

PROJECT REPORT ON

OTT MEDIA PLATFORMS: EDA ANALYSIS

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT OF UNIVERSITY OF
MUMBAI FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SUBMITTED BY

PAWAN AJAY GOSAVI

SUPERVISED BY

PROF. SAGAR KULKARNI

AND

PROF. ABHIJEET SALVI



DEPARTMENT OF INFORMATION TECHNOLOGY

PILLAI COLLEGE OF ARTS, COMMERCE AND SCIENCE, NEW PANVEL – 410206

UNIVERSITY OF MUMBAI

ACADEMIC YEAR: 2020 – 2021

CERTIFICATE

This Is to Certify That the Project Entitled, "**OTT MEDIA PLATFORMS: EDA ANALYSIS**", Is Successful Completed by **MR. PAWAN GOSAVI** As Per the Syllabus and In Partial Fulfilment for The Completion of M.Sc. In Information Technology of University of Mumbai.

It Is Also to Certify That This Is the Work of The Candidate One During the Academic Year 2020-2021.

Exam Seat No: 6909 Place: Pillai's ACS College, New Panvel

Signature of Principal / Co-Coordinator Signature of External Examiner

SEAL

DECLARATION

I Hereby Declare That the Project Entitled, "**Ott Media Platforms: Eda Analysis**" Done at Pillai's College of Arts, Commerce and Science, Has Not Been in Any Case Duplicated to Submit to Any Other University for The Award of Any Degree. To the Best of My Knowledge Other Than Me, No One Has Submitted to Any Other University.

The Project Is Done in Partial Fulfilment of The Requirement for The Award of Degree of **Master of Science (Information Technology)** To Be Submitted as Final Semester Project as Part of Our Curricula.

**MR. PAWAN AJAY GOSAVI
MSC IT PART 2 - 6909**

ACKNOWLEDGEMENT

When I Got the Opportunity to Do A Work On "**OTT MEDIA PLATFORMS: EDA ANALYSIS**" Using Machine Learning It was bit a Challenge at first for Me but was for to Do Better in Development Field.

I Took This Challenge and Finished the Work with My Hard Working. There are few Names I would like to Share that helped to Achieve this Goal. I Dedicate This Page to Thanks All of Them who Helped in in Any Way Possible, In Any Capacity.

A special thanks to our project guide, **PROF. SAGAR KULKARNI**, whose encouragement and timely assistance, helped me to co-ordinate my project report well. Choosing the domain is very important and crucial phase in any project start. Hence I would like to mention that Prof. Sagar Kulkarni helped me in choosing the right project domain. He also helped me in searching of research papers from IEEE and their studying tactics.

I Also would also like to give a special thanks to **PROF. ABHIJEET SALVI**. He has indeed been a lighthouse for me in this entire journey.

But there is More! In this Academic Year, I've made few Friends Too, Which Later then Transformed into an Unshakable Team of 4 Members – **Smart Sachin, Expert Sadiq, Genius Samidha & Me!** The Team is Called "**The Quad Squad !**" They Helped me in so Many Ways, by Giving me Ideas to QuickStart, Sachin helped me to Solve Errors in My Code, Sadiq guided in any Last-Minute Changes, & Samidha Helped me in Project Documentation, and Much More!

So, it had been Such an Honour to Work and Talk with You Guys!

Still I am well Conscious and well Aware That My Project Work Rather Than This Acknowledgement Will Be More Appropriate Way to Express Our Gratitude Towards the People Mentioned Above.

PAWAN GOSAVI, MSC. IT PART 2 - 6909

Table of Contents

#	TITLE	PAGE NO
1	CHAPTER: ONE	
	ABSTRACT	1
2	CHAPTER: TWO	
	Are the OTT's the New Future?	2
	Introduction	3
	OTT Services	4
	OTT Apps Contents and Message Distribution	4
	OTT Content & It's Distribution	5
	Fundamentals of OTT Platforms	5
	There is a Shift!	6
	Why this Shift?	7
3	CHAPTER: THREE	
	Review of Literature	8
	Introduction	8
	Conclusion	15
4	CHAPTER: FOUR	
	SURVEY	16
	Validate the Emergence of OTT	16
5	CHAPTER: FIVE	
	RESEARCH METHODOLOGY	18
	Research design	18
	Population and sampling	18
	Materials	18
	Hypothesis	18
	Significance	19
	Objectives	19

Scope	20
Primary method & Process of data collection	20
Process of data analysis	20
Limitations of this study	20
Conclusion	20
6 CHAPTER: SIX	
PYTHON / COLLAB NOTEPAD FILES	21
MOVIES	21
TV Shows	206
7 CHAPTER: SEVEN	
COLLAB NOTEPAD OUTPUTS	550
MOVIES	550
TV Shows	601
8 CHAPTER: EIGHT	
EXPLORATORY DATA ANALYSIS	645
Introduction	645
Data Sets	646
Movie.csv	646
TV.csv	646
MISSING VALUES	647
PLATFORMS	648
TITLE	649
YEAR	650
AGE	651
IMDb	652
DIRECTORS	653
GENRES	655
COUNTRY	658
LANGUAGE	659
RUNTIME	661
Language Vs Platforms	663
Language Vs Ratings	664

Genre Vs Platforms	665
Genre Vs Ratings	666
Director Vs Platforms	667
Director Vs Ratings	668
Indian Language Movies	669
Indian Movies vs OTT platforms	669
Netflix Vs Prime Movies	670
9	CHAPTER: NINE
Conclusion	673
Findings	673
Research limitations	674
Conclusion	674
10	CHAPTER: TEN
APPLICATIONS OF OTT	676
Viewster: Breaking Free	676
Voddler: Shelving & Sharing	677
Dailymotion: Open House	678
11	CHAPTER: ELEVEN
LIST OF FIGURES	679
12	CHAPTER: TWELVE
REFERENCES	681

CHAPTER: ONE

ABSTRACT

Are the OTT's the New Future?

This research aims to study the growth and future of OTT platforms. To know their future reach, it is important to know the extent of the increase in the popularity of OTT platforms during the pandemic.

It is obvious that OTT platforms have only experienced an upward curve in their popularity and use since their launch, but due to the pandemic, its popularity has increased exponentially due to the shift in consumption habits of people for entertainment across different media platforms.

This study conducted a survey and examined the views of OTT platforms, their consumption habits, and compared to cinema to see if OTT platforms were slowly taking over the most common traditional entertainment medium.

It was found that people used OTT more to spend their time or for entertainment than any other TV and YouTube channels.

EDA also carried out this research on Movies & TV Data Sets Extracted from Kaggle Through different graphs and formats, we tried to find out which platform is ideal for whom, and which platform is the best in terms of quality and quantity content.

Most of them experienced an improvement in their consumption times and looked forward to films being released at the same time as in cinemas on OTT.

People were also happy with watching movies rather than movies on OTT. But for others it was dependent on the film.

This study has shown that in the future there is a large potential for OTT channels, and the pandemic has played an important role in it.

Keywords: OTT, Pandemic, Entertainment, Movies, Television, EDA

CHAPTER: TWO

Are the OTT's the New Future?



It's a little jarring to know that next year is going to be the first year that we spend more time on the internet than watching TV.

We're in the middle of a paradigm shift. What was sold during the dynasty of cable television no longer attracts the new audience, and we have no choice but to embrace and push forward our broadcasting archaisms as casualties.

The media and broadcasting ecosystem must be considered a rolling stone, a constant and frenetic movement with a destination that we have not yet established, but enough momentum to continue to drive forward.

These platforms are becoming more and more customer oriented and has been consistent innovations to the delight of the user. One of the major innovations that OTT platforms have brought about is a feature where movies and TV shows are suggested to the user based on the content that they have watched previously with the help of analytics.

In other words, we know we are moving away from conventional bundling of cable boxes and networks, and we have uncovered a powerful alternative, but in a continuing transformation, this eclectic world is just a pit stop.

Where we are now: we're Over the Top.

Introduction:

Over-the-Top (OTT) channels are a natural evolution. They are a by-product of technological disruption, a way of combining one of the most popular, everyday conveniences with the digital revolution.

The OTT umbrella facilitates the internet distribution of film and TV content, a mix of TV and digital video to create a special, rotating source of content.

An entire lineage of OTT related products, applications, apps, content, and marketing strategies has been giving way to developments in data processing, cloud-based storage, and streaming.

Before we put a name to them there is one force for companies pondering OTT concepts that eventually shook their ideas loose and pushed them into action-Network. Like a bullet, Netflix came barreling through the architecture we designed for the distribution of content and made us say to ourselves, what now?

Well over 181 million individuals now regularly connect with OTT channels.

The rest of the broadcasting industry faced a crossroads when Netflix traded in their DVD rental service for a digital content library. Follow in their footsteps and launch similar OTT platforms, or like Blockbuster, get left behind.

The word Over-the-Top (OTT) means the distribution via the internet of audio and video streaming content to consumers without subscribing to a conventional provider of satellite services.

Netflix, Amazon Prime, Hulu, etc.; audio streaming services such as Apple Music and Spotify; and messaging apps such as Facebook, Skype, WeChat, and WhatsApp are some of the OTT video streaming platforms.

Basically, OTT is the technical equivalent of OTC (over-the-counter) medicine.

Traditionally, for their package of goods, a company offering media or telecommunications services will charge a monthly fee, prescribing what products and services you get.

Today, the a la carte style of the metaphorical media pharmacy is being opened up for clients to choose and choose which goods and services they want to pay for.

These industries are improving at a rapid pace with the day-by-day progress of technology. There were both cases of cut-throat rivalry and cases that saw businesses trying to generate goodwill in the market.

The film industry is volatile because for example, if a movie fetches more than 100-200cr, the main profits depend on the TRP of a specific television channel, or the box office collection of some movie, etc. It is assumed to be a blockbuster and hit movie, although often it is assumed to be average for those who could not meet this bar at the box office, despite the fact that certain movies may have an outstanding script or direction, etc.

There are so many ways in which this entertainment can be viewed in different forms (as described above). Even the advertisers have now grasped this idea a day and are adamant in producing fresh and eye-catching commercials for the same.

Consumers are looking for innovation and ideas out of the box that please them and make them think and remember a specific brand or product in a particular way. For instance, typically we are reminded of the pizza chain, Dominos ', when we see two domino dices. Similarly, three parallel vertical lines signify the Adidas brand when we see them.

OTT Services

Services that were historically operated offline, using tangible and even physical materials, are being disrupted by applications that are going over the top.

Historically, the stockbroker would need to be physically present at the exchange if you wanted to purchase a share of stock, to proclaim the number of shares and the price they were prepared to buy or sell. If the price was right, a counterparty will approve it.

Today, inventory and bonds are bought online and confirmed almost immediately, surpassing the conventional model and saving the vocal cords of countless traders at the same time.

Hailing a taxi is another instance of how the conventional model has been undermined by smartphone OTT apps.

There was a time when you had to physically hail a taxi to get a lift. Today, this move has been eliminated by ride-sharing apps and thus disrupted the private transportation industry.

OTT Apps Contents and Message Distribution

OTT has definitely had its effect on the companies for which the word, cable providers, was first coined.

For millions of people "cutting the cord," streaming sites such as Hulu, YouTube TV, and Netflix have been accountable, circumventing the need to pay cable operators for content distribution.

Often these are referred to as VOD (video on demand), but later on, more on that.

The Internet is used by other OTT applications as a means for connecting phone calls and sending MMS and SMS messages.

Text messages have historically been sent using cellular networks, for example, but now iMessage, WhatsApp, and others allow users to send messages anytime the computer is connected to the internet.

OTT Content & It's Distribution

OTT networks follow a distribution system based on the internet. To access the content, a customer needs compatible hardware and internet connectivity.

The following devices are needed by the content delivery system:

1. Mobile Device: It is possible to install OTT software on mobile devices.
2. Smart TV: There are pre-installed OTT apps on new smart TVs. On these TVs, you can also download such games.
3. PC: Via browsers or applications, most PCs can allow access to OTT content.
4. Digital Video Player: A number of OTT solutions are provided by Apple TV and other third-party devices. On the new gaming consoles, such content can also be accessed.

The ways in which OTT services have been monetized are different. It covers paid subscriptions, in-app advertisements, and in-app purchases.

Fundamentals of OTT Platforms

1. **Encoders:** Software encoders are used to generate adaptive bitrate (ABR) streams in formats such as MPEG, ready to be distributed on the web or tablet.
2. **Content Delivery Network (CDN):** Video content is distributed through a Content Delivery Network (CDN) to users, which is a web server network. You may opt to either purchase a third-party CDN or create your own.
3. **Content Management System (CMS):** To streamline workflows and help control customers, their subscriptions, payment gateways, monetization models including advertising, content publications, and syndication, a content management system is used.
4. **Digital Rights Management (DRM):** DRM is a method of digital licensing that is used for piracy prevention. DRM guarantees that the content is stored and distributed in an encrypted format, so that the content can be accessed only by approved users.
5. **Recommendation Engine and Analytics Engine:** In an OTT platform, an analytics engine is used to get a 360-degree view of client activities. A recommendation engine further uses this analytics data to suggest to users what to watch next.

There is a Shift!

OTT streaming networks are seen as a challenge to the satellite broadcasting industry as cord-cutting is motivated by the increased acceptance of linear OTT services. The penetration of the television market in India is at 64 per cent and is highly competitive, according to The Hindu Business Line & Economic Times.

Television networks such as Zee, Sun and Star continue to expand their programming hours in many formats, providing new material and TV shows.

However more people are watching video content online via cell phones and tablets rather than on television, with cheaper internet and less costly mobile data plans.

A recent study by KPMG-Eros Now says that almost 87 percent of daily online video content is watched through mobile phones.

Over-the-top channels like Netflix, Amazon Prime and Hotstar are seeking to attract more eyeballs to challenge the supremacy of conventional TV platforms, but this might not be as easy as it appears for them.

According to Business Wire, the total number of OTT users would hit 915 million by 2023, most of whom will be subscribing to at least one SVoD (subscription-based video-on-demand) service.

Why this Shift?

The incremental surge in over-the-top content consumption can be largely attributed to the following factors:

- Unlimited streaming material for audio and video
- Affordable rate, flexible pricing models focused either on pay-as-you-go, freemium or subscription (limited access to free content)
- A wide variety of gadgets such as smartphones, laptops, desktops, smart TVs, connected devices and set-top boxes have been offered on the go.
- It is easy to subscribe to OTT services and unsubscribe
- Access to original content for film, such as web series and recent movies
- Content-on-demand available instead of waiting to air on TV for a show or movie

So now comes the Main Question!

Are the OTT's the New Future?

The Answer is **It's Complicated!**

When Apple CEO Tim Cook announced in 2015 that "TV's future is apps," he pointed to a future where a majority of TV viewing via online apps is streamed over-the-top (OTT).

Oh, but how? & Why? Why? And why not DTH? This Paper will be answering all these Issues.

CHAPTER: THREE

Review of Literature

Introduction

The review of literature is an essential component of a research investigation which gives necessary inputs for the researcher to frame the research study on the elected topic. The basic objective of this chapter is to analyze the previous findings so that it will help to know the gap in earlier studies and to justify the research problem selected by the researcher for the study purpose.

