```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True)
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
df_prime = pd.read_csv('/content/amazon_prime_titles.csv')
```

```
df_prime.head()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	The Grand Seduction	Don McKellar	Brendan Gleeson, Taylor Kitsch, Gordon Pinsent	Canada	March 30, 2021	2014	NaN	113 min	Comedy, Drama	A small fishing village must procure a local d...
1	s2	Movie	Take Care Good Night	Girish Joshi	Mahesh Manjrekar, Abhay Mahajan, Sachin Khedekar	India	March 30, 2021	2018	13+	110 min	Drama, International	A Metro Family decides to fight a Cyber Crimin...
2	s3	Movie	Secrets of Deception	Josh Webber	Tom Sizemore, Lorenzo Lamas, Robert LaSardo, R...	United States	March 30, 2021	2017	NaN	74 min	Action, Drama, Suspense	After a man discovers his wife is cheating on ...
3	s4	Movie	Pink: Staying True	Sonia Anderson	Interviews with: Pink, Adele, Beyoncé, Britney...	United States	March 30, 2021	2014	NaN	69 min	Documentary	Pink breaks the mold once again, bringing her ...
4	s5	Movie	Monster Maker	Giles Foster	Harry Dean Stanton, Kieran	United Kingdom	March 30, 2021	1989	NaN	45 min	Drama, Fantasy	Teenage Matt Banting wants to

Next steps: Generate code with df_prime | New interactive sheet

```
df_prime.tail()
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	descript
9663	s9664	Movie	Pride Of The Bowery	Joseph H. Lewis	Leo Gorcey, Bobby Jordan	NaN	NaN	1940	7+	60 min	Comedy	New Y City st principles an East
9664	s9665	TV Show	Planet Patrol	NaN	DICK VOSBURGH, RONNIE STEVENS, LIBBY MORRIS, M...	NaN	NaN	2018	13+	4 Seasons	TV Shows	This is Ea 2100A and th are adv
9665	s9666	Movie	Outpost	Steve Barker	Ray Stevenson, Julian Wadham, Richard Brake, M...	NaN	NaN	2008	R	90 min	Action	In war-t East Europ world-we gro
9666	s9667	TV Show	Maradona: Blessed Dream	NaN	Esteban Recagno, Ezequiel Stremiz, Luciano Vit...	NaN	NaN	2021	TV-MA	1 Season	Drama, Sports	The se tells the s of Di Maradona
9667	s9668	Movie	Harry Brown	Daniel Barber	Michael Caine, Emily Mortimer, Joseph	NaN	NaN	2010	R	103 min	Action, Drama, Suspense	Harry Bro starring t t Acade

```
# Check the shape of the DataFrame df_prime
print("Number of Rows",df_prime.shape[0])
print("Number of Columns",df_prime.shape[1])
```

```
Number of Rows 9668
Number of Columns 12
```

```
# Print the column names of the DataFrame df_prime
print("Columns Names : ", df_prime.columns)
```

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

```
