Netflix Movies and TV shows analysis and visualization project

This is my final notebook for this project. For this project, I chose a dataset from kaggle.com called Netflix Movies and TV Shows (link to dataset: https://www.kaggle.com/shivamb/netflix-shows (https://www.kaggle.com/shivamb/netflix-shows)), which includes movies and TV shows available on Netflix as of 2019. The dataset is collected from Flixable which is a third-party Netflix search engine. I also used other datsets from google trends, which are the datasets of netflix trending scores of the US versus the world.

For this dataset, some of my questions are:

1. What are the countries that produce most movies and TV shows?
2. How many movies and TV shows are added into Netflix per year?
3. What is the average length of movie and tv show titles per year?
4. What is the distribution of the duration of movies and TV shows? Is there a relationship between duration and average title length?
5. What is the distribution of the age ratings for Netflix movies and TV shows?
6. What is the difference between netflix US trending scores and netflix worldwide trending scores from 2015 until present?

In [1]:
```
import pandas as pd
import altair as alt
import numpy as np
```

In [2]:
```
nf_show = pd.read_csv('netflix_shows.csv')
```

In [3]:
```
nf_show.shape
```

Out[3]:
```
(6234, 12)
```

In [4]:
```
nf_show.head()
```

Out[4]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	dura
0	81145628	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	9-Sep-19	2019	TV-PG	90
1	80117401	Movie	Jandino: Whatever it Takes	NaN	Jandino Asporaat	United Kingdom	9-Sep-16	2016	TV-MA	94
2	70234439	TV Show	Transformers Prime	NaN	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	8-Sep-18	2013	TV-Y7-FV	Sea
3	80058654	TV Show	Transformers: Robots in Disguise	NaN	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	8-Sep-18	2016	TV-Y7	Sea
4	80125979	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	8-Sep-17	2017	TV-14	99

In [5]:
```
nf_show.columns
#The columns mean:
#'show_id': id of a movie/TV show (it can be used to trace IMDB scores, etc.,)
#'type': there are only 2 categories - movie or TV show
#'title': name of the movie/TV show
#'director' and 'cast': names of the directors and the main casts of the movies and
 TV shows
#'country': names of the countries that produce each of the movie/TV show
#'date_added': day, month, and year a movie/TV show is added to Netflix
#'release_year': when the movie/TV show is originally released
#'rating': age rating
#'duration': the length of a movie (minutes)/ TV show (seasons)
#'listed_in': genres of a movie/TV show
#'description': the content brief of a movie/TV show
```

Out[5]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

In [6]:
```
nf_show = nf_show.drop(['show_id','description'],axis=1).fillna('Null')
nf_show.head()
#I dropped 2 columns called "show_id" and "description" because they aren't necessar
y for my analysis questions.
#I also noticed that there were several blank cells in the dataframe. Therefore, I u
sed the fillna method to fill in the cells. I chose to fill in with the string "Nul
l"
```

Out[6]:

	type	title	director	cast	country	date_added	release_year	rating	duration	liste
0	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	9-Sep-19	2019	TV-PG	90 min	Chil & Fa Mo Come
1	Movie	Jandino: Whatever it Takes	Null	Jandino Asporaat	United Kingdom	9-Sep-16	2016	TV-MA	94 min	Stan Cor
2	TV Show	Transformers Prime	Null	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	8-Sep-18	2013	TV-Y7-FV	1 Season	Kids
3	TV Show	Transformers: Robots in Disguise	Null	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	8-Sep-18	2016	TV-Y7	1 Season	Kids
4	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	8-Sep-17	2017	TV-14	99 min	Come

Here is my first question:

- What are the countries that produce most movies and TV shows?

In [7]:
```
from collections import Counter
country_data = nf_show['country']
country_count = pd.Series(Counter(','.join(country_data).replace(' ,',',').replace(
', ',',').split(',')),name = 'counts'
).sort_values(ascending=False)
country_count = country_count.drop('Null')
country_count_df = country_count.to_frame().reset_index()
country_count_df.shape
#Here I created a series with the list of unique countries and the counts of their a
ppearance in the "country" column of the dataframe. I used the Counter sub-class fro
m the Collections module in order to count hashable objects.
#I also sorted the series so that the counts are sorted in a descending order.
#I changed the series into a dataframe for later visualization.
```

Out[7]: (111, 2)

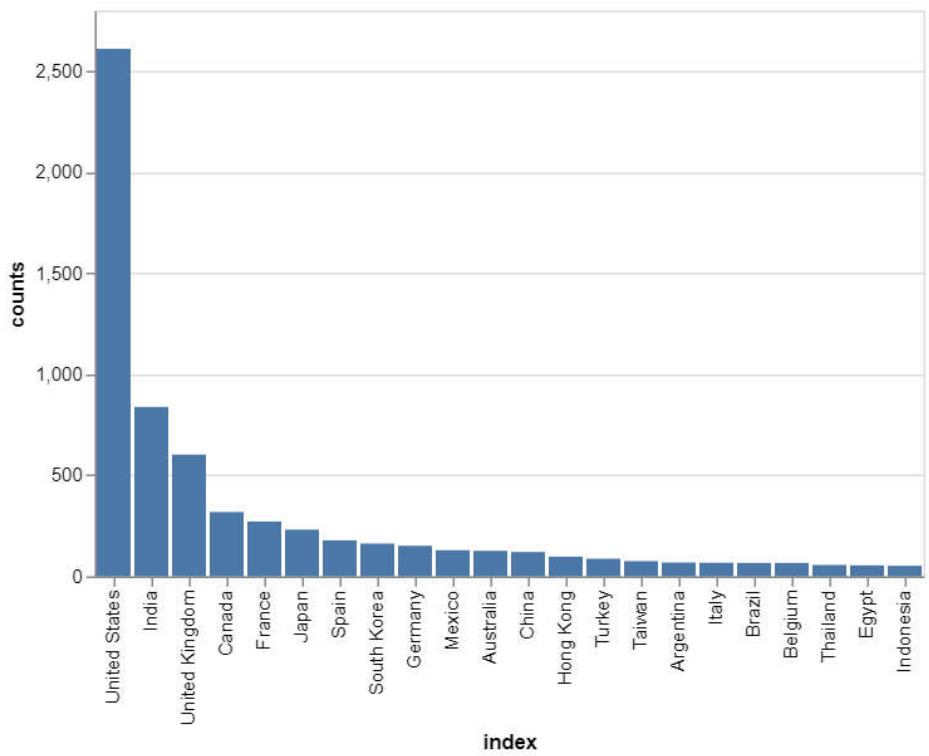
In [8]:
```
top_countries_df = country_count_df[country_count_df.counts >= 50]
top_countries_df.shape
#There are a lot of countries that are involved in making movies and shows in this d
ataset, but I'm only interested in the countries that porduce most movies and tv sho
ws.
#Therefore, I created a dataframe with only the countries that have the counts equal
or over 50.
```

