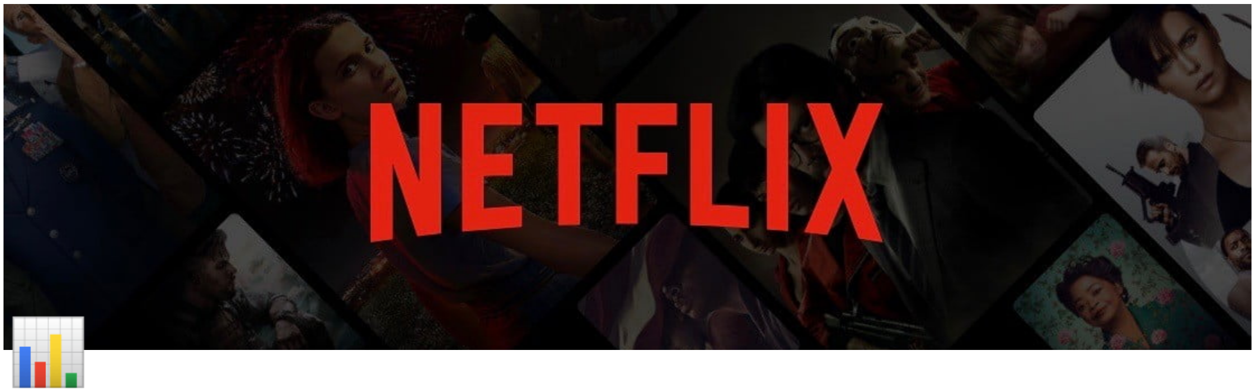


EDA of Netflix Dataset using Plotly

We are going to talk about some of the advanced and most used charts in Plotly while doing analysis. All you need to know is Plotly for visualization!

- Description of Dataset
- Data Exploration
- Data Cleaning
- Data visualization
- The questions which we are going to answer with the charts
    - Correlation between the features
    - Most watched shows on the Netflix
    - Distribution of Ratings
    - Which has the highest rating Tv show or Movies
    - Finding the best Month for releasing content
    - Highest watched genres on Netflix
    - Released movie over the years

Netflix is an application that keeps growing exponentially whole around the world and it is the most famous streaming platform.

> **Dataset** : https://www.kaggle.com/shivamb/netflix-shows

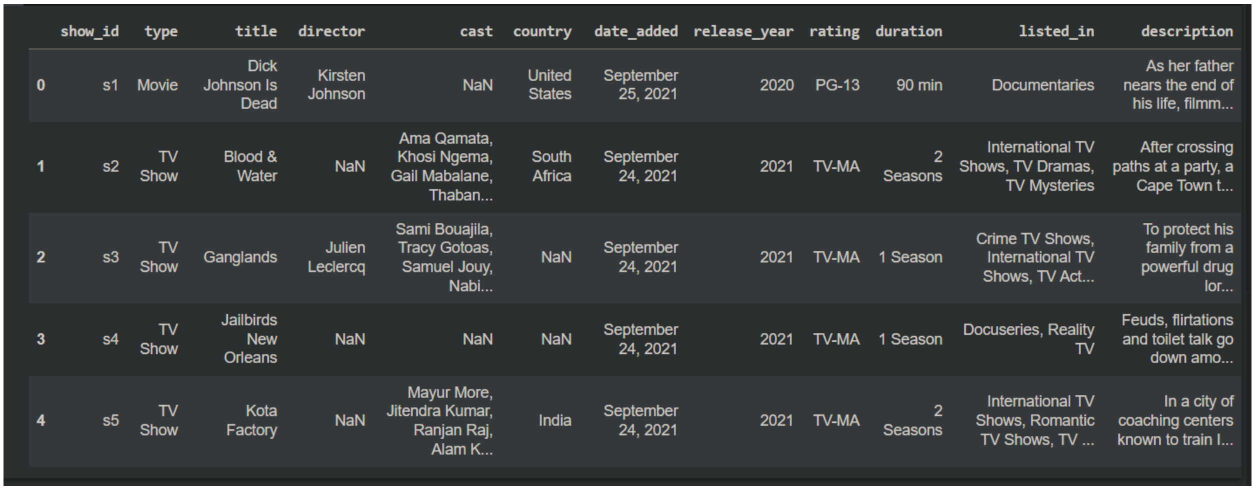
Let's create an EDA through this data with beautiful charts and visuals to gain some insights.

Importing Libraries

```
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.figure_factory as ff
```

Let's see the Netflix data:

```
netflix_data = pd.read_csv('/content/netflix_titles.csv')
netflix_data.head()
```



	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To protect his family from a powerful drug lor...
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down amo...
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a city of coaching centers known to train I...

Dataset

Description of Netflix Dataset

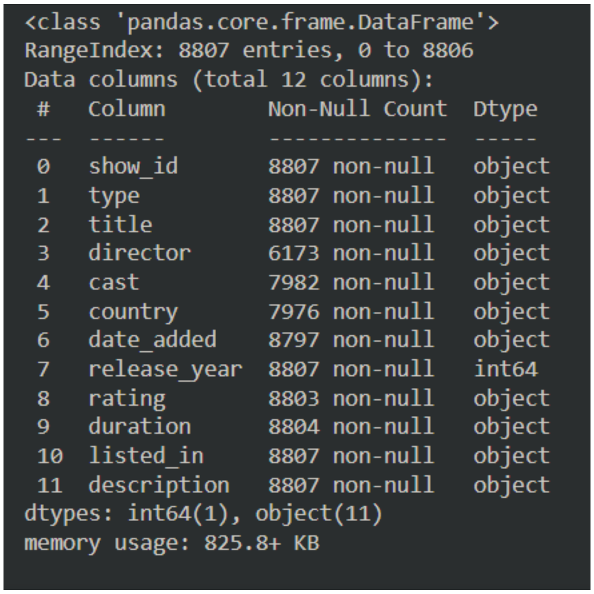
This dataset contains data collected from Netflix of different TV shows and movies from the year 2008 to 2021.