Keeping in mind the objectives of the research, the review of related literature is organized in the following way:

2.1.1 Factors influencing the shift from traditional TV to OTT platforms in India, by Rohit Jacob Jose (Year 2020) [\[1\]](#)

This research paper concludes that user friendliness and content richness are important factors affecting the switch of customers from television to India's OTT platforms, though cost is not an important factor. While most individuals rely on TV, we can certainly see a pattern of individuals moving to OTT platforms. This change is overwhelmingly dominated by society's young people, the middle class and upper class. For OTT channels, this indicates a great future and undoubtedly the slow death of the conventional television system.

This research also highlights the factors on which OTT platforms should focus more on enhancing their satisfaction with customers and raking in more subscribers. While most platforms have continuously focused and invested in enhancing content richness and user friendliness, more needs to be done to make cost a major factor in affecting a greater change.

2.1.2 Research on the Relationship between the Growth of OTT Service Market and the Change in the Structure of the Pay-Tv Market, by Park, Sungwook; Kwon, Youngsun (Year 2019) [\[2\]](#)

As a result of analyzing the cases of major broadcasting countries, this paper found that OTT operators' typical strategies are localization strategy, collaboration strategy, content differentiation strategy, revenue enhancement strategy, and service optimization strategy.

The findings of the network effects of fixed broadband have a direct impact on the structural shift of the pay-tv industry and on cord cutting are also driven by this paper. However, the relationship between fixed broadband subscriptions and market concentration shows an inverted U-shape in two dimensions when the square independent variable 'fixed broadband subscriptions' is used in the regression test, whereas the relationship between fixed broadband subscriptions and pay-tv take-up shows a U-shape. These findings show that after a certain stage, there are opposite effects.

2.1.3 Digital Media: Rise of On-demand Content, by Hemant Joshi (Year 2019) [\[3\]](#)

This paper notes that India has the world's largest young population, which is driving India's consumption of digital media. Mobile internet users are pushing internet traffic in India. The key explanation for this would be the accessibility of cost-effective smartphones in India, expanding internet coverage for 3G and 4G and increasingly reducing data prices. This has contributed to the need for digital entertainment services such as audio and video streaming on-demand. Monetization models are still emerging for these services, however. Ecosystem players fail to define the correct models that can be scaled and experiment with different levers to arrive at the most feasible alternative, such as price points, value offers and mixed model approaches. Hybrid models have been introduced by leading digital media players where they offer a lot of content free of cost, but charge for their premium content.

This paper concludes that India's video industry is also seeing a transition to digital content. Younger generations are leading India's video consumption. In addition to standard definition video streaming online, demand for HD and UHD video content is expected to increase with increased network speeds. Like digital music players, digital video players are now embracing both subscription and ad monetization models and delivering customized services to increase adoption.

2.1.4 The Burgeoning Digital Media Consumption: A Challenge for Traditional Television and Advertising Industries, by Ritu Bhavsar (Year 2018) [\[4\]](#)

Studies have shown that the burgeoning use of digital media has a powerful impact on shifting customer tastes, perceptions and behaviours. In contrast to conventional platforms, more and more media consumption is happening on digital platforms. In addition, it can be said that advancements in technology for mobile devices and internet connectivity have given audiences the option of viewing digital media on the go. Research also advocates that OTT and VOD services have emerged as the leading online traffic drivers and are projected to increase their share of the pie by growing internet penetration and acceptance of services. Marketers are changing spending on budgets in line with the change in audience preference from conventional media to digital media.

The study noted the driving factors that are rising the consumption of digital media. India has the world's largest young population, which is driving the consumption of digital media. Mobile internet users are driving Internet traffic in India and the key explanation for this is the availability of cost-efficient smartphones and data plans, enhanced 3G and 4G high-speed mobile technology network coverage. The change in customer preference to digital media opens a new door of opportunities for advertisers, suppliers of content and new players in the media. This raises the need for more personalized, individual-centred content that is highly engaging and innovative.

2.1.5 How OTT platforms can remain 'on-demand ready', by Apurva Dixit & 17 Others (Year 2017) [\[5\]](#)

We are now in a relentless climate of transition, according to this article, which requires companies to constantly reconsider their procedures and strategies. Digital transition is not a one-off effort. The digital evolutionary path and ideal digital end state will have to be periodically evaluated by every organization considering digital transformation.

The progress of digital transformation is primarily determined by the ability of the company to keep pace with technological advancement, evolving consumer demands and last but not least, the capacity of top management to achieve a radical shift in the DNA of the organization. It is no longer possible to treat digital as an additional feature, but digital strategy is rather inherent in business strategy.

2.1.6 Comparative Study of Viewers' Behaviour Over Traditional Television Channels and Over Ott Video Platforms in Maharashtra, by Ripal Madhani & Vidya Nakhate (Year 2020) [\[6\]](#)

OTT video channels are significantly becoming part of the entertainment time of audiences, according to this article, and they are giving conventional modes tough competition.

The flexibility of time and place, the availability of reliable and affordable data connections, the penetration of smart phones, the availability of cheap and even free access to OTT video platforms, the sheer variety of content to choose from and the quality of content are some of the key factors motivating viewers to migrate to OTT video platforms.

Traditional TV channels, however, will not be totally replaced by OTT video networks, at least in the near future, and they will co-exist. Traditional TV channels still have a chink of loyal viewers, with some improvements in the quality of programming and tactics that can still draw customers and thrive in the competitive period.

2.1.7 The Apple of Digital Television, by Twentify (Year 2018) [\[7\]](#)

3 Key Observations This Article Points out, first one, Netflix is king, so it's best to pay attention. We see that Netflix has a stickiness score of 79 percent when we look at the stickiness of OTT services, led by Amazon with 68 percent and Hulu with 50 percent. On the other hand, we see that 94.37% of respondents had previously used Netflix, where Hulu was used by just 66% of them, and Amazon Prime Video was used by 52.52%.

In order to draw the young crowd, Amazon should revise its content strategy. Amazon has achieved the peak audience penetration between the ages of 36-44. The penetration rate of Amazon in the age group 25-35 is 52.81 percent, but it is 45 percent at 18 < and 18-24. We see a full shift when comparing these numbers with Netflix.

Usage trends for Netflix begin to decline after age 35. Yeah, older adults may have trouble migrating from cable TV to OTT media services, but this tells us another thing: for the young crowd, Amazon is not enticing enough.

Their stickiness must work on Hulu and YouTube. YouTube TV and Hulu are aggregators, both of which are seeking to extend original material to their libraries. We see a 66.00 percent scope, with 50 percent stickiness, when we search individuals who have used Hulu at least once. We see a 49.70 percent scope, and 35 percent stickiness, when YouTube TV is measured.

2.1.8 The Rise of Over-the-Top Content: Implications for Television Advertising in a Direct-to-Consumer World, by Tata Consultancy Services (Year 2017) [\[8\]](#)

According to this report, attracting customer interest is the holy grail for marketers in an excessively competitive business climate that exists today. And marketers need to reimagine their customer outreach campaigns, with over-the-top television helping viewers access their favourite content on demand.

Until growing into a mature field, OTT advertising is still in a nascent stage, and has a long way to go. In all likelihood, some of the basic standard television advertising techniques will underpin it thus integrating the new aspects of media measurement across different multichannel platforms.

The true currency that will push customer interest in the future is relevance, both on the content and advertisement fronts. Therefore, for better performance, OTT operators need to work in tandem with advertisers to test, iterate, and improve their ad rendering capabilities.

2.1.9 Indian OTT platforms: Streaming the new age narrative, by MICA (Year 2018) [\[9\]](#)

According to this paper, India's total entertainment and media spending has risen at a compound annual growth rate (CAGR) of 11.6 percent over the last five years, narrowly ahead of China's 10.9 percent, and more than double the 5.0 percent CAGR expected worldwide for entertainment and media. India's growth in spending will be driven by digitally driven industries such as Internet ads (20.4 percent CAGR), video games (16.7 percent) and Internet connectivity (15.0 percent). India is increasingly becoming a market where the new and the old co-exist. While the categories of television and filmed entertainment have been increasing in the country, digital advertising has also shown growth in the past year. While the majority of digital revenue comes from ads, thanks to international players and their network strategies, subscription revenue has seen steady growth.

OTT video channels are significantly becoming part of the entertainment time of audiences, according to this article, and they are giving conventional modes tough competition. The flexibility of time and place, the availability of reliable and affordable data connections, the penetration of smart phones, the availability of cheap and even free access to OTT video platforms, the sheer variety of content to choose from and the quality of content are some of the key factors motivating viewers to migrate to OTT video platforms.

Traditional TV channels, however, will not be totally replaced by OTT video networks, at least in the near future, and they will co-exist. Traditional TV channels still have a chink of loyal viewers, with some improvements in the quality of programming and tactics that can still draw customers and thrive in the competitive period.

2.1.10 The emergence of OTT platforms during the pandemic and its future scope, by Bhargav Pancholi (Year 2020) [\[10\]](#)

According to this paper, In the preceding paragraphs, the data obtained from the questionnaire survey was interpreted and analysed. The thesis on the subject 'The advent of OTT platforms & their future reach' has been completed. The theory that there was a large increase in OTT platform consumption during the pandemic has been proved correct. Since individuals in the lockdown began to consume OTT more and more. It has been demonstrated that OTT platforms have undergone major inorganic growth by eating up the market share of other platforms.

Although the researcher could not prove the hypothesis that OTT will overtake cinema, we can see that there is suddenly an increase in the use of OTT over other media and people have positive responses to movies that will be released at the same time as in cinemas on OTTs. This shows that even though cinema cannot be replaced by OTT platforms, it

definitely creates its own segment. We may assume that there will be very few people in the future who would choose OTT over cinema.

2.1.11 Understanding adoption factors of Over-The-Top video services among millennial consumers, by Sabyasachi Dasgupta & Priya Grover (Year 2019) [\[11\]](#)

According to this paper research: 25 out of 35 respondents said OTT channels were much better than conventional media vehicles for video consumption, such as television. OTT's future looks promising with the internet and smartphone penetration growing by the day. It is extremely difficult for OTT channels to sustain a steady streamline of sales and are therefore highly dependent on paying subscribers for their profits. According to Toni Kothari, marketers need to tell customers that Ott channels are part of their convenience need that has become part and parcel of daily life.

OTT channels are more aligned with the millennial lifestyle. Television will redefine another target demographic as a market that automatically leaves Young viewers on-the-go media like OTT. There is a significant need for the entire OTT business structure in India to be streamlined. OTT networks may not always be able to provide an integrated forum for video content free of charge.

2.1.12 Impact of Over the Top (OTT) Services on Telecom Service Providers, by Joshi Sujata & 5 others (Year 2015) [\[12\]](#)

According to this paper, Traditional telecom operator revenue streams, focused mainly on subscriptions and metered services, show signs of being obsolete. OTT services such as WhatsApp, Skype and Netflix do not contribute to the direct profits of providers of connectivity or to the tax revenues of the government. However, they allow use of communication networks which need additional investment in the network.

The government and regulatory stance towards them, the authors conclude, will be the most decisive factor in the growth of the OTT service. The authors claim that high-speed Internet access, the possibilities it provides for new business models such as OTTs to grow, practically forecast a technological transition that the government should embrace, they conclude. The writers suggest that this process should be encouraged by governments and should not be enforced by policies that could impede it.

2.1.13 The Future of Online OTT Entertainment Services in India, by Quresh Moochhala (Year 2018) [\[13\]](#)

According to this paper, India is a country sensitive to prices and culture, and the price gap between OTT and cable TV is important, which is why television viewership will also

continue to rise at the same time. The paper highlights the fact that by concentrating on regional content, the secret to unlocking the digital market in India is that the audience of programs in English is smaller.

With the huge Indian market far from saturation, the growth of the user base and digital ads promises to be good for all players in the entertainment industry. Consistent development over the past few years means that there are OTT streaming services here to continue offering Indians a new alternative to conventional cable television and cinema. Entertainment is obviously in the process of being rapidly re-branded in India for digital entertainment.

2.1.14 Proliferation of OTT apps in India: an empirical study of OTT apps and its impact on college students, by Reshma & Chaithra (Year 2020) [\[14\]](#)

It was concluded that, according to the findings, one of the biggest pivots in the entertainment world this century was the rise of streaming apps, especially among students. For both their personal and academic purposes, the students used the OTT platform.

The results have shown that streaming movies and online shows have become a culture among students. And all these factors are focused on the services offered to consumers and the day-to-day addition of more and more subscribers as the rivalry has intensified and people's psychology is understood to draw their attention to their streaming apps. In terms of entertainment and leisure spending, streaming apps will build a great legacy and continue to remain a top choice.

2.1.15 OTT Viewership in “Lockdown” and Viewer’s Dynamic Watching Experience, by Manoj Kumar Patel & 2 Others (Year 2020) [\[15\]](#)

According to this Author, the new generation does not have the ability to wait on a linear channel like television for a show or movie to air. Just like Maggi noodles - Masaledaar, instant and on-demand this viewer wants her stuff. That is what our study shows that the growth of OTT will only increase in India due to some such factors. The only thing we have known is that the OTT video streaming service will continue to expand its feet in India, and it will have a significant effect on our conventional medium, such as TV and Cinema Hall. All the studies and articles we have read and go through. This reality has further reinforced the strong connection of OTT to the audience in the lockdown era. Smartphone penetration, foreign alliances between media moguls and the media's digital quality. One of the factors behind the growth of streaming media in India is cost-effectiveness and freedom of access (anywhere at any time).

Dynamic viewing habits can alter the collective watching experience in other findings about the viewers and also affect the future footfalls of the cinema hall. As shown by the report, audiences have the same way of thinking about the cinema hall's future. There may be decrease or may be not in the future footfalls of cinema hall after this lockdown time, it all depends on the viewers who are still in dilemma about this. We have to say that the owners of the cinema hall should think of more convenience and deals, and it must also be cost-worthy. As we see OTT spreading very rapidly, its influence on other conventional mediums will be very profound.

Conclusion

The above chapter discusses the studies carried out on a subject almost identical to that taken by the researcher, and the above studies have provided some new aspects of the issue. Studies have presented figures and information about the processing of their data and thus provide good justification for their results.

CHAPTER: FOUR

SURVEY

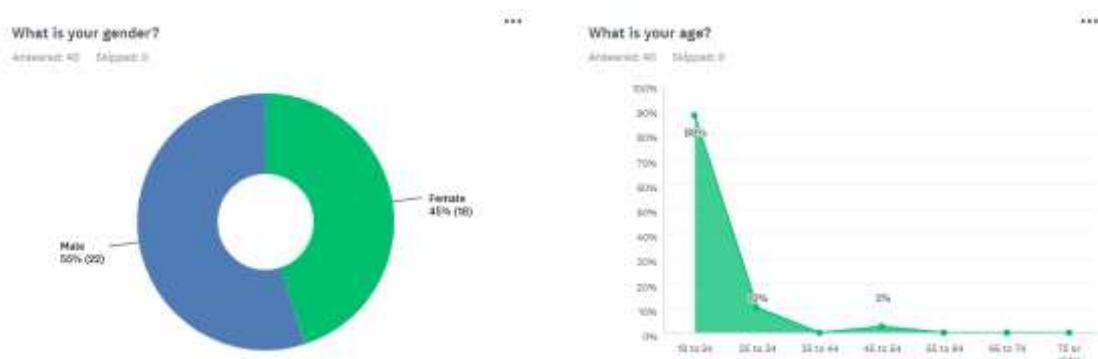
Validate the Emergence of OTT

A personal Level Survey was also conducted to Validate the Superiority of OTT's, the significance of the study is to highlight the nature and scale of the consumption of OTT platforms among people. This research will help us to understand how OTT outlets during a pandemic have taken over the global entertainment landscape. It will help us to understand how OTT has evolved inorganically and gradually taken over other entertainment media.