Out[8]: (22, 2)

In [9]:
```
#I got 22 countries that appear more than 50 times in the dataset (23 includes the t
itle column).
```

In [10]:
```
chart_1 = alt.Chart(top_countries_df).mark_bar(
).encode(x= alt.X('index', sort = 'x'), y= 'counts',tooltip = 'counts'
).interactive()
chart_1
```

Out[10]:



The bar chart here visualizes the dataframe of the top 22 countries above. According to the chart, the United States produces most movies and tv shows. Coming in the second place is India, and the third place is the UK.

My second question is:

- How many movies and TV shows are added into Netflix per year?

In [11]:
```
alt.data_transformers.disable_max_rows()
```

Out[11]:
```
DataTransformerRegistry.enable('default')
```

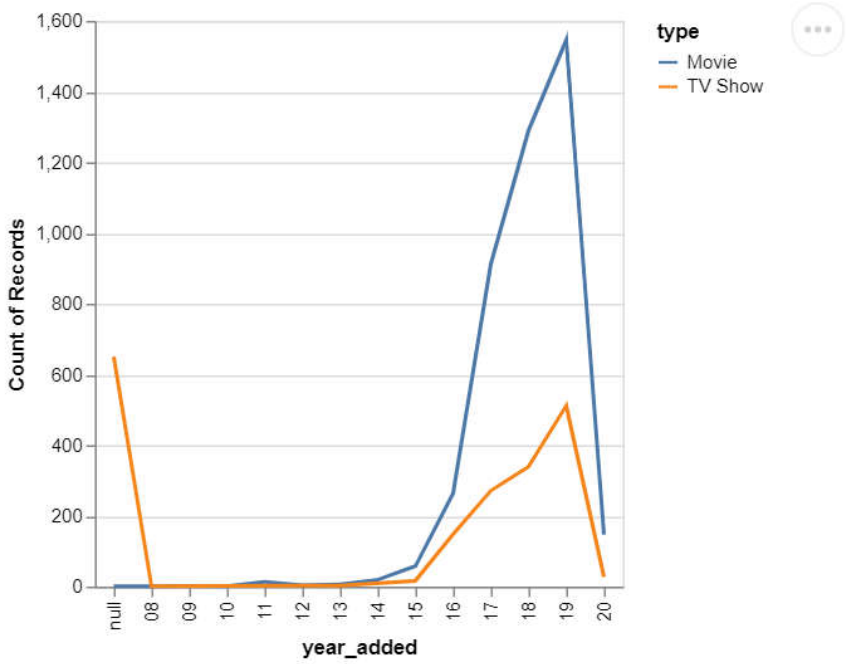
In [12]:
```
nf_show[['day_added','month_added','year_added']] = nf_show['date_added'].str.split(
"-", expand=True)
nf_show = nf_show.drop(['date_added','day_added','month_added'], axis = 1)
nf_show.head()

#For the original "date_added" column of the original dataframe, I split the date st
ring into three parts - day, month, and year.
#I'm only interested in the year the movies and TV shows are added, therefore I drop
ped the original date_added column and the columns for day and month added.
```

Out[12]:

	type	title	director	cast	country	release_year	rating	duration	listed_in	year_ad
0	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	2019	TV-PG	90 min	Children & Family Movies, Comedies	
1	Movie	Jandino: Whatever it Takes	Null	Jandino Asporaat	United Kingdom	2016	TV-MA	94 min	Stand-Up Comedy	
2	TV Show	Transformers Prime	Null	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	2013	TV-Y7-FV	1 Season	Kids' TV	
3	TV Show	Transformers: Robots in Disguise	Null	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	2016	TV-Y7	1 Season	Kids' TV	
4	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	2017	TV-14	99 min	Comedies	

In [13]:
```
chart_2 = alt.Chart(nf_show).mark_line(
).encode(x= 'year_added:O', y='count():Q', color = 'type:N',tooltip = 'count()'
).interactive()
chart_2
```

Out[13]:



According to the line graph, there weren't many movies and and TV shows added into Netflix before 2015.

After 2015, there was a soar in the number of movies and TV shows on Netflix.

However, the increase in movies tends to slow down gradually after mid-2016, while the increase in TV shows becomes more dramatic since 2018. It seems that Netflix is focusing more on TV shows than movies recently.

I ignored the numbers in 2020 because the year is not over, hence the results do not reflect anything.

My third question is:

- What is the average length of movie and TV show titles per year?