- **type:** Gives information about 2 different unique values one is TV Show and another is Movie
- **title:** Gives information about the title of Movie or TV Show
- **director:** Gives information about the director who directed the Movie or TV Show
- **cast:** Gives information about the cast who plays role in Movie or TV Show
- **release_year:** Gives information about the year when Movie or TV Show was released
- **rating:** Gives information about the Movie or TV Show are in which category (eg like the movies are only for students, or adults, etc)
- **duration:** Gives information about the duration of Movie or TV Show
- **listed_in:** Gives information about the genre of Movie or TV Show
- **description:** Gives information about the description of Movie or TV Show

Data Exploration

Exploraing the data

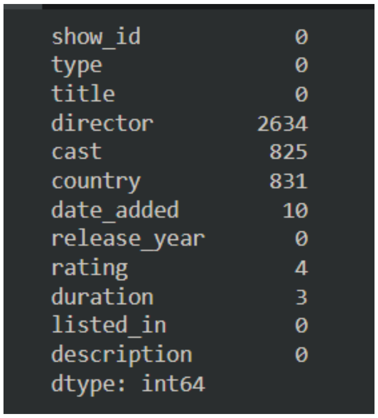
```
netflix_data.info()
```



```
<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
```

Finding if the dataset contains null values or not.

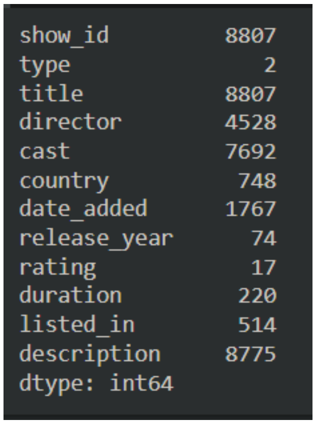
```
netflix_data.isnull().sum() #returns the number of missing values
```



```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

Finding how many unique vales are present in the dataset.

```
netflix_data.nunique()
```



```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

Data Cleaning

Dropping the cast and director features because we are not going to use those features right now.

```
netflix_data = netflix_data.dropna(how='any', subset=['cast', 'director'])
netflix_data = netflix_data.dropna()
```

Replacing null values with ‘missing’

```
#Replacing Null values with 'missing'
netflix_data['country'].fillna('missing', inplace=True )
netflix_data['date_added'].fillna('missing', inplace=True )
netflix_data['rating'].fillna('missing', inplace=True )
netflix_data.isnull().sum().sum()
```

Converting into a proper date-time format and adding two more features year and month.

```
# Converting date_added into proper format
netflix_data["date_added"] = pd.to_datetime(netflix_data['date_added'])
netflix_data['year_added'] = netflix_data['date_added'].dt.year
netflix_data['month_added'] = netflix_data['date_added'].dt.month
```

Finding seasons from durations

```
# finding seasons from durations
netflix_data['season_count'] = netflix_data.apply(lambda x : x['duration'].split(" ")[0] if "Season" in x['duration'] else "", axis = 1)
netflix_data['duration'] = netflix_data.apply(lambda x : x['duration'].split(" ")[0] if "Season" not in x['duration'] else "", axis = 1)
```

Renaming the ‘listed_in’ feature to the genre for easy use.

```
#Renaming the ‘listed_in’ feature to the genre for easy use.
netflix_data = netflix_data.rename(columns={"listed_in":"genre"})
netflix_data['genre'] = netflix_data['genre'].apply(lambda x: x.split(",")[0])
```

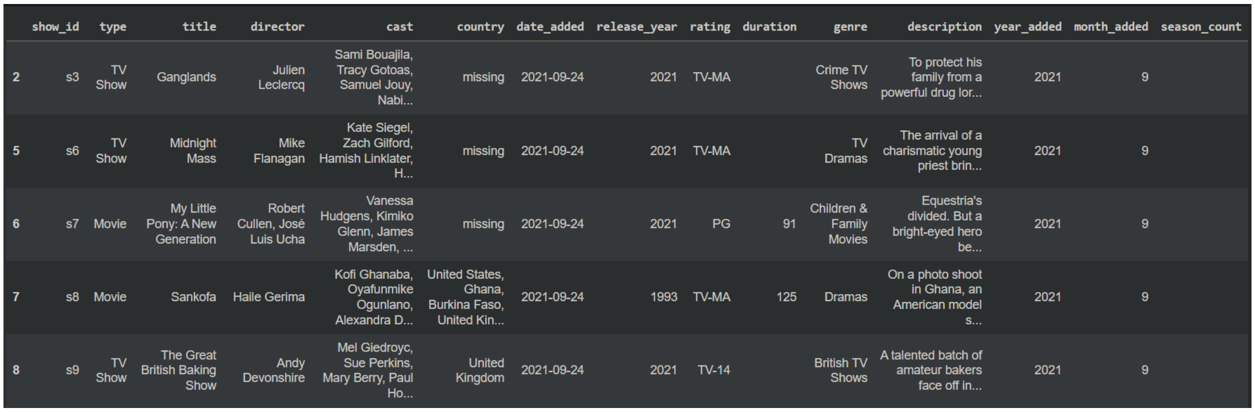
What Are Lambda Functions | Lambda Function In Python

For loops are the antithesis of efficient programming. They're still necessary and are the first conditional loops taught to Python beginners but in my opinion, they leave a lot to be desired. These for loops can be cumbersome and can make our Python code bulky and untidy.

