Netflix data analysis with python

Import Necessary libraries

In [41]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats
from math import sqrt
import statsmodels.api as sm
import os
import math
from datetime import datetime
from datetime import timedelta
import plotly.express as px
import ast
import random
```

This Python 3 environment comes with numerous supportive analytics libraries installed

It is characterized by the kaggle/python Docker picture: https://github.com/kaggle/docker-python

For illustration, here's a few supportive bundles to load

direct algebra # information preparing, CSV record I/O (e.g. pd.read_csv)

In [42]:
```
import numpy as np
import pandas as pd
```

Input data files ar out there within the read-only "../input/" directory

for instance, running this (by clicking run or pressing Shift+Enter) can list all files underneath the input directory

In [43]:
```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [ ]:
```
# upload dataset
```

In [45]:
```
df=pd.read_csv('credits.csv')
df=pd.read_csv('titles.csv')
```

explore the data

In [46]:
```
df.head()
```

Out[46]:

	id	title	type	description	release_year	age_certification	runtime	genre
0	ts300399	Five Came Back: The Reference Films	SHOW	This collection includes 12 World War II-era p...	1945	TV-MA	48	['documentation
1	tm84618	Taxi Driver	MOVIE	A mentally unstable Vietnam War veteran works ...	1976	R	113	['crime', 'drama
2	tm127384	Monty Python and the Holy Grail	MOVIE	King Arthur, accompanied by his squire, recrui...	1975	PG	91	['comedy 'fantasy
3	tm70993	Life of Brian	MOVIE	Brian Cohen is an average young Jewish man, bu...	1979	R	94	['comedy
4	tm190788	The Exorcist	MOVIE	12-year-old Regan MacNeil begins to adapt an e...	1973	R	133	['horror

In [47]:
```
df.columns
```

Out[47]:
```
Index(['id', 'title', 'type', 'description', 'release_year',
       'age_certification', 'runtime', 'genres', 'production_countries',
       'seasons', 'imdb_id', 'imdb_score', 'imdb_votes', 'tmdb_popularity',
       'tmdb_score'],
      dtype='object')
```

Unload and repair the production_countries and sort column values which are right now arrays

In [48]:
```
def repair_array_bound_categories(arr):
    arr = ast.literal_eval(arr)

    if len(arr) == 0:
        return np.nan

    elif len(arr) == 1:
        return arr[0]

    else:
        return random.choice(arr)
```

In [49]:
```
df["production_countries"] = df["production_countries"].apply(repair_array_bound_categ
df["genres"] = df["genres"].apply(repair_array_bound_categories)
```

In [50]:
```
df.head()
```

Out[50]:

	id	title	type	description	release_year	age_certification	runtime	genres
0	ts300399	Five Came Back: The Reference Films	SHOW	This collection includes 12 World War II-era p...	1945	TV-MA	48	documentation
1	tm84618	Taxi Driver	MOVIE	A mentally unstable Vietnam War veteran works ...	1976	R	113	crime
2	tm127384	Monty Python and the Holy Grail	MOVIE	King Arthur, accompanied by his squire, recrui...	1975	PG	91	comedy
3	tm70993	Life of Brian	MOVIE	Brian Cohen is an average young Jewish man, bu...	1979	R	94	comedy
4	tm190788	The Exorcist	MOVIE	12-year-old Regan MacNeil begins to adapt an e...	1973	R	133	horror

In [51]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5806 entries, 0 to 5805
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   id                    5806 non-null   object 
 1   title                 5805 non-null   object 
 2   type                  5806 non-null   object 
 3   description           5788 non-null   object 
 4   release_year          5806 non-null   int64  
 5   age_certification     3196 non-null   object 
 6   runtime               5806 non-null   int64  
 7   genres                5738 non-null   object 
 8   production_countries  5574 non-null   object 
 9   seasons               2047 non-null   float64
 10  imdb_id               5362 non-null   object 
 11  imdb_score            5283 non-null   float64
 12  imdb_votes            5267 non-null   float64
 13  tmdb_popularity       5712 non-null   float64
 14  tmdb_score            5488 non-null   float64
dtypes: float64(5), int64(2), object(8)
memory usage: 680.5+ KB
```

let's check the duplicated

In [52]:
```
df.duplicated().sum()
```

Out[52]:
```
0
```

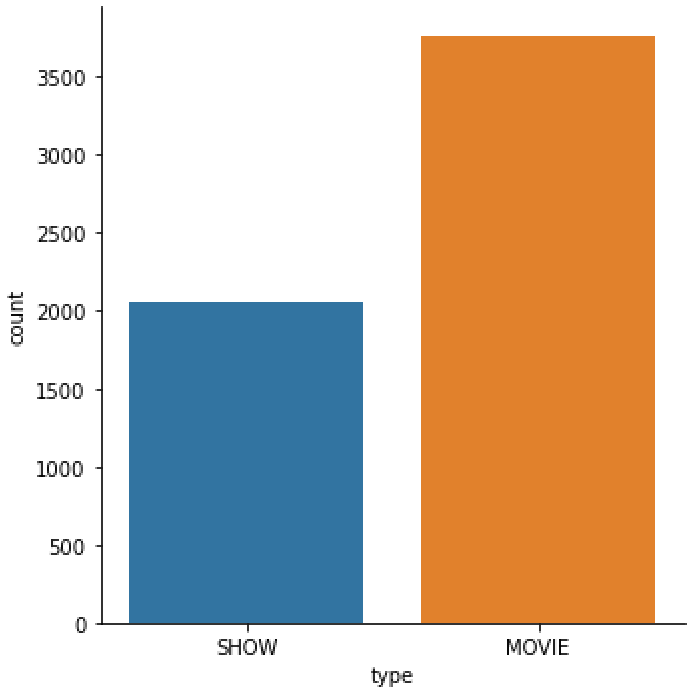
In [53]:
```
df.isnull().sum()
```

Out[53]:
```
id                         0
title                      1
type                       0
description               18
release_year               0
age_certification       2610
runtime                    0
genres                    68
production_countries     232
seasons                 3759
imdb_id                  444
imdb_score               523
imdb_votes               539
tmdb_popularity           94
tmdb_score               318
dtype: int64
```

let's descover the sort in case it show up or movise

In [54]:
```
sns.catplot(x='type',kind='count',data=df)
```

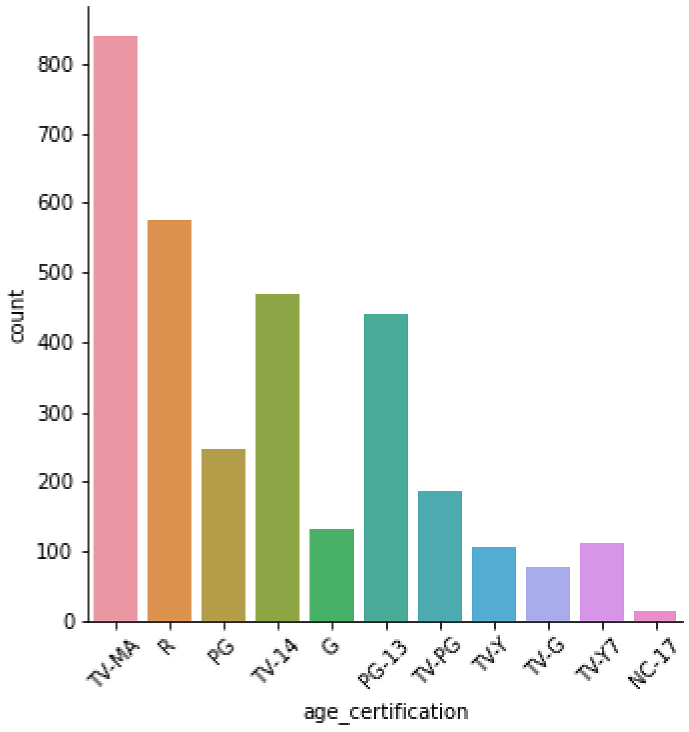
Out[54]:
```
<seaborn.axisgrid.FacetGrid at 0x1609c87e5e0>
```



most of the type is movie

In [55]:
```
sns.catplot(x='age_certification',kind="count",data=df)
plt.xticks(rotation=45)
```

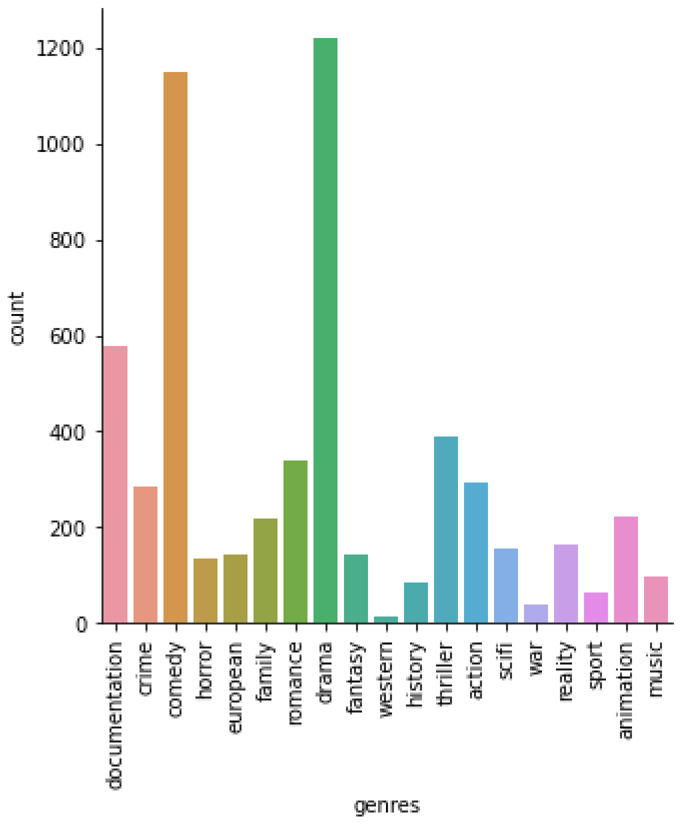
Out[55]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10]),
 [Text(0, 0, 'TV-MA'),
  Text(1, 0, 'R'),
  Text(2, 0, 'PG'),
  Text(3, 0, 'TV-14'),
  Text(4, 0, 'G'),
  Text(5, 0, 'PG-13'),
  Text(6, 0, 'TV-PG'),
  Text(7, 0, 'TV-Y'),
  Text(8, 0, 'TV-G'),
  Text(9, 0, 'TV-Y7'),
  Text(10, 0, 'NC-17')])
```