df_prime.isnull().sum()
```

	0
show_id	0
type	0
title	0
director	2083
cast	1233
country	8996
date_added	9513
release_year	0
rating	337
duration	0
listed_in	0
description	0

dtype: int64

```
df_prime['rating'].fillna(df_prime['rating'].mode()[0], inplace=True)
```

```
df_prime['release_year'].fillna(df_prime['release_year'].mode()[0], inplace=True)
```

```
df_prime['description'].replace(np.nan, 'No Data', inplace=True)
```

```
df_prime['director'].replace(np.nan, 'No Data', inplace=True)
```

```
df_prime['cast'].replace(np.nan, 'No Data', inplace=True)
```

```
df_prime['country'].replace(np.nan, 'No Data', inplace=True)
```

```
df_prime['date_added'].replace(np.nan, 'No Data', inplace=True)
```

```
cleaned_missing_values = df_prime.isnull().sum()
```

```
print("Missing values after cleaning:")
print(cleaned_missing_values)
```

```
Missing values after cleaning:
show_id         0
type            0
title           0
director        0
cast            0
country         0
date_added      0
release_year    0
rating          0
duration        0
listed_in       0
description     0
dtype: int64
```

```
num_duplicates = len(df_prime[df_prime.duplicated()])
```

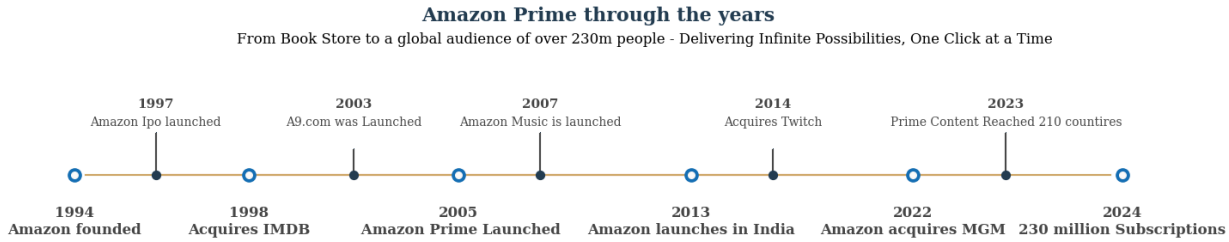
```
print("Number of duplicated rows:", num_duplicates)
```

```
Number of duplicated rows: 0
```

```
df_prime.drop_duplicates(inplace=True)
```

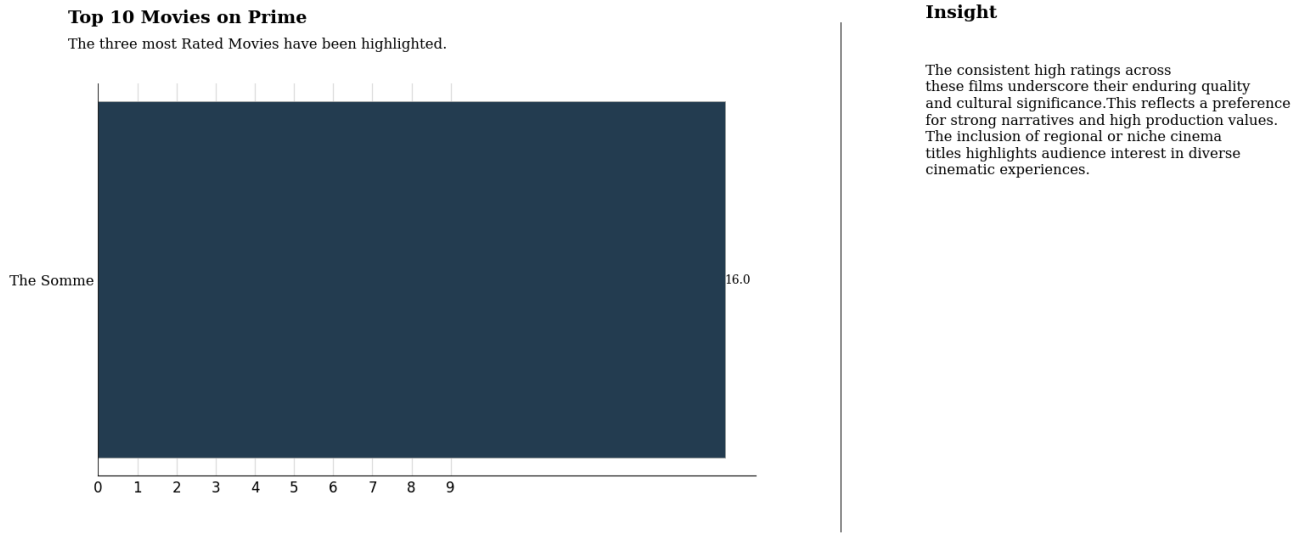
```
tl_dates = ['1994\nAmazon founded',
            '1998\nAcquires IMDB',
            '2005\n Amazon Prime Launched',
            '2013\nAmazon launches in India',
            '2022\nAmazon acquires MGM',
            '2024\n230 million Subscriptions'

]

tl_x = [1,2.5,4.3,6.3,8.2,10]

## these go on the numbers
tl_sub_x = [1.7,3.4,5,7,9]


tl_sub_times = [
    "1997","2003","2007","2014",'2023'
]

tl_text = [
    "Amazon Ipo launched",
    "A9.com was Launched","Amazon Music is launched","Acquires Twitch","Prime Content Reached 210 countires"]

fig,ax = plt.subplots(figsize = (15,4),constrained_layout=True)
ax.set_ylim(-2,1.75)
ax.set_xlim(0,11)

# Timeline Line
ax.axhline(0,xmin= 0.1,xmax=0.9,color = '#CEA968',zorder = 1)

#Timeline Datepoints
ax.scatter(tl_x,np.zeros(len(tl_x)),s = 120,color ='#146eb4',zorder = 2)
ax.scatter(tl_x,np.zeros(len(tl_x)),s = 30,color ='#fafafa',zorder = 3)

#Timeline Subpoints
ax.scatter(tl_sub_x,np.zeros(len(tl_sub_x)),s = 50,color ='#233D53',zorder = 4)

#Dates
for x, date in zip(tl_x,tl_dates):
  ax.text(x, -0.70, date, ha='center',
