Streaming company

Netflix, Amazon Prime and Disney + Movies and TV Shows

Current sprint: **1**

Objectives

After completing this notebook we will be able to:

- Offer to streaming company best products to offer

members: German Briz, Carlo Marmolejo

Table of Contents

Explore data:

In [1]:
```
import pandas as pd
import numpy as np
import plotly.express as px
import re
```

In [2]:
```
df = pd.read_csv('archive/netflix_titles.csv')
```

In [3]:

```
def print_details(df):
    print(df.info())
    print("*" * 50)
    print(df.describe())
    print("*" * 50)
    print(df.isnull().sum().sort_values())
def plot_by_cats(df):
    for cat in ['type','country', 'rating', 'release_year', 'rating', 'duration', 'listed_in']:
        fig = px.histogram(df, x=cat, barmode='group',title=f'Count plot for given {cat}', height=700)
        fig.show()
def get_num_artist(df):
    df_artist = df['cast'].str.split(',', expand = True)#.fillna('notFound')
    df_artist.columns = ['artist_{}'.format(col) for col in df_artist.columns]
    df['num_artist'] = df_artist.count(axis=1)
    return df
def date_columns_for_df(df):
    df['date_added'] = pd.to_datetime(df['date_added'])
    df['year'] = df['date_added'].dt.year
    df['month'] = df['date_added'].dt.month
    df['day'] = df['date_added'].dt.day
    return df
def create_range_for_movies(time_movie):
    if (time_movie < 60):
        return 'A'
    elif (time_movie >= 60) &  (time_movie <= 90):
        return 'B'
    elif (time_movie > 90) & (time_movie <= 120):
        return 'C'
    else: return 'D'
def create_range_for_seasons(seasons):
    if (seasons < 3):
        return 'E'
    elif (seasons >= 3) &  (seasons <= 7):
        return 'F'
    elif (seasons > 7) & (seasons <= 10):
        return 'G'
    else: return 'H'
def create_ranges(duration):
    if ('min' in duration):
        duration =  int(re.findall("\d+", duration)[0])
        return create_range_for_movies(duration)
    elif (('Season' in duration) or ('Seasons' in duration)):
        duration = int(re.findall("\d+", duration)[0])
        return create_range_for_seasons(duration)
    else:
        return 'nan'
```

In [4]:
```
print_details(df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
None
**************************************************
       release_year
count   8807.000000
mean    2014.180198
std        8.819312
min     1925.000000
25%     2013.000000
50%     2017.000000
75%     2019.000000
max     2021.000000
**************************************************
show_id            0
type               0
title              0
release_year       0
listed_in          0
description        0
duration           3
rating             4
date_added        10
cast             825
country          831
director        2634
dtype: int64
```

In [5]:
```
plot_by_cats(df)
```

Data Cleaning and task To-Do:

Comments init:

- There are more movies than tv shows
- show_id is id from row
- there are 2 types of media : serie or movie
- For series duration measure is season, for movie minutes

Commnets version 3

- We can analyze tv shows
- We will analyze Prime data, on kaggle there's a similar data like Netflix

Tasks todo

- Create range for duration
- change date format
- Count num of artist in casting

In [6]:
```
df = get_num_artist(df)
```

In [7]:
```
df = date_columns_for_df(df)
```

In [8]:
```
df[df.duration.isnull()]
```

Out[8]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	num_artist	y
5541	s5542	Movie	Louis C.K. 2017	Louis C.K.	Louis C.K.	United States	2017-04-04	2017	74 min	NaN	Movies	Louis C.K. muses on religion, eternal love, gi...	1	201
5794	s5795	Movie	Louis C.K.: Hilarious	Louis C.K.	Louis C.K.	United States	2016-09-16	2010	84 min	NaN	Movies	Emmy-winning comedy writer Louis C.K. brings h...	1	201
5813	s5814	Movie	Louis C.K.: Live at the Comedy Store	Louis C.K.	Louis C.K.	United States	2016-08-15	2015	66 min	NaN	Movies	The comic puts his trademark hilarious/thought...	1	201

In [9]:
```
df.loc[df.duration.isnull(), 'duration'] = df.loc[df.duration.isnull(), 'rating']
```