the preeminent of age certification is TV_MA TV_MA:esigned to be seen by grown-ups and in this way may be unsatisfactory for children underneath 17 and let's see the preeminent genders

```
In [56]: sns.catplot(x='genres', kind="count", data=df)
         plt.xticks(rotation=90)
```

```
Out[56]: (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
                 17, 18]),
          [Text(0, 0, 'documentation'),
           Text(1, 0, 'crime'),
           Text(2, 0, 'comedy'),
           Text(3, 0, 'horror'),
           Text(4, 0, 'european'),
           Text(5, 0, 'family'),
           Text(6, 0, 'romance'),
           Text(7, 0, 'drama'),
           Text(8, 0, 'fantasy'),
           Text(9, 0, 'western'),
           Text(10, 0, 'history'),
           Text(11, 0, 'thriller'),
           Text(12, 0, 'action'),
           Text(13, 0, 'scifi'),
           Text(14, 0, 'war'),
           Text(15, 0, 'reality'),
           Text(16, 0, 'sport'),
           Text(17, 0, 'animation'),
           Text(18, 0, 'music')])
```



appear is the preeminent sexual orientation we need to know the preeminent era countries so let's check it

```
In [57]: shows_countries= df.production_countries.value_counts()
         shows_countries = pd.DataFrame(shows_countries)
```

we got to see the first era nations so i separated the preeminent 15 countries

```
In [58]: shows_countries= df.production_countries.value_counts()
         shows_countries = pd.DataFrame(shows_countries)
```

```
In [59]: shows_countries= shows_countries.head(15)
         shows_countries
```

Out[59]:

	production_countries
US	2111
IN	617
GB	290
JP	277
KR	213
ES	183
FR	171
CA	149
MX	110
DE	91
BR	89
CN	84
PH	82
TR	79
NG	73

In [60]:

```
labels = ['US','IN','JP','GB','KR','ES','FR','CA','MX','BR','PH','TR','NG','DE','AU']
values = [1950, 605, 266, 219,210,159,124,103,95,86,80,79,67,65,62]
```

In [28]:

```
release_year_count= df.release_year.value_counts()

release_year_count = pd.DataFrame(release_year_count)

release_year_count
```

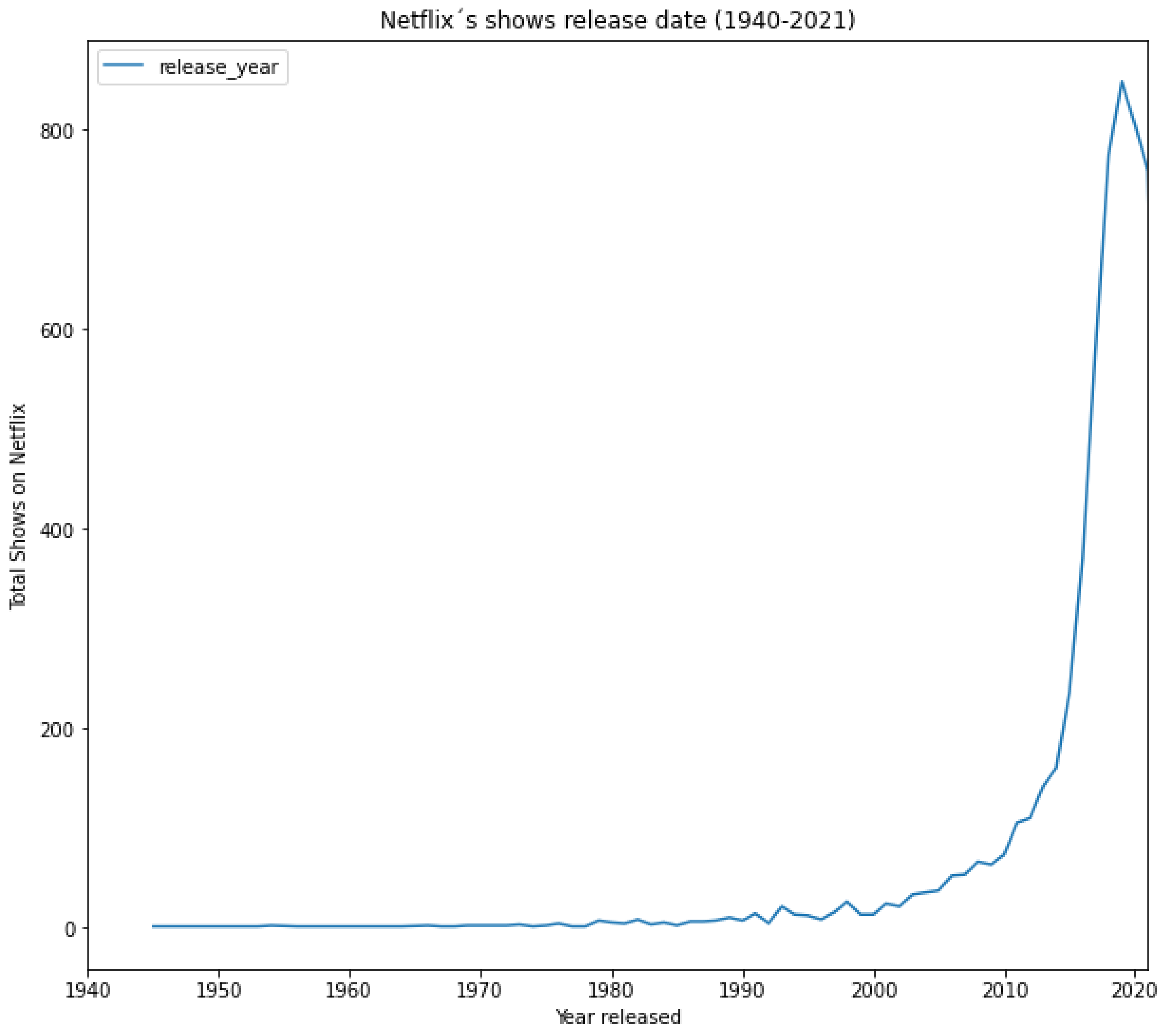
Out[28]:

	release_year
2019	848
2020	805
2018	774
2021	758
2017	580
...	...
1974	1
1959	1
1962	1
1978	1
1945	1