https://www.analyticsvidhya.com/blog/2020/03/what-are-lambda-functions-in-python/

After cleaning the dataset, data will look like as follows:

```
netflix_data.head()
```



	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	genre	description	year_added	month_added	season_count
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	missing	2021-09-24	2021	TV-MA		Crime TV Shows	To protect his family from a powerful drug lor...	2021	9	
5	s6	TV Show	Midnight Mass	Mike Flanagan	Kate Siegel, Zach Gilford, Hamish Linklater, H...	missing	2021-09-24	2021	TV-MA		TV Dramas	The arrival of a charismatic young priest brin...	2021	9	
6	s7	Movie	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha	Vanessa Hudgens, Kimiko Glenn, James Marsden, ...	missing	2021-09-24	2021	PG	91	Children & Family Movies	Equestria's divided. But a bright-eyed hero be...	2021	9	
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States, Ghana, Burkina Faso, United Kin...	2021-09-24	1993	TV-MA	125	Dramas	On a photo shoot in Ghana, an American model s...	2021	9	
8	s9	TV Show	The Great British Baking Show	Andy Devonshire	Mel Giedroyc, Sue Perkins, Mary Berry, Paul Ho...	United Kingdom	2021-09-24	2021	TV-14		British TV Shows	A talented batch of amateur bakers face off in...	2021	9	

Data Visualization

Before going to deep dive into visualization, let's first get clear with some Parameters which are commonly used.

Basic Parameters

- **height, width:** Giving height & width of the chart respectively
- **color_discrete_sequence:** Specifying which colors you want in your chart
- **margin:** By setting values of Top, Bottom, Left, and Right you can change the margin of the graph
- **plot_bgcolor:** Plotly provides different templates and themes you can use that also or you can create your own theme.
- **showgrid:** The grid lines which are present in the chart. You can hide it by showgrid=False

Advanced Parameters

- **categoryorder:** If you want your chart to show in ascending or decsending order you can specify values here
- **hovermode:** While hovering on the bar you can specify in which mode you want like x hovermode or y hovermode.
- **hovertemplate:** While hovering on the data you can specify a format like how you want to look the data
- **xaxis_title, yaxis_title:** Specify the xaxis title & yaxis title respectively
- **title:** Giving a Title of the chart
- **title_font**
  - **size:** Setting the font size of the title
  - **color:** Setting the color of the title
  - **family:** Setting the font-family of the title
- **font**
  - **size:** Setting the font size of font in chart
  - **color:** Setting the color of font in chart

- **family:** Setting the font-family of the font in chart

Now let’s start the fun and gain some insights from the data.

If we look at the data, we have some questions ready like

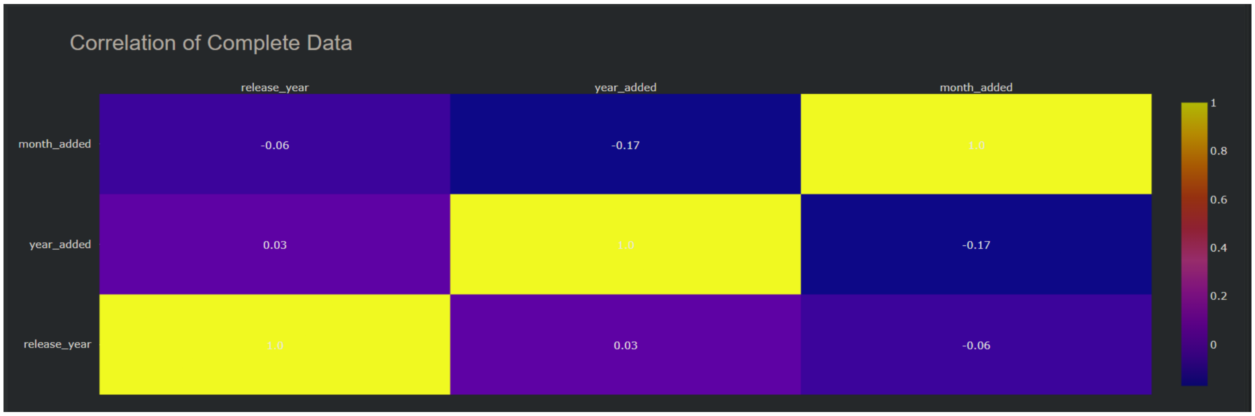
- What is the ratio of Movie and TV Shows on Netflix
- Distribution of Rating in Netflix
- Which has the highest rating Tv Shows or Movies on Netflix

and many more questions...

Chart 01

Viewing the correlation between the features.

```
# Heatmap
# Correlation between the feature show with the help of visualization
correlation_data = netflix_data.corr()
fig_heatmap = ff.create_annotated_heatmap(
    z = correlation_data.values,
    x = list(correlation_data.columns),
    y = list(correlation_data.index),
    annotation_text = correlation_data
    .round(2).values,
    showscale = True
)

fig_heatmap.update_layout(title = 'Correlation of Complete Data',
                          plot_bgcolor = '#2d3035', paper_bgcolor = '#2d3035',
                          title_font = dict(size = 25, color = '#a5a7ab', family = "Muli, sans-serif"),
                          font = dict(color = '#131414'))
```