In [10]:
```
df['duration_range'] = [create_ranges(duration) for duration in df.duration]
```

Data cleaning after first analytics

In [11]:
```
df['show_id'].str[0].unique()
```

Out[11]:
```
array(['s'], dtype=object)
```

In [12]:
```
df['show_id'] = df['show_id'].str[1:]
```

In [13]:
```
df = df[df['release_year'] != 2021]
```

In [14]:
```
df_netflix = df.copy()
df_netflix.duration = df_netflix.duration.str.extract('(\d+)')
```

In [15]:
```
df_netflix.loc[:,'data_source'] = 'Netflix'
```

Read Amazon

In [16]:
```
df = pd.read_csv('archive/amazon_prime_titles.csv')
df.tail(4)
```

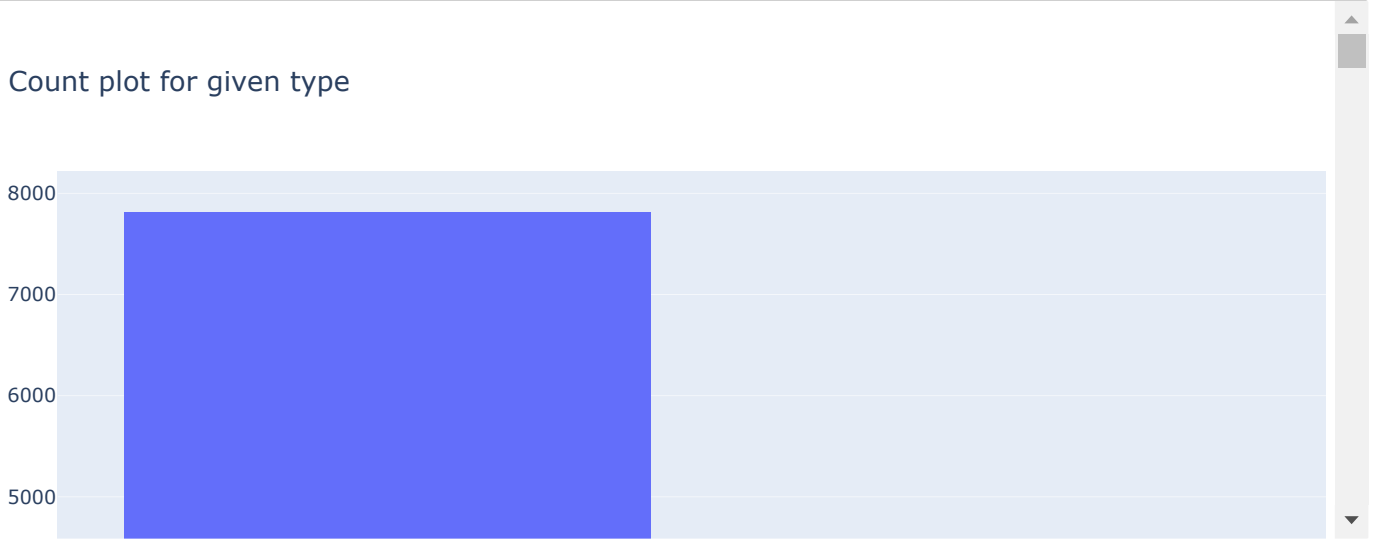
Out[16]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
9664	s9665	TV Show	Planet Patrol	NaN	DICK VOSBURGH, RONNIE STEVENS, LIBBY MORRIS, M...	NaN	NaN	2018	13+	4 Seasons	TV Shows	This is Earth, 2100AD - and these are the adve...
9665	s9666	Movie	Outpost	Steve Barker	Ray Stevenson, Julian Wadham, Richard Brake, M...	NaN	NaN	2008	R	90 min	Action	In war-torn Eastern Europe, a world-weary grou...
9666	s9667	TV Show	Maradona: Blessed Dream	NaN	Esteban Recagno, Ezequiel Stremiz, Luciano Vit...	NaN	NaN	2021	TV-MA	1 Season	Drama, Sports	The series tells the story of Diego Maradona, ...
9667	s9668	Movie	Harry Brown	Daniel Barber	Michael Caine, Emily Mortimer, Joseph Gilgun, ...	NaN	NaN	2010	R	103 min	Action, Drama, Suspense	Harry Brown, starring two-time Academy Award w...