In [14]:
```
nf_show["title_length"]= nf_show["title"].str.len()
nf_show.head()
#I created a new column that counts the character length of each of the movie title.
```

Out[14]:

	type	title	director	cast	country	release_year	rating	duration	listed_in	year_ac
0	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	2019	TV-PG	90 min	Children & Family Movies, Comedies	
1	Movie	Jandino: Whatever it Takes	Null	Jandino Asporaat	United Kingdom	2016	TV-MA	94 min	Stand-Up Comedy	
2	TV Show	Transformers Prime	Null	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	2013	TV-Y7-FV	1 Season	Kids' TV	
3	TV Show	Transformers: Robots in Disguise	Null	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	2016	TV-Y7	1 Season	Kids' TV	
4	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	2017	TV-14	99 min	Comedies	

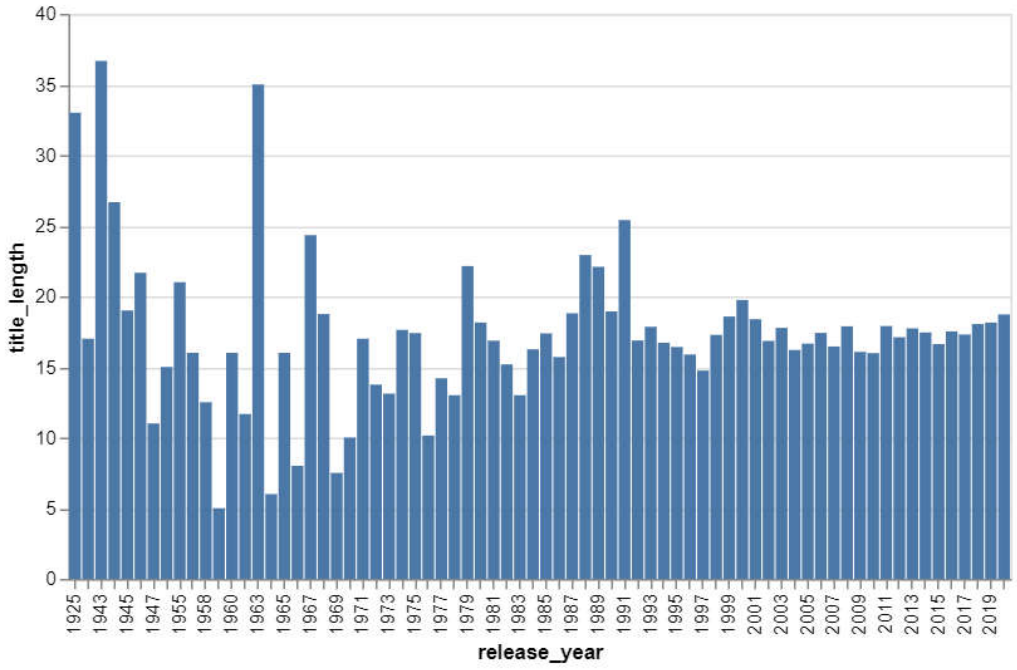
In [15]:
```
avg_len = nf_show.groupby('release_year')['title_length'].mean()
avg_len = avg_len.to_frame().reset_index()
avg_len.head()
#I used the split-apply-combine operation to calculate the average movie and TV show
title length (in characters) by each year
```

Out[15]:

	release_year	title_length
0	1925	33.000000
1	1942	17.000000
2	1943	36.666667
3	1944	26.666667
4	1945	19.000000

In [16]:
```
chart_3 = alt.Chart(avg_len, width = 500).mark_bar(
).encode(x= 'release_year:O', y='title_length:Q', tooltip = 'title_length'
).interactive()
chart_3
```

Out[16]:



According to the graph, most of the movies and TV shows have the title length less than 20 characters. There are some older movies and shows that have more than 30 characters in their titles.

My fourth question is:

- What is the distribution of the duration of movies and TV shows?

For this question, I decided to created two dataframes that include either only movies or only TV shows.

In [17]:
```
movie_df = nf_show[nf_show['type'] == 'Movie']
movie_df.shape
```

Out[17]: (4265, 11)

In [19]:
```
movie_df[['duration(min)','Minutes']] = movie_df['duration'].str.split(" ", expand=True)
movie_df.head()
#Since I want only integers for the movie duration, I created new columns called duration(min) and minutes that contain the split strings from the original "duration" column.
```

Out[19]:

	type	title	director	cast	country	release_year	rating	duration	listed_in	year_a
0	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	2019	TV-PG	90 min	Children & Family Movies, Comedies	
1	Movie	Jandino: Whatever it Takes	Null	Jandino Asporaat	United Kingdom	2016	TV-MA	94 min	Stand-Up Comedy	
4	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	2017	TV-14	99 min	Comedies	
6	Movie	Automata	Gabe Ibáñez	Antonio Banderas, Dylan McDermott, Melanie Gri...	Bulgaria, United States, Spain, Canada	2014	R	110 min	International Movies, Sci-Fi & Fantasy, Thrillers	
7	Movie	Fabrizio Copano: Solo pienso en mi	Rodrigo Toro, Francisco Schultz	Fabrizio Copano	Chile	2017	TV-MA	60 min	Stand-Up Comedy	

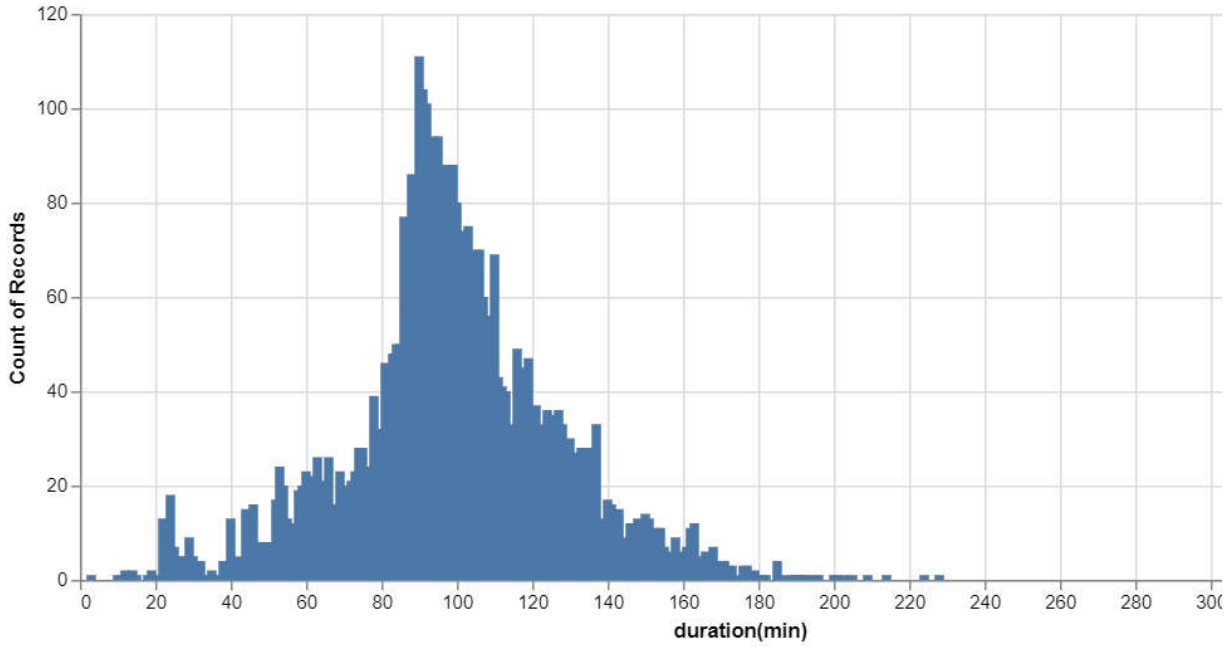
In [20]:
```
movie_df = movie_df.drop(['duration','Minutes'], axis = 1)
movie_df.head()
convert_dict = {'duration(min)': int}
movie_df = movie_df.astype(convert_dict)
movie_df.head()
#I dropped the two unecessary columns which are the original "duration" column and t
he "Minutes" column.
#The numbers in the duration(min) column still have the string format, therefore I c
onverted the strings to the int type.
```