           fontfamily='serif', fontweight='bold',
```

```
            color='#4a4a4a',fontsize=12)
#Stem Plot
levels = np.zeros(len(tl_sub_x))
levels[::2] = 0.5
levels[1::2] = 0.3
markerline, stemline, baseline = ax.stem(tl_sub_x, levels)
plt.setp(baseline, zorder=0)
plt.setp(markerline, marker=',', color='#233D53')
plt.setp(stemline, color='#4a4a4a')

#Text
for idx, x, time, txt in zip(range(1, len(tl_sub_x)+1), tl_sub_x, tl_sub_times, tl_text):
    ax.text(x, 1.3-0.5, time, ha='center',
            fontfamily='serif', fontweight='bold',
            color='#4a4a4a', fontsize=11)

    ax.text(x, 1.3-0.6, txt, va='top', ha='center',
        fontfamily='serif',color='#4a4a4a')

#Spines
for spine in ["left", "top", "right", "bottom"]:
    ax.spines[spine].set_visible(False)

ax.set_xticks([])
ax.set_yticks([])

ax.set_title("Amazon Prime through the years", fontweight="bold", fontfamily='serif', fontsize=16, color='#233D53')
ax.text(2.4,1.57,"From Book Store to a global audience of over 230m people - Delivering Infinite Possibilities, One Click at a

plt.show()
```



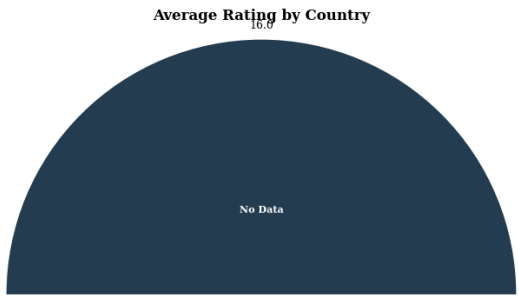

```
# Convert 'rating' to numeric, coercing errors
df_prime['rating'] = pd.to_numeric(df_prime['rating'], errors='coerce')

# Group by 'title' and calculate mean rating for the top 10
top_movies = df_prime.groupby('title')['rating'].mean().sort_values(ascending=False)[:10]
color_map = ['#CEA968' for _ in range(10)]
color_map[0] = color_map[1] = color_map[2] = '#233D53'



fig, ax = plt.subplots(1, 1, figsize=(10, 6))
ax.barh(top_movies.index, top_movies, height=0.5, edgecolor='darkgrey', linewidth=0.6, color=color_map)

for i in top_movies.index:
    ax.annotate(f'{top_movies[i]}', xy=(top_movies[i], i), va='center', ha='left', fontweight='light',
                fontfamily='serif')

for s in ['top', 'left', 'right']:
    ax.spines[s].set_visible(False)

ax.set_yticklabels(top_movies.index, fontfamily='serif', rotation=0)
fig.text(0.09, 1, 'Top 10 Movies on Prime', fontsize=15, fontweight='bold', fontfamily='serif')
fig.text(0.09, 0.95, 'The three most Rated Movies have been highlighted.', fontsize=12, fontweight='light',
         fontfamily='serif')
fig.text(1.1, 1.01, 'Insight', fontsize=15, fontweight='bold', fontfamily='serif')

fig.text(1.1, 0.67, '''
The consistent high ratings across
these films underscore their enduring quality
and cultural significance.This reflects a preference
for strong narratives and high production values.