In [17]:
```
print_details(df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9668 entries, 0 to 9667
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       9668 non-null   object
 1   type          9668 non-null   object
 2   title         9668 non-null   object
 3   director      7586 non-null   object
 4   cast          8435 non-null   object
 5   country       672 non-null    object
 6   date_added    155 non-null    object
 7   release_year  9668 non-null   int64 
 8   rating        9331 non-null   object
 9   duration      9668 non-null   object
 10  listed_in     9668 non-null   object
 11  description   9668 non-null   object
dtypes: int64(1), object(11)
memory usage: 906.5+ KB
None
**************************************************
       release_year
count   9668.000000
mean    2008.341849
std       18.922482
min     1920.000000
25%     2007.000000
50%     2016.000000
75%     2019.000000
max     2021.000000
**************************************************
show_id            0
type               0
title              0
release_year       0
duration           0
listed_in          0
description        0
rating           337
cast            1233
director        2082
country         8996
date_added      9513
dtype: int64
```

In [18]:
```
plot_by_cats(df)
```



In [19]:
```
df = get_num_artist(df)
df = date_columns_for_df(df)
```

In [20]:
```
df[df.duration.isnull()]
```

Out[20]:

show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	num_artist	year	month	day

In [21]:
```
df['duration_range'] = [create_ranges(duration) for duration in df.duration]
df['show_id'] = df['show_id'].str[1:]
df = df[df['release_year'] != 2021]
```

In [22]:
```
df_prime = df.copy()
```

In [23]:
```
df_prime.loc[:,'data_source'] = 'Prime'
```

In [24]:
```
df_netflix.shape, df_prime.shape
```

Out[24]:
```
((8215, 18), (8226, 18))
```

In [25]:
```
df_prime.groupby('type').count()
```

Out[25]:

type	show_id	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	num_artist	year	month	day
Movie	6675	6675	6463	6048	528	16	6675	6357	6675	6675	6675	6675	16	16	16
TV Show	1551	1551	0	1155	85	86	1551	1549	1551	1551	1551	1551	86	86	86

In [26]:
```
df_prime.listed_in
```

Out[26]:
```
0                  Comedy, Drama
1           Drama, International
2        Action, Drama, Suspense
3                   Documentary
4                 Drama, Fantasy
                  ...
9661                    TV Shows
9663                      Comedy
9664                    TV Shows
9665                      Action
9667     Action, Drama, Suspense
Name: listed_in, Length: 8226, dtype: object
```

Concat Netflix + Prime

In [27]:
```
df_final = pd.concat([df_netflix,df_prime], ignore_index = True)
```

Netflix Revenue

In [28]:
```
df = pd.read_csv('archive/DataNetflixRevenue2020_V2.csv')
```

In [29]:
```
print_details(df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40 entries, 0 to 39
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   Area     40 non-null     object
 1   Years    40 non-null     object
 2   Revenue  40 non-null     int64 
dtypes: int64(1), object(2)
memory usage: 1.1+ KB
None
**************************************************
            Revenue
count  4.000000e+01
mean   1.176952e+09
std    8.246170e+08
min    1.991170e+08
25%    5.567758e+08
50%    8.400510e+08
75%    1.913442e+09
max    2.839670e+09
**************************************************
Area       0
Years      0
Revenue    0
dtype: int64
```

In [30]:
```
df['Quarter'] = df['Years'].str.split('-',expand = True)[0]
```

In [31]:
```
df['Year'] = df['Years'].str.split('-',expand = True)[1]
```

In [32]:
```
df.head(1)
```

Out[32]:

	Area	Years	Revenue	Quarter	Year
0	United States and Canada	Q1 - 2018	1976157000	Q1	2018

In [33]:
```
df.to_csv('netflix_revenue.csv', index=None)
```

Disney plus

In [34]:
```
df = pd.read_csv('archive/disney_plus_titles.csv')
df.tail(2)
```

Out[34]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
1448	s1449	Movie	Bend It Like Beckham	Gurinder Chadha	Parminder Nagra, Keira Knightley, Jonathan Rhy...	United Kingdom, Germany, United States	September 18, 2020	2003	PG-13	112 min	Buddy, Comedy, Coming of Age	Despite the wishes of their traditional famili...
1449	s1450	Movie	Captain Sparky vs. The Flying Saucers	Mark Waring	Charlie Tahan	United States	April 1, 2020	2012	TV-G	2 min	Action-Adventure, Animals & Nature, Animation	View one of Sparky's favorite home movies.