Out[20]:

	type	title	director	cast	country	release_year	rating	listed_in	year_added	titl
0	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole...	United States, India, South Korea, China	2019	TV-PG	Children & Family Movies, Comedies	19	
1	Movie	Jandino: Whatever it Takes	Null	Jandino Asporaat	United Kingdom	2016	TV-MA	Stand-Up Comedy	16	
4	Movie	#realityhigh	Fernando Lebrija	Nesta Cooper, Kate Walsh, John Michael Higgins...	United States	2017	TV-14	Comedies	17	
6	Movie	Automata	Gabe Ibáñez	Antonio Banderas, Dylan McDermott, Melanie Gri...	Bulgaria, United States, Spain, Canada	2014	R	International Movies, Sci-Fi & Fantasy, Thrillers	17	
7	Movie	Fabrizio Copano: Solo pienso en mi	Rodrigo Toro, Francisco Schultz	Fabrizio Copano	Chile	2017	TV-MA	Stand-Up Comedy	17	

In [36]:
```
chart_4 = alt.Chart(movie_df,width=700).mark_bar(
).encode(x= 'duration(min):Q', y= 'count()', tooltip = 'count()'
).interactive()
chart_4
```

Out[36]:



According to the histogram, the duration of movies throughout the years have a normal distribution that centers at around 90 minutes.

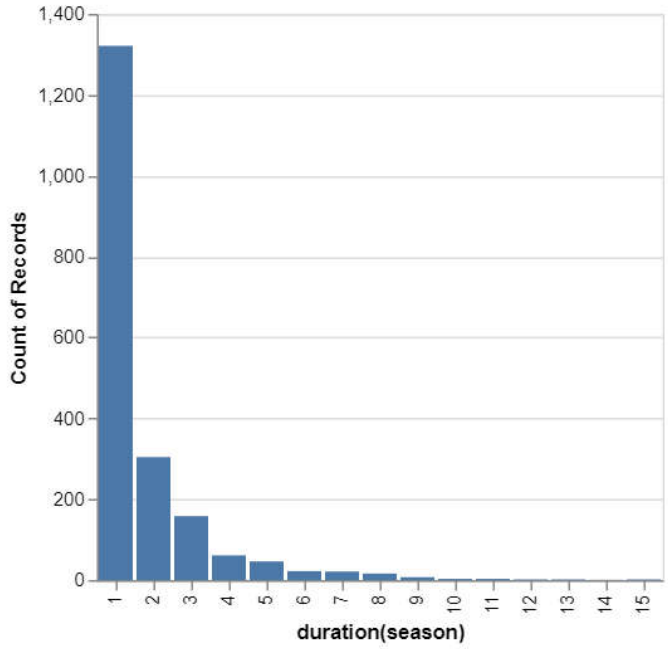
In [22]:
```
tv_show_df = nf_show[nf_show['type'] == 'TV Show']
tv_show_df[['duration(season)','season']] = tv_show_df['duration'].str.split(" ", ex
pand=True)
tv_show_df = tv_show_df.drop(['duration','season'], axis=1)
convert_dict1 = {'duration(season)':int}
tv_show_df = tv_show_df.astype(convert_dict1)
tv_show_df.head()

#I also did the same for the TV show dataframe. I only kept the numbers in a column
 called "duration(season)", and converted the strings to integers.
```

Out[22]:

	type	title	director	cast	country	release_year	rating	listed_in	year_added
2	TV Show	Transformers Prime	Null	Peter Cullen, Sumalee Montano, Frank Welker, J...	United States	2013	TV-Y7-FV	Kids' TV	18
3	TV Show	Transformers: Robots in Disguise	Null	Will Friedle, Darren Criss, Constance Zimmer, ...	United States	2016	TV-Y7	Kids' TV	18
5	TV Show	Apaches	Null	Alberto Ammann, Eloy Azorín, Verónica Echegui,...	Spain	2016	TV-MA	Crime TV Shows, International TV Shows, Spanis...	17
8	TV Show	Fire Chasers	Null	Null	United States	2017	TV-MA	Docuseries, Science & Nature TV	17
26	TV Show	Castle of Stars	Null	Chaiyapol Pupart, Jintanutda Lummakanon, Worra...	Null	2015	TV-14	International TV Shows, Romantic TV Shows, TV ...	18

In [29]:
```
chart_5 = alt.Chart(tv_show_df).mark_bar(
).encode(x= 'duration(season):N', y= 'count()', tooltip = 'count()'
).interactive()
chart_5
```

Out[29]:



According to the histogram, the distribution of the TV shows throughout the years is skewed to the right. Most of the shows only have 1 season.

Follow up question: Is there a relationship between average duration per year and average title length per year?