HeatMap in Plotly

Chart 02

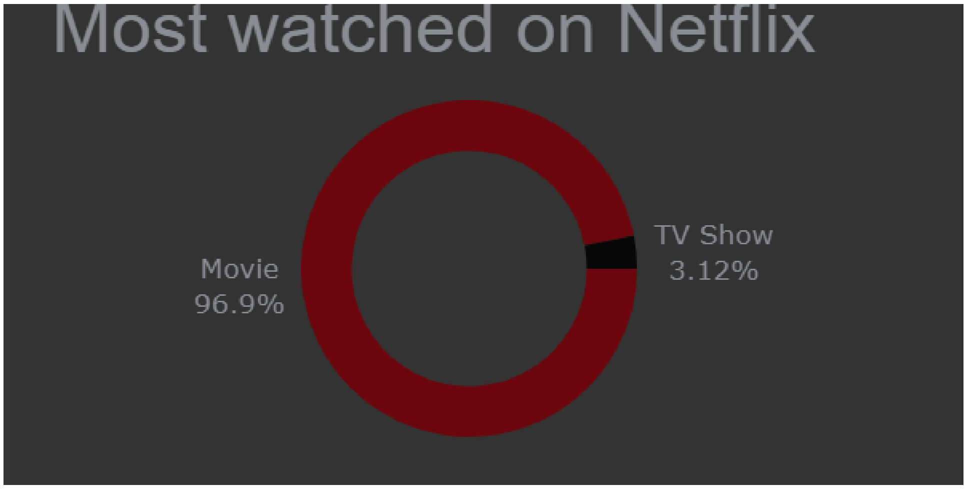
The reason for plotting this chart....?

This will tell us about the most watched shows on Netflix, what exactly audience prefers to watch whether it's TV shows or series. So that, Netflix can decide what type of content they should publish to make the audience happy and they can make profit.

Parameters:

- **textposition:** The values of Movie & TV Show are shown outside of the chart by giving textposition=outside
- **textinfo:** What information you want to shown while hovering on the chart is specified here
- **rotation:** Giving different rotation to the chart to make chart presentable
- **pull:** What data and percent of data you want to pull you specified here
- **hole:** You can specify any values between 0 to 1 to make hole.

```
fig_donut = px.pie(netflix_data, names='type', height= 300, width= 600, hole= 0.7,
                   title = 'Most watched on Netflix',
                   color_discrete_sequence = ['#6b070c', '#080707'])
#updating figure
fig_donut.update_traces(hovertemplate = None, textposition = 'outside',
                        textinfo = 'percent + label', rotation = 90)

fig_donut.update_layout(margin = dict(t=60, b= 30, l=0, r=0), showlegend = False,
                        plot_bgcolor = '#333', paper_bgcolor = '#333',
                        title_font=dict(size=45, color='#8a8d93', family="Lato, sans-serif"),
                        font=dict(size=17, color='#8a8d93'),
                        hoverlabel=dict(bgcolor="#444", font_size=13, font_family="Lato, sans-serif")
                        )
```



Donut chart in Plotly

Audience prefers Movies over TV Shows as 97% audience like Movie.

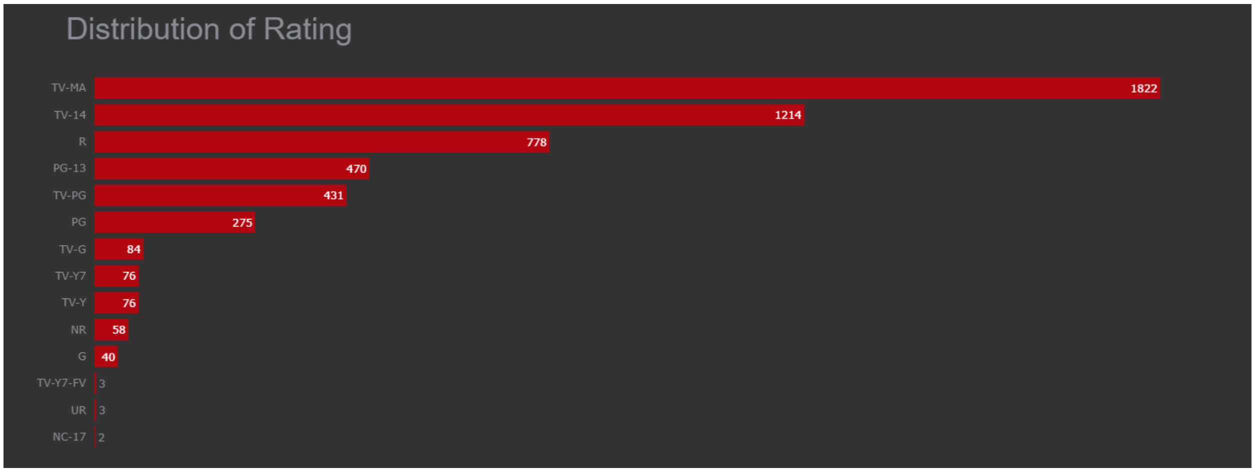
Chart 03

Distribution of Rating and finding what audience prefer to watch.

The reason for plotting the chart?

To know which type of content is most watched by the audience so that Netflix can decide what type of content to be released next. It helps Netflix to understand the most and least favourite content watched by an audience.

```
netflix_rating = pd.DataFrame(netflix_data['rating'].value_counts()).reset_index().rename(columns={'index':'rating','rating':'count'})

fig_bar = px.bar(netflix_rating, y='rating', x='count', title='Distribution of Rating',
                 color_discrete_sequence=['#b20710'], text='count')
fig_bar.update_xaxes(visible=False)
fig_bar.update_yaxes(showgrid=False, categoryorder='total ascending', ticksuffix='  ', showline=False)
fig_bar.update_traces(hovertemplate=None, marker=dict(line=dict(width=0)))
fig_bar.update_layout(margin=dict(t=80, b=20, l=70, r=40),
                      hovermode="y unified",
                      xaxis_title=' ', yaxis_title=" ",
                      plot_bgcolor='#333', paper_bgcolor='#333',
                      title_font=dict(size=35, color='#8a8d93', family="Lato, sans-serif"),
                      font=dict(color='#8a8d93'),
                      legend=dict(orientation="h", yanchor="bottom", y=1, xanchor="center", x=0.5),
                      hoverlabel=dict(bgcolor="black", font_size=13, font_family="Lato, sans-serif"))
```



Bar Chart

The audience prefers TV-MA and TV-14 shows more and the least preferred rating shows are Nc-17. Most of the content watched by the audience is for a mature audience. The TV-MA rating is a type of rating given by the TV parental guidelines to a television program.