67 rows × 1 columns

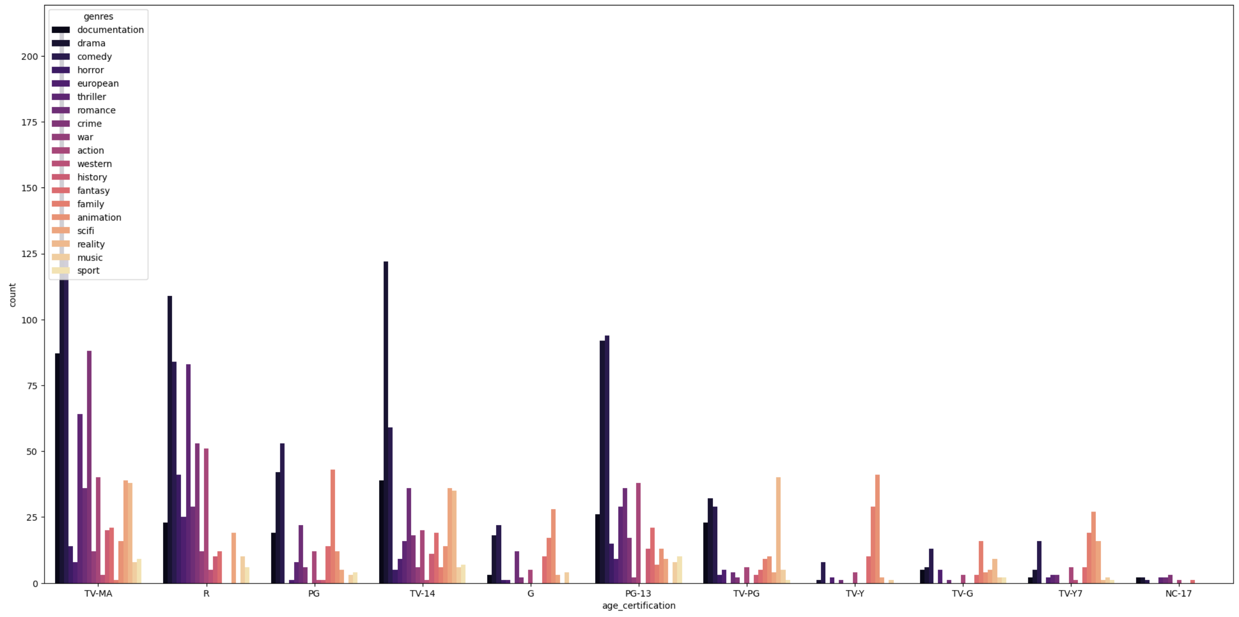
In [29]:

```
plt.figure(figsize=(10, 9))
sns.lineplot(data= release_year_count)
plt.title('Netflix´s shows release date (1940-2021)')
plt.xlim(1940, 2021)
plt.xlabel('Year released')
plt.ylabel('Total Shows on Netflix')
plt.show()
```



```
In [30]: plt.figure(figsize=(24, 12), dpi=100)
         sns.countplot(data=df, x="age_certification", hue="genres",palette = 'magma')
```

```
Out[30]: <AxesSubplot:xlabel='age_certification', ylabel='count'>
```



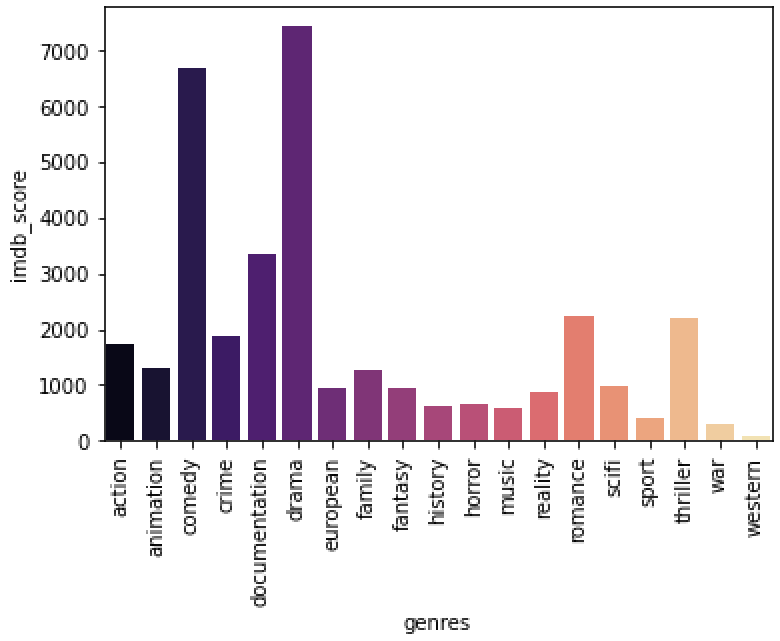
In [31]:
```
b= df.groupby('genres')[['imdb_score']].sum().reset_index()
b
```

Out[31]:

	genres	imdb_score
0	action	1726.3
1	animation	1302.2
2	comedy	6674.6
3	crime	1872.9
4	documentation	3345.9
5	drama	7437.8
6	european	959.2
7	family	1277.9
8	fantasy	936.0
9	history	605.9
10	horror	673.2
11	music	601.8
12	reality	884.5
13	romance	2229.4
14	scifi	965.0
15	sport	389.4
16	thriller	2195.3
17	war	311.2
18	western	77.1

In [32]:
```
sns.barplot(x='genres',y='imdb_score',data=b,palette = 'magma')
plt.xticks(rotation=90)
```

```
Out[32]: (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
                 17, 18]),
          [Text(0, 0, 'action'),
           Text(1, 0, 'animation'),
           Text(2, 0, 'comedy'),
           Text(3, 0, 'crime'),
           Text(4, 0, 'documentation'),
           Text(5, 0, 'drama'),
           Text(6, 0, 'european'),
           Text(7, 0, 'family'),
           Text(8, 0, 'fantasy'),
           Text(9, 0, 'history'),
           Text(10, 0, 'horror'),
           Text(11, 0, 'music'),
           Text(12, 0, 'reality'),
           Text(13, 0, 'romance'),
           Text(14, 0, 'scifi'),
           Text(15, 0, 'sport'),
           Text(16, 0, 'thriller'),
           Text(17, 0, 'war'),
           Text(18, 0, 'western')])
```



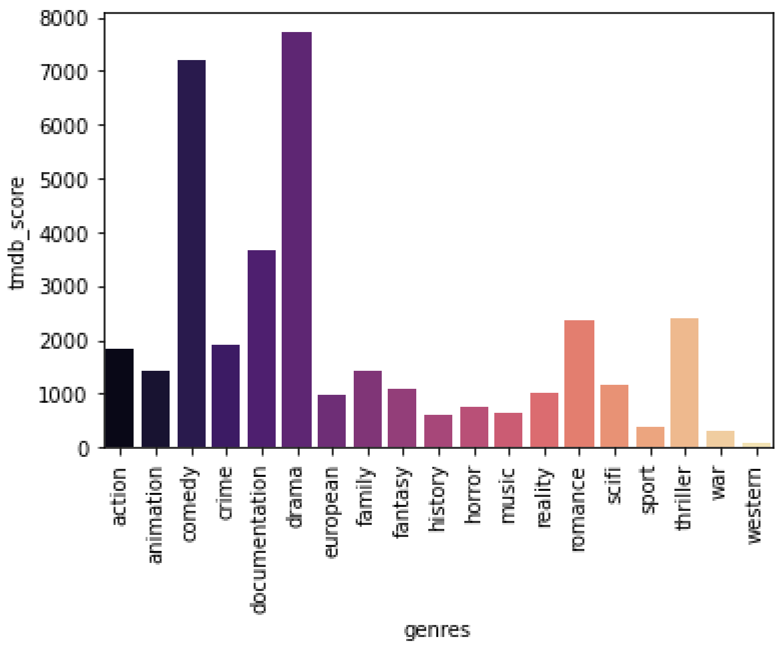
```
In [33]: b= df.groupby('genres')[['tmdb_score']].sum().reset_index()
         b
```