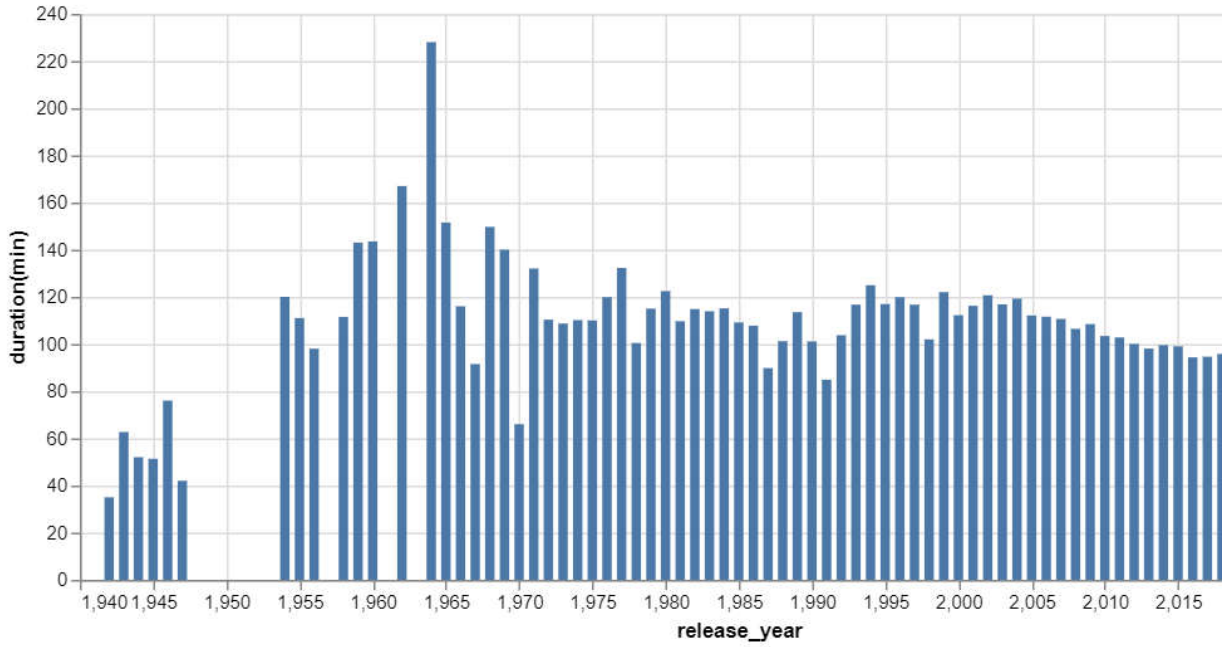
In [37]:
```
avg_movie_duration = movie_df.groupby('release_year').mean()
avg_movie_duration = avg_movie_duration.reset_index()
avg_movie_duration.head()
```

Out[37]:

	release_year	title_length	duration(min)
0	1942	17.000000	35.000000
1	1943	36.666667	62.666667
2	1944	26.666667	52.000000
3	1945	19.000000	51.333333
4	1946	15.000000	76.000000

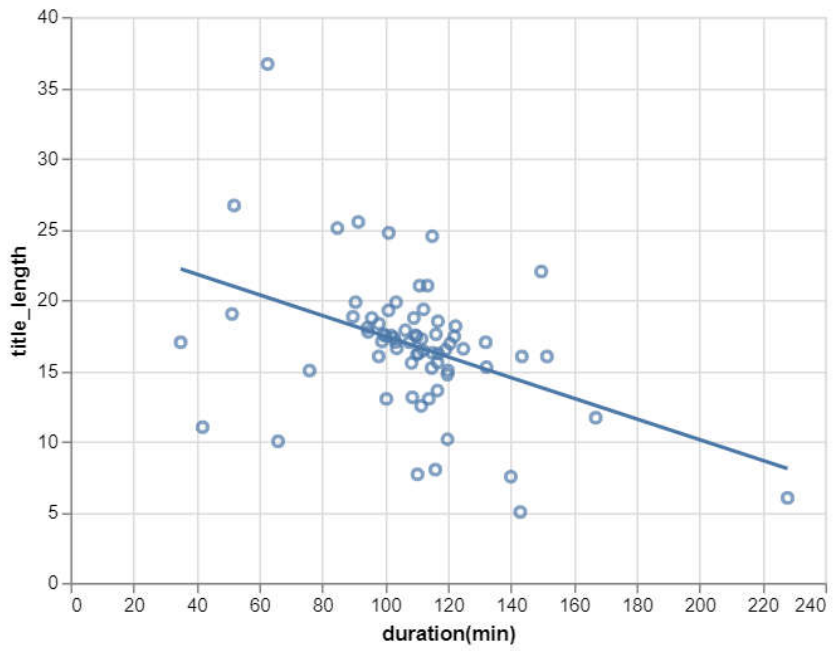
In [38]:
```
chart_9  =alt.Chart(avg_movie_duration, width  = 700).mark_bar(
).encode(x= 'release_year', y= 'duration(min)', tooltip = 'duration(min)'
).interactive()
chart_9
```

Out[38]:



In [39]:
```
chart_6 = alt.Chart(avg_movie_duration).mark_point(
).encode(x= 'duration(min)', y= 'title_length')
chart_6 + chart_6.transform_regression('duration(min)', 'title_length').mark_line()
```

Out[39]:



According to the graph, there seems to be a weak negative relationship between average movie duration and average movie title length.