The second largest type of rating watched by the audience is TV-14 which is inappropriate for children younger than age 14. The conclusion is drawn here is most of the audience is of mature age.

Chart 04

We will see which has the highest rating TV Shows or Movies.

The reason for plotting the chart?

Bidirectional bar chart compares between the TV shows and Movies vs Ratings.

Creating two different bar charts one for TV Show and another for Movie doesn’t make sense but combining a user can easily understand the difference.

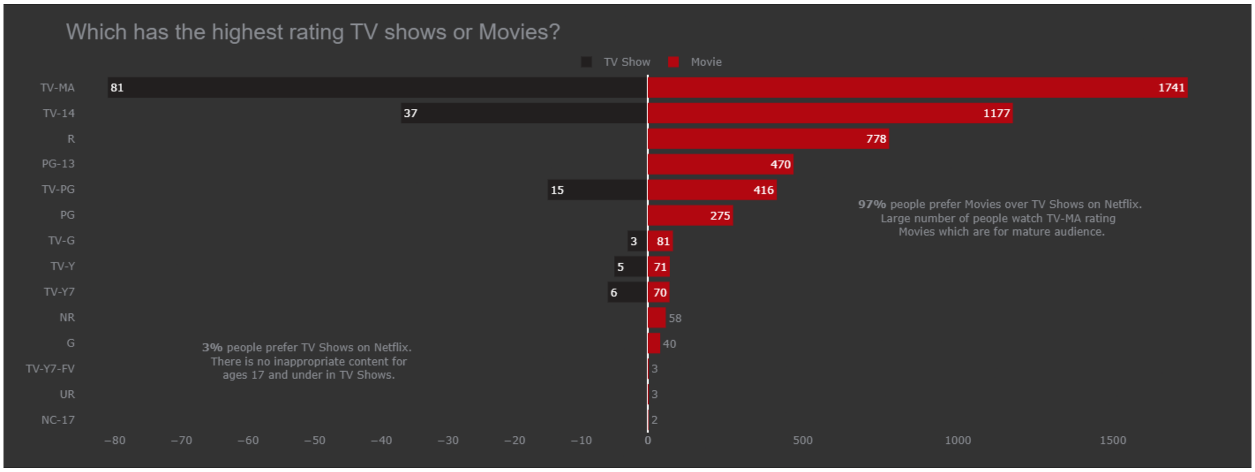
```
# making a copy of netflix_data
dff = netflix_data.copy()
```

```
# making 2 df one for tv show and another for movie with rating
df_tv_show = dff[dff['type']=='TV Show'][['rating', 'type']].rename(columns={'type':'tv_show'})
df_movie = dff[dff['type']=='Movie'][['rating', 'type']].rename(columns={'type':'movie'})
df_tv_show = pd.DataFrame(df_tv_show.rating.value_counts()).reset_index().rename(columns={'index':'tv_show'})
df_tv_show['rating_final'] = df_tv_show['rating']
# making rating column value negative
df_tv_show['rating'] *= -1
df_movie = pd.DataFrame(df_movie.rating.value_counts()).reset_index().rename(columns={'index':'movie'})
```

Here for the bi-directional chart, we will make 2 different data frames one for Movies and another one for TV Shows having ratings in them.

We are making 2 subplots and they are sharing the y-axis.

```
fig = make_subplots(rows=1, cols=2, specs=[[{}, {}]], shared_yaxes=True, horizontal_spacing=0)
# bar plot for tv shows
fig.append_trace(go.Bar(x=df_tv_show.rating, y=df_tv_show.tv_show, orientation='h', showlegend=True,
                        text=df_tv_show.rating_final, name='TV Show', marker_color='#221f1f'), 1, 1)
# bar plot for movies
fig.append_trace(go.Bar(x=df_movie.rating, y=df_movie.movie, orientation='h', showlegend=True, text=df_movie.rating,
                        name='Movie', marker_color='#b20710'), 1, 2)
fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False, categoryorder='total ascending', ticksuffix=' ', showline=False)
fig.update_traces(hovertemplate=None, marker=dict(line=dict(width=0)))
fig.update_layout(title='Which has the highest rating TV shows or Movies?',
                  margin=dict(t=80, b=0, l=70, r=40),
                  hovermode="y unified",
                  xaxis_title=' ', yaxis_title=" ",
                  plot_bgcolor='#333', paper_bgcolor='#333',
                  title_font=dict(size=25, color='#8a8d93', family="Lato, sans-serif"),
                  font=dict(color='#8a8d93'),
                  legend=dict(orientation="h", yanchor="bottom", y=1, xanchor="center", x=0.5),
                  hoverlabel=dict(bgcolor="black", font_size=13, font_family="Lato, sans-serif"))
```



Bi-directional Bar Chart

Tips and tricks for Plotly Bar Chart

You never knew you needed Plotly! Plotly is like chocolate for visualization you can't get enough of it. The best library has the best user interaction charts. Here I am going to talk about different types of Bar charts in Plotly. Data is taken from Kaggle and the whole notebook is available here.

https://pub.towardsai.net/tips-and-tricks-for-plotly-bar-chart-71261391c57b

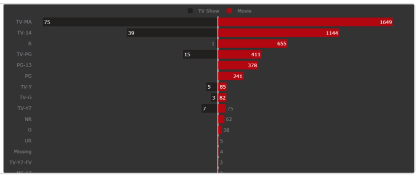


Chart 05

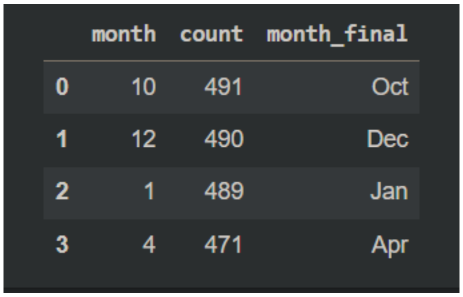
If a producer wants to release a show which month is the best month to release it.