By using the quantitative approach, primary data was obtained through Monkey Survey. There was a survey performed. The respondents are from the 18-65+ age group. The overall sample size taken into account in the analysis was 100 responses. The questionnaire was divided into segments such as: demographics and behaviour of OTT consumption, which included questions related to the use of OTT, past habits and their opinions.

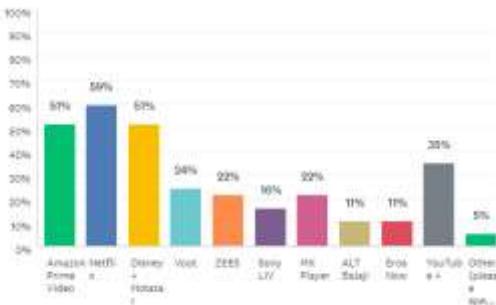
- Q1 What is your gender?
- Q2 What is your age?
- Q3 OTT Platforms Means Over-The-Top Video Streaming Services like Netflix, Prime Video etc. Which of these Services You are familiar with & Use or Have Paid Subscription?
- Q4 Which Kind of Network Connection do You Prefer?
- Q5 Which kind of Device you will Prefer to Stream or Watch Content?
- Q6 What kind of Streaming Content You Watch / Prefer?
- Q7 How much do you spend on the Subscriptions per Month?
- Q8 How often do you use Streaming Services?
- Q9 Do Movie / TV Show's Ratings Matters to You?
- Q10 Which Video Streaming Service is Best According to Your Knowledge?

The Results are as Follows:



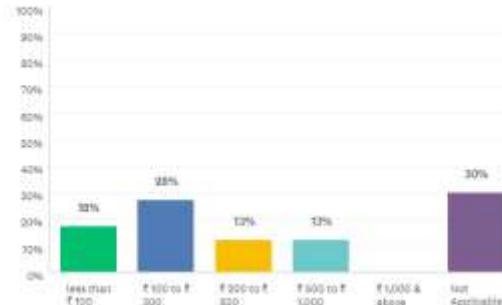
OTT Platforms Means Over-The-Top Video Streaming Services like ...

Answered: 27 Skipped: 3



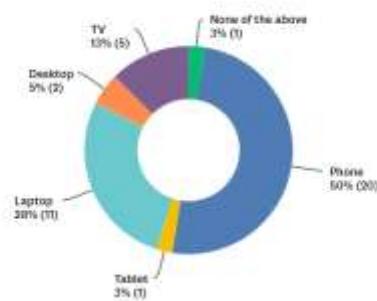
How much do you spend on the Subscriptions per Month?

Answered: 40 Skipped: 0



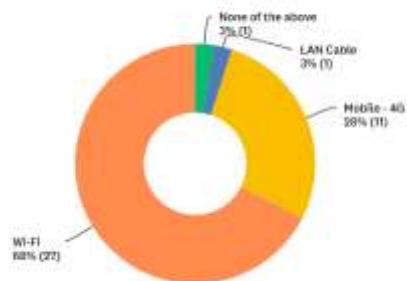
Which kind of Device you will Prefer to Stream or Watch Content?

Answered: 40 Skipped: 0



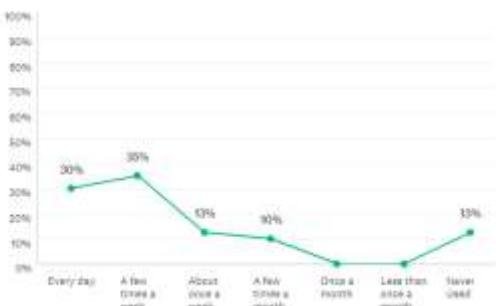
Which Kind of Network Connection do You Prefer?

Answered: 40 Skipped: 0



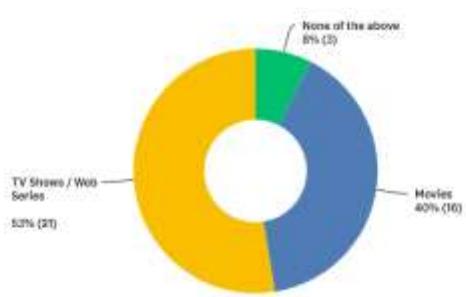
How often do you use Streaming Services?

Answered: 40 Skipped: 0



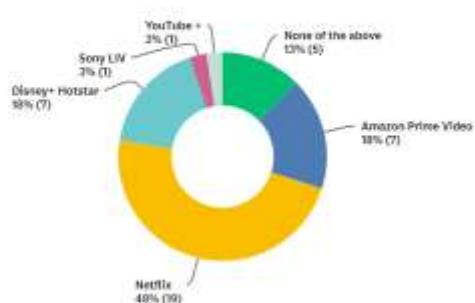
What kind of Streaming Content You Watch / Prefer?

Answered: 40 Skipped: 0



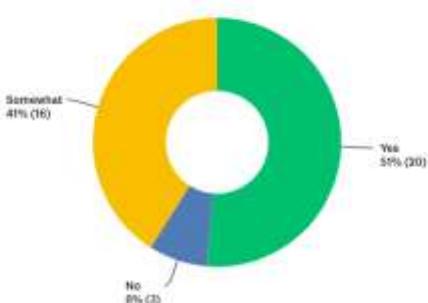
Which Video Streaming Service Is Best According to Your Knowledge...

Answered: 40 Skipped: 0



Do Movie / TV Show's Ratings Matters to You?

Answered: 39 Skipped: 1



CHAPTER: FIVE

RESEARCH METHODOLOGY

Research design

The purpose of this study was to understand the emergence of OTT platforms and its future scope and to achieve this, hypothesis was established. EDA method was selected for the same. The focus was on the Data Set Retrieved from Kaggle.

Population and sampling

The Data was Retrieved for OTT Services like Netflix Amazon Prime, Hulu & Hotstar. IMDB & Rotten Tomato's Rating was also picked up.

Although the Data has some Poor Entries. that was Cleaned through Data Cleaning process.

Materials

The qualitative and quantitative research sample in which we carried out the study using the form of the survey was adopted.

For data collection, a questionnaire was distributed among the volunteers consisting of 10 questions related to the studies.

There were several choices in the test and the respondent had to tick one of them, which was most applicable to them.

Hypothesis

OTT platforms have experienced a substantial inorganic growth by taking up the market share of other platforms. OTT platform will soon become a platform for releasing movies or maybe even surpass cinema. (which earlier release in cinema)

The entire analysis is focused on the search volume of the system level and users for OTT applications. India is the smart phone industry's second largest and fastest growing

market. Overall, the competitive dominance of OTT TV surpasses that of all aspects of conventional TV.

Finally, in terms of program styles, news, movies, and sports effectively predict the satisfaction of cable TV users, whereas dramas and films predict the satisfaction of OTT TV users.

Significance

The significance of the study is to highlight the scope of OTT platforms and the extent of their consumption amongst the people. These studies will help us in understanding how OTT platforms have taken over the world of digital entertainment during a pandemic. It will help us understand how OTT has inorganically growing and gradually taking over other mediums of entertainment.

Objectives

1. To find out and understand the increase in consumption of OTT platforms.
2. To understand the future scope of the OTT platforms.
3. To study the shift in consumption from old entertainment to new.
4. Perform a Survey on OTT Platforms.
5. Use the Data and do Statistical Analysis.
6. Gain raw data sets for Few Services from Kaggle.
7. Perform Machine learning on raw data.
8. Give Visual Analysis by doing EDA on the Same data.
9. Select few Metrix to decide Performance of Services.
10. Predict whether this Service will be the Future or not by current Insights.
11. Tell which OTT is the Beast and Provide Better Conclusion.

Scope

The goal of this paper is to explore and understand the definition of OTT apps and their rapid growth in India, to define the relationship between Freemium and premium and to define the leading OTT app, to identify and analyze the effect of OTT apps on college students (academic and personal life), to decide which regional or international shows among students are more preferred.

Primary method & Process of data collection

Primary data was collected by applying the quantitative method. the Raw data was downloaded from Kaggle. for Each Streaming Services the Data like Title, Year was used. then that data was combined with Others Streams Data in a single file.

Process of data analysis

The Data was then Processed on the Jupyter / Python. the Data was then cleaned and after that We separated TV Data from Movie Data.

Limitations of this study

There are certain limitations of the tool which are as follows: -

1. Data collection was restricted to third Party Availability.
2. The major limitation of this study is due to time constraint and also a limited data is available for current Year.

Conclusion

This chapter includes all the aspects that were taken into consideration for data collection and analysis. Objectives and hypothesis of the research have also been explained here.

CHAPTER: SIX

PYTHON / COLLAB NOTEPAD FILES

MOVIES

ottmovies_age.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```

# In[4]:


nltk.download('all')


# In[5]:


# path = '/content/drive/MyDrive/Files/'


path = 'C:\\Users\\pawan\\OneDrive\\Desktop\\ott\\Data\\'


df_movies = pd.read_csv(path + 'ottmovies.csv')


df_movies.head()


# In[6]:


# profile = ProfileReport(df_movies)
# profile


# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('*'*25)
    print('Columns Names : \n', df.columns)
    print('*'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('*'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('*'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('*'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()


# In[8]:


data_investigate(df_movies)


# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'

```

```

# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider

```

```

df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix', 'Prime Video', 'Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]: 

data_investigate(df_movies)

# In[11]: 

df_movies.head()

# In[12]: 

df_movies.describe()

# In[13]: 

df_movies.corr()

# In[14]: 

# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]: 

# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]: 

# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]: 

```

```

df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] ==
0) & (df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]
```

In[18]:

```
df_movies_age = df_movies.copy()
```

In[19]:

```
df_movies_age.drop(df_movies_age.loc[df_movies_age['Age'] == "NA"].index, inplace = True)
df_movies_age.drop(df_movies_age.loc[df_movies_age['Age'] == "NR"].index, inplace = True)
# df_movies_age = df_movies_age[df_movies_age.Age != "NA"]
df_movies_age['Age'] = df_movies_age['Age'].astype(int)
```

In[20]:

```
# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_age_movies = df_movies_age.loc[df_movies_age['Netflix'] == 1]
hulu_age_movies = df_movies_age.loc[df_movies_age['Hulu'] == 1]
prime_video_age_movies = df_movies_age.loc[df_movies_age['Prime Video'] == 1]
disney_age_movies = df_movies_age.loc[df_movies_age['Disney+'] == 1]
```

In[21]:

```
df_movies_age_group = df_movies_age.copy()
```

In[22]:

```
plt.figure(figsize = (10, 10))
corr = df_movies_age.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

In[23]:

```
df.age_all_movies = df_movies_age
print('\nMovies with Age Rating are : \n')
```

```

df_age_all_movies.head(5)

# In[24]: 

df_age_0_movies = df_movies_age.loc[df_movies_age['Age'] == 0]
print('\nMovies with All Age Rating are : \n')
df_age_0_movies.head(5)

# In[25]: 

df_age_7_movies = df_movies_age.loc[df_movies_age['Age'] == 7]
print('\nMovies with 7+ Age Rating are : \n')
df_age_7_movies.head(5)

# In[26]: 

df_age_13_movies = df_movies_age.loc[df_movies_age['Age'] == 13]
print('\nMovies with 13+ Age Rating are : \n')
df_age_13_movies.head(5)

# In[27]: 

df_age_16_movies = df_movies_age.loc[df_movies_age['Age'] == 16]
print('\nMovies with 16+ Age Rating are : \n')
df_age_16_movies.head(5)

# In[28]: 

df_age_18_movies = df_movies_age.loc[df_movies_age['Age'] == 18]
print('\nMovies with 18+ Age Rating are : \n')
df_age_18_movies.head(5)

# In[29]: 

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_age['Age'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_movies_age['Age'], ax = ax[1])
plt.show()

# In[30]: 

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Age s Per Platform')

# Plotting the information from each dataset into a histogram

```

```

sns.histplot(prime_video_age_movies['Age'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_age_movies['Age'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_age_movies['Age'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_age_movies['Age'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[31]:


def round_val(data):
    if str(data) != 'nan':
        return round(data)

# In[32]:


df_movies_age_group['Age Group'] = df_movies_age['Age'].apply(round_val)

age_values = df_movies_age_group['Age Group'].value_counts().sort_index(ascending = False).tolist()
age_index = df_movies_age_group['Age Group'].value_counts().sort_index(ascending = False).index

# age_values, age_index

# In[33]:


age_group_count = df_movies_age_group.groupby('Age Group')['Title'].count()
age_group_movies = df_movies_age_group.groupby('Age Group')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
age_group_data_movies = pd.concat([age_group_count, age_group_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count'})
age_group_data_movies = age_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[34]:


# Age Group with Movies Counts - All Platforms Combined
age_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[35]:


age_group_data_movies.sort_values(by = 'Age Group', ascending = False)

# In[36]:


fig = px.bar(y = age_group_data_movies['Movies Count'],
              x = age_group_data_movies['Age Group'],
              color = age_group_data_movies['Age Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Age : '},
              title = 'Movies with Group Age : All Platforms')

```

```

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[37]:


fig = px.pie(age_group_data_movies,
              names = age_group_data_movies['Age Group'],
              values = age_group_data_movies['Movies Count'],
              color = age_group_data_movies['Movies Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on Age Group')
fig.show()

# In[38]:


df_age_group_high_movies = age_group_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_age_group_high_movies = df_age_group_high_movies.drop(['index'], axis = 1)
# filter = (age_group_data_movies['Movies Count'] == (age_group_data_movies['Movies Count'].max()))
# df_age_group_high_movies = age_group_data_movies[filter]

# highest_rated_movies = age_group_data_movies.loc[age_group_data_movies['Movies Count'].idxmax()]

# print('\nAge with Highest Ever Movies Count are : All Platforms Combined\n')
df_age_group_high_movies.head(5)

# In[39]:


df_age_group_low_movies = age_group_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_age_group_low_movies = df_age_group_low_movies.drop(['index'], axis = 1)
# filter = (age_group_data_movies['Movies Count'] == (age_group_data_movies['Movies Count'].min()))
# df_age_group_low_movies = age_group_data_movies[filter]

# print('\nAge with Lowest Ever Movies Count are : All Platforms Combined\n')
df_age_group_low_movies.head(5)

# In[40]:


print(f'''
      Total '{df_movies_age['Age'].count()}' Titles are available on All Platforms, out of
      which\n
      You Can Choose to see Movies from Total '{age_group_data_movies['Age Group'].unique().shape[0]}' Age Group, They were Like this, \n
      {age_group_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Age Group']].unique()} etc. \n
      The Age Group with Highest Movies Count have '{age_group_data_movies['Movies Count'].max()}' Movies Available is '{df_age_group_high_movies['Age Group'][0]}', &\n
      The Age Group with Lowest Movies Count have '{age_group_data_movies['Movies Count'].min()}' Movies Available is '{df_age_group_low_movies['Age Group'][0]}'
      ''')