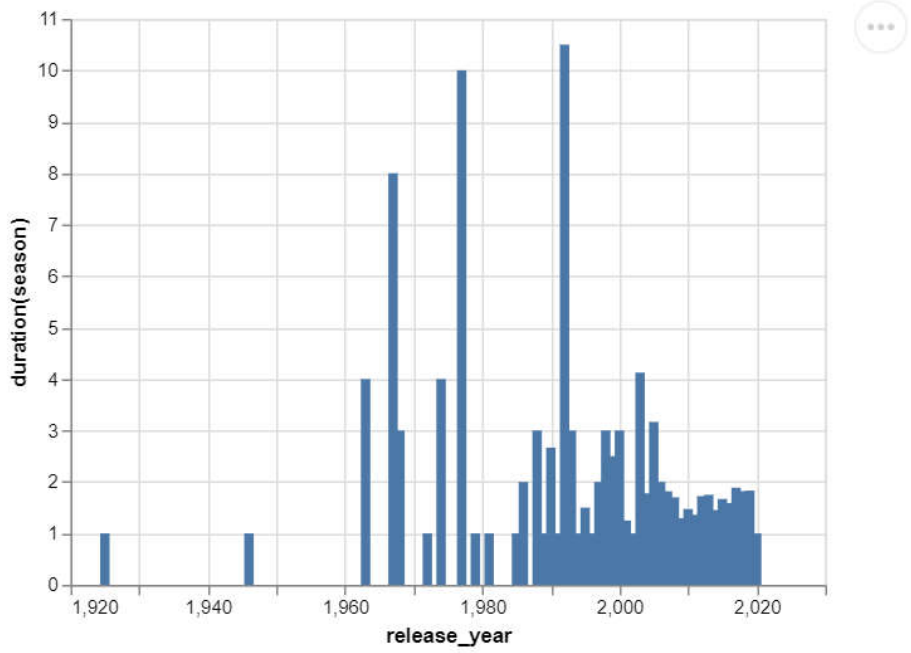
In [40]:
```
avg_tvshow_duration = tv_show_df.groupby('release_year').mean()
avg_tvshow_duration = avg_tvshow_duration.reset_index()
avg_tvshow_duration.head()
```

Out[40]:

	release_year	title_length	duration(season)
0	1925	33.0	1.0
1	1946	35.0	1.0
2	1963	35.0	4.0
3	1967	22.0	8.0
4	1968	9.0	3.0

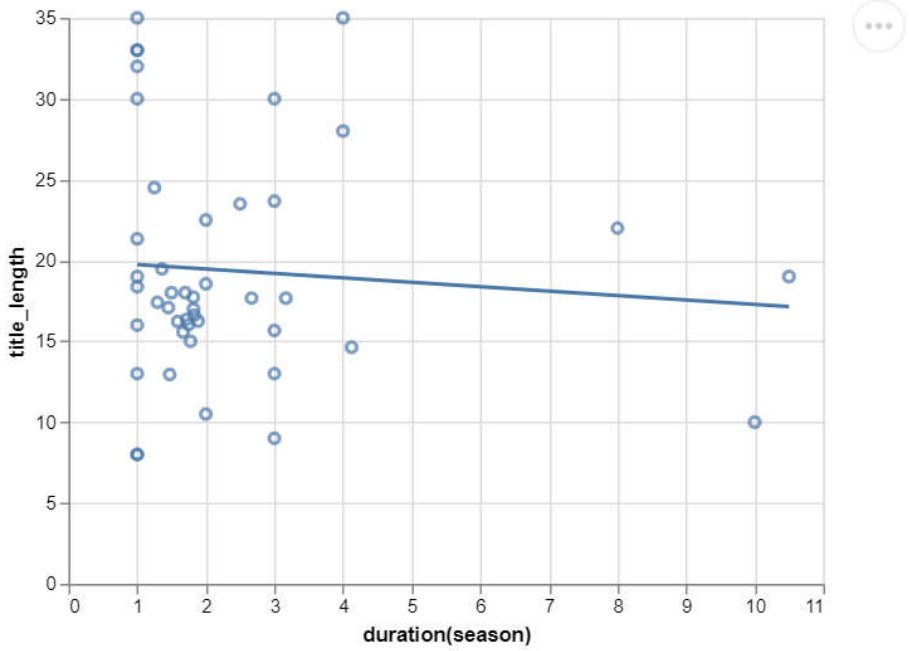
In [41]:
```
chart_10 = alt.Chart(avg_tvshow_duration).mark_bar(
).encode(x= 'release_year' , y= 'duration(season)',tooltip = 'duration(season)').int
eractive()
chart_10
```

Out[41]:



In [42]:
```
chart_7 = alt.Chart(avg_tvshow_duration).mark_point(
).encode(x= 'duration(season)', y= 'title_length')
chart_7 + chart_7.transform_regression('duration(season)','title_length').mark_line
()
```

Out[42]:



There seems to be no relationship between average TV show duration and average TV show title length.

My fifth question is:

- What is the distribution of the age ratings for Netflix movies and TV shows?

In [43]:
```
rating_data_1 = movie_df['rating']
rating_count = pd.Series(Counter(','.join(rating_data_1).replace(' ,',',').replace(
', ',',').split(','))
,name = 'movie_count').sort_values(ascending=False)
rating_count.drop(['Null'], axis=0, inplace=True)
rating_count1_df = rating_count.to_frame().reset_index()
rating_count1_df.head()

#Since I want to know how many times each of the ratings appear in the movie-only da
taframe, I used the counter sub-class to count for the occurrence of uniques age rat
ings.
```

Out[43]:

	index	movie_count
0	TV-MA	1348
1	TV-14	1038
2	R	506
3	TV-PG	432
4	PG-13	286

In [44]:
```
rating_data_2 = tv_show_df['rating']
rating_count = pd.Series(Counter(','.join(rating_data_2).replace(' ,',',').replace(
', ',',').split(','))
,name = 'tv_count').sort_values(ascending=False)
rating_count.drop(['Null'], axis=0, inplace=True)
rating_count2_df = rating_count.to_frame().reset_index()
rating_count2_df.head()
#I did the same thing to calculate the occurrence of unique age ratings in the TV sh
ow-only dataframe.
```