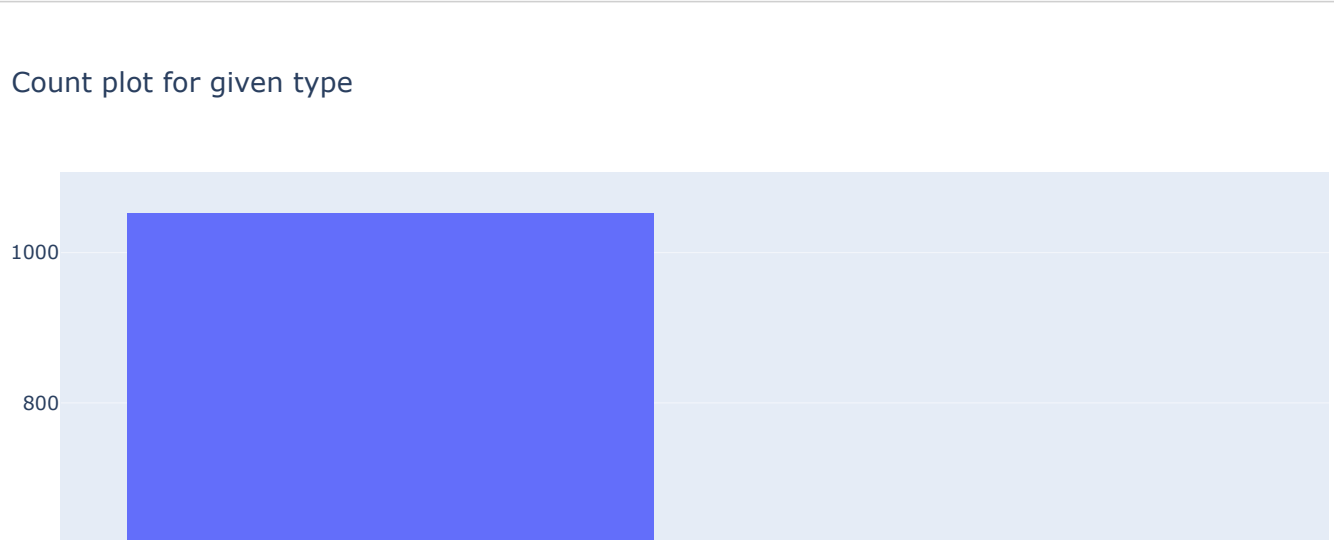
In [35]:
```
df.type.unique()
```

Out[35]:
```
array(['Movie', 'TV Show'], dtype=object)
```

In [36]:
```
print_details(df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1450 entries, 0 to 1449
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       1450 non-null   object
 1   type          1450 non-null   object
 2   title         1450 non-null   object
 3   director      977 non-null    object
 4   cast          1260 non-null   object
 5   country       1231 non-null   object
 6   date_added    1447 non-null   object
 7   release_year  1450 non-null   int64 
 8   rating        1447 non-null   object
 9   duration      1450 non-null   object
 10  listed_in     1450 non-null   object
 11  description   1450 non-null   object
dtypes: int64(1), object(11)
memory usage: 136.1+ KB
None
**************************************************
       release_year
count   1450.000000
mean    2003.091724
std       21.860162
min     1928.000000
25%     1999.000000
50%     2011.000000
75%     2018.000000
max     2021.000000
**************************************************
show_id           0
type              0
title             0
release_year      0
duration          0
listed_in         0
description       0
date_added        3
rating            3
cast            190
country         219
director        473
dtype: int64
```

In [37]:
```
plot_by_cats(df)
```



In [38]:
```
df['duration_range'] = [create_ranges(duration) for duration in df.duration]
```