```

```
'''
```

```
# In[41]:
```

```
netflix_age_group_movies = age_group_data_movies[age_group_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_age_group_movies = netflix_age_group_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_age_group_high_movies = df_age_group_high_movies.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_age_group_high_movies = netflix_age_group_high_movies.drop(['index'], axis = 1)

netflix_age_group_low_movies = df_age_group_high_movies.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_age_group_low_movies = netflix_age_group_low_movies.drop(['index'], axis = 1)

netflix_age_group_high_movies.head(5)
```

```
# In[42]:
```

```
hulu_age_group_movies = age_group_data_movies[age_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_age_group_movies = hulu_age_group_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_age_group_high_movies = df_age_group_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_age_group_high_movies = hulu_age_group_high_movies.drop(['index'], axis = 1)

hulu_age_group_low_movies = df_age_group_high_movies.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_age_group_low_movies = hulu_age_group_low_movies.drop(['index'], axis = 1)

hulu_age_group_high_movies.head(5)
```

```
# In[43]:
```

```
prime_video_age_group_movies = age_group_data_movies[age_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_age_group_movies = prime_video_age_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_age_group_high_movies = df_age_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_age_group_high_movies = prime_video_age_group_high_movies.drop(['index'], axis = 1)

prime_video_age_group_low_movies = df_age_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_age_group_low_movies = prime_video_age_group_low_movies.drop(['index'], axis = 1)

prime_video_age_group_high_movies.head(5)
```

```
# In[44]:
```

```

disney_age_group_movies = age_group_data_movies[age_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_age_group_movies = disney_age_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_age_group_high_movies = df_age_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_age_group_high_movies = disney_age_group_high_movies.drop(['index'], axis = 1)

disney_age_group_low_movies = df_age_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_age_group_low_movies = disney_age_group_low_movies.drop(['index'], axis = 1)

disney_age_group_high_movies.head(5)

```

In[45]:

```

print(f'''
    The Age Group with Highest Movies Count Ever Got is '{df_age_group_high_movies['Age Group'][0]}' : '{df_age_group_high_movies['Movies Count'].max()}'\n
    The Age Group with Lowest Movies Count Ever Got is '{df_age_group_low_movies['Age Group'][0]}' : '{df_age_group_low_movies['Movies Count'].min()}'\n

    The Age Group with Highest Movies Count on 'Netflix' is
'{netflix_age_group_high_movies['Age Group'][0]}' :
'{netflix_age_group_high_movies['Netflix'].max()}'\n
    The Age Group with Lowest Movies Count on 'Netflix' is
'{netflix_age_group_low_movies['Age Group'][0]}' :
'{netflix_age_group_low_movies['Netflix'].min()}'\n

    The Age Group with Highest Movies Count on 'Hulu' is
'{hulu_age_group_high_movies['Age Group'][0]}' :
'{hulu_age_group_high_movies['Hulu'].max()}'\n
    The Age Group with Lowest Movies Count on 'Hulu' is '{hulu_age_group_low_movies['Age Group'][0]}' : '{hulu_age_group_low_movies['Hulu'].min()}'\n

    The Age Group with Highest Movies Count on 'Prime Video' is
'{prime_video_age_group_high_movies['Age Group'][0]}' :
'{prime_video_age_group_high_movies['Prime Video'].max()}'\n
    The Age Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_age_group_low_movies['Age Group'][0]}' :
'{prime_video_age_group_low_movies['Prime Video'].min()}'\n

    The Age Group with Highest Movies Count on 'Disney+' is
'{disney_age_group_high_movies['Age Group'][0]}' :
'{disney_age_group_high_movies['Disney+'].max()}'\n
    The Age Group with Lowest Movies Count on 'Disney+' is
'{disney_age_group_low_movies['Age Group'][0]}' :
'{disney_age_group_low_movies['Disney+'].min()}'\n
    ''')

```

In[46]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.barplot(x = netflix_age_group_movies['Age Group'], y =
netflix_age_group_movies['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_a_ax2 = sns.barplot(x = hulu_age_group_movies['Age Group'], y =
hulu_age_group_movies['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_a_ax3 = sns.barplot(x = prime_video_age_group_movies['Age Group'], y =
prime_video_age_group_movies['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])

```

```

d_a_ax4 = sns.barplot(x = disney_age_group_movies['Age Group'], y =
disney_age_group_movies['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])
h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])
d_a_ax4.title.set_text(labels[3])

plt.show()

```

In[47]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = age_group_data_movies['Age Group'], y = age_group_data_movies['Netflix'],
color = 'red')
sns.lineplot(x = age_group_data_movies['Age Group'], y = age_group_data_movies['Hulu'],
color = 'lightgreen')
sns.lineplot(x = age_group_data_movies['Age Group'], y = age_group_data_movies['Prime
Video'], color = 'lightblue')
sns.lineplot(x = age_group_data_movies['Age Group'], y = age_group_data_movies['Disney+'],
color = 'darkblue')
plt.xlabel('Age Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

```

In[48]:

```

print(f'''
    Across All Platforms Total Count of Age Group is '{age_group_data_movies['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Netflix' is '{netflix_age_group_movies['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Hulu' is '{hulu_age_group_movies['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Prime Video' is '{prime_video_age_group_movies['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Disney+' is '{disney_age_group_movies['Age
Group'].unique().shape[0]}'
    ''')

```

In[49]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.lineplot(y = age_group_data_movies['Age Group'], x =
age_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_a_ax2 = sns.lineplot(y = age_group_data_movies['Age Group'], x =
age_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_a_ax3 = sns.lineplot(y = age_group_data_movies['Age Group'], x =
age_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_a_ax4 = sns.lineplot(y = age_group_data_movies['Age Group'], x =
age_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])
h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])

```

```

d_a_ax4.title.set_text(labels[3])
plt.show()

# In[50]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.barplot(x = age_group_data_movies['Age Group'], y =
age_group_data_movies['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_a_ax2 = sns.barplot(x = age_group_data_movies['Age Group'], y =
age_group_data_movies['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_a_ax3 = sns.barplot(x = age_group_data_movies['Age Group'], y =
age_group_data_movies['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_a_ax4 = sns.barplot(x = age_group_data_movies['Age Group'], y =
age_group_data_movies['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])
h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])
d_a_ax4.title.set_text(labels[3])

plt.show()

```

ottmovies_cast.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np

```

```

import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_movies = pd.read_csv(path + 'ottmovies.csv')

df_movies.head()

# In[6]:


# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))

```

```

sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

# In[8]:

data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Casts
# df_movies = df_movies.drop(['Casts'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)

```

```

df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

```

```

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# In[16]:


# df_netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df_hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df_prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df_disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]


# In[17]:


df_netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df_disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]


# In[18]:


df_movies_casts = df_movies.copy()


# In[19]:


df_movies_casts.drop(df_movies_casts.loc[df_movies_casts['Cast'] == "NA"].index, inplace = True)
# df_movies_casts = df_movies_casts[df_movies_casts.Cast != "NA"]
# df_movies_casts['Cast'] = df_movies_casts['Cast'].astype(str)


# In[20]:


df_movies_count_casts = df_movies_casts.copy()


# In[21]:


df_movies_cast = df_movies_casts.copy()


# In[22]:


# Create casts dict where key=name and value = number of casts

casts = {}

for i in df_movies_count_casts['Cast'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        casts[i] = len(i.split(','))
    else:

```

```

casts[i] = 0

# Add this information to our dataframe as a new column

df_movies_count_casts['Number of Casts'] =
df_movies_count_casts['Cast'].map(casts).astype(int)

# In[23]:


df_movies_mixed_casts = df_movies_count_casts.copy()

# In[24]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_casts_movies = df_movies_count_casts.loc[df_movies_count_casts['Netflix'] == 1]
hulu_casts_movies = df_movies_count_casts.loc[df_movies_count_casts['Hulu'] == 1]
prime_video_casts_movies = df_movies_count_casts.loc[df_movies_count_casts['Prime Video']
== 1]
disney_casts_movies = df_movies_count_casts.loc[df_movies_count_casts['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_movies_count_casts.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_casts_most_movies = df_movies_count_casts.sort_values(by = 'Number of Casts', ascending
= False).reset_index()
df_casts_most_movies = df_casts_most_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_casts['Number of Casts'] == (df_movies_count_casts['Number of
Casts'].max()))
# df_casts_most_movies = df_movies_count_casts[filter]

# mostest_rated_movies = df_movies_count_casts.loc[df_movies_count_casts['Number of
Casts'].idxmax()]

print('\nMovies with Highest Ever Number of Casts are : \n')
df_casts_most_movies.head(5)

# In[27]:


fig = px.bar(y = df_casts_most_movies['Title'][:15],
              x = df_casts_most_movies['Number of Casts'][:15],

```

```

color = df_casts_most_movies['Number of Casts'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
title = 'Movies with Highest Number of Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[28]:


df_casts_least_movies = df_movies_count_casts.sort_values(by = 'Number of Casts', ascending = True).reset_index()
df_casts_least_movies = df_casts_least_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_casts['Number of Casts'] == (df_movies_count_casts['Number of Casts'].min()))
# df_casts_least_movies = df_movies_count_casts[filter]

print('\nMovies with Lowest Ever Number of Casts are : \n')
df_casts_least_movies.head(5)

# In[29]:


fig = px.bar(y = df_casts_least_movies['Title'][:15],
              x = df_casts_least_movies['Number of Casts'][:15],
              color = df_casts_least_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Lowest Number of Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
      Total '{df_movies_count_casts['Number of Casts'].unique().shape[0]}' unique Number of Casts s were Given, They were Like this,\n
      {df_movies_count_casts.sort_values(by = 'Number of Casts', ascending = False)['Number of Casts'].unique()}\n\n

      The Highest Number of Casts Ever Any Movie Got is
      '{df_casts_most_movies['Title'][0]}' : '{df_casts_most_movies['Number of Casts'].max()}'\n

      The Lowest Number of Casts Ever Any Movie Got is
      '{df_casts_least_movies['Title'][0]}' : '{df_casts_least_movies['Number of Casts'].min()}'\n
      ''')

# In[31]:


netflix_casts_most_movies =
df_casts_most_movies.loc[df_casts_most_movies['Netflix']==1].reset_index()
netflix_casts_most_movies = netflix_casts_most_movies.drop(['index'], axis = 1)

netflix_casts_least_movies =
df_casts_least_movies.loc[df_casts_least_movies['Netflix']==1].reset_index()
netflix_casts_least_movies = netflix_casts_least_movies.drop(['index'], axis = 1)

```

```

netflix_casts_most_movies.head(5)

# In[32]:


fig = px.bar(y = netflix_casts_most_movies['Title'][:15],
              x = netflix_casts_most_movies['Number of Casts'][:15],
              color = netflix_casts_most_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Highest Number of Casts : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_casts_least_movies['Title'][:15],
              x = netflix_casts_least_movies['Number of Casts'][:15],
              color = netflix_casts_least_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Lowest Number of Casts : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


hulu_casts_most_movies =
df_casts_most_movies.loc[df_casts_most_movies['Hulu']==1].reset_index()
hulu_casts_most_movies = hulu_casts_most_movies.drop(['index'], axis = 1)

hulu_casts_least_movies =
df_casts_least_movies.loc[df_casts_least_movies['Hulu']==1].reset_index()
hulu_casts_least_movies = hulu_casts_least_movies.drop(['index'], axis = 1)

hulu_casts_most_movies.head(5)

# In[35]:


fig = px.bar(y = hulu_casts_most_movies['Title'][:15],
              x = hulu_casts_most_movies['Number of Casts'][:15],
              color = hulu_casts_most_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Highest Number of Casts : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_casts_least_movies['Title'][:15],
              x = hulu_casts_least_movies['Number of Casts'][:15],
              color = hulu_casts_least_movies['Number of Casts'][:15],

```

```

color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
title = 'Movies with Lowest Number of Casts : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_casts_most_movies = df_casts_most_movies.loc[df_casts_most_movies['Prime
Video']==1].reset_index()
prime_video_casts_most_movies = prime_video_casts_most_movies.drop(['index'], axis = 1)

prime_video_casts_least_movies = df_casts_least_movies.loc[df_casts_least_movies['Prime
Video']==1].reset_index()
prime_video_casts_least_movies = prime_video_casts_least_movies.drop(['index'], axis = 1)

prime_video_casts_most_movies.head(5)

# In[38]:


fig = px.bar(y = prime_video_casts_most_movies['Title'][:15],
              x = prime_video_casts_most_movies['Number of Casts'][:15],
              color = prime_video_casts_most_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Highest Number of Casts : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[39]:


fig = px.bar(y = prime_video_casts_least_movies['Title'][:15],
              x = prime_video_casts_least_movies['Number of Casts'][:15],
              color = prime_video_casts_least_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Lowest Number of Casts : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


disney_casts_most_movies =
df_casts_most_movies.loc[df_casts_most_movies['Disney+']==1].reset_index()
disney_casts_most_movies = disney_casts_most_movies.drop(['index'], axis = 1)

disney_casts_least_movies =
df_casts_least_movies.loc[df_casts_least_movies['Disney+']==1].reset_index()
disney_casts_least_movies = disney_casts_least_movies.drop(['index'], axis = 1)

disney_casts_most_movies.head(5)

# In[41]:

```

```

fig = px.bar(y = disney_casts_most_movies['Title'][:15],
              x = disney_casts_most_movies['Number of Casts'][:15],
              color = disney_casts_most_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Highest Number of Casts : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_casts_least_movies['Title'][:15],
              x = disney_casts_least_movies['Number of Casts'][:15],
              color = disney_casts_least_movies['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Casts'},
              title = 'Movies with Lowest Number of Casts : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''The Movie with Highest Number of Casts Ever Got is
'{df_casts_most_movies['Title'][0]}' : '{df_casts_most_movies['Number of Casts'].max()}'\n
The Movie with Lowest Number of Casts Ever Got is
'{df_casts_least_movies['Title'][0]}' : '{df_casts_least_movies['Number of Casts'].min()}'\n

The Movie with Highest Number of Casts on 'Netflix' is
'{netflix_casts_most_movies['Title'][0]}' : '{netflix_casts_most_movies['Number of Casts'].max()}'\n
The Movie with Lowest Number of Casts on 'Netflix' is
'{netflix_casts_least_movies['Title'][0]}' : '{netflix_casts_least_movies['Number of Casts'].min()}'\n

The Movie with Highest Number of Casts on 'Hulu' is
'{hulu_casts_most_movies['Title'][0]}' : '{hulu_casts_most_movies['Number of Casts'].max()}'\n
The Movie with Lowest Number of Casts on 'Hulu' is
'{hulu_casts_least_movies['Title'][0]}' : '{hulu_casts_least_movies['Number of Casts'].min()}'\n

The Movie with Highest Number of Casts on 'Prime Video' is
'{prime_video_casts_most_movies['Title'][0]}' : '{prime_video_casts_most_movies['Number of Casts'].max()}'\n
The Movie with Lowest Number of Casts on 'Prime Video' is
'{prime_video_casts_least_movies['Title'][0]}' : '{prime_video_casts_least_movies['Number of Casts'].min()}'\n

The Movie with Highest Number of Casts on 'Disney+' is
'{disney_casts_most_movies['Title'][0]}' : '{disney_casts_most_movies['Number of Casts'].max()}'\n
The Movie with Lowest Number of Casts on 'Disney+' is
'{disney_casts_least_movies['Title'][0]}' : '{disney_casts_least_movies['Number of Casts'].min()}'\n
'''')