Out[44]:

	index	tv_count
0	TV-MA	679
1	TV-14	660
2	TV-PG	269
3	TV-Y	102
4	TV-Y7	100

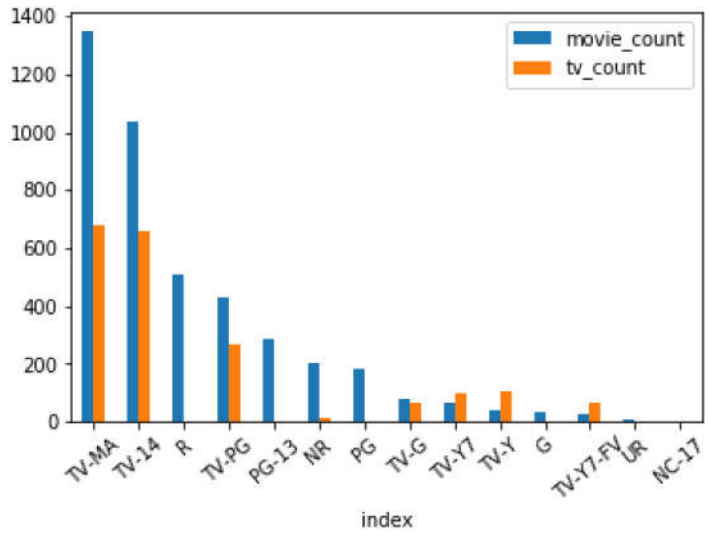
In [45]:
```
rating_df = pd.merge(rating_count1_df, rating_count2_df,how='left')
rating_df= rating_df.fillna(0)
rating_df = rating_df.set_index('index')
rating_df.head()

#Since I want to compare the distribution of the age ratings for movies and TV shows
side-by-side, I merged two dataframes with the index as the age ratings.
```

Out[45]:

index	movie_count	tv_count
TV-MA	1348	679.0
TV-14	1038	660.0
R	506	2.0
TV-PG	432	269.0
PG-13	286	0.0

In [47]:
```
chart_8 = rating_df.plot.bar(rot=40)
chart_8
```

Out[47]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x176d8444288>
```



Both the movies and TV shows are more likely to be rated TV-MA and TV-14, which are for mature audiences and teenagers being at least 14 years old. We can see that Netflix seems to be a streaming app that are more dedicated to teenagers and adults rather than children.

My sixth question is:

- What is the difference between netflix US trending scores and netflix worldwide trending scores from 2015 until present?

In [53]:
```
worldwide_trend = pd.read_csv('worldwide trend.csv')
worldwide_trend.head()
#I imported another csv file called "worldwide trend", which shows the popularity sc
ores of Netflix worldwide from 2015 to present.
```

Out[53]:

	Category: All categories	Unnamed: 1
0	NaN	NaN
1	Week	netflix: (Worldwide)
2	3/5/2015	27
3	10/5/2015	28
4	17/5/2015	27

In [54]:
```
worldwide_trend = worldwide_trend.dropna(how = 'any')
worldwide_trend = worldwide_trend.drop(worldwide_trend.index[0]
).rename(columns = {"Category: All categories":"Week","Unnamed: 1": "World Score"})
worldwide_trend.head()

#I dropped the rows with NaN values by the dropna function and how='any' means that
 the fuction would drop any rows that contain NaN values.
#I reset the column indexes to "Week" and "World Score"
#The weeks are from May 2015 to the present, and the scores range from 0 to 100, wit
h 100 indicating the highest popularity scores Netflix can achieve, and 0 means non
e.
```

Out[54]:

	Week	World Score
2	3/5/2015	27
3	10/5/2015	28
4	17/5/2015	27
5	24/5/2015	30
6	31/5/2015	31

In [55]:
```
worldwide_trend['Week'] = pd.to_datetime(worldwide_trend.Week,errors = 'coerce')
convert_dict3 = {'World Score':int}
worldwide_trend = worldwide_trend.astype(convert_dict3)
worldwide_trend.head()

#The format of the "Week" column is string. I converted strings to datetime type bec
ause I realized that altair would not graph the date-time entries chronologically if
I don't change them into datetime type.
#The format of the "World Score" column is also string, therefore I had to convert t
he strings into integers for later visualization.
```

Out[55]:

	Week	World Score
2	2015-03-05	27
3	2015-10-05	28
4	2015-05-17	27
5	2015-05-24	30
6	2015-05-31	31

In [76]:
```
us_trend = pd.read_csv('US trend.csv')
us_trend.head()
#To compare to the worldwide Netflix trending score, I imported another csv file cal
led "US trend", which shows the popularity of Netflix in the US from May 2015 to the
present.
```

Out[76]:

	Category: All categories	Unnamed: 1
0	NaN	NaN
1	Week	netflix: (United States)
2	3/5/2015	50
3	10/5/2015	54
4	17/5/2015	55

In [81]:
```
us_trend = us_trend.dropna(how='any')
us_trend = us_trend.drop(us_trend.index[0]
).rename(columns={'Category: All categories': 'Week',"Unnamed: 1":'US Score'})
us_trend['Week'] = pd.to_datetime(us_trend['Week'],errors = 'coerce')
convert_dict4 = {'US Score':int}
us_trend = us_trend.astype(convert_dict4)
us_trend.head()

#I did the same thing to this dataframe as I did to the worldwide trend dataframe, w
hich are dropping NaN values, changing the column indexes to "Week" and "US Score",
 and converting the strings to datetime and integer formats
```

Out[81]:

	Week	US Score
3	2015-10-05	54
4	2015-05-17	55
5	2015-05-24	60
6	2015-05-31	60
7	2015-07-06	63

In [82]:
```
compare_score= pd.merge(worldwide_trend,us_trend)
compare_score.head()

#To make a graph that can compare the trending scores of the world and the US, I mer
ged the two dataframes and called the new one "compare_score".
```

Out[82]:

	Week	World Score	US Score
0	2015-10-05	28	54
1	2015-05-17	27	55
2	2015-05-24	30	60
3	2015-05-31	31	60
4	2015-07-06	32	63

In [88]:
```
uk_trend = pd.read_csv('UK trend.csv')
uk_trend.head()
#To compare to the worldwide Netflix trending score, I imported another csv file cal
led "UK trend", which shows the popularity of Netflix in the UK from May 2015 to the
present.
```

Out[88]:

	Category: All categories
Week	netflix: (United Kingdom)
2015-05-03	45
2015-05-10	43
2015-05-17	43
2015-05-24	47

In [89]:
```
uk_trend = uk_trend.dropna(how='any')
uk_trend = uk_trend.drop(uk_trend.index[0]
).reset_index().rename(columns={'index': 'Week','Category: All categories':'UK Scor
e'})
uk_trend['Week'] = pd.to_datetime(uk_trend['Week'],errors = 'coerce')
convert_dict4 = {'UK Score':int}
uk_trend = uk_trend.astype(convert_dict4)
uk_trend.head()