In [39]:
```
df = get_num_artist(df)
df = date_columns_for_df(df)
```

In [40]:
```
df['show_id'] = df['show_id'].str[1:]
df = df[df['release_year'] != 2021]
```

In [41]:
```
df_disney = df.copy()
df_disney.loc[:,'data_source'] = 'Disney+'
```

In [42]:
```
df_final = pd.concat([df_final,df_disney], ignore_index = True)
```

In [43]:
```
df_final.to_csv('df_netflix_prime_disney.csv', index = None)
print('guardado')
```

```
guardado
```

In [44]:
```
df_final.groupby(['release_year','data_source']).count()
```

Out[44]:

		show_id	type	title	director	cast	country	date_added	rating	duration	listed_in	description	num_artist	year
release_year	**data_source**													
1920	**Prime**	3	3	3	3	3	0	0	3	3	3	3	3	0
1922	**Prime**	2	2	2	2	2	0	0	2	2	2	2	2	0
1923	**Prime**	1	1	1	1	1	0	0	1	1	1	1	1	0
1924	**Prime**	1	1	1	1	1	0	0	1	1	1	1	1	0
1925	**Netflix**	1	1	1	0	0	0	1	1	1	1	1	1	1
...	**...**	...	...	...	...	...	...	...	...	...	...	...	...	...
2019	**Netflix**	1030	1030	1030	629	917	913	1030	1030	1030	1030	1030	1030	1030
	Prime	929	929	929	691	735	111	12	899	929	929	929	929	12
2020	**Disney+**	114	114	114	53	88	90	114	114	114	114	114	114	114
	Netflix	953	953	953	548	827	852	953	953	953	953	953	953	953
	Prime	962	962	962	705	838	91	12	923	962	962	962	962	12

261 rows × 16 columns

In [45]:
```
print_details(df_final)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17766 entries, 0 to 17765
Data columns (total 18 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   show_id         17766 non-null  object
 1   type            17766 non-null  object
 2   title           17766 non-null  object
 3   director        13265 non-null  object
 4   cast            15854 non-null  object
 5   country         9406 non-null   object
 6   date_added      9629 non-null   datetime64[ns]
 7   release_year    17766 non-null  int64
 8   rating          17441 non-null  object
 9   duration        17766 non-null  object
 10  listed_in       17766 non-null  object
 11  description     17766 non-null  object
 12  num_artist      17766 non-null  int64
 13  year            9629 non-null   float64
 14  month           9629 non-null   float64
 15  day             9629 non-null   float64
 16  duration_range  17766 non-null  object
 17  data_source     17766 non-null  object
dtypes: datetime64[ns](1), float64(3), int64(2), object(12)
memory usage: 2.4+ MB
None
**************************************************
       release_year    num_artist         year        month          day
count  17766.000000  17766.000000  9629.000000  9629.000000  9629.000000
mean    2009.269278      5.785039  2018.860733     7.046318    12.362654
std       16.467833      4.971170     1.476173     3.522221     9.553479
min     1920.000000      0.000000  2008.000000     1.000000     1.000000
25%     2008.000000      2.000000  2018.000000     4.000000     2.000000
50%     2016.000000      5.000000  2019.000000     7.000000    12.000000
75%     2018.000000      8.000000  2020.000000    10.000000    20.000000
max     2020.000000     76.000000  2021.000000    12.000000    31.000000
**************************************************
show_id              0
num_artist           0
description          0
listed_in            0
duration             0
duration_range       0
release_year         0
data_source          0
title                0
type                 0
rating             325
cast              1912
director          4501
year              8137
month             8137
day               8137
date_added        8137
country           8360
dtype: int64
```

Steps After Data cleaning and Tableau analisis:

De acuerdo al análisis expuesto en Tableau, haremos un sistema de recomendación de series entre las 3 fuentes

- De acuerdo a la descripción

Cosas por hacer

- eliminar columnas; director , cast, num_artis, country, date_added(y sus divisiones) mientras analizamos como llenarlas

In [46]:
```
df = df_final.dropna(how = 'any', subset =['cast','day','month','year','director'])
```