```

```
# In[44]:
```

```
print(f'''  
    Accross All Platforms the Average Number of Casts is  
'{round(df_movies_count_casts['Number of Casts'].mean(), ndigits = 2)}'\n        The Average Number of Casts on 'Netflix' is '{round(netflix_casts_movies['Number of  
Casts'].mean(), ndigits = 2)}'\n        The Average Number of Casts on 'Hulu' is '{round(hulu_casts_movies['Number of  
Casts'].mean(), ndigits = 2)}'\n        The Average Number of Casts on 'Prime Video' is  
'{round(prime_video_casts_movies['Number of Casts'].mean(), ndigits = 2)}'\n        The Average Number of Casts on 'Disney+' is '{round(disney_casts_movies['Number of  
Casts'].mean(), ndigits = 2)}'  
''')
```

```
# In[45]:
```

```
print(f'''  
    Accross All Platforms Total Count of Cast is '{df_movies_count_casts['Number of  
Casts'].max()}'\n        Total Count of Cast on 'Netflix' is '{netflix_casts_movies['Number of  
Casts'].max()}'\n        Total Count of Cast on 'Hulu' is '{hulu_casts_movies['Number of Casts'].max()}'\n        Total Count of Cast on 'Prime Video' is '{prime_video_casts_movies['Number of  
Casts'].max()}'\n        Total Count of Cast on 'Disney+' is '{disney_casts_movies['Number of  
Casts'].max()}'  
''')
```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))  
sns.distplot(df_movies_count_casts['Number of Casts'], bins = 20, kde = True, ax = ax[0])  
sns.boxplot(df_movies_count_casts['Number of Casts'], ax = ax[1])  
plt.show()
```

```
# In[47]:
```

```
# Defining plot size and title  
plt.figure(figsize = (20, 5))  
plt.title('Number of Casts s Per Platform')  
  
# Plotting the information from each dataset into a histogram  
sns.histplot(prime_video_casts_movies['Number of Casts'], color = 'lightblue', legend =  
True, kde = True)  
sns.histplot(netflix_casts_movies['Number of Casts'], color = 'red', legend = True, kde =  
True)  
sns.histplot(hulu_casts_movies['Number of Casts'], color = 'lightgreen', legend = True, kde =  
True)  
sns.histplot(disney_casts_movies['Number of Casts'], color = 'darkblue', legend = True, kde =  
True)  
  
# Setting the legend  
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])  
plt.show()
```

```
# In[48]:
```

```

df_lan = df_movies_cast['Cast'].str.split(',').apply(pd.Series).stack()
del df_movies_cast['Cast']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Cast'
df_movies_cast = df_movies_cast.join(df_lan)
df_movies_cast.drop_duplicates(inplace = True)

# In[49]:
```

```

df_movies_cast.head(5)
```

```

# In[50]:
```

```

cast_count = df_movies_cast.groupby('Cast')['Title'].count()
cast_movies = df_movies_cast.groupby('Cast')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
cast_data_movies = pd.concat([cast_count, cast_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
cast_data_movies = cast_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[51]:
```

```

# Cast with Movies Counts - All Platforms Combined
cast_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]
```

```

# In[52]:
```

```

fig = px.bar(x = cast_data_movies['Cast'][:50],
y = cast_data_movies['Movies Count'][:50],
color = cast_data_movies['Movies Count'][:50],
color_continuous_scale = 'Teal_r',
labels = { 'x' : 'Cast', 'y' : 'Movies Count'},
title = 'Major Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[53]:
```

```

df_cast_high_movies = cast_data_movies.sort_values(by = 'Movies Count', ascending =
False).reset_index()
df_cast_high_movies = df_cast_high_movies.drop(['index'], axis = 1)
# filter = (cast_data_movies['Movies Count'] == (cast_data_movies['Movies Count'].max()))
# df_cast_high_movies = cast_data_movies[filter]

# highestRated_movies = cast_data_movies.loc[cast_data_movies['Movies Count'].idxmax()]

print('\nCast with Highest Ever Movies Count are : All Platforms Combined\n')
df_cast_high_movies.head(5)

# In[54]:
```

```

fig = px.bar(y = df_cast_high_movies['Cast'][:15],
```

```

x = df_cast_high_movies['Movies Count'][:15],
color = df_cast_high_movies['Movies Count'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
title = 'Cast with Highest Movies : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[55]:


df_cast_low_movies = cast_data_movies.sort_values(by = 'Movies Count', ascending =
True).reset_index()
df_cast_low_movies = df_cast_low_movies.drop(['index'], axis = 1)
# filter = (cast_data_movies['Movies Count'] == (cast_data_movies['Movies Count'].min()))
# df_cast_low_movies = cast_data_movies[filter]

print('\nCast with Lowest Ever Movies Count are : All Platforms Combined\n')
df_cast_low_movies.head(5)

# In[56]:


fig = px.bar(y = df_cast_low_movies['Cast'][:15],
              x = df_cast_low_movies['Movies Count'][:15],
              color = df_cast_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
              title = 'Cast with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[57]:


print(f'''
      Total '{cast_data_movies['Cast'].unique().shape[0]}' unique Cast Count s were Given,
      They were Like this,\n

      {cast_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Cast'].unique()[:5]}\n

      The Highest Ever Movies Count Ever Any Movie Got is
      '{df_cast_high_movies['Cast'][0]}' : '{df_cast_high_movies['Movies Count'].max()}'\n

      The Lowest Ever Movies Count Ever Any Movie Got is '{df_cast_low_movies['Cast'][0]}'
      : '{df_cast_low_movies['Movies Count'].min()}'\n
      ''')

# In[58]:


fig = px.pie(cast_data_movies[:10], names = 'Cast', values = 'Movies Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Cast')
fig.show()

# In[59]:

```

```

# netflix_cast_movies = cast_data_movies[cast_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_cast_movies = netflix_cast_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_cast_high_movies = df_cast_high_movies.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_cast_high_movies = netflix_cast_high_movies.drop(['index'], axis = 1)

netflix_cast_low_movies = df_cast_high_movies.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_cast_low_movies = netflix_cast_low_movies.drop(['index'], axis = 1)

netflix_cast_high_movies.head(5)

```

In[60]:

```

fig = px.bar(x = netflix_cast_high_movies['Cast'][:15],
              y = netflix_cast_high_movies['Netflix'][:15],
              color = netflix_cast_high_movies['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
              title = 'Cast with Highest Movies : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[61]:

```

# hulu_cast_movies = cast_data_movies[cast_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_cast_movies = hulu_cast_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_cast_high_movies = df_cast_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_cast_high_movies = hulu_cast_high_movies.drop(['index'], axis = 1)

hulu_cast_low_movies = df_cast_high_movies.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_cast_low_movies = hulu_cast_low_movies.drop(['index'], axis = 1)

hulu_cast_high_movies.head(5)

```

In[62]:

```

fig = px.bar(x = hulu_cast_high_movies['Cast'][:15],
              y = hulu_cast_high_movies['Hulu'][:15],
              color = hulu_cast_high_movies['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
              title = 'Cast with Highest Movies : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[63]:

```

# prime_video_cast_movies = cast_data_movies[cast_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_cast_movies = prime_video_cast_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_cast_high_movies = df_cast_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_cast_high_movies = prime_video_cast_high_movies.drop(['index'], axis = 1)

prime_video_cast_low_movies = df_cast_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_cast_low_movies = prime_video_cast_low_movies.drop(['index'], axis = 1)

prime_video_cast_high_movies.head(5)

```

In[64]:

```

fig = px.bar(x = prime_video_cast_high_movies['Cast'][:15],
              y = prime_video_cast_high_movies['Prime Video'][:15],
              color = prime_video_cast_high_movies['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
              title = 'Cast with Highest Movies : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[65]:

```

# disney_cast_movies = cast_data_movies[cast_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_cast_movies = disney_cast_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_cast_high_movies = df_cast_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_cast_high_movies = disney_cast_high_movies.drop(['index'], axis = 1)

disney_cast_low_movies = df_cast_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_cast_low_movies = disney_cast_low_movies.drop(['index'], axis = 1)

disney_cast_high_movies.head(5)

```

In[66]:

```

fig = px.bar(x = disney_cast_high_movies['Cast'][:15],
              y = disney_cast_high_movies['Disney+'][:15],
              color = disney_cast_high_movies['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'Movies Count'},
              title = 'Cast with Highest Movies : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[67]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(cast_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(cast_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_cast_movies = cast_data_movies[cast_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_cast_movies = netflix_cast_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_cast_movies = cast_data_movies[cast_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_cast_movies = hulu_cast_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_cast_movies = cast_data_movies[cast_data_movies['Prime Video'] != 0].sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_cast_movies = prime_video_cast_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_cast_movies = cast_data_movies[cast_data_movies['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
disney_cast_movies = disney_cast_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Cast Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_cast_movies['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_cast_movies['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_cast_movies['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_cast_movies['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''
    The Cast with Highest Movies Count Ever Got is '{df_cast_high_movies['Cast'][0]}' :
'{df_cast_high_movies['Movies Count'].max()}'\n
    The Cast with Lowest Movies Count Ever Got is '{df_cast_low_movies['Cast'][0]}' :
'{df_cast_low_movies['Movies Count'].min()}'\n

```

```

    The Cast with Highest Movies Count on 'Netflix' is
'{netflix_cast_high_movies['Cast'][0]} : '{netflix_cast_high_movies['Netflix'].max()}'\n
    The Cast with Lowest Movies Count on 'Netflix' is
'{netflix_cast_low_movies['Cast'][0]} : '{netflix_cast_low_movies['Netflix'].min()}'\n

    The Cast with Highest Movies Count on 'Hulu' is '{hulu_cast_high_movies['Cast'][0]}'
: '{hulu_cast_high_movies['Hulu'].max()}'\n
    The Cast with Lowest Movies Count on 'Hulu' is '{hulu_cast_low_movies['Cast'][0]} : '
'{hulu_cast_low_movies['Hulu'].min()}'\n

    The Cast with Highest Movies Count on 'Prime Video' is
'{prime_video_cast_high_movies['Cast'][0]} : '{prime_video_cast_high_movies['Prime
Video'].max()}'\n
    The Cast with Lowest Movies Count on 'Prime Video' is
'{prime_video_cast_low_movies['Cast'][0]} : '{prime_video_cast_low_movies['Prime
Video'].min()}'\n

    The Cast with Highest Movies Count on 'Disney+' is
'{disney_cast_high_movies['Cast'][0]} : '{disney_cast_high_movies['Disney+'].max()}'\n
    The Cast with Lowest Movies Count on 'Disney+' is
'{disney_cast_low_movies['Cast'][0]} : '{disney_cast_low_movies['Disney+'].min()}'\n
'''')

```

In[71]:

```

# Distribution of movies cast in each platform
plt.figure(figsize = (20, 5))
plt.title('Cast with Movies Count for All Platforms')
sns.violinplot(x = cast_data_movies['Movies Count'][:100], color = 'gold', legend = True,
kde = True, shade = False)
plt.show()

```

In[72]:

```

# Distribution of Cast Movies Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_cast_movies['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_cast_movies['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_cast_movies['Prime Video'][:100], color = 'lightblue', ax =
ax2[0])
sns.violinplot(x = disney_cast_movies['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()

```

In[73]:

```

print(f'''
    Across All Platforms the Average Movies Count of Cast is
'{round(cast_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Cast on 'Netflix' is
'{round(netflix_cast_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Cast on 'Hulu' is
'{round(hulu_cast_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Cast on 'Prime Video' is
'{round(prime_video_cast_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Cast on 'Disney+' is
'{round(disney_cast_movies['Disney+'].mean(), ndigits = 2)}'\n
''')

```

```
# In[74]:
print(f'''
    Accross All Platforms Total Count of Cast is
'{cast_data_movies['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Netflix' is
'{netflix_cast_movies['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Hulu' is '{hulu_cast_movies['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Prime Video' is
'{prime_video_cast_movies['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Disney+' is
'{disney_cast_movies['Cast'].unique().shape[0]}'
''')


# In[75]:
plt.figure(figsize = (20, 5))
sns.lineplot(x = cast_data_movies['Cast'][:10], y = cast_data_movies['Netflix'][:10], color = 'red')
sns.lineplot(x = cast_data_movies['Cast'][:10], y = cast_data_movies['Hulu'][:10], color = 'lightgreen')
sns.lineplot(x = cast_data_movies['Cast'][:10], y = cast_data_movies['Prime Video'][:10], color = 'lightblue')
sns.lineplot(x = cast_data_movies['Cast'][:10], y = cast_data_movies['Disney+'][:10], color = 'darkblue')
plt.xlabel('Cast', fontsize = 20)
plt.ylabel('Movies Count', fontsize = 20)
plt.show()

# In[76]:
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_c_ax1 = sns.lineplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_c_ax2 = sns.lineplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_c_ax3 = sns.lineplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_c_ax4 = sns.lineplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

# In[77]:
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_c_ax1 = sns.barplot(y = netflix_cast_movies['Cast'][:10], x = netflix_cast_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
```

```

h_c_ax2 = sns.barplot(y = hulu_cast_movies['Cast'][:10], x = hulu_cast_movies['Hulu'][:10],
palette = 'Greens_r', ax = axes[0, 1])
p_c_ax3 = sns.barplot(y = prime_video_cast_movies['Cast'][:10], x =
prime_video_cast_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_c_ax4 = sns.barplot(y = disney_cast_movies['Cast'][:10], x =
disney_cast_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

# In[78]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Cast Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_cast_movies['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_cast_movies['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_cast_movies['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_cast_movies['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[79]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_c_ax1 = sns.barplot(y = cast_data_movies['Cast'][:10], x =
cast_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_c_ax2 = sns.barplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Hulu'][:10],
palette = 'Greens_r', ax = axes[0, 1])
p_c_ax3 = sns.barplot(y = cast_data_movies['Cast'][:10], x = cast_data_movies['Prime
Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_c_ax4 = sns.barplot(y = cast_data_movies['Cast'][:10], x =
cast_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

# In[80]:


df_movies_mixed_casts.drop(df_movies_mixed_casts.loc[df_movies_mixed_casts['Cast'] ==
"NA"].index, inplace = True)
# df_movies_mixed_casts = df_movies_mixed_casts[df_movies_mixed_casts.Cast != "NA"]