The reason for plotting the chart.?

The best month to release content so the producer can gain much revenue. Most of the holidays came in December month so to releases a Movie or TV show in December is the best way to earn a lot of profit as the whole family will be spending time with each other and watching shows.

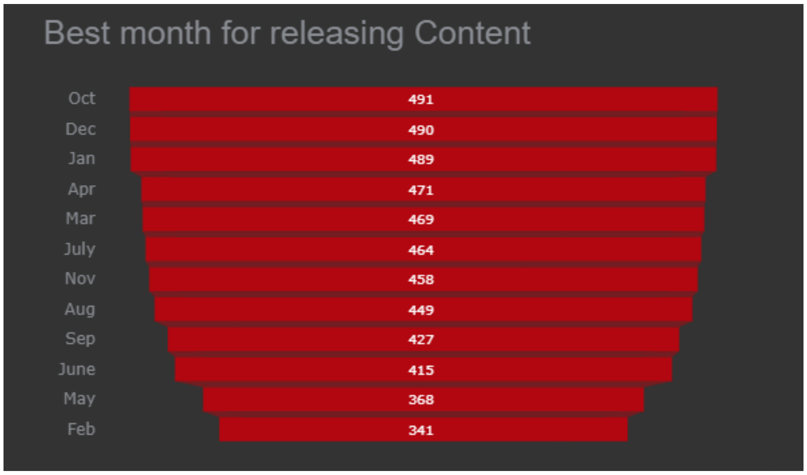
```
netflix_month = pd.DataFrame(netflix_data.month_added.value_counts()).reset_index().rename(columns={'index':'month','month_added':'count'})
# Replacing month number to month name for a better visualization
```

```
netflix_month['month_final'] = netflix_month['month'].replace({1:'Jan', 2:'Feb', 3:'Mar', 4:'Apr', 5:'May', 6:'June', 7:'July', 8:'Aug', 9:
netflix_month[:4]
```



	month	count	month_final
0	10	491	Oct
1	12	490	Dec
2	1	489	Jan
3	4	471	Apr

```
fig_month = px.funnel(netflix_month, x='count', y='month_final', title='Best month for releasing Content',
                     height=350, width=600, color_discrete_sequence=['#b20710'])
fig_month.update_xaxes(showgrid=False, ticksuffix=' ', showline=True)
fig_month.update_traces(hovertemplate=None, marker=dict(line=dict(width=0)))
fig_month.update_layout(margin=dict(t=60, b=20, l=70, r=40),
                       xaxis_title=' ', yaxis_title=" ",
                       plot_bgcolor='#333', paper_bgcolor='#333',
                       title_font=dict(size=25, color='#8a8d93', family="Lato, sans-serif"),
                       font=dict(color='#8a8d93'),
                       hoverlabel=dict(bgcolor="black", font_size=13, font_family="Lato, sans-serif"))
```



Funnel Chart

> **Parameters:**
>
> **add_annotationx, y:** Setting the values of x,y for showing the text**text:** Add the text which you want to show on the chart

Ending and starting of the year December and January is the best month to release content. The best 4 months to release content are October, November, December, and January.

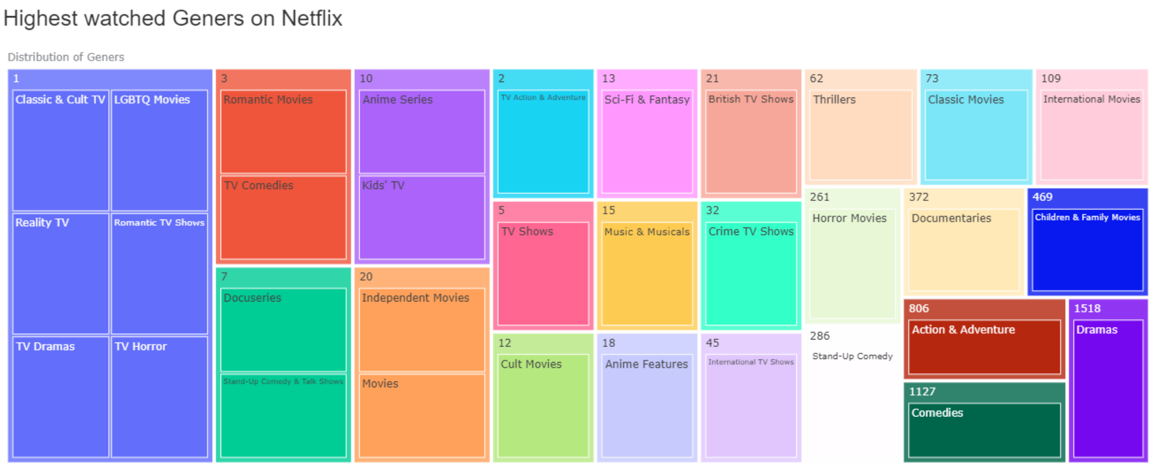
Chart 06

Finding the highest watched genres on Netflix.

The reason for plotting the chart?

