapes of Wrath', 'Babam ve Oglum',
'Amores perros', 'Rebecca', 'Cool Hand Luke', 'Ah-ga-ssi', 'Les quatre cen
ts coups', 'The Sound of Music', 'It Happened One Night', 'Persona', 'Life
of Brian', 'The Iron Giant', 'Dersu Uzala', 'The Help', 'Aladdin', 'Gandh
i', 'Dances with Wolves']
```

In [46]:

```
rating_=[]
ratings=soup.find("tbody",class_="lister-list").find_all("tr")
for rating in ratings:
    rating_.append(rating.find("td",class_="ratingColumn").strong.text)

print(rating_)
```

```
['9.2', '9.2', '9.0', '9.0', '9.0', '8.9', '8.9', '8.8', '8.8', '8.8', '8.
8', '8.7', '8.7', '8.7', '8.7', '8.7', '8.7', '8.6', '8.6', '8.6', '8.6',
'8.6', '8.6', '8.6', '8.6', '8.6', '8.6', '8.5', '8.5', '8.5', '8.5', '8.
5', '8.5', '8.5', '8.5', '8.5', '8.5', '8.5', '8.5', '8.5', '8.5', '8.5',
'8.5', '8.5', '8.5', '8.5', '8.4', '8.4', '8.4', '8.4', '8.4', '8.4', '8.
4', '8.4', '8.4', '8.4', '8.4', '8.4', '8.4', '8.4', '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.
3', '8.3', '8.3', '8.3', '8.3', '8.3', '8.3', '8.3', '8.3', '8.3', '8.3',
'8.3', '8.3', '8.3', '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', '8.2', '8.2', '8.2', '8.2',
'8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.
2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2',
'8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.
2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.2', '8.1', '8.1', '8.1', '8.1',
'8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.
1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1',
'8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.
1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1',
'8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.
1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.1', '8.0', '8.0',
'8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.
0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0',
'8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.
0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0', '8.0']
```

In [ ]:

In [47]:

```
years_=[]
years=soup.find("tbody",class_="lister-list").find_all("tr")
for year in years:
    years_.append(year.find("td",class_="titleColumn").span.text.strip("()"))
print(years_)
```

```
['1994', '1972', '2008', '1974', '1957', '1993', '2003', '1994', '2001',
'1966', '1994', '1999', '2002', '2010', '1980', '1999', '1990', '1975', '1
995', '1954', '1946', '1991', '1998', '2002', '2014', '1997', '1999', '197
7', '1991', '1985', '2001', '2002', '1960', '2019', '1994', '1994', '200
0', '1998', '2006', '1995', '2006', '2014', '1942', '1962', '1988', '201
1', '1936', '1968', '1954', '1988', '1979', '1931', '1979', '2000', '201
2', '1981', '2008', '2006', '1950', '1957', '1980', '1940', '2018', '195
7', '1986', '2018', '1999', '1964', '2012', '2003', '2009', '1984', '201
7', '1995', '2019', '1995', '1981', '2019', '1997', '1984', '1997', '201
6', '1952', '2009', '2000', '1963', '2010', '2018', '1983', '2004', '196
8', '1992', '1985', '2012', '1941', '1931', '1962', '1959', '1952', '195
8', '1960', '2001', '1971', '1944', '1987', '1983', '2020', '2010', '202
2', '1962', '1995', '1973', '2009', '2011', '1927', '1976', '1997', '198
8', '2000', '1989', '1948', '2007', '2019', '2004', '2016', '1965', '200
5', '1921', '1959', '2020', '1950', '2013', '2018', '1961', '1985', '199
5', '1998', '2006', '2007', '1992', '1999', '2010', '2001', '1993', '196
1', '1948', '1975', '1963', '2007', '2003', '1950', '2021', '1982', '200
3', '1980', '1974', '1980', '2005', '1939', '1998', '2015', '1954', '200
9', '2004', '2017', '1957', '1996', '2013', '2011', '1996', '2008', '198
8', '2002', '2004', '1997', '1982', '1925', '1995', '2019', '2013', '201
1', '1954', '1959', '2014', '2014', '1957', '1926', '1949', '1993', '197
5', '1978', '2016', '1953', '2003', '1924', '2014', '1939', '2015', '195
7', '2009', '2010', '2015', '2001', '1975', '1989', '1998', '1953', '192
8', '2019', '2004', '1976', '1986', '2007', '2015', '1984', '2017', '198
6', '2013', '1976', '2004', '2007', '1939', '1955', '1993', '1946', '197
3', '1942', '2004', '1995', '2003', '1966', '2021', '2009', '1940', '200
5', '2000', '1940', '1967', '2016', '1959', '1965', '1934', '1966', '197
9', '1999', '1975', '2011', '1992', '1982', '1990']
```

In [48]:

```
ranks_=[]

ranks=soup.find("tbody",class_="lister-list").find_all("tr")

for rank in ranks:
    ranks_.append(rank.find("td",class_="titleColumn").get_text(strip=True).split(".")[0
print(ranks_)
```

```
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '1
4', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '2
6', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '3
8', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '5
0', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '6
2', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73', '7
4', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '8
6', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96', '97', '9
8', '99', '100', '101', '102', '103', '104', '105', '106', '107', '108',
'109', '110', '111', '112', '113', '114', '115', '116', '117', '118', '11
9', '120', '121', '122', '123', '124', '125', '126', '127', '128', '129',
'130', '131', '132', '133', '134', '135', '136', '137', '138', '139', '14
0', '141', '142', '143', '144', '145', '146', '147', '148', '149', '150',
'151', '152', '153', '154', '155', '156', '157', '158', '159', '160', '16
1', '162', '163', '164', '165', '166', '167', '168', '169', '170', '171',
'172', '173', '174', '175', '176', '177', '178', '179', '180', '181', '18
2', '183', '184', '185', '186', '187', '188', '189', '190', '191', '192',
'193', '194', '195', '196', '197', '198', '199', '200', '201', '202', '20
3', '204', '205', '206', '207', '208', '209', '210', '211', '212', '213',
'214', '215', '216', '217', '218', '219', '220', '221', '222', '223', '22
4', '225', '226', '227', '228', '229', '230', '231', '232', '233', '234',
'235', '236', '237', '238', '239', '240', '241', '242', '243', '244', '24
5', '246', '247', '248', '249', '250']
```

In [49]:

```
import pandas as pd
df=pd.DataFrame({"Rank":ranks_,"Movie_name":name,"year":years_,"ratings":rating_})
```

In [50]:

```
df
```

Out[50]:

	Rank	Movie_name	year	ratings
0	1	The Shawshank Redemption	1994	9.2
1	2	The Godfather	1972	9.2
2	3	The Dark Knight	2008	9.0
3	4	The Godfather Part II	1974	9.0
4	5	12 Angry Men	1957	9.0
...	...	...	...	...
245	246	Dersu Uzala	1975	8.0
246	247	The Help	2011	8.0
247	248	Aladdin	1992	8.0
248	249	Gandhi	1982	8.0
249	250	Dances with Wolves	1990	8.0

250 rows × 4 columns

In [56]:

```
df.to_excel("IMDB_Movies.xlsx",sheet_name='sheet1', index=False)
```

In [66]:

```
#number of movies each year
a=df["year"].value_counts()
a
```

Out[66]:

```
1995    8
2004    7
2009    6
1957    6
2003    6
       ..
1941    1
2022    1
1958    1
1987    1
1934    1
Name: year, Length: 86, dtype: int64
```

In [69]:

```
#number of movies for rating
b=df["ratings"].value_counts()
b
```

Out[69]:

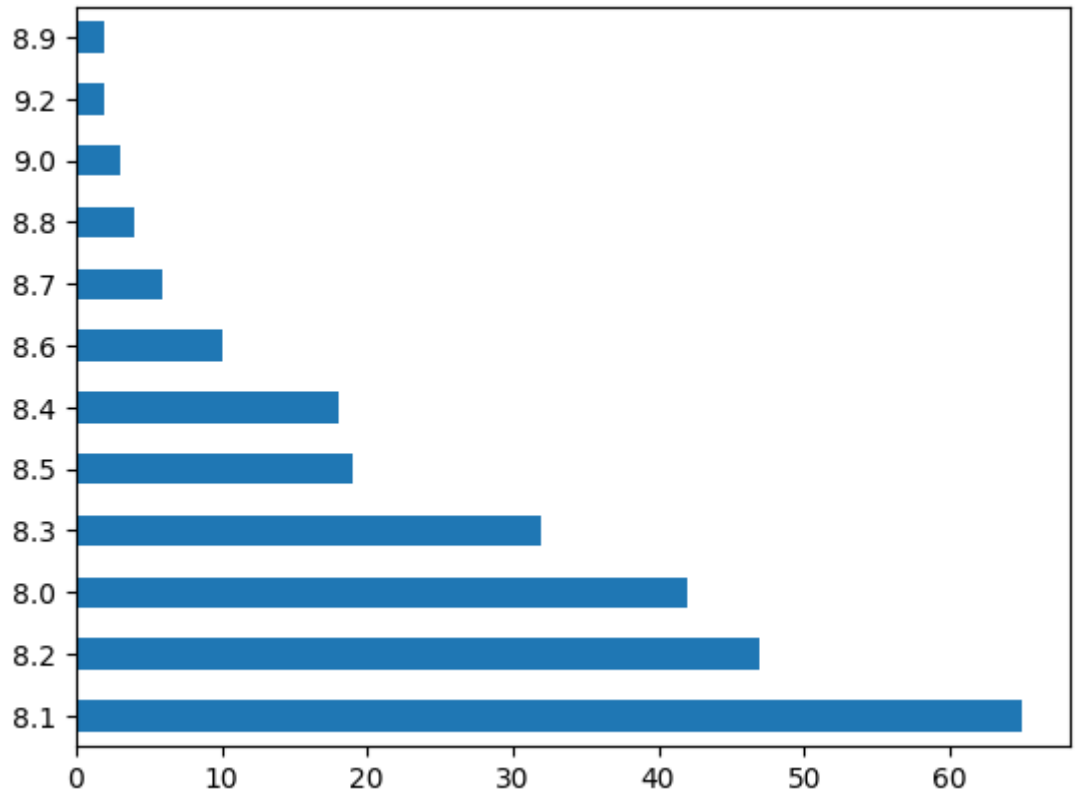
```
8.1    65
8.2    47
8.0    42
8.3    32
8.5    19
8.4    18
8.6    10
8.7     6
8.8     4
9.0     3
9.2     2
8.9     2
Name: ratings, dtype: int64
```

In [70]:

```
b.plot.barh()
```

Out[70]:

```
<AxesSubplot: >
```



In [77]:

```
#rating grater than 8.5
df["ratings"].astype(float)#converting to float
a=df[df["ratings"].astype(float)>8.5]
a
```

Out[77]:

	Rank	Movie_name	year	ratings
0	1	The Shawshank Redemption	1994	9.2
1	2	The Godfather	1972	9.2
2	3	The Dark Knight	2008	9.0
3	4	The Godfather Part II	1974	9.0
4	5	12 Angry Men	1957	9.0
5	6	Schindler's List	1993	8.9
6	7	The Lord of the Rings: The Return of the King	2003	8.9
7	8	Pulp Fiction	1994	8.8
8	9	The Lord of the Rings: The Fellowship of the Ring	2001	8.8
9	10	Il buono, il brutto, il cattivo	1966	8.8
10	11	Forrest Gump	1994	8.8
11	12	Fight Club	1999	8.7
12	13	The Lord of the Rings: The Two Towers	2002	8.7
13	14	Inception	2010	8.7
14	15	The Empire Strikes Back	1980	8.7
15	16	The Matrix	1999	8.7
16	17	GoodFellas	1990	8.7
17	18	One Flew Over the Cuckoo's Nest	1975	8.6
18	19	Se7en	1995	8.6
19	20	Shichinin no samurai	1954	8.6
20	21	It's a Wonderful Life	1946	8.6
21	22	The Silence of the Lambs	1991	8.6
22	23	Saving Private Ryan	1998	8.6
23	24	Cidade de Deus	2002	8.6
24	25	Interstellar	2014	8.6
25	26	La vita è bella	1997	8.6
26	27	The Green Mile	1999	8.6

In [88]:

```
#the year where number of movies having rating greater than 8.5 for a particuar year
b=a["year"].value_counts()
b[b>=2]
```

Out[88]:

```
1994    3
1999    3
2002    2
Name: year, dtype: int64
```

In [ ]:

```

```