In [47]:
```
df.isna().sum()
```

Out[47]:
```
show_id            0
type               0
title              0
director           0
cast               0
country          298
date_added         0
release_year       0
rating            17
duration           0
listed_in          0
description        0
num_artist         0
year               0
month              0
day                0
duration_range     0
data_source        0
dtype: int64
```

Ilenar con la moda country y rating

In [48]:
```
df.loc[:,'country'] = df.loc[:,'country'].fillna(df['country'].mode()[0])
df.loc[:,'rating'] = df.loc[:,'rating'].fillna(df['rating'].mode()[0])
```

```
C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:966: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
```

In [49]:
```
df.shape
```

Out[49]:
```
(6293, 18)
```

In [50]:
```
df.groupby('data_source').count()
```

Out[50]:

data_source	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description	num_artist	year
Disney+	833	833	833	833	833	833	833	833	833	833	833	833	833	833
Netflix	5444	5444	5444	5444	5444	5444	5444	5444	5444	5444	5444	5444	5444	5444
Prime	16	16	16	16	16	16	16	16	16	16	16	16	16	16

Aquí sí vamos a usar Disney+ como producto recomendado, aunque en Tableau no se analizó su crecimiento de producciones creadas

In [51]:
```
category_list = list(df['listed_in'].apply(lambda x: x.split(',')[0]).unique())
print(sorted(category_list))
```

```
['Action', 'Action & Adventure', 'Action-Adventure', 'Adventure', 'Animals & Nature', 'Animation', 'Anime Features', 'Anime Series', 'Anthology', 'Biographical', 'British TV Shows', 'Buddy', 'Children & Family Movies', 'Classic & Cult TV', 'Classic Movies', 'Comedies', 'Comedy', 'Coming of Age', 'Concert Film', 'Crime', 'Crime TV Shows', 'Cult Movies', 'Dance', 'Documentaries', 'Documentary', 'Docuseries', 'Drama', 'Dramas', 'Family', 'Fantasy', 'Horror', 'Horror Movies', 'Independent Movies', 'International Movies', 'International TV Shows', "Kids' TV", 'LGBTQ Movies', 'Movies', 'Music & Musicals', 'Musical', 'Reality TV', 'Romantic Movies', 'Sci-Fi & Fantasy', 'Sports', 'Stand-Up Comedy', 'Stand-Up Comedy & Talk Shows', 'TV Action & Adventure', 'TV Comedies', 'TV Horror', 'TV Shows', 'Thrillers']
```

Same values for:

- Action & Adventure', 'Action-Adventure' -> Action & Adventure
- 'Anime Features', 'Anime Series' -> Anime Series
- 'Crime', 'Crime TV Shows' -> Crime
- 'Comedy' , Comedies', 'TV Comedies' -> Comedy
- 'Drama', 'Dramas' -> Drama
- 'Music & Musicals', 'Musical' -> Music & Musicals
- 'Stand-Up Comedy', 'Stand-Up Comedy & Talk Shows' -> Stand-Up Comedy & Talk Shows

In [52]:
```
df.loc[:,'category'] = df['listed_in'].apply(lambda x: x.split(',')[0])
```

```
C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:845: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:966: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
```

In [53]:
```
clean_cat_dict = { 'Action & Adventure': 'Action & Adventure',
 'Action-Adventure' : 'Action & Adventure',
 'Anime Features': 'Anime Series',
 'Anime Series': 'Anime Series',
 'Crime':'Crime',
 'Crime TV Shows':'Crime',
 'Comedy':'Comedy',
    'Comedies':'Comedy',
 'TV Comedies':'Comedy',
 'Drama':'Drama', 'Dramas':'Drama',
 'Music & Musicals':'Music & Musicals', 'Musical':'Music & Musicals',
 'Stand-Up Comedy & Talk Shows':'Stand-Up Comedy'
}
```

In [54]:
```
df.loc[:,'category'] = df['category'].map(clean_cat_dict).fillna(df['category'])
```

In [55]:
```
df1 = df.drop(columns = ['cast','day','month','year','num_artist','data_source','date_added','listed_in']).reset_inde
df1.isnull().sum()
```