```

```

df_movies_mixed_casts.drop(df_movies_mixed_casts.loc[df_movies_mixed_casts['Number of Casts'] == 1].index, inplace = True)

# In[81]:
df_movies_mixed_casts.head(5)

# In[82]:
mixed_casts_count = df_movies_mixed_casts.groupby('Cast')['Title'].count()
mixed_casts_movies = df_movies_mixed_casts.groupby('Cast')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_casts_data_movies = pd.concat([mixed_casts_count, mixed_casts_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count', 'Cast' : 'Mixed Cast'})
mixed_casts_data_movies = mixed_casts_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[83]:
mixed_casts_data_movies.head(5)

# In[84]:
# Mixed Cast with Movies Counts - All Platforms Combined
mixed_casts_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[85]:
df_mixed_casts_high_movies = mixed_casts_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_mixed_casts_high_movies = df_mixed_casts_high_movies.drop(['index'], axis = 1)
# filter = (mixed_casts_data_movies['Movies Count'] == (mixed_casts_data_movies['Movies Count'].max()))
# df_mixed_casts_high_movies = mixed_casts_data_movies[filter]

# highest_rated_movies = mixed_casts_data_movies.loc[mixed_casts_data_movies['Movies Count'].idxmax()]

print('\nMixed Cast with Highest Ever Movies Count are : All Platforms Combined\n')
df_mixed_casts_high_movies.head(5)

# In[86]:
fig = px.bar(y = df_mixed_casts_high_movies['Mixed Cast'][:15],
              x = df_mixed_casts_high_movies['Movies Count'][:15],
              color = df_mixed_casts_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Cast'},
              title = 'Movies with Highest Number of Mixed Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[87]:


df_mixed_casts_low_movies = mixed_casts_data_movies.sort_values(by = 'Movies Count',
ascending = True).reset_index()
df_mixed_casts_low_movies = df_mixed_casts_low_movies.drop(['index'], axis = 1)
# filter = (mixed_casts_data_movies['Movies Count'] == (mixed_casts_data_movies['Movies Count'].min()))
# df_mixed_casts_low_movies = mixed_casts_data_movies[filter]

print('\nMixed Cast with Lowest Ever Movies Count are : All Platforms Combined\n')
df_mixed_casts_low_movies.head(5)

# In[88]:


fig = px.bar(y = df_mixed_casts_low_movies['Mixed Cast'][:15],
              x = df_mixed_casts_low_movies['Movies Count'][:15],
              color = df_mixed_casts_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Cast'},
              title = 'Movies with Lowest Number of Mixed Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[89]:


print(f'''
      Total '{df_movies_casts['Cast'].count()}' Titles are available on All Platforms, out
      of which\n
      You Can Choose to see Movies from Total '{mixed_casts_data_movies['Mixed
      Cast'].unique().shape[0]}' Mixed Cast, They were Like this, \n
      {mixed_casts_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Mixed
      Cast']].head(5).unique()} etc. \n
      The Mixed Cast with Highest Movies Count have '{mixed_casts_data_movies['Movies
      Count'].max()}' Movies Available is '{df_mixed_casts_high_movies['Mixed Cast'][0]}', &\n
      The Mixed Cast with Lowest Movies Count have '{mixed_casts_data_movies['Movies
      Count'].min()}' Movies Available is '{df_mixed_casts_low_movies['Mixed Cast'][0]}'
      ''')

# In[90]:


fig = px.pie(mixed_casts_data_movies[:10], names = 'Mixed Cast', values = 'Movies Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Mixed Cast')
fig.show()

# In[91]:


# netflix_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Netflix']
!= 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_mixed_casts_movies = netflix_mixed_casts_movies.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

```

```

netflix_mixed_casts_high_movies = df_mixed_casts_high_movies.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_mixed_casts_high_movies = netflix_mixed_casts_high_movies.drop(['index'], axis = 1)

netflix_mixed_casts_low_movies = df_mixed_casts_high_movies.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_mixed_casts_low_movies = netflix_mixed_casts_low_movies.drop(['index'], axis = 1)

netflix_mixed_casts_high_movies.head(5)

# In[92]:


# hulu_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_mixed_casts_movies = hulu_mixed_casts_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_casts_high_movies = df_mixed_casts_high_movies.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_mixed_casts_high_movies = hulu_mixed_casts_high_movies.drop(['index'], axis = 1)

hulu_mixed_casts_low_movies = df_mixed_casts_high_movies.sort_values(by = 'Hulu', ascending =
= True).reset_index()
hulu_mixed_casts_low_movies = hulu_mixed_casts_low_movies.drop(['index'], axis = 1)

hulu_mixed_casts_high_movies.head(5)

# In[93]:


# prime_video_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_casts_movies = prime_video_mixed_casts_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_casts_high_movies = df_mixed_casts_high_movies.sort_values(by = 'Prime Video',
ascending = False).reset_index()
prime_video_mixed_casts_high_movies = prime_video_mixed_casts_high_movies.drop(['index'], axis = 1)

prime_video_mixed_casts_low_movies = df_mixed_casts_high_movies.sort_values(by = 'Prime Video', ascending =
= True).reset_index()
prime_video_mixed_casts_low_movies = prime_video_mixed_casts_low_movies.drop(['index'], axis = 1)

prime_video_mixed_casts_high_movies.head(5)

# In[94]:


# disney_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_casts_movies = disney_mixed_casts_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_mixed_casts_high_movies = df_mixed_casts_high_movies.sort_values(by = 'Disney+', ascending =
= False).reset_index()
disney_mixed_casts_high_movies = disney_mixed_casts_high_movies.drop(['index'], axis = 1)

disney_mixed_casts_low_movies = df_mixed_casts_high_movies.sort_values(by = 'Disney+', ascending =
= True).reset_index()
disney_mixed_casts_low_movies = disney_mixed_casts_low_movies.drop(['index'], axis = 1)

```

```

disney_mixed_casts_high_movies.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_casts_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_casts_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_casts_movies = netflix_mixed_casts_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_casts_movies = hulu_mixed_casts_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_casts_movies = prime_video_mixed_casts_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_mixed_casts_movies = mixed_casts_data_movies[mixed_casts_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_casts_movies = disney_mixed_casts_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

# In[97]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Cast Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_casts_movies['Prime Video'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_mixed_casts_movies['Netflix'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_mixed_casts_movies['Hulu'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_mixed_casts_movies['Disney+'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[98]:


print(f'''
```

```

        The Mixed Cast with Highest Movies Count Ever Got is
'{df_mixed_casts_high_movies['Mixed Cast'][0]} : '{df_mixed_casts_high_movies['Movies
Count'].max()}'\n
        The Mixed Cast with Lowest Movies Count Ever Got is
'{df_mixed_casts_low_movies['Mixed Cast'][0]} : '{df_mixed_casts_low_movies['Movies
Count'].min()}'\n

        The Mixed Cast with Highest Movies Count on 'Netflix' is
'{netflix_mixed_casts_high_movies['Mixed Cast'][0]} :
'{netflix_mixed_casts_high_movies['Netflix'].max()}'\n
        The Mixed Cast with Lowest Movies Count on 'Netflix' is
'{netflix_mixed_casts_low_movies['Mixed Cast'][0]} :
'{netflix_mixed_casts_low_movies['Netflix'].min()}'\n

        The Mixed Cast with Highest Movies Count on 'Hulu' is
'{hulu_mixed_casts_high_movies['Mixed Cast'][0]} :
'{hulu_mixed_casts_high_movies['Hulu'].max()}'\n
        The Mixed Cast with Lowest Movies Count on 'Hulu' is
'{hulu_mixed_casts_low_movies['Mixed Cast'][0]} :
'{hulu_mixed_casts_low_movies['Hulu'].min()}'\n

        The Mixed Cast with Highest Movies Count on 'Prime Video' is
'{prime_video_mixed_casts_high_movies['Mixed Cast'][0]} :
'{prime_video_mixed_casts_high_movies['Prime Video'].max()}'\n
        The Mixed Cast with Lowest Movies Count on 'Prime Video' is
'{prime_video_mixed_casts_low_movies['Mixed Cast'][0]} :
'{prime_video_mixed_casts_low_movies['Prime Video'].min()}'\n

        The Mixed Cast with Highest Movies Count on 'Disney+' is
'{disney_mixed_casts_high_movies['Mixed Cast'][0]} :
'{disney_mixed_casts_high_movies['Disney+'].max()}'\n
        The Mixed Cast with Lowest Movies Count on 'Disney+' is
'{disney_mixed_casts_low_movies['Mixed Cast'][0]} :
'{disney_mixed_casts_low_movies['Disney+'].min()}'\n
        ''')

```

In[99]:

```

print(f'''
    Accross All Platforms the Average Movies Count of Mixed Cast is
'{round(mixed_casts_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Cast on 'Netflix' is
'{round(netflix_mixed_casts_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Cast on 'Hulu' is
'{round(hulu_mixed_casts_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Cast on 'Prime Video' is
'{round(prime_video_mixed_casts_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Cast on 'Disney+' is
'{round(disney_mixed_casts_movies['Disney+'].mean(), ndigits = 2)}'\n
        ''')

```

In[100]:

```

print(f'''
    Accross All Platforms Total Count of Mixed Cast is '{mixed_casts_data_movies['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Netflix' is '{netflix_mixed_casts_movies['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Hulu' is '{hulu_mixed_casts_movies['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Prime Video' is '{prime_video_mixed_casts_movies['Mixed
Cast'].unique().shape[0]}'\n
        ''')

```

```
Total Count of Mixed Cast on 'Disney+' is '{disney_mixed_casts_movies['Mixed Cast'].unique().shape[0]}'\n''')
```

```
# In[101]:
```

```
plt.figure(figsize = (20, 5))  
sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:5], y =  
mixed_casts_data_movies['Netflix'][:5], color = 'red')  
sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:5], y =  
mixed_casts_data_movies['Hulu'][:5], color = 'lightgreen')  
sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:5], y =  
mixed_casts_data_movies['Prime Video'][:5], color = 'lightblue')  
sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:5], y =  
mixed_casts_data_movies['Disney+'][:5], color = 'darkblue')  
plt.xlabel('Mixed Cast', fontsize = 15)  
plt.ylabel('Movies Count', fontsize = 15)  
plt.show()
```

```
# In[102]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))  
  
n_c_ax1 = sns.barplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])  
h_c_ax2 = sns.barplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])  
p_c_ax3 = sns.barplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])  
d_c_ax4 = sns.barplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_c_ax1.title.set_text(labels[0])  
h_c_ax2.title.set_text(labels[1])  
p_c_ax3.title.set_text(labels[2])  
d_c_ax4.title.set_text(labels[3])  
  
plt.show()
```

```
# In[103]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))  
  
n_mc_ax1 = sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])  
h_mc_ax2 = sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])  
p_mc_ax3 = sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])  
d_mc_ax4 = sns.lineplot(x = mixed_casts_data_movies['Mixed Cast'][:10], y =  
mixed_casts_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_mc_ax1.title.set_text(labels[0])  
h_mc_ax2.title.set_text(labels[1])  
p_mc_ax3.title.set_text(labels[2])  
d_mc_ax4.title.set_text(labels[3])
```

```

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Cast Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_casts_movies['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_casts_movies['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_casts_movies['Prime Video'][:50], color = 'lightblue', legend = True)
sns.kdeplot(disney_mixed_casts_movies['Disney+'][:50], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()


# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_mc_ax1 = sns.barplot(x = netflix_mixed_casts_movies['Mixed Cast'][:10], y =
netflix_mixed_casts_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mc_ax2 = sns.barplot(x = hulu_mixed_casts_movies['Mixed Cast'][:10], y =
hulu_mixed_casts_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mc_ax3 = sns.barplot(x = prime_video_mixed_casts_movies['Mixed Cast'][:10], y =
prime_video_mixed_casts_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_mc_ax4 = sns.barplot(x = disney_mixed_casts_movies['Mixed Cast'][:10], y =
disney_mixed_casts_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mc_ax1.title.set_text(labels[0])
h_mc_ax2.title.set_text(labels[1])
p_mc_ax3.title.set_text(labels[2])
d_mc_ax4.title.set_text(labels[3])

plt.show()


# In[106]:


fig = go.Figure(go.Funnel(y = mixed_casts_data_movies['Mixed Cast'][:10], x =
mixed_casts_data_movies['Movies Count'][:10]))
fig.show()

```

ottmovies_country.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy

```

```
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_movies = pd.read_csv(path + 'ottmovies.csv')
df_movies.head()
```

```
# In[6]:
```

```

# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)

```

```

# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

```

```

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:
# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:
df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] ==
0) & (df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]

# In[18]:
df_movies_countries = df_movies.copy()

# In[19]:
df_movies_countries.drop(df_movies_countries.loc[df_movies_countries['Country'] ==
"NA"].index, inplace = True)
# df_movies_countries = df_movies_countries[df_movies_countries.Country != "NA"]

```

```

# df_movies_countries['Country'] = df_movies_countries['Country'].astype(str)

# In[20]:


df_movies_count_countries = df_movies_countries.copy()

# In[21]:


df_movies_country = df_movies_countries.copy()

# In[22]:


# Create countries dict where key=name and value = number of countries
countries = {}

for i in df_movies_count_countries['Country'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        countries[i] = len(i.split(','))
    else:
        countries[i] = 0

# Add this information to our dataframe as a new column

df_movies_count_countries['Number of Countries'] =
df_movies_count_countries['Country'].map(countries).astype(int)

# In[23]:


df_movies_mixed_countries = df_movies_count_countries.copy()

# In[24]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_countries_movies =
df_movies_count_countries.loc[df_movies_count_countries['Netflix'] == 1]
hulu_countries_movies = df_movies_count_countries.loc[df_movies_count_countries['Hulu'] ==
1]
prime_video_countries_movies =
df_movies_count_countries.loc[df_movies_count_countries['Prime Video'] == 1]
disney_countries_movies =
df_movies_count_countries.loc[df_movies_count_countries['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_movies_count_countries.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks

```

```

plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_countries_most_movies = df_movies_count_countries.sort_values(by = 'Number of Countries', ascending = False).reset_index()
df_countries_most_movies = df_countries_most_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_countries['Number of Countries'] ==
(df_movies_count_countries['Number of Countries'].max()))
# df_countries_most_movies = df_movies_count_countries[filter]

# mostest_rated_movies = df_movies_count_countries.loc[df_movies_count_countries['Number of Countries'].idxmax()]

print('\nMovies with Highest Ever Number of Countries are : \n')
df_countries_most_movies.head(5)

# In[27]:


fig = px.bar(y = df_countries_most_movies['Title'][:15],
              x = df_countries_most_movies['Number of Countries'][:15],
              color = df_countries_most_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Highest Number of Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[28]:


df_countries_least_movies = df_movies_count_countries.sort_values(by = 'Number of Countries', ascending = True).reset_index()
df_countries_least_movies = df_countries_least_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_countries['Number of Countries'] ==
(df_movies_count_countries['Number of Countries'].min()))
# df_countries_least_movies = df_movies_count_countries[filter]

print('\nMovies with Lowest Ever Number of Countries are : \n')
df_countries_least_movies.head(5)

# In[29]:


fig = px.bar(y = df_countries_least_movies['Title'][:15],
              x = df_countries_least_movies['Number of Countries'][:15],
              color = df_countries_least_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Lowest Number of Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[30]:


print(f'''
    Total '{df_movies_count_countries['Number of Countries'].unique().shape[0]}' unique
Number of Countries s were Given, They were Like this,\n

    {df_movies_count_countries.sort_values(by = 'Number of Countries', ascending =
False)['Number of Countries'].unique()}\n

    The Highest Number of Countries Ever Any Movie Got is
'{df_countries_most_movies['Title'][0]}' : '{df_countries_most_movies['Number of
Countries'].max()}'\n

    The Lowest Number of Countries Ever Any Movie Got is
'{df_countries_least_movies['Title'][0]}' : '{df_countries_least_movies['Number of
Countries'].min()}'\n
''')