Out[33]:

	genres	tmdb_score
0	action	1850.0
1	animation	1440.2
2	comedy	7213.4
3	crime	1908.7
4	documentation	3674.5
5	drama	7712.8
6	european	988.9
7	family	1441.3
8	fantasy	1097.2
9	history	593.4
10	horror	769.0
11	music	647.2
12	reality	1026.0
13	romance	2368.5
14	scifi	1169.7
15	sport	399.8
16	thriller	2391.0
17	war	325.9
18	western	88.1

In [34]:
```
sns.barplot(x='genres',y='tmdb_score',data=b,palette = 'magma')
plt.xticks(rotation=90)
```

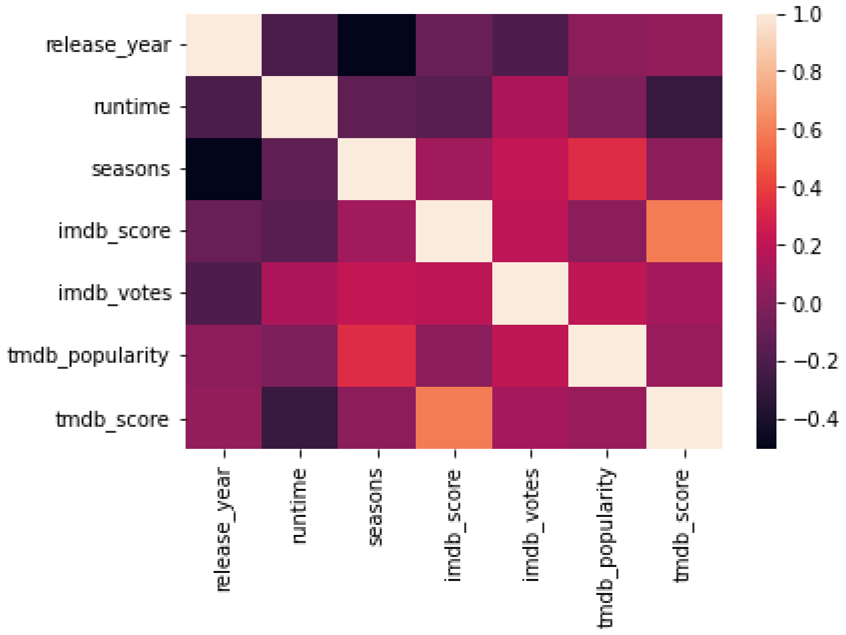
Out[34]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18]),
 [Text(0, 0, 'action'),
  Text(1, 0, 'animation'),
  Text(2, 0, 'comedy'),
  Text(3, 0, 'crime'),
  Text(4, 0, 'documentation'),
  Text(5, 0, 'drama'),
  Text(6, 0, 'european'),
  Text(7, 0, 'family'),
  Text(8, 0, 'fantasy'),
  Text(9, 0, 'history'),
  Text(10, 0, 'horror'),
  Text(11, 0, 'music'),
  Text(12, 0, 'reality'),
  Text(13, 0, 'romance'),
  Text(14, 0, 'scifi'),
  Text(15, 0, 'sport'),
  Text(16, 0, 'thriller'),
  Text(17, 0, 'war'),
  Text(18, 0, 'western')])
```



In [35]:
```
df_num = df[['release_year', 'runtime',   'seasons',  'imdb_score', 'imdb_votes', 'tmc
print(df_num.corr())
sns.heatmap(df_num.corr())
```

```
                 release_year   runtime   seasons  imdb_score  imdb_votes  \
release_year         1.000000 -0.211076 -0.505831   -0.102849   -0.196988
runtime             -0.211076  1.000000 -0.132740   -0.159297    0.138610
seasons             -0.505831 -0.132740  1.000000    0.097727    0.212645
imdb_score          -0.102849 -0.159297  0.097727    1.000000    0.189954
imdb_votes          -0.196988  0.138610  0.212645    0.189954    1.000000
tmdb_popularity      0.025628 -0.027493  0.331362    0.023159    0.201813
tmdb_score           0.049107 -0.285232  0.026796    0.587675    0.109720

                 tmdb_popularity  tmdb_score
release_year            0.025628    0.049107
runtime                -0.027493   -0.285232
seasons                 0.331362    0.026796
imdb_score              0.023159    0.587675
imdb_votes              0.201813    0.109720
tmdb_popularity         1.000000    0.068405
tmdb_score              0.068405    1.000000
```

Out[35]:
```
<AxesSubplot:>
```



In [71]:
```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [75]:
```
df_title=pd.read_csv('titles.csv')
```

In [76]:
```
df_title.head()
```

Out[76]:

	id	title	type	description	release_year	age_certification	runtime	genre
0	ts300399	Five Came Back: The Reference Films	SHOW	This collection includes 12 World War II-era p...	1945	TV-MA	48	['documentation
1	tm84618	Taxi Driver	MOVIE	A mentally unstable Vietnam War veteran works ...	1976	R	113	['crime', 'drama
2	tm127384	Monty Python and the Holy Grail	MOVIE	King Arthur, accompanied by his squire, recrui...	1975	PG	91	['comedy 'fantasy
3	tm70993	Life of Brian	MOVIE	Brian Cohen is an average young Jewish man, bu...	1979	R	94	['comedy
4	tm190788	The Exorcist	MOVIE	12-year-old Regan MacNeil begins to adapt an e...	1973	R	133	['horror

EDA Assignments:

How various movement pictures are inside the dataset and list the titles?

In [77]:
```
movies = df_title[df_title['type']=='MOVIE']['title']
print(f'The dataset has {len(movies)} movies')
movies[:20]
```

```
The dataset has 3759 movies
```

Out[77]:
```
1                                  Taxi Driver
2                Monty Python and the Holy Grail
3                                Life of Brian
4                                 The Exorcist
6                                  Dirty Harry
7                                  My Fair Lady
8                              The Blue Lagoon
9                             Bonnie and Clyde
10                           The Professionals
11                        The Guns of Navarone
12     Lupin the Third: The Castle of Cagliostro
13               Richard Pryor: Live in Concert
14                             The Long Riders
15                             White Christmas
16                               Cairo Station
17                                   The Queen
18                            Hitler: A Career
19                                         FTA
20                        Saladin the Victorious
21                                   Singapore
Name: title, dtype: object
```

How numerous show up are inside the dataset and list the titles?