Out[55]:
```
show_id          0
type             0
title            0
director         0
country          0
release_year     0
rating           0
duration         0
description      0
duration_range   0
category         0
dtype: int64
```

Recommendation system:

In [56]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
```

In [57]:
```
tfidf = TfidfVectorizer(stop_words = 'english')
```

In [58]:
```
features = ['title','director','rating','description','duration_range','category']
filters = df1[features]
```

In [59]:
```
def format_column(x):
    return str.lower(x.replace(" ",""))
def create_soup(x): #
    return  x['title'] + ' '+ x['director'] + ' ' +x['rating'] + ' ' + x['description'] + ' ' + x['duration_range'] +
```

In [60]:
```
for f in features:
    filters.loc[:,f] = filters.loc[:,f].apply(format_column)
```

```
C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:1048: SettingWithCopyWarnin
g:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returnin
g-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versu
s-a-copy)
```

In [61]:
```
filters.loc[:,'soup'] = filters.apply(create_soup, axis = 1)
```

```
C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:845: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returnin
g-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versu
s-a-copy)

C:\Users\Csanch05\AppData\Roaming\Python\Python37\site-packages\pandas\core\indexing.py:966: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returnin
g-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versu
s-a-copy)
```

In [62]:
```
tfidf_vector = tfidf.fit_transform(filters['soup'])
coseno_similitud = linear_kernel(tfidf_vector,tfidf_vector)
series = pd.Series(filters.index, index = filters['title'])
```

In [63]:
```
def get_recomendation(title,coseno_similitud  ):
    title = title.lower().replace(' ','')
    idx = series[title]
    sim_scores = list(enumerate(coseno_similitud[idx]))
    sim_scores = sorted(sim_scores, key = lambda x: x[1], reverse = True)
    top_2 = [i[0] for i in sim_scores[1:3]]
    others = [i[0] for i in sim_scores[3:]]
    top_2_df = df1.iloc[top_2]
    aux_df = df1.iloc[others]
    aux_df = aux_df.loc[aux_df['type'] == 'TV Show']
    return pd.concat([top_2_df,aux_df.head(8)], ignore_index = True)
```

Results:

In [64]:
```
get_recomendation('Green Lantern',coseno_similitud  )
```

Out[64]:

	show_id	type	title	director	country	release_year	rating	duration	description	duration_range	category
0	6867	Movie	GoldenEye	Martin Campbell	United Kingdom, United States	1995	PG-13	130	Pierce Brosnan takes his first turn as debonai...	D	Action & Adventure
1	8414	Movie	The Mask of Zorro	Martin Campbell	United States, Germany, Mexico	1998	PG-13	138	An aging Zorro passes the torch to a young suc...	D	Action & Adventure
2	6811	TV Show	Frozen Planet	Alastair Fothergill	United Kingdom, United States, Spain, Germany,...	2011	TV-PG	1	Go on a journey through the Arctic and Antarct...	E	British TV Shows
3	4263	TV Show	Watership Down	Noam Murro	United Kingdom, Ireland, United States	2018	TV-PG	1	A warren of rabbits battles many threats on th...	E	British TV Shows
4	7410	TV Show	Mars	Everardo Gout	United States	2018	TV-PG	2	Fact meets fiction in this docudrama chronicli...	E	Docuseries
5	2406	TV Show	DC's Legends of Tomorrow	Rob Seidenglanz	United States	2020	TV-14	5	A mysterious "time master" from the future uni...	F	TV Action & Adventure
6	7749	TV Show	Planet Earth: The Complete Collection	Alastair Fothergill	United Kingdom	2006	TV-PG	1	This landmark series transports nature lovers ...	E	British TV Shows
7	5112	TV Show	Myths & Monsters	Daniel Kontur	United Kingdom	2017	TV-PG	1	This documentary series takes us to the mythic...	E	British TV Shows