# In[31]:


netflix_countries_most_movies =
df_countries_most_movies.loc[df_countries_most_movies['Netflix']==1].reset_index()
netflix_countries_most_movies = netflix_countries_most_movies.drop(['index'], axis = 1)

netflix_countries_least_movies =
df_countries_least_movies.loc[df_countries_least_movies['Netflix']==1].reset_index()
netflix_countries_least_movies = netflix_countries_least_movies.drop(['index'], axis = 1)

netflix_countries_most_movies.head(5)

# In[32]:


fig = px.bar(y = netflix_countries_most_movies['Title'][:15],
              x = netflix_countries_most_movies['Number of Countries'][:15],
              color = netflix_countries_most_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Highest Number of Countries : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_countries_least_movies['Title'][:15],
              x = netflix_countries_least_movies['Number of Countries'][:15],
              color = netflix_countries_least_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Lowest Number of Countries : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:

```

```

hulu_countries_most_movies =
df_countries_most_movies.loc[df_countries_most_movies['Hulu']==1].reset_index()
hulu_countries_most_movies = hulu_countries_most_movies.drop(['index'], axis = 1)

hulu_countries_least_movies =
df_countries_least_movies.loc[df_countries_least_movies['Hulu']==1].reset_index()
hulu_countries_least_movies = hulu_countries_least_movies.drop(['index'], axis = 1)

hulu_countries_most_movies.head(5)

# In[35]:


fig = px.bar(y = hulu_countries_most_movies['Title'][:15],
              x = hulu_countries_most_movies['Number of Countries'][:15],
              color = hulu_countries_most_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Highest Number of Countries : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_countries_least_movies['Title'][:15],
              x = hulu_countries_least_movies['Number of Countries'][:15],
              color = hulu_countries_least_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Lowest Number of Countries : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_countries_most_movies =
df_countries_most_movies.loc[df_countries_most_movies['Prime Video']==1].reset_index()
prime_video_countries_most_movies = prime_video_countries_most_movies.drop(['index'], axis = 1)

prime_video_countries_least_movies =
df_countries_least_movies.loc[df_countries_least_movies['Prime Video']==1].reset_index()
prime_video_countries_least_movies = prime_video_countries_least_movies.drop(['index'], axis = 1)

prime_video_countries_most_movies.head(5)

# In[38]:


fig = px.bar(y = prime_video_countries_most_movies['Title'][:15],
              x = prime_video_countries_most_movies['Number of Countries'][:15],
              color = prime_video_countries_most_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Highest Number of Countries : Prime Video')

fig.update_layout(plot_bgcolor = 'white')

```

```

fig.show()

# In[39]:


fig = px.bar(y = prime_video_countries_least_movies['Title'][:15],
              x = prime_video_countries_least_movies['Number of Countries'][:15],
              color = prime_video_countries_least_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Lowest Number of Countries : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


disney_countries_most_movies =
df_countries_most_movies.loc[df_countries_most_movies['Disney+']==1].reset_index()
disney_countries_most_movies = disney_countries_most_movies.drop(['index'], axis = 1)

disney_countries_least_movies =
df_countries_least_movies.loc[df_countries_least_movies['Disney+']==1].reset_index()
disney_countries_least_movies = disney_countries_least_movies.drop(['index'], axis = 1)

disney_countries_most_movies.head(5)

# In[41]:


fig = px.bar(y = disney_countries_most_movies['Title'][:15],
              x = disney_countries_most_movies['Number of Countries'][:15],
              color = disney_countries_most_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Highest Number of Countries : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_countries_least_movies['Title'][:15],
              x = disney_countries_least_movies['Number of Countries'][:15],
              color = disney_countries_least_movies['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Countries'},
              title = 'Movies with Lowest Number of Countries : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''
      The Movie with Highest Number of Countries Ever Got is
'{df_countries_most_movies['Title'][0]}' : '{df_countries_most_movies['Number of Countries'].max()}'\n

```

```

    The Movie with Lowest Number of Countries Ever Got is
'{df_countries_least_movies['Title'][0]} : '{df_countries_least_movies['Number of
Countries'].min()}'\n

    The Movie with Highest Number of Countries on 'Netflix' is
'{netflix_countries_most_movies['Title'][0]} : '{netflix_countries_most_movies['Number of
Countries'].max()}'\n
    The Movie with Lowest Number of Countries on 'Netflix' is
'{netflix_countries_least_movies['Title'][0]} : '{netflix_countries_least_movies['Number of
Countries'].min()}'\n

    The Movie with Highest Number of Countries on 'Hulu' is
'{hulu_countries_most_movies['Title'][0]} : '{hulu_countries_most_movies['Number of
Countries'].max()}'\n
    The Movie with Lowest Number of Countries on 'Hulu' is
'{hulu_countries_least_movies['Title'][0]} : '{hulu_countries_least_movies['Number of
Countries'].min()}'\n

    The Movie with Highest Number of Countries on 'Prime Video' is
'{prime_video_countries_most_movies['Title'][0]} :
'{prime_video_countries_most_movies['Number of Countries'].max()}'\n
    The Movie with Lowest Number of Countries on 'Prime Video' is
'{prime_video_countries_least_movies['Title'][0]} :
'{prime_video_countries_least_movies['Number of Countries'].min()}'\n

    The Movie with Highest Number of Countries on 'Disney+' is
'{disney_countries_most_movies['Title'][0]} : '{disney_countries_most_movies['Number of
Countries'].max()}'\n
    The Movie with Lowest Number of Countries on 'Disney+' is
'{disney_countries_least_movies['Title'][0]} : '{disney_countries_least_movies['Number of
Countries'].min()}'\n
    ''')

```

In[44]:

```

print(f'''
    Accross All Platforms the Average Number of Countries is
'{round(df_movies_count_countries['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Netflix' is
'{round(netflix_countries_movies['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Hulu' is '{round(hulu_countries_movies['Number of
Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Prime Video' is
'{round(prime_video_countries_movies['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Disney+' is
'{round(disney_countries_movies['Number of Countries'].mean(), ndigits = 2)}'\n
    ''')

```

In[45]:

```

print(f'''
    Accross All Platforms Total Count of Country is '{df_movies_count_countries['Number
of Countries'].max()}'\n
    Total Count of Country on 'Netflix' is '{netflix_countries_movies['Number of
Countries'].max()}'\n
    Total Count of Country on 'Hulu' is '{hulu_countries_movies['Number of
Countries'].max()}'\n
    Total Count of Country on 'Prime Video' is '{prime_video_countries_movies['Number of
Countries'].max()}'\n
    Total Count of Country on 'Disney+' is '{disney_countries_movies['Number of
Countries'].max()}'\n
    ''')

```

```
# In[46]:  
  
f, ax = plt.subplots(1, 2 , figsize = (20, 5))  
sns.distplot(df_movies_count_countries['Number of Countries'],bins = 20, kde = True, ax = ax[0])  
sns.boxplot(df_movies_count_countries['Number of Countries'], ax = ax[1])  
plt.show()
```

```
# In[47]:
```

```
# Defining plot size and title  
plt.figure(figsize = (20, 5))  
plt.title('Number of Countries s Per Platform')  
  
# Plotting the information from each dataset into a histogram  
sns.histplot(prime_video_countries_movies['Number of Countries'], color = 'lightblue',  
legend = True, kde = True)  
sns.histplot(netflix_countries_movies['Number of Countries'], color = 'red', legend = True,  
kde = True)  
sns.histplot(hulu_countries_movies['Number of Countries'], color = 'lightgreen', legend =  
True, kde = True)  
sns.histplot(disney_countries_movies['Number of Countries'], color = 'darkblue', legend =  
True, kde = True)  
  
# Setting the legend  
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])  
plt.show()
```

```
# In[48]:
```

```
df_lan = df_movies_country['Country'].str.split(',').apply(pd.Series).stack()  
del df_movies_country['Country']  
df_lan.index = df_lan.index.droplevel(-1)  
df_lan.name = 'Country'  
df_movies_country = df_movies_country.join(df_lan)  
df_movies_country.drop_duplicates(inplace = True)
```

```
# In[49]:
```

```
df_movies_country.head(5)
```

```
# In[50]:
```

```
country_count = df_movies_country.groupby('Country')['Title'].count()  
country_movies = df_movies_country.groupby('Country')[['Netflix', 'Hulu', 'Prime Video',  
'Disney+']].sum()  
country_data_movies = pd.concat([country_count, country_movies], axis =  
1).reset_index().rename(columns = {'Title' : 'Movies Count'})  
country_data_movies = country_data_movies.sort_values(by = 'Movies Count', ascending =  
False)
```

```
# In[51]:
```

```

# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_country_movies = country_data_movies[country_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_country_movies = netflix_country_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_country_movies = country_data_movies[country_data_movies['Hulu'] != 0].sort_values(by =
= 'Hulu', ascending = False).reset_index()
hulu_country_movies = hulu_country_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

prime_video_country_movies = country_data_movies[country_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_country_movies = prime_video_country_movies.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'Movies Count'], axis = 1)

disney_country_movies = country_data_movies[country_data_movies['Disney+'] != 0].sort_values(by =
= 'Disney+', ascending = False).reset_index()
disney_country_movies = disney_country_movies.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'Movies Count'], axis = 1)

# In[52]:
```

```

# Country with Movies Counts - All Platforms Combined
country_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]
```

```
# In[53]:
```

```

fig = px.bar(x = country_data_movies['Country'][:50],
              y = country_data_movies['Movies Count'][:50],
              color = country_data_movies['Movies Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Country', 'y' : 'Movies Count'},
              title = 'Major Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[54]:
```

```

fig = px.choropleth(data_frame = country_data_movies, locations = 'Country', locationmode =
= 'country names', color = 'Movies Count', color_continuous_scale = 'deep')

fig.show()
```

```
# In[55]:
```

```

df_country_high_movies = country_data_movies.sort_values(by = 'Movies Count', ascending =
False).reset_index()
df_country_high_movies = df_country_high_movies.drop(['index'], axis = 1)
# filter = (country_data_movies['Movies Count'] == (country_data_movies['Movies
Count'].max()))
# df_country_high_movies = country_data_movies[filter]

# highestRatedMovies = country_data_movies.loc[country_data_movies['Movies
Count'].idxmax()]
```

```
print('\nCountry with Highest Ever Movies Count are : All Platforms Combined\n')
df_country_high_movies.head(5)
```

```
# In[56]:
```

```
fig = px.bar(y = df_country_high_movies['Country'][:15],
              x = df_country_high_movies['Movies Count'][:15],
              color = df_country_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Highest Movies : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[57]:
```

```
df_country_low_movies = country_data_movies.sort_values(by = 'Movies Count', ascending =
True).reset_index()
df_country_low_movies = df_country_low_movies.drop(['index'], axis = 1)
# filter = (country_data_movies['Movies Count'] == (country_data_movies['Movies
Count'].min()))
# df_country_low_movies = country_data_movies[filter]

print('\nCountry with Lowest Ever Movies Count are : All Platforms Combined\n')
df_country_low_movies.head(5)
```

```
# In[58]:
```

```
fig = px.bar(y = df_country_low_movies['Country'][:15],
              x = df_country_low_movies['Movies Count'][:15],
              color = df_country_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[59]:
```

```
print(f'''
    Total '{country_data_movies['Country'].unique().shape[0]}' unique Country Count s
were Given, They were Like this,\n

    {country_data_movies.sort_values(by = 'Movies Count', ascending =
False)[['Country'].unique()[:5}}\n

    The Highest Ever Movies Count Ever Any Movie Got is
'{df_country_high_movies['Country'][0]}' : '{df_country_high_movies['Movies
Count'].max()}'\n

    The Lowest Ever Movies Count Ever Any Movie Got is
'{df_country_low_movies['Country'][0]}' : '{df_country_low_movies['Movies Count'].min()}'\n'''
```

```
# In[60]:
```

```

fig = px.pie(country_data_movies[:10], names = 'Country', values = 'Movies Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Country')
fig.show()

# In[61]:


# netflix_country_movies = country_data_movies[country_data_movies['Netflix'] !=
0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_country_movies = netflix_country_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

netflix_country_high_movies = df_country_high_movies.sort_values(by = 'Netflix', ascending =
False).reset_index()
netflix_country_high_movies = netflix_country_high_movies.drop(['index'], axis = 1)

netflix_country_low_movies = df_country_high_movies.sort_values(by = 'Netflix', ascending =
True).reset_index()
netflix_country_low_movies = netflix_country_low_movies.drop(['index'], axis = 1)

netflix_country_high_movies.head(5)

# In[62]:


fig = px.bar(x = netflix_country_high_movies['Country'][:15],
              y = netflix_country_high_movies['Netflix'][:15],
              color = netflix_country_high_movies['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Highest Movies : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


fig = px.choropleth(data_frame = netflix_country_movies, locations = 'Country',
locationmode = 'country names', color = 'Netflix', color_continuous_scale = 'Reds')

fig.show()

# In[64]:


# hulu_country_movies = country_data_movies[country_data_movies['Hulu'] !=
0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_country_movies = hulu_country_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_country_high_movies = df_country_high_movies.sort_values(by = 'Hulu', ascending =
False).reset_index()
hulu_country_high_movies = hulu_country_high_movies.drop(['index'], axis = 1)

hulu_country_low_movies = df_country_high_movies.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_country_low_movies = hulu_country_low_movies.drop(['index'], axis = 1)

```

```

hulu_country_high_movies.head(5)

# In[65]:


fig = px.bar(x = hulu_country_high_movies['Country'][:15],
              y = hulu_country_high_movies['Hulu'][:15],
              color = hulu_country_high_movies['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Highest Movies : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[66]:


fig = px.choropleth(data_frame = hulu_country_movies, locations = 'Country', locationmode =
'country names', color = 'Hulu', color_continuous_scale = 'Greens')

fig.show()

# In[67]:


# prime_video_country_movies = country_data_movies[country_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_country_movies = prime_video_country_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_country_high_movies = df_country_high_movies.sort_values(by = 'Prime Video',
ascending = False).reset_index()
prime_video_country_high_movies = prime_video_country_high_movies.drop(['index'], axis = 1)

prime_video_country_low_movies = df_country_high_movies.sort_values(by = 'Prime Video',
ascending = True).reset_index()
prime_video_country_low_movies = prime_video_country_low_movies.drop(['index'], axis = 1)

prime_video_country_high_movies.head(5)

# In[68]:


fig = px.bar(x = prime_video_country_high_movies['Country'][:15],
              y = prime_video_country_high_movies['Prime Video'][:15],
              color = prime_video_country_high_movies['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Highest Movies : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[69]:


fig = px.choropleth(data_frame = prime_video_country_movies, locations = 'Country',
locationmode = 'country names', color = 'Prime Video', color_continuous_scale = 'Blues')

```

```

fig.show()

# In[70]:


# disney_country_movies = country_data_movies[country_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_country_movies = disney_country_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_country_high_movies = df_country_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_country_high_movies = disney_country_high_movies.drop(['index'], axis = 1)

disney_country_low_movies = df_country_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_country_low_movies = disney_country_low_movies.drop(['index'], axis = 1)

disney_country_high_movies.head(5)

# In[71]:


fig = px.bar(x = disney_country_high_movies['Country'][:15],
              y = disney_country_high_movies['Disney+'][:15],
              color = disney_country_high_movies['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'Movies Count'},
              title = 'Country with Highest Movies : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[72]:


fig = px.choropleth(data_frame = disney_country_movies, locations = 'Country', locationmode = 'country names', color = 'Disney+', color_continuous_scale = 'BuPu')

fig.show()