In [78]:
```
shows = df_title[df_title['type']=='SHOW']['title']
print(f'The dataset has {len(shows)} movies')
shows[:20]
```

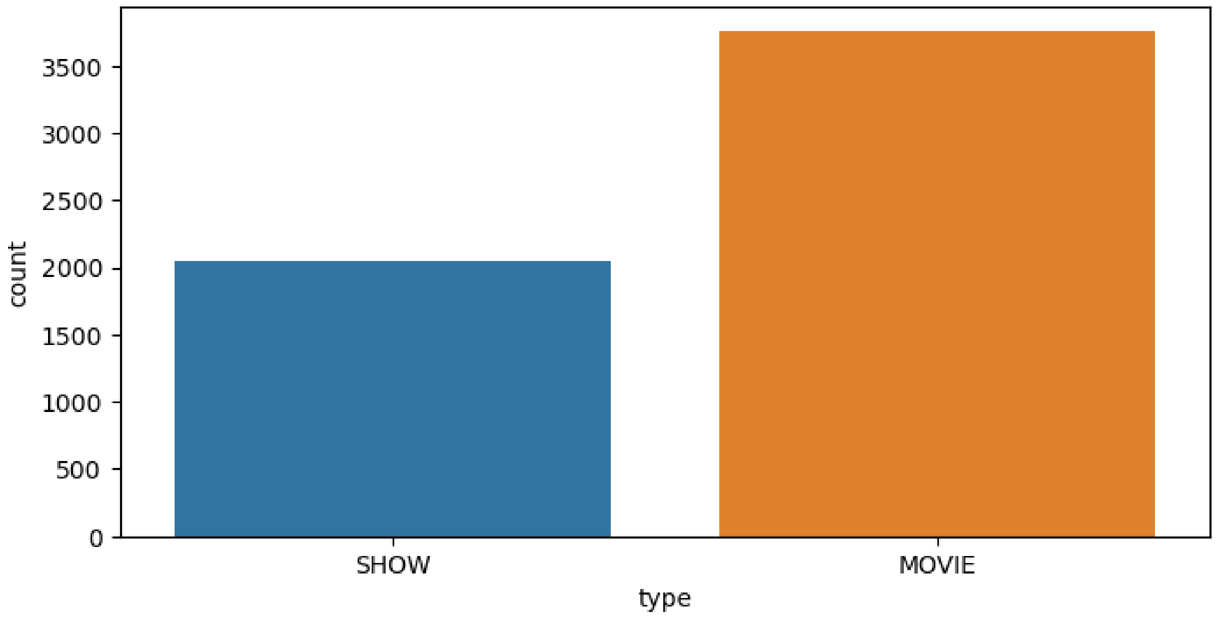
```
The dataset has 2047 movies
```

Out[78]:
```
0      Five Came Back: The Reference Films
5             Monty Python's Flying Circus
29          Monty Python's Fliegender Zirkus
47                                Seinfeld
55                            Knight Rider
57                        Thomas & Friends
60                       Saved by the Bell
64                        Wheel of Fortune
65                               Major Dad
66                             Fireman Sam
67                            Danger Mouse
98                               High Risk
105                               Survivor
106                          Stargate SG-1
107                                Pokémon
110                              One Piece
112                           Cowboy Bebop
113           Star Trek: Deep Space Nine
114                          The Challenge
116                          Gilmore Girls
Name: title, dtype: object
```

Plot the dispersal of movies/shows How various movies/shows are released each year?

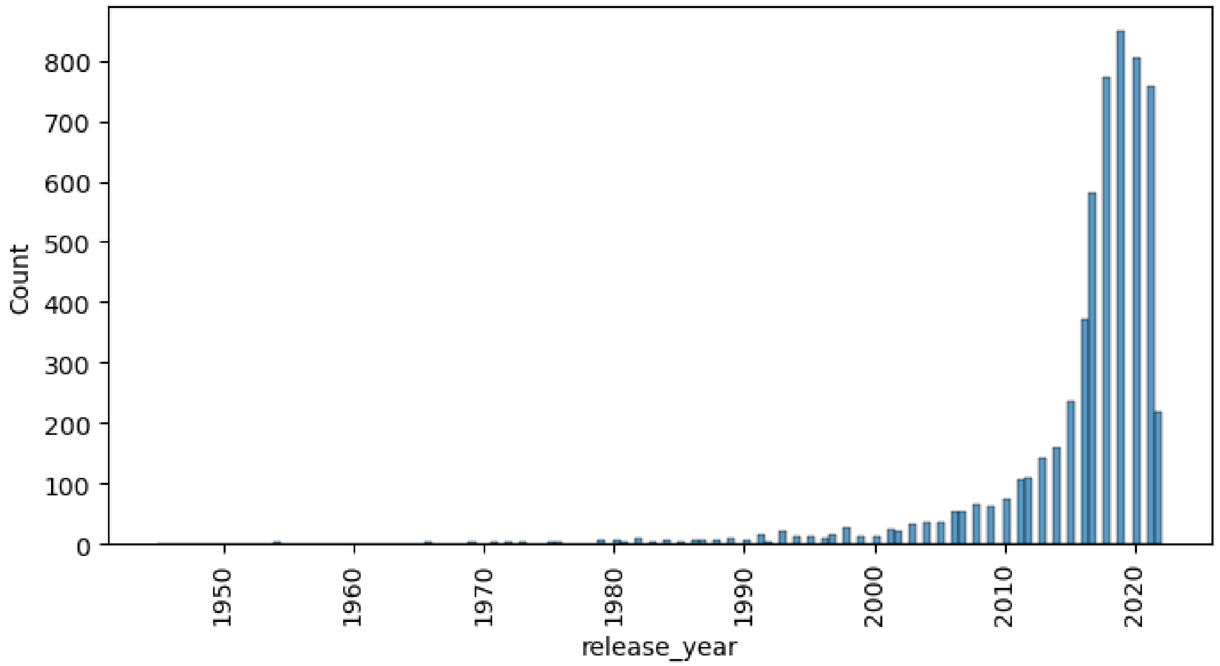
In [79]:
```python
plt.figure(figsize=(8,4),dpi=100)
sns.countplot(data=df_title,x='type')
```

Out[79]:
```
<AxesSubplot:xlabel='type', ylabel='count'>
```



How various moveis was released in 1976? How various shows up was released in 1945?

In [80]:
```
plt.figure(figsize=(8,4),dpi=100)
sns.histplot(data=df_title,x='release_year')
plt.xticks(rotation=90);
```



In [81]:
```
px.histogram(df_title, x = "release_year", color = "release_year")
```

How many shows was released in 1976?

In [82]:
```
movies = df_title[(df_title['type']=='MOVIE') & (df_title['release_year']==1976)]
print(f'{len(movies)} movies was released in 1976')
movies['title']
```

```
4 movies was released in 1976
```

Out[82]:
```
1                               Taxi Driver
30          The Return of the Prodigal Son
33     The Witness Who Didn't See Anything
43                    Chadi Jawani Budhe Nu
Name: title, dtype: object
```

Which Movement pictures is the longest and which one is the most limited?

In [ ]:
```
sss = df_title[df_title['type']=='MOVIE']
longest = sss[['title','runtime']].sort_values('runtime',ascending=False)[:1]

sss = df_title[df_title['type']=='MOVIE']
shortest = sss[['title','runtime']].sort_values('runtime')[:1]
```

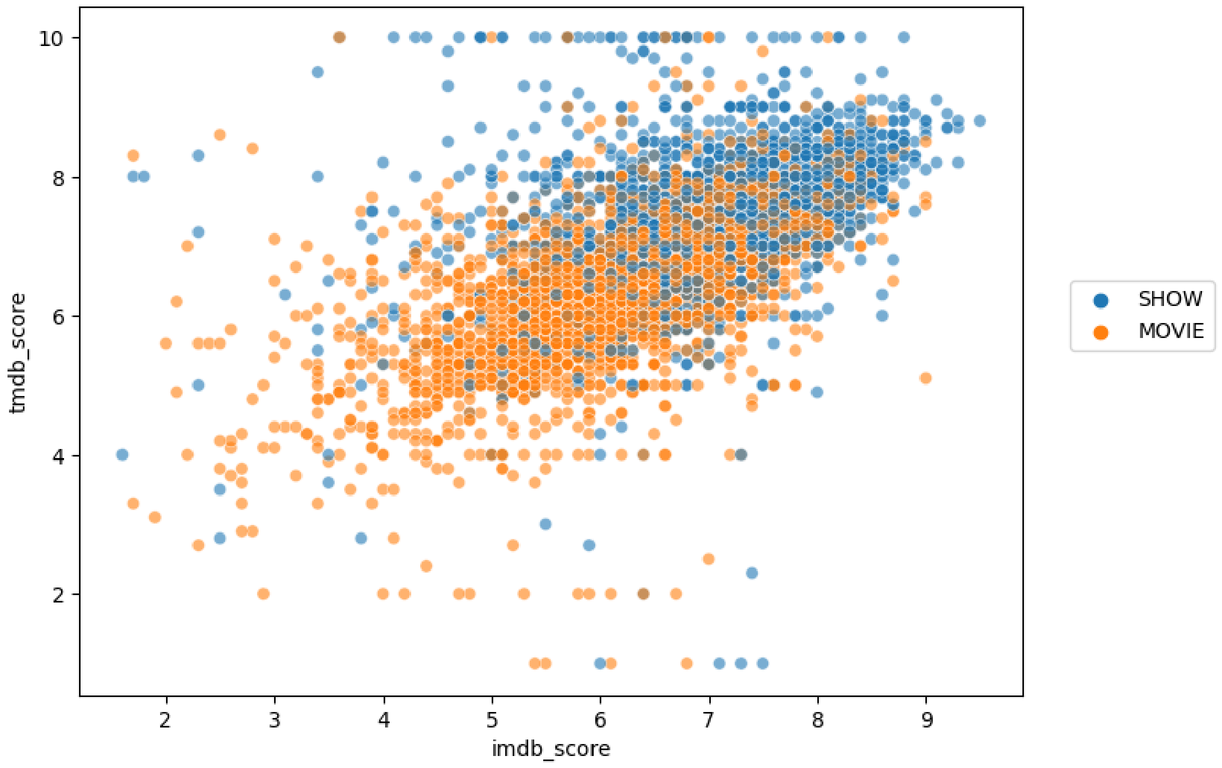
```
print(f'longest Movie:\n {longest}\n')
print(f'Shortest Movie:\n {shortest}')
```

```
longest Movie:
                        title  runtime
34  The School of Mischief      251

Shortest Movie:
        title  runtime
1115  Silent        3
```