The inclusion of regional or niche cinema
titles highlights audience interest in diverse
cinematic experiences.
```

```
'''
, fontsize=12, fontweight='light', fontfamily='serif')

ax.grid(axis='x', linestyle='-', alpha=0.4)

grid_x_ticks = np.arange(0, 10, 1)  # x ticks, min, max, then step
ax.set_xticks(grid_x_ticks)
ax.set_axisbelow(True)

plt.axvline(x=0, color='black', linewidth=1.3, alpha=.9)
ax.tick_params(axis='both', which='major', labelsize=12)

import matplotlib.lines as lines

l1 = lines.Line2D([1, 1], [0, 1], transform=fig.transFigure, figure=fig, color='black', lw=0.7)
fig.lines.extend([l1])

ax.tick_params(axis=u'both', which=u'both', length=0)

plt.show()
```




```
# Convert 'rating' to numeric, coercing errors
df_prime['rating'] = pd.to_numeric(df_prime['rating'], errors='coerce')

# Group by 'country' and calculate mean rating for the top 10 countries, dropping NaN values
country_ratings = df_prime.groupby('country')['rating'].mean().dropna().sort_values(ascending=False)[:10]

# Compute the number of bars
num_bars = len(country_ratings)

# Set the angles for each bar
angles = np.linspace(0, 2 * np.pi, num_bars, endpoint=False).tolist()

# The width of each bar
width = np.pi / num_bars

# Initialize the figure with one subplot
fig, ax = plt.subplots(1, 1, figsize=(12, 8), subplot_kw=dict(polar=True))

# Remove the axis
ax.axis('off')

# Plot each bar on the polar plot
for angle, value, country_name in zip(angles, country_ratings, country_ratings.index):
    # Rating labels
    rating_label = f'{value:.1f}'
    if value > country_ratings.mean(): # Highlight countries above average rating
        bar_color = '#08AAE3'
    else:
        bar_color = '#233D53'
    ax.bar(angle, value, width=width, color=bar_color)
    # Country name labels
    ax.text(angle, 5, country_name, ha='center', va='bottom', fontsize=8, fontfamily='serif', color='white', fontweight='bold
    # Rating labels above the axis
    ax.text(angle, value + 0.5, rating_label, ha='center', va='bottom', fontsize=9, fontfamily='serif', color='black')

# Set the label position
```

```
ax.set_theta_offset(np.pi / 2)
ax.set_theta_direction(-1)
ax.set_xticks(angles)
ax.set_xticklabels(country_ratings.index, fontsize=10, fontfamily='serif')

# Add text for the insight
fig.text(1.1, 1.01, 'Insight', fontsize=15, fontweight='bold', fontfamily='serif')
fig.text(1.1, 0.67, '''
Based on available ratings, this plot shows the
average rating for content from the top 10 countries.
Countries with higher average ratings are highlighted.
This can provide insights into the perceived quality
of content originating from different regions.