To know more about the distribution of genres and see which type of content do audience prefers to watch. So Netflix can decide and take movies or tv shows of the highest watched genres which will benefit Netflix in a long run.

```
df_genre = pd.DataFrame(netflix_data.genre.value_counts()).reset_index().rename(columns={'index':'genre', 'genre':'count'})
fig_tree = px.treemap(df_genre, path=[px.Constant("Distribution of Geners"), 'count','genre'])
fig_tree.update_layout(title='Highest watched Geners on Netflix',
                  margin=dict(t=50, b=0, l=70, r=40),
                  plot_bgcolor='#fff', paper_bgcolor='#fff',
                  title_font=dict(size=25, color='#333', family="Lato, sans-serif"),
                  font=dict(color='#8a8d93'),
                  hoverlabel=dict(bgcolor="#444", font_size=13, font_family="Lato, sans-serif"))
```



TreeMap

<u>Drama is the highest preferred show by the audience then comes the comedy show and action show, the least preferred show is of LGBTQ movies.</u>

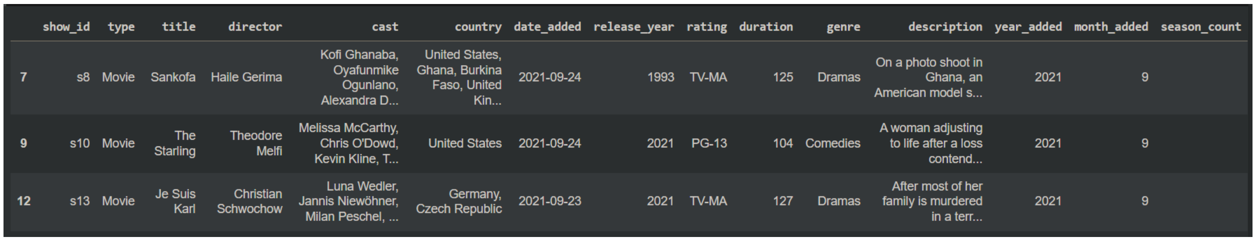
Chart 07

Let's see how many movies were released over the years with a waterfall chart.

The reason for plotting the chart?

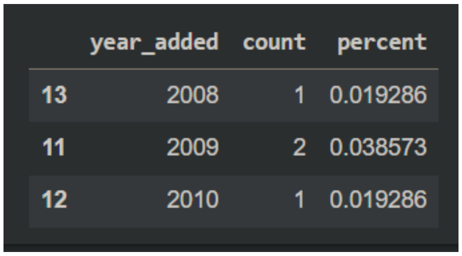
We want to see what is the distribution of releases of a movie or tv show in terms of years. Do the releases increase or decrease with the years. We can easily compare if the release of movies is decreasing or increasing by year by year with the help of a waterfall chart.

```
# creating a dataframe which only consists of Movie shows
d2 = netflix_data[netflix_data["type"] == "Movie"]
d2[:3]
```



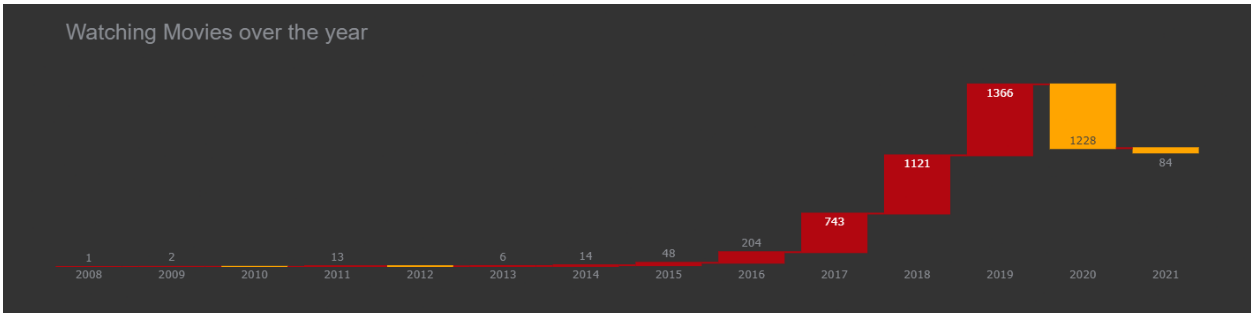
	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	genre	description	year_added	month_added	season_count
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States, Ghana, Burkina Faso, United Kin...	2021-09-24	1993	TV-MA	125	Dramas	On a photo shoot in Ghana, an American model s...	2021	9	
9	s10	Movie	The Starling	Theodore Melfi	Melissa McCarthy, Chris O'Dowd, Kevin Kline, T...	United States	2021-09-24	2021	PG-13	104	Comedies	A woman adjusting to life after a loss contend...	2021	9	
12	s13	Movie	Je Suis Karl	Christian Schwochow	Luna Wedler, Jannis Niewöhner, Milan Peschel, ...	Germany, Czech Republic	2021-09-23	2021	TV-MA	127	Dramas	After most of her family is murdered in a terr...	2021	9	

```
# creating a dataframe which shows how many movies were released each year
col = "year_added"

vc2 = d2[col].value_counts().reset_index().rename(columns = {col : "count", "index" : col})
vc2['percent'] = vc2['count'].apply(lambda x : 100*x/sum(vc2['count']))
vc2 = vc2.sort_values(col)
vc2[:3]
```