Plot imdb_score vs tmdb_score

In [84]:
```
plt.figure(figsize=(8,6),dpi=100)
sns.scatterplot(data=df_title,x='imdb_score',y='tmdb_score',hue='type',alpha=0.6)
plt.legend(loc=(1.05,0.5))
```

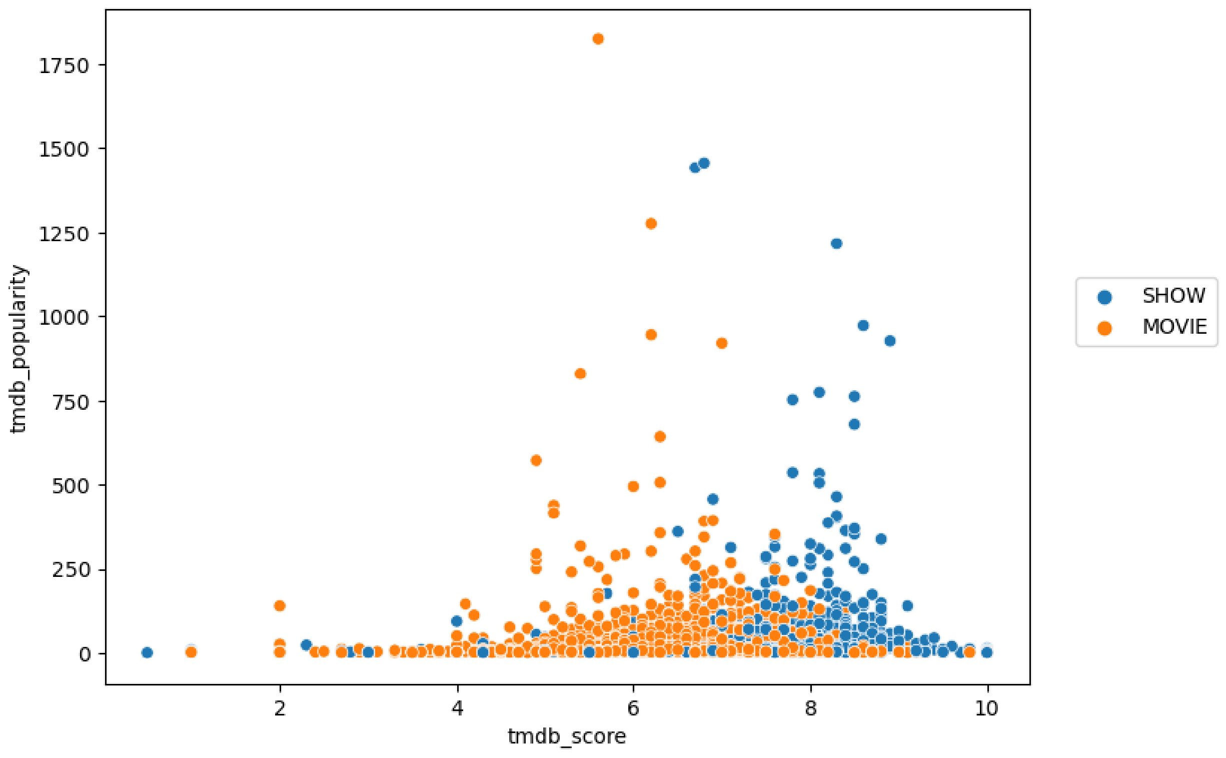
Out[84]:
```
<matplotlib.legend.Legend at 0x160b8d61430>
```



plot tmdb_score vs tmdb_score_popularity

In [85]:
```
plt.figure(figsize=(8,6),dpi=100)
sns.scatterplot(data=df_title,y='tmdb_popularity',x='tmdb_score',hue='type')
plt.legend(loc=(1.05,0.5))
```

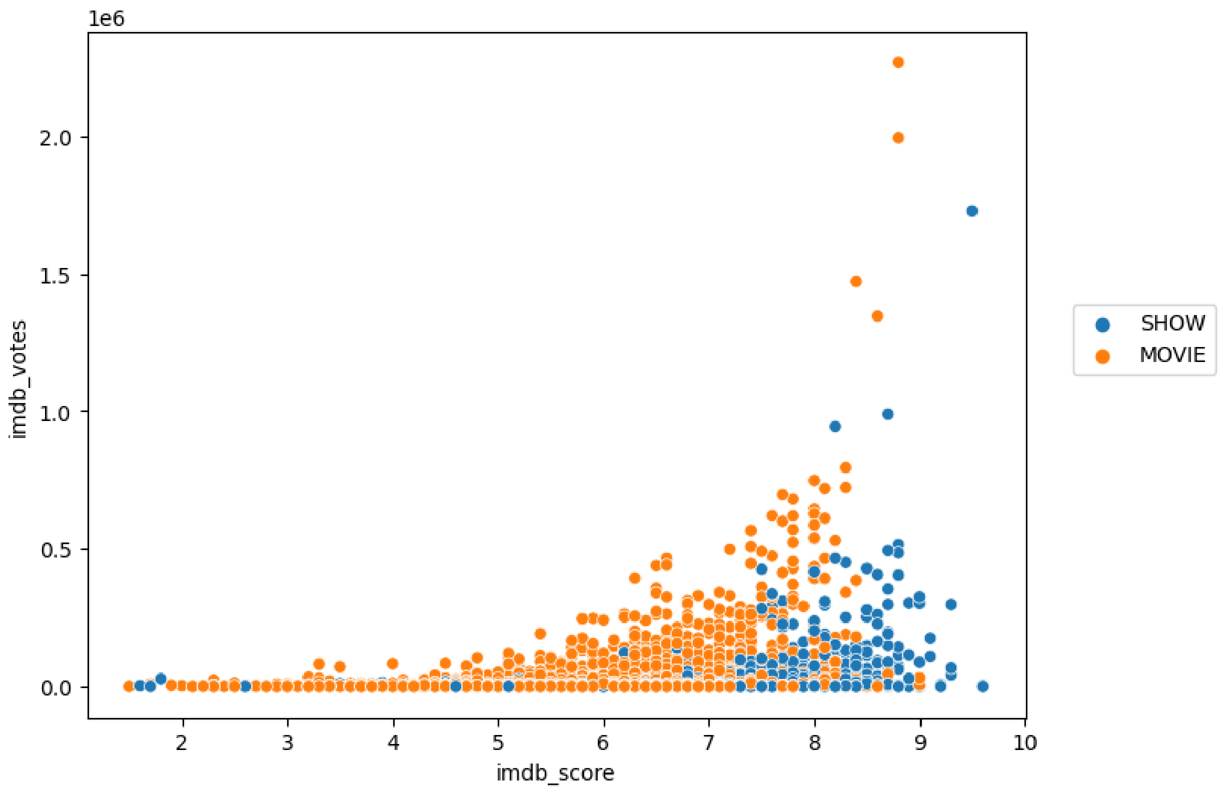
Out[85]:
```
<matplotlib.legend.Legend at 0x160b8f282e0>
```



plot imdb_score vs imdb_votes

In [87]:
```
plt.figure(figsize=(8,6),dpi=100)
sns.scatterplot(data=df_title,y='imdb_votes',x='imdb_score',hue='type')
plt.legend(loc=(1.05,0.5))
```