'''
, fontsize=12, fontweight='light', fontfamily='serif')

import matplotlib.lines as lines

l1 = lines.Line2D([1, 1], [0, 1], transform=fig.transFigure, figure=fig, color='black', lw=0.7)
fig.lines.extend([l1])

ax.tick_params(axis=u'both', which=u'both', length=0)
plt.title('Average Rating by Country', fontsize=12, fontweight='bold', fontfamily='serif', loc='center')

plt.show()
```

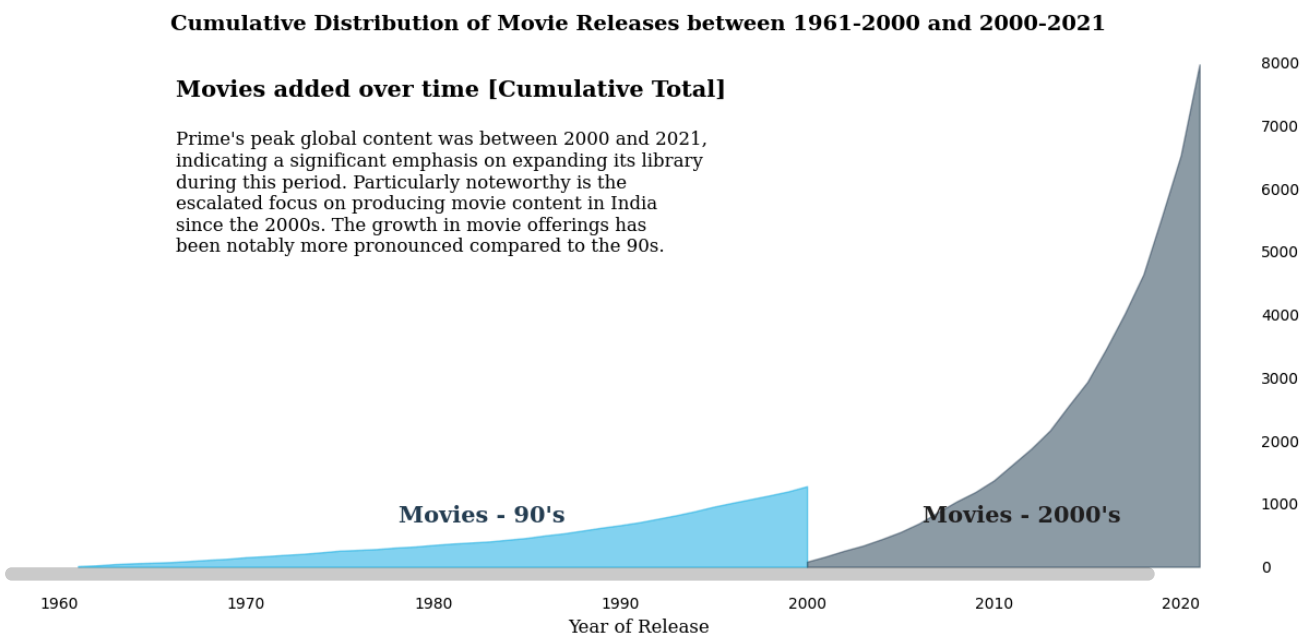


Insight

Based on available ratings, this plot shows the average rating for content from the top 10 countries. Countries with higher average ratings are highlighted. This can provide insights into the perceived quality of content originating from different regions.