	year_added	count	percent
13	2008	1	0.019286
11	2009	2	0.038573
12	2010	1	0.019286

```
# Waterfall Chart
fig2 = go.Figure(go.Waterfall(
    name = "Movie", orientation = "v",
    x = ["2008", "2009", "2010", "2011", "2012", "2013", "2014", "2015", "2016", "2017", "2018", "2019", "2020", "2021"],
    textposition = "auto",
    text = ["1", "2", "1", "13", "3", "6", "14", "48", "204", "743", "1121", "1366", "1228", "84"],
    y = [1, 2, -1, 13, -3, 6, 14, 48, 204, 743, 1121, 1366, -1228, -84],
    connector = {"line":{"color":"#b20710"}},
    increasing = {"marker":{"color":"#b20710"}},
    decreasing = {"marker":{"color":"orange"}},

))


fig2.update_xaxes(showgrid=False)
fig2.update_yaxes(showgrid=False, visible=False)
fig2.update_traces(hovertemplate=None)
fig2.update_layout(title='Watching Movies over the year', height=350,
                   margin=dict(t=80, b=20, l=50, r=50),
                   hovermode="x unified",
                   xaxis_title=' ', yaxis_title=" ",
                   plot_bgcolor='#333', paper_bgcolor='#333',
                   title_font=dict(size=25, color='#8a8d93', family="Lato, sans-serif"),
                   font=dict(color='#8a8d93'))
```



Waterfall Chart

Here yellow color shows the decrement and the red color shows the increment.

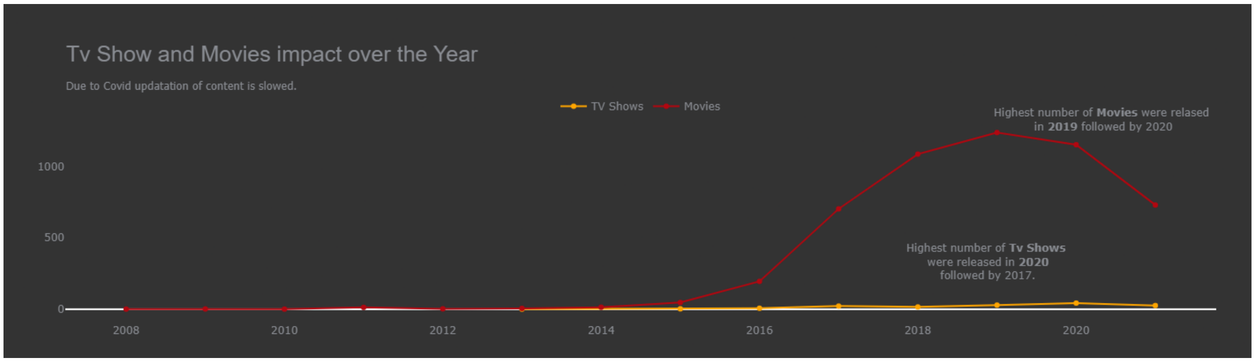
> **Parameters:**
>
> **connector:** Specify the color for connecting two bar increasing, decreasing: Specify the colors for the increasing and decreasing bar.

The highest number of movies were released in 2019 and 2018 due to the covid releasing of movies were significantly dropped.

Chart 08

What is the impact of Netflix TV Shows or Movies over the years by comparing both.

```
d1 = netflix_data[netflix_data["type"] == "TV Show"]
d2 = netflix_data[netflix_data["type"] == "Movie"]

col = "year_added"

vc1 = d1[col].value_counts().reset_index().rename(columns = {col : "count", "index" : col})
vc1['percent'] = vc1['count'].apply(lambda x : 100*x/sum(vc1['count']))
vc1 = vc1.sort_values(col)

vc2 = d2[col].value_counts().reset_index().rename(columns = {col : "count", "index" : col})
vc2['percent'] = vc2['count'].apply(lambda x : 100*x/sum(vc2['count']))
vc2 = vc2.sort_values(col)

trace1 = go.Scatter(x=vc1[col], y=vc1["count"], name="TV Shows", marker=dict(color="orange"), )
trace2 = go.Scatter(x=vc2[col], y=vc2["count"], name="Movies", marker=dict(color="#b20710"))
data = [trace1, trace2]
fig_line = go.Figure(data)

fig_line.update_traces(hovertemplate=None)
fig_line.update_xaxes(showgrid=False)
fig_line.update_yaxes(showgrid=False)

large_title_format = 'Tv Show and Movies impact over the Year'
small_title_format = "<span style='font-size:13px; font-family:Tahoma'>Due to Covid updatation of content is slowed."
fig_line.update_layout(title=large_title_format + "<br>" + small_title_format, height=400,
                  margin=dict(t=130, b=0, l=70, r=40),
                  hovermode="x unified",
                  xaxis_title=' ', yaxis_title=" ",
                  plot_bgcolor='#333', paper_bgcolor='#333',
                  title_font=dict(size=25, color='#8a8d93', family="Lato, sans-serif"),
                  font=dict(color='#8a8d93'),
                  legend=dict(orientation="h", yanchor="bottom", y=1, xanchor="center", x=0.5))

fig_line.add_annotation(dict(x=0.8, y=0.3, ax=0, ay=0,
                    xref = "paper", yref = "paper",
                    text= "Highest number of <b>Tv Shows</b><br> were released in <b>2020</b><br> followed by 2017."
                  ))
fig_line.add_annotation(dict(x=0.9, y=1, ax=0, ay=0,
                    xref = "paper", yref = "paper",
                    text= "Highest number of <b>Movies</b> were relased<br> in <b>2019</b> followed by 2020"
                  ))
fig_line.show()
```



After the year 2019 covid came that badly affects Netflix for producing content. Movies have exponential growth from the start but due to covid, it is going downwards.

We explored the Netflix dataset and saw how to clean the data and then jump into how to visualize the data. We saw some basic and advanced level charts of Plotly like Heatmap, Donut chart, Bar chart, Funnel chart, Bi-directional Bar chart, Treemap, Waterfall chart, Line chart.

Google Colaboratory

https://colab.research.google.com/drive/1UKDiQwATkBuMko3Oc2DV-gMRzAtZfBnY?authuser=1#scrollTo=RWK9pxwbHIbX