Out[87]:
```
<matplotlib.legend.Legend at 0x160b8de4f40>
```

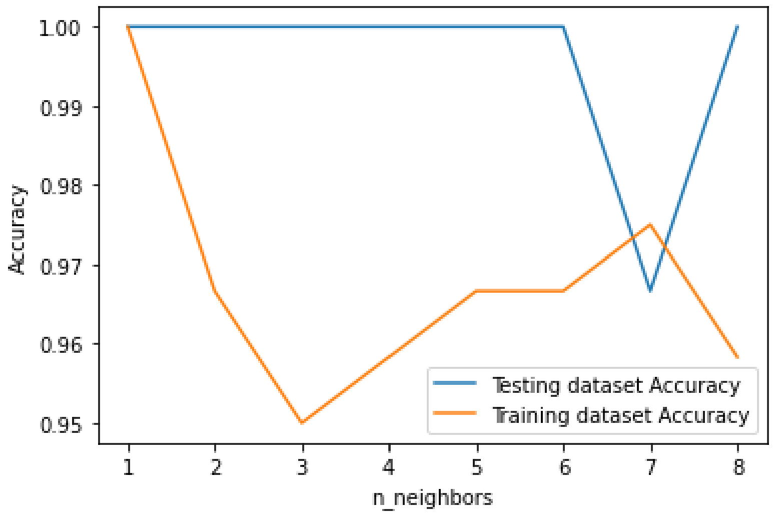


In [ ]:

In [89]:
```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

# Loading data
irisData = load_iris()

# Create feature and target arrays
X = irisData.data
y = irisData.target

# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(
                        X, y, test_size = 0.2, random_state=42)

knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(X_train, y_train)

# Predict on dataset which model has not seen before
print(knn.predict(X_test))
```

```
[1 0 2 1 1 0 1 2 2 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0]
```

In [90]:
```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
import numpy as np
import matplotlib.pyplot as plt

irisData = load_iris()

# Create feature and target arrays
X = irisData.data
y = irisData.target

# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(
                        X, y, test_size = 0.2, random_state=42)

neighbors = np.arange(1, 9)
train_accuracy = np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))

# Loop over K values
for i, k in enumerate(neighbors):
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)

        # Compute training and test data accuracy
        train_accuracy[i] = knn.score(X_train, y_train)
        test_accuracy[i] = knn.score(X_test, y_test)

# Generate plot
plt.plot(neighbors, test_accuracy, label = 'Testing dataset Accuracy')
plt.plot(neighbors, train_accuracy, label = 'Training dataset Accuracy')
plt.legend()
plt.xlabel('n_neighbors')
plt.ylabel('Accuracy')
plt.show()
```



In [1]:
```
pip install -U notebook-as-pdf
```

```
Collecting notebook-as-pdf
  Downloading notebook_as_pdf-0.5.0-py3-none-any.whl (6.5 kB)
Collecting PyPDF2
  Downloading PyPDF2-2.7.0-py3-none-any.whl (202 kB)
Collecting pyppeteer
  Downloading pyppeteer-1.0.2-py3-none-any.whl (83 kB)
Requirement already satisfied: nbconvert in c:\users\shaki\anaconda3\lib\site-package
s (from notebook-as-pdf) (6.4.4)
Requirement already satisfied: beautifulsoup4 in c:\users\shaki\anaconda3\lib\site-pa
ckages (from nbconvert->notebook-as-pdf) (4.11.1)
Requirement already satisfied: pygments>=2.4.1 in c:\users\shaki\anaconda3\lib\site-p
ackages (from nbconvert->notebook-as-pdf) (2.11.2)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\shaki\anaconda3\lib\sit
e-packages (from nbconvert->notebook-as-pdf) (0.4)
Requirement already satisfied: bleach in c:\users\shaki\anaconda3\lib\site-packages
(from nbconvert->notebook-as-pdf) (4.1.0)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\shaki\anaconda3\lib\s
ite-packages (from nbconvert->notebook-as-pdf) (1.5.0)
Requirement already satisfied: nbformat>=4.4 in c:\users\shaki\anaconda3\lib\site-pac
kages (from nbconvert->notebook-as-pdf) (5.3.0)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in c:\users\shaki\anaconda3\lib
\site-packages (from nbconvert->notebook-as-pdf) (0.5.13)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\shaki\anaconda3\lib\site
-packages (from nbconvert->notebook-as-pdf) (0.8.4)
Requirement already satisfied: defusedxml in c:\users\shaki\anaconda3\lib\site-packag
es (from nbconvert->notebook-as-pdf) (0.7.1)
Requirement already satisfied: jupyter-core in c:\users\shaki\anaconda3\lib\site-pack
ages (from nbconvert->notebook-as-pdf) (4.10.0)
Requirement already satisfied: jupyterlab-pygments in c:\users\shaki\anaconda3\lib\si
te-packages (from nbconvert->notebook-as-pdf) (0.1.2)
Requirement already satisfied: testpath in c:\users\shaki\anaconda3\lib\site-packages
(from nbconvert->notebook-as-pdf) (0.5.0)
Requirement already satisfied: jinja2>=2.4 in c:\users\shaki\anaconda3\lib\site-packa
ges (from nbconvert->notebook-as-pdf) (2.11.3)
Requirement already satisfied: traitlets>=5.0 in c:\users\shaki\anaconda3\lib\site-pa
ckages (from nbconvert->notebook-as-pdf) (5.1.1)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\shaki\anaconda3\lib\site-
packages (from jinja2>=2.4->nbconvert->notebook-as-pdf) (2.0.1)
Requirement already satisfied: nest-asyncio in c:\users\shaki\anaconda3\lib\site-pack
ages (from nbclient<0.6.0,>=0.5.0->nbconvert->notebook-as-pdf) (1.5.5)
Requirement already satisfied: jupyter-client>=6.1.5 in c:\users\shaki\anaconda3\lib
\site-packages (from nbclient<0.6.0,>=0.5.0->nbconvert->notebook-as-pdf) (6.1.12)
Requirement already satisfied: pyzmq>=13 in c:\users\shaki\anaconda3\lib\site-package
s (from jupyter-client>=6.1.5->nbclient<0.6.0,>=0.5.0->nbconvert->notebook-as-pdf) (2
2.3.0)
Requirement already satisfied: tornado>=4.1 in c:\users\shaki\anaconda3\lib\site-pack
ages (from jupyter-client>=6.1.5->nbclient<0.6.0,>=0.5.0->nbconvert->notebook-as-pdf)
(6.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\shaki\anaconda3\lib\s
ite-packages (from jupyter-client>=6.1.5->nbclient<0.6.0,>=0.5.0->nbconvert->notebook
-as-pdf) (2.8.2)
Requirement already satisfied: pywin32>=1.0 in c:\users\shaki\anaconda3\lib\site-pack
ages (from jupyter-core->nbconvert->notebook-as-pdf) (302)
Requirement already satisfied: jsonschema>=2.6 in c:\users\shaki\anaconda3\lib\site-p
ackages (from nbformat>=4.4->nbconvert->notebook-as-pdf) (4.4.0)
Requirement already satisfied: fastjsonschema in c:\users\shaki\anaconda3\lib\site-pa
ckages (from nbformat>=4.4->nbconvert->notebook-as-pdf) (2.15.1)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in c:\us
ers\shaki\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat>=4.4->nbconvert
->notebook-as-pdf) (0.18.0)
```

```
Requirement already satisfied: attrs>=17.4.0 in c:\users\shaki\anaconda3\lib\site-pac
kages (from jsonschema>=2.6->nbformat>=4.4->nbconvert->notebook-as-pdf) (21.4.0)
Requirement already satisfied: six>=1.5 in c:\users\shaki\anaconda3\lib\site-packages
(from python-dateutil>=2.1->jupyter-client>=6.1.5->nbclient<0.6.0,>=0.5.0->nbconvert-
>notebook-as-pdf) (1.16.0)
Requirement already satisfied: soupsieve>1.2 in c:\users\shaki\anaconda3\lib\site-pac
kages (from beautifulsoup4->nbconvert->notebook-as-pdf) (2.3.1)
Requirement already satisfied: packaging in c:\users\shaki\anaconda3\lib\site-package
s (from bleach->nbconvert->notebook-as-pdf) (21.3)
Requirement already satisfied: webencodings in c:\users\shaki\anaconda3\lib\site-pack
ages (from bleach->nbconvert->notebook-as-pdf) (0.5.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\shaki\anaconda3\l
ib\site-packages (from packaging->bleach->nbconvert->notebook-as-pdf) (3.0.4)
Requirement already satisfied: typing-extensions in c:\users\shaki\anaconda3\lib\site
-packages (from PyPDF2->notebook-as-pdf) (4.1.1)
Requirement already satisfied: tqdm<5.0.0,>=4.42.1 in c:\users\shaki\anaconda3\lib\si
te-packages (from pyppeteer->notebook-as-pdf) (4.64.0)
Requirement already satisfied: urllib3<2.0.0,>=1.25.8 in c:\users\shaki\anaconda3\lib
\site-packages (from pyppeteer->notebook-as-pdf) (1.26.9)
Collecting pyee<9.0.0,>=8.1.0
  Downloading pyee-8.2.2-py2.py3-none-any.whl (12 kB)
Collecting websockets<11.0,>=10.0
  Downloading websockets-10.3-cp39-cp39-win_amd64.whl (98 kB)
Requirement already satisfied: appdirs<2.0.0,>=1.4.3 in c:\users\shaki\anaconda3\lib
\site-packages (from pyppeteer->notebook-as-pdf) (1.4.4)
Requirement already satisfied: certifi>=2021 in c:\users\shaki\anaconda3\lib\site-pac
kages (from pyppeteer->notebook-as-pdf) (2022.5.18.1)
Requirement already satisfied: importlib-metadata>=1.4 in c:\users\shaki\anaconda3\li
b\site-packages (from pyppeteer->notebook-as-pdf) (4.11.3)
Requirement already satisfied: zipp>=0.5 in c:\users\shaki\anaconda3\lib\site-package
s (from importlib-metadata>=1.4->pyppeteer->notebook-as-pdf) (3.8.0)
Requirement already satisfied: colorama in c:\users\shaki\anaconda3\lib\site-packages
(from tqdm<5.0.0,>=4.42.1->pyppeteer->notebook-as-pdf) (0.4.4)
Installing collected packages: websockets, pyee, pyppeteer, PyPDF2, notebook-as-pdf
Successfully installed PyPDF2-2.7.0 notebook-as-pdf-0.5.0 pyee-8.2.2 pyppeteer-1.0.2
websockets-10.3
Note: you may need to restart the kernel to use updated packages.
```