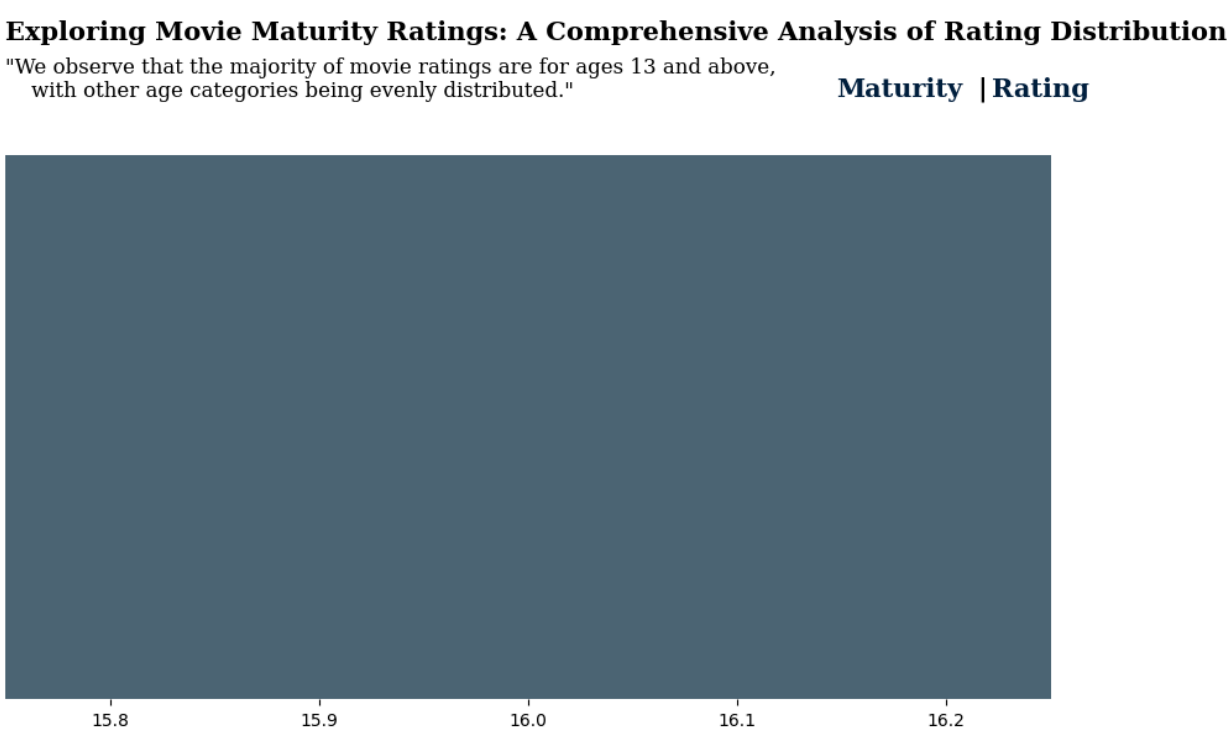
```
import matplotlib.pyplot as plt

# Filter the DataFrame for movies released between 1961 and 2000
df_1961_2000 = df_prime[df_prime['release_year'].between(1961, 2000)]

# Filter the DataFrame for movies released between 2000 and 2021
df_2000_2021 = df_prime[df_prime['release_year'].between(2000, 2021)]

# Count the number of releases for each time period
count_1961_2000 = len(df_1961_2000)
count_2000_2021 = len(df_2000_2021)

# Calculate the ratio of releases between the two time periods
ratio_1961_2000 = count_1961_2000 / (count_1961_2000 + count_2000_2021)
ratio_2000_2021 = count_2000_2021 / (count_1961_2000 + count_2000_2021)

# Group by 'release_year' and count the number of releases for each time period
count_by_year_1961_2000 = df_1961_2000.groupby('release_year').size().cumsum()
count_by_year_2000_2021 = df_2000_2021.groupby('release_year').size().cumsum()

# Plotting the area chart
fig, ax = plt.subplots(figsize=(12, 6))
```

```
ax.fill_between(count_by_year_1961_2000.index, count_by_year_1961_2000, color='#08AAE3', alpha=0.5, label='1961-2000')
ax.fill_between(count_by_year_2000_2021.index, count_by_year_2000_2021, color='#233D53', alpha=0.5, label='2000-2021')

# Adding labels and title
ax.set_title('Cumulative Distribution of Movie Releases between 1961-2000 and 2000-2021', fontsize=14, fontweight='bold', fon
ax.set_xlabel('Year of Release', fontsize=12, fontfamily='serif')

# Remove grid lines and spines
ax.grid(False)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.spines['bottom'].set_visible(False)
ax.spines['left'].set_visible(False)

# Add additional text
fig.text(0.13, 0.85, 'Movies added over time [Cumulative Total]', fontsize=15, fontweight='bold', fontfamily='serif')
fig.text(0.13, 0.58,
'''Prime's peak global content was between 2000 and 2021,
indicating a significant emphasis on expanding its library
during this period. Particularly noteworthy is the
escalated focus on producing movie content in India
since the 2000s. The growth in movie offerings has
been notably more pronounced compared to the 90s.