#I did the same thing to this dataframe as I did to the worldwide trend dataframe, w
hich are dropping NaN values, changing the column indexes to "Week" and "US Score",
 and converting the strings to datetime and integer formats
```

Out[89]:

	Week	UK Score
0	2015-05-03	45
1	2015-05-10	43
2	2015-05-17	43
3	2015-05-24	47
4	2015-05-31	47

In [90]:
```
compare_score= pd.merge(compare_score,uk_trend)
compare_score.head()

#To make a graph that can compare the trending scores of the world, the US, and the
 UK, I merged the three dataframes and called the new one "compare_score".
```

Out[90]:

	Week	World Score	US Score	UK Score
0	2015-05-17	27	55	43
1	2015-05-24	30	60	47
2	2015-05-31	31	60	47
3	2015-06-14	33	64	53
4	2015-06-21	33	64	50

In [91]:
```
india_trend = pd.read_csv('India trend.csv')
india_trend.head()
#To compare to the worldwide Netflix trending score, I imported another csv file cal
led "India trend", which shows the popularity of Netflix in the India from May 2015
 to the present.
```

Out[91]:

	Category: All categories
Week	netflix: (India)
2015-05-03	2
2015-05-10	2
2015-05-17	2
2015-05-24	2

In [93]:
```
india_trend = india_trend.dropna(how='any')
india_trend = india_trend.drop(india_trend.index[0]
).reset_index().rename(columns={'index': 'Week','Category: All categories':'India Sc
ore'})
india_trend['Week'] = pd.to_datetime(india_trend['Week'],errors = 'coerce')
convert_dict4 = {'India Score':int}
india_trend = india_trend.astype(convert_dict4)
india_trend.head()

#I did the same thing to this dataframe as I did to the worldwide trend dataframe, w
hich are dropping NaN values, changing the column indexes to "Week" and "US Score",
 and converting the strings to datetime and integer formats
```

Out[93]:

	Week	India Score
0	2015-05-10	2
1	2015-05-17	2
2	2015-05-24	2
3	2015-05-31	2
4	2015-06-07	2

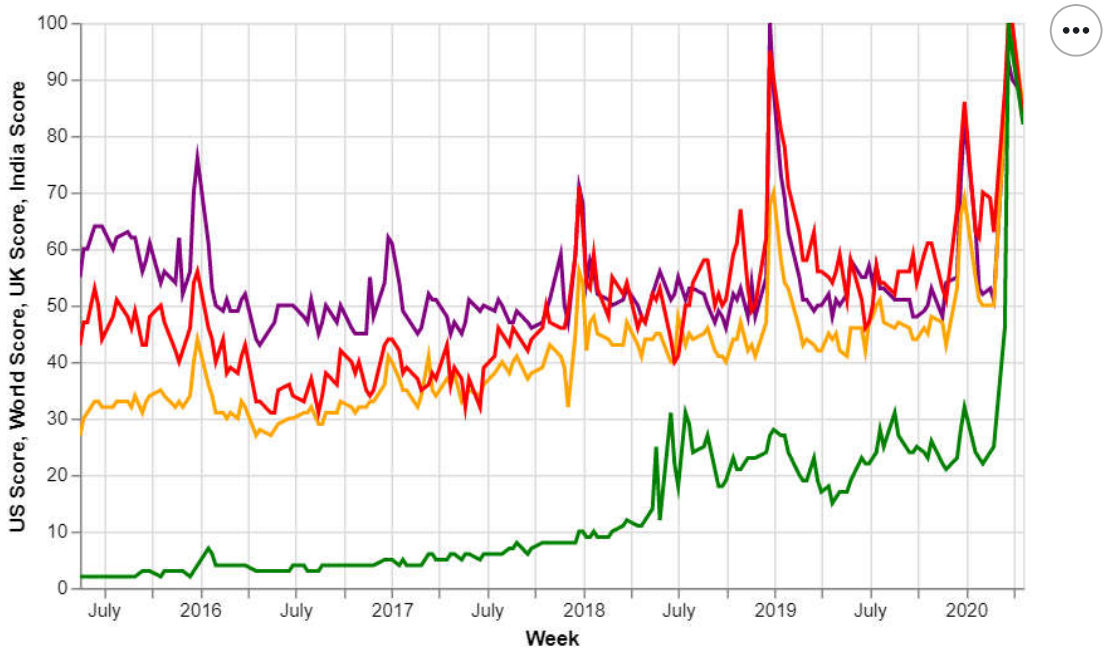
In [94]:
```
compare_score= pd.merge(compare_score,india_trend)
compare_score.head()

#To make a graph that can compare the trending scores of the world, the US, the UK,
 and India, I merged the three dataframes and called the new one "compare_score".
```

Out[94]:

	Week	World Score	US Score	UK Score	India Score
0	2015-05-17	27	55	43	2
1	2015-05-24	30	60	47	2
2	2015-05-31	31	60	47	2
3	2015-06-14	33	64	53	2
4	2015-06-21	33	64	50	2

In [96]:
```
base = alt.Chart(compare_score.reset_index(),width=500).mark_line().encode(x='Week')
alt.layer(
    base.mark_line(color='purple').encode(y='US Score'),
    base.mark_line(color='orange').encode(y='World Score'),
    base.mark_line(color='red').encode(y='UK Score'),
    base.mark_line(color='green').encode(y='India Score')
).interactive()
```

Out[96]:



The purple line represents US trending score, the red line represents UK trending scores, the green line represents India trending scores, and the orange line represents the world score.

All scores have an increasing trend, which indicates that Netflix is getting more popular over time.