In [7]:

```python
import nbconvert
```

In [9]:

```
pip install LaTeX
```

```
Collecting LaTeX
  Downloading latex-0.7.0.tar.gz (6.5 kB)
Collecting tempdir
  Downloading tempdir-0.7.1.tar.gz (5.9 kB)
Collecting data
  Downloading data-0.4.tar.gz (7.0 kB)
Requirement already satisfied: future in c:\users\shaki\anaconda3\lib\site-packages
(from LaTeX) (0.18.2)
Collecting shutilwhich
  Downloading shutilwhich-1.1.0.tar.gz (2.3 kB)
Requirement already satisfied: six in c:\users\shaki\anaconda3\lib\site-packages (fro
m data->LaTeX) (1.16.0)
Requirement already satisfied: decorator in c:\users\shaki\anaconda3\lib\site-package
s (from data->LaTeX) (5.1.1)
Collecting funcsigs
  Downloading funcsigs-1.0.2-py2.py3-none-any.whl (17 kB)
Building wheels for collected packages: LaTeX, data, shutilwhich, tempdir
  Building wheel for LaTeX (setup.py): started
  Building wheel for LaTeX (setup.py): finished with status 'done'
  Created wheel for LaTeX: filename=latex-0.7.0-py3-none-any.whl size=7604 sha256=633
43b4a1f7400717488b53a67b152f845f221ef71c263c18c46ab70d5bb6f02
  Stored in directory: c:\users\shaki\appdata\local\pip\cache\wheels\94\84\e5\5ce5825
23fd479d00356867953085a67c47fbbc86506aa92f8
  Building wheel for data (setup.py): started
  Building wheel for data (setup.py): finished with status 'done'
  Created wheel for data: filename=data-0.4-py3-none-any.whl size=7247 sha256=73cb2f2
d3ffd3f6d91b7b27e3b2114929a0d802339fd2668d0f178309ad29c61
  Stored in directory: c:\users\shaki\appdata\local\pip\cache\wheels\8a\0b\a3\37ca07d
5a2838bba2e475e8090455e40b94631bd57a99a35f4
  Building wheel for shutilwhich (setup.py): started
  Building wheel for shutilwhich (setup.py): finished with status 'done'
  Created wheel for shutilwhich: filename=shutilwhich-1.1.0-py3-none-any.whl size=278
1 sha256=2bd5cfe3a478ae899c34f040ff1981bf1539074861c662c4519de828c9e05361
  Stored in directory: c:\users\shaki\appdata\local\pip\cache\wheels\84\c7\f5\fed66dc
e1ed897b44e0da776b6a592dfad0a70f7dd61f73a9d
  Building wheel for tempdir (setup.py): started
  Building wheel for tempdir (setup.py): finished with status 'done'
  Created wheel for tempdir: filename=tempdir-0.7.1-py3-none-any.whl size=2214 sha256
=b9c391f659d1564bdfecb50deb4241e03a70cbd394b36a42d9962187c74241a9
  Stored in directory: c:\users\shaki\appdata\local\pip\cache\wheels\31\7b\e3\af441c2
f71a48c30809aada978c1433b163a0747e73b5805ca
Successfully built LaTeX data shutilwhich tempdir
Installing collected packages: funcsigs, tempdir, shutilwhich, data, LaTeX
Successfully installed LaTeX-0.7.0 data-0.4 funcsigs-1.0.2 shutilwhich-1.1.0 tempdir-
0.7.1
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
import latex
```

In [2]:
```
pip install pandoc
```

```
Collecting pandoc
  Downloading pandoc-2.2.tar.gz (29 kB)
Collecting plumbum
  Downloading plumbum-1.7.2-py2.py3-none-any.whl (117 kB)
Requirement already satisfied: ply in c:\users\shaki\anaconda3\lib\site-packages (fro
m pandoc) (3.11)
Requirement already satisfied: pywin32 in c:\users\shaki\anaconda3\lib\site-packages
(from plumbum->pandoc) (302)
Building wheels for collected packages: pandoc
  Building wheel for pandoc (setup.py): started
  Building wheel for pandoc (setup.py): finished with status 'done'
  Created wheel for pandoc: filename=pandoc-2.2-py3-none-any.whl size=29557 sha256=1c
ff2b636d5b2bdb6df12b44dad1a23f24b0648cefb2efb0526fcd44975a6d2f
  Stored in directory: c:\users\shaki\appdata\local\pip\cache\wheels\2d\da\b1\54ff040
1ef9b07b60c7fc9cffe616f243cf27dc3d04bd5d5ef
Successfully built pandoc
Installing collected packages: plumbum, pandoc
Successfully installed pandoc-2.2 plumbum-1.7.2
Note: you may need to restart the kernel to use updated packages.
```

In [3]:

```
import pandoc
```

In [ ]:

```
```