''', fontsize=12, fontweight='light', fontfamily='serif')

fig.text(0.3, 0.2, "Movies - 90's", fontweight="bold", fontfamily='serif', fontsize=15, color='#233D53')
fig.text(0.7, 0.2, "Movies - 2000's", fontweight="bold", fontfamily='serif', fontsize=15, color='#221f1f')

ax.tick_params(axis=u'both', which=u'both', length=0)

# Move x-axis ticks to the right
ax.yaxis.tick_right()

# Display the plot
plt.tight_layout()
plt.show()
```




```
# Grouping by Maturity Rating and counting the number of movies in each rating category
rating = df_prime.groupby('rating')['title'].count()

# Creating the plot
fig, ax = plt.subplots(1, 1, figsize=(12, 6))

# Plotting the bar chart with dark blue bars for movies released before 2000 and different color for movies released beyond 200
colors = ['#233D53' if rating.index[i]!= '13+' else '#08AAE3' for i in range(len(rating))]
ax.bar(rating.index, rating, width=0.5, color=colors, alpha=0.8, label='Movie')

# Annotating each bar with the count
for rating_label, count in rating.items():
    ax.annotate(f"{count}",
                xy=(rating_label, count + 60),
                va='center', ha='center', fontweight='light', fontfamily='serif',
                color='#4a4a4a')
```

```
ax.yaxis.set_visible(False)

# Removing the spines
for s in ['top', 'left', 'right', 'bottom']:
    ax.spines[s].set_visible(False)

# Displaying the plot
ax.legend().set_visible(False)
fig.text(0.16, 1, 'Exploring Movie Maturity Ratings: A Comprehensive Analysis of Rating Distribution', fontsize=15, fontweight=
fig.text(0.16, 0.89,
'''"We observe that the majority of movie ratings are for ages 13 and above,
    with other age categories being evenly distributed."
'''

, fontsize=12, fontweight='light', fontfamily='serif')


fig.text(0.720,0.924,"Maturity", fontweight="bold", fontfamily='serif', fontsize=15, color='#001f3f')
fig.text(0.815,0.924,"|", fontweight="bold", fontfamily='serif', fontsize=15, color='black')
fig.text(0.825,0.924,"Rating", fontweight="bold", fontfamily='serif', fontsize=15, color='#001f3f')

plt.show()
```




```
from wordcloud import WordCloud
import random
from PIL import Image
import matplotlib


text = str(list(df_prime['title'])).replace(',', '').replace('[', '').replace("'", '').replace(']', '').replace('.', '')

cmap = matplotlib.colors.LinearSegmentedColormap.from_list("", ['#233D53', '#08AAE3'])

mask = np.array(Image.open('/content/prime.jpeg'))


wordcloud = WordCloud(background_color = 'white', width = 1000,  height = 627,max_words = 350, mask = mask,contour_color= '#0

plt.figure( figsize=(12,6))
plt.imshow(wordcloud)
plt.axis('off')
plt.tight_layout(pad=0)
plt.show()
```


Kid
Love
Story
World
4K UHD
Christmas
Night
Life
Road
Death
Legend
King
Classic
Summer
Killer
Day
Dead
America
Blue
Blood
Time
Girl
Dog
House
Friend
Live
Queen
Murder
Telugu
Robot
Boy
Animal
American
Shadow
Adventure
Little
Good
Baby
Bum
Great
Song
Big
Car
Year
War
Bad
Black
Monster
Secret
Morphle
White
Little Baby
Yoga
Show
Ghost
Serie
Comedy
Learn
Nursery Rhyme
Sleep Street
Workout Minute
City
Kids Song
Tamil
Water
Mountain
Fire
Father
Dragon
Mother
Kill
Gold
Moon
Pinkfong
Baby Shark
Paradise
Hindi
Light
Wild
Island
Men
Go Buster