8	7919	TV Show	Sadqay Tumhare	Ehtesham Uddin	Pakistan	2014	TV-PG	1	An arranged engagement between a village girl ...	E	International TV Shows
9	7865	TV Show	Revolting Rhymes	Jani Lachauer, Jakob Schuh	United Kingdom	2017	TV-PG	1	Popular fairy tales take on a darkly comic edg...	E	British TV Shows

In [65]:
```
get_recomendation('Y Tu Mamá También',coseno_similitud  )
```

Out[65]:

	show_id	type	title	director	country	release_year	rating	duration	description	duration_range	category
0	8051	Movie	Solo Con Tu Pareja	Alfonso Cuarón	Mexico	1991	NR	94	A yuppie playboy looks for a quick death after...	C	Comedy
1	4307	Movie	ROMA	Alfonso Cuarón	Mexico, United States	2018	R	135	Director Alfonso Cuarón delivers a vivid, emot...	D	Drama
2	48	TV Show	The Smart Money Woman	Bunmi Ajakaiye	United States	2020	TV-MA	1	Five glamorous millennials strive for success ...	E	International TV Shows
3	149	TV Show	HQ Barbers	Gerhard Mostert	United States	2020	TV-14	1	When a family run barber shop in the heart of ...	E	TV Shows
4	317	TV Show	Office Girls	Hsu Fu-chun	Taiwan	2011	TV-14	1	A department store mogul has his son work inco...	E	International TV Shows
5	588	TV Show	Quarantine Tales	Sidharta Tata, Aco Tenriyagelli, Dian Sastrowa...	United States	2020	TV-MA	1	Traversing genres, five separate stories offer...	E	International TV Shows
6	622	TV Show	Legend of Exorcism	Shen Leping	United States	2020	TV-14	1	After leaving Yaojin Palace, Kong Hongjun arri...	E	Anime Series
7	667	TV Show	Gameboys Level-Up Edition	Ivan Andrew Payawal	United States	2020	TV-14	1	In this recut of the popular web series, live-...	E	International TV Shows
8	677	TV Show	Riverdale	Rob Seidenglanz	United States	2019	TV-14	4	While navigating the troubled waters of sex, r...	F	Crime
9	682	TV Show	They've Gotta Have Us	Simon Frederick	United Kingdom	2018	TV-MA	1	Powered by candid recollections from esteemed ...	E	British TV Shows

Next Steps:

- Try to add to soup country, and split it
- Model must be recommend cheapest series
- Get avg time subscription for user to analyze freq deploy new series

Q&A

In [ ]:
```

```

In [ ]:
```

```

More Info

Author

Carlo Marmolejo (mailto: carlo.marmolejo.mx@hotmail.com)

Change Log

Date (YYYY-MM-DD)	Time (hh:mm)	Version	Changed By	Change Description
2020-05-11	08:23	5.1	Carlo	Clean notebook, add index and subtitles, export as pdf, add next steps
2020-05-11	08:06	5.0	Carlo	Clean notebook, and test with 2 movies
2020-05-11	07:58	4.5	Carlo	Recomendation workign, change en function split top2 + next top 8 series
2020-05-11	07:01	4.4	Carlo	Edit predict function
2020-05-10	22:21	4.3	Carlo	Add functions for Model tested and get better for next
2020-05-10	13:06	4.2	Carlo	Pause data cleaning, we clean category for df
2020-05-10	12:12	4.1	Carlo	Análisis para modelo
2020-05-09	14:21	4.0	Carlo	Add disney + as data source, cleaning process and concat netflix + prime + disney plus
2020-05-09	13:24	3.3	Carlo	split yers column to quarter and year
2020-05-09	12:53	3.2	Carlo	new Netflix revenue datasource
2020-05-08	20:07	3.1	Carlo	new columns for prime, we will take from netflix last analisis
2020-05-08	19:58	3.1	Carlo	join tables to get Netflix + Prime table, also change codelines to foo(s)
2020-05-08	19:11	3.0	Carlo	Change PrimeVideo as data source and add new one (similar as netflix)
2020-05-08	14:52	2.1	Carlo	Cleaning PrimeVideo and clean netflix to concat them
2020-05-08	13:53	2.0	Carlo	Add Prime video as datasource from kaggle and read main info
2020-05-08	11:10	1.2	Carlo	Drop data from release year 2021, it's incomplted
2020-05-08	08:40	1.1	Carlo	Explore data , create new columns for tableau
2020-05-06	19:20	1.0	Carlo	Get datascource from kaggle and read stuff

Carlo Marmolejo

Click for more info