# In[73]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(country_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(country_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[74]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Country Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_country_movies['Disney+'][:50], color = 'darkblue', legend = True, kde = True)

```

```

sns.histplot(prime_video_country_movies['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_country_movies['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_country_movies['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[75]:


print(f'''
    The Country with Highest Movies Count Ever Got is
'{df_country_high_movies['Country'][0]}' : '{df_country_high_movies['Movies Count'].max()}'\n
    The Country with Lowest Movies Count Ever Got is
'{df_country_low_movies['Country'][0]}' : '{df_country_low_movies['Movies Count'].min()}'\n

    The Country with Highest Movies Count on 'Netflix' is
'{netflix_country_high_movies['Country'][0]}' :
'{netflix_country_high_movies['Netflix'].max()}'\n
    The Country with Lowest Movies Count on 'Netflix' is
'{netflix_country_low_movies['Country'][0]}' :
'{netflix_country_low_movies['Netflix'].min()}'\n

    The Country with Highest Movies Count on 'Hulu' is
'{hulu_country_high_movies['Country'][0]}' : '{hulu_country_high_movies['Hulu'].max()}'\n
    The Country with Lowest Movies Count on 'Hulu' is
'{hulu_country_low_movies['Country'][0]}' : '{hulu_country_low_movies['Hulu'].min()}'\n

    The Country with Highest Movies Count on 'Prime Video' is
'{prime_video_country_high_movies['Country'][0]}' :
'{prime_video_country_high_movies['Prime Video'].max()}'\n
    The Country with Lowest Movies Count on 'Prime Video' is
'{prime_video_country_low_movies['Country'][0]}' : '{prime_video_country_low_movies['Prime Video'].min()}'\n

    The Country with Highest Movies Count on 'Disney+' is
'{disney_country_high_movies['Country'][0]}' :
'{disney_country_high_movies['Disney+'].max()}'\n
    The Country with Lowest Movies Count on 'Disney+' is
'{disney_country_low_movies['Country'][0]}' :
'{disney_country_low_movies['Disney+'].min()}'\n
''')


# In[76]:


# Distribution of movies country in each platform
plt.figure(figsize = (20, 5))
plt.title('Country with Movies Count for All Platforms')
sns.violinplot(x = country_data_movies['Movies Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

# In[77]:


# Distribution of Country Movies Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))

```

```

sns.violinplot(x = netflix_country_movies['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_country_movies['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_country_movies['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_country_movies['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()

```

In[78]:

```

print(f'''
    Across All Platforms the Average Movies Count of Country is
'{round(country_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Country on 'Netflix' is
'{round(netflix_country_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Country on 'Hulu' is
'{round(hulu_country_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Country on 'Prime Video' is
'{round(prime_video_country_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Country on 'Disney+' is
'{round(disney_country_movies['Disney+'].mean(), ndigits = 2)}'
    ''')

```

In[79]:

```

print(f'''
    Across All Platforms Total Count of Country is
'{country_data_movies['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Netflix' is
'{netflix_country_movies['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Hulu' is
'{hulu_country_movies['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Prime Video' is
'{prime_video_country_movies['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Disney+' is
'{disney_country_movies['Country'].unique().shape[0]}'
    ''')

```

In[80]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = country_data_movies['Country'][:10], y =
country_data_movies['Netflix'][:10], color = 'red')
sns.lineplot(x = country_data_movies['Country'][:10], y = country_data_movies['Hulu'][:10],
color = 'lightgreen')
sns.lineplot(x = country_data_movies['Country'][:10], y = country_data_movies['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = country_data_movies['Country'][:10], y =
country_data_movies['Disney+'][:10], color = 'darkblue')
plt.xlabel('Country', fontsize = 20)
plt.ylabel('Movies Count', fontsize = 20)
plt.show()

```

In[81]:

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))
```

```

n_co_ax1 = sns.lineplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_co_ax2 = sns.lineplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_co_ax3 = sns.lineplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_co_ax4 = sns.lineplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()

```

In[82]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_co_ax1 = sns.barplot(y = netflix_country_movies['Country'][:10], x =
netflix_country_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_co_ax2 = sns.barplot(y = hulu_country_movies['Country'][:10], x =
hulu_country_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_co_ax3 = sns.barplot(y = prime_video_country_movies['Country'][:10], x =
prime_video_country_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_co_ax4 = sns.barplot(y = disney_country_movies['Country'][:10], x =
disney_country_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()

```

In[83]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Country Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_country_movies['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_country_movies['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_country_movies['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_country_movies['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

```

In[84]:

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))
```

```

n_co_ax1 = sns.barplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_co_ax2 = sns.barplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_co_ax3 = sns.barplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_co_ax4 = sns.barplot(y = country_data_movies['Country'][:10], x =
country_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()

# In[85]:


df_movies_mixed_countries.drop(df_movies_mixed_countries.loc[df_movies_mixed_countries['Country'] == "NA"].index, inplace = True)
# df_movies_mixed_countries = df_movies_mixed_countries[df_movies_mixed_countries.Country != "NA"]
df_movies_mixed_countries.drop(df_movies_mixed_countries.loc[df_movies_mixed_countries['Number of Countries'] == 1].index, inplace = True)

# In[86]:


df_movies_mixed_countries.head(5)

# In[87]:


mixed_countries_count = df_movies_mixed_countries.groupby('Country')['Title'].count()
mixed_countries_movies = df_movies_mixed_countries.groupby('Country')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_countries_data_movies = pd.concat([mixed_countries_count, mixed_countries_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count', 'Country' : 'Mixed Country'})
mixed_countries_data_movies = mixed_countries_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[88]:


mixed_countries_data_movies.head(5)

# In[89]:


# Mixed Country with Movies Counts - All Platforms Combined
mixed_countries_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[90]:

```

```

df_mixed_countries_high_movies = mixed_countries_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_mixed_countries_high_movies = df_mixed_countries_high_movies.drop(['index'], axis = 1)
# filter = (mixed_countries_data_movies['Movies Count'] == (mixed_countries_data_movies['Movies Count'].max()))
# df_mixed_countries_high_movies = mixed_countries_data_movies[filter]

# highest_rated_movies =
mixed_countries_data_movies.loc[mixed_countries_data_movies['Movies Count'].idxmax()]

print('\nMixed Country with Highest Ever Movies Count are : All Platforms Combined\n')
df_mixed_countries_high_movies.head(5)

# In[91]:


fig = px.bar(y = df_mixed_countries_high_movies['Mixed Country'][:15],
              x = df_mixed_countries_high_movies['Movies Count'][:15],
              color = df_mixed_countries_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Country'},
              title = 'Movies with Highest Number of Mixed Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[92]:


df_mixed_countries_low_movies = mixed_countries_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_mixed_countries_low_movies = df_mixed_countries_low_movies.drop(['index'], axis = 1)
# filter = (mixed_countries_data_movies['Movies Count'] == (mixed_countries_data_movies['Movies Count'].min()))
# df_mixed_countries_low_movies = mixed_countries_data_movies[filter]

print('\nMixed Country with Lowest Ever Movies Count are : All Platforms Combined\n')
df_mixed_countries_low_movies.head(5)

# In[93]:


fig = px.bar(y = df_mixed_countries_low_movies['Mixed Country'][:15],
              x = df_mixed_countries_low_movies['Movies Count'][:15],
              color = df_mixed_countries_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Country'},
              title = 'Movies with Lowest Number of Mixed Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[94]:


print(f'''
      Total '{df_movies_countries['Country'].count()}' Titles are available on All Platforms, out of which
      You Can Choose to see Movies from Total '{mixed_countries_data_movies['Mixed Country'].unique().shape[0]}' Mixed Country, They were Like this, \n
    ''')

```

```

{mixed_countries_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Mixed Country'].head(5).unique()} etc. \n

The Mixed Country with Highest Movies Count have
'{mixed_countries_data_movies['Movies Count'].max()}' Movies Available is
'{df_mixed_countries_high_movies['Mixed Country'][0]}', &\n
The Mixed Country with Lowest Movies Count have '{mixed_countries_data_movies['Movies
Count'].min()}' Movies Available is '{df_mixed_countries_low_movies['Mixed Country'][0]}'
''')

```

In[95]:

```

fig = px.pie(mixed_countries_data_movies[:10], names = 'Mixed Country', values = 'Movies
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Mixed Country')
fig.show()

```

In[96]:

```

# netflix_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
# netflix_mixed_countries_movies = netflix_mixed_countries_movies.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_mixed_countries_high_movies = df_mixed_countries_high_movies.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_countries_high_movies = netflix_mixed_countries_high_movies.drop(['index'],
axis = 1)

netflix_mixed_countries_low_movies = df_mixed_countries_low_movies.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_mixed_countries_low_movies = netflix_mixed_countries_low_movies.drop(['index'],
axis = 1)

netflix_mixed_countries_high_movies.head(5)

```

In[97]:

```

# hulu_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_mixed_countries_movies = hulu_mixed_countries_movies.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_countries_high_movies = df_mixed_countries_high_movies.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_mixed_countries_high_movies = hulu_mixed_countries_high_movies.drop(['index'], axis =
1)

hulu_mixed_countries_low_movies = df_mixed_countries_low_movies.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_countries_low_movies = hulu_mixed_countries_low_movies.drop(['index'], axis = 1)

hulu_mixed_countries_high_movies.head(5)

```

In[98]:

```

# prime_video_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_countries_movies = prime_video_mixed_countries_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_countries_high_movies = df_mixed_countries_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_countries_high_movies =
prime_video_mixed_countries_high_movies.drop(['index'], axis = 1)

prime_video_mixed_countries_low_movies = df_mixed_countries_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_mixed_countries_low_movies =
prime_video_mixed_countries_low_movies.drop(['index'], axis = 1)

prime_video_mixed_countries_high_movies.head(5)

```

In[99]:

```

# disney_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_countries_movies = disney_mixed_countries_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_mixed_countries_high_movies = df_mixed_countries_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_countries_high_movies = disney_mixed_countries_high_movies.drop(['index'], axis = 1)

disney_mixed_countries_low_movies = df_mixed_countries_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_countries_low_movies = disney_mixed_countries_low_movies.drop(['index'], axis = 1)

disney_mixed_countries_high_movies.head(5)

```

In[100]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_countries_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_countries_data_movies['Movies Count'], ax = ax[1])
plt.show()

```

In[101]:

```

# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_countries_movies = netflix_mixed_countries_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()

```

```

hulu_mixed_countries_movies = hulu_mixed_countries_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_countries_movies = prime_video_mixed_countries_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_mixed_countries_movies =
mixed_countries_data_movies[mixed_countries_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_countries_movies = disney_mixed_countries_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

```

In[102]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Country Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_countries_movies['Prime Video'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_mixed_countries_movies['Netflix'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_mixed_countries_movies['Hulu'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_mixed_countries_movies['Disney+'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[103]:

```

print(f'''
    The Mixed Country with Highest Movies Count Ever Got is
'{df_mixed_countries_high_movies['Mixed Country'][0]}' :
'{df_mixed_countries_high_movies['Movies Count'].max()}'\n
    The Mixed Country with Lowest Movies Count Ever Got is
'{df_mixed_countries_low_movies['Mixed Country'][0]}' :
'{df_mixed_countries_low_movies['Movies Count'].min()}'\n

    The Mixed Country with Highest Movies Count on 'Netflix' is
'{netflix_mixed_countries_high_movies['Mixed Country'][0]}' :
'{netflix_mixed_countries_high_movies['Netflix'].max()}'\n
    The Mixed Country with Lowest Movies Count on 'Netflix' is
'{netflix_mixed_countries_low_movies['Mixed Country'][0]}' :
'{netflix_mixed_countries_low_movies['Netflix'].min()}'\n

    The Mixed Country with Highest Movies Count on 'Hulu' is
'{hulu_mixed_countries_high_movies['Mixed Country'][0]}' :
'{hulu_mixed_countries_high_movies['Hulu'].max()}'\n
    The Mixed Country with Lowest Movies Count on 'Hulu' is
'{hulu_mixed_countries_low_movies['Mixed Country'][0]}' :
'{hulu_mixed_countries_low_movies['Hulu'].min()}'\n

```

```
The Mixed Country with Highest Movies Count on 'Prime Video' is
'{prime_video_mixed_countries_high_movies['Mixed Country'][0]}':
'{prime_video_mixed_countries_high_movies['Prime Video'].max()}'\n
The Mixed Country with Lowest Movies Count on 'Prime Video' is
'{prime_video_mixed_countries_low_movies['Mixed Country'][0]}':
'{prime_video_mixed_countries_low_movies['Prime Video'].min()}'\n
```

```
The Mixed Country with Highest Movies Count on 'Disney+' is
'{disney_mixed_countries_high_movies['Mixed Country'][0]}':
'{disney_mixed_countries_high_movies['Disney+'].max()}'\n
The Mixed Country with Lowest Movies Count on 'Disney+' is
'{disney_mixed_countries_low_movies['Mixed Country'][0]}':
'{disney_mixed_countries_low_movies['Disney+'].min()}'\n
''')
```

```
# In[104]:
```

```
print(f'''
    Accross All Platforms the Average Movies Count of Mixed Country is
'{round(mixed_countries_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Country on 'Netflix' is
'{round(netflix_mixed_countries_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Country on 'Hulu' is
'{round(hulu_mixed_countries_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Country on 'Prime Video' is
'{round(prime_video_mixed_countries_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Country on 'Disney+' is
'{round(disney_mixed_countries_movies['Disney+'].mean(), ndigits = 2)}'\n
'''')
```

```
# In[105]:
```

```
print(f'''
    Accross All Platforms Total Count of Mixed Country is
'{mixed_countries_data_movies['Mixed Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Netflix' is '{netflix_mixed_countries_movies['Mixed
Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Hulu' is '{hulu_mixed_countries_movies['Mixed
Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Prime Video' is
'{prime_video_mixed_countries_movies['Mixed Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Disney+' is '{disney_mixed_countries_movies['Mixed
Country'].unique().shape[0]}'\n
'''')
```

```
# In[106]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_countries_data_movies['Mixed Country'][:5], y =
mixed_countries_data_movies['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_countries_data_movies['Mixed Country'][:5], y =
mixed_countries_data_movies['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_countries_data_movies['Mixed Country'][:5], y =
mixed_countries_data_movies['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_countries_data_movies['Mixed Country'][:5], y =
mixed_countries_data_movies['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Country', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()
```

```
# In[107]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_co_ax1 = sns.barplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_co_ax2 = sns.barplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_co_ax3 = sns.barplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_co_ax4 = sns.barplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[108]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_mco_ax1 = sns.lineplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_mco_ax2 = sns.lineplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_mco_ax3 = sns.lineplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_mco_ax4 = sns.lineplot(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mco_ax1.title.set_text(labels[0])
h_mco_ax2.title.set_text(labels[1])
p_mco_ax3.title.set_text(labels[2])
d_mco_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[109]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Country Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_countries_movies['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_countries_movies['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_countries_movies['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_countries_movies['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
```

```

plt.show()

# In[110]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_mco_ax1 = sns.barplot(y = netflix_mixed_countries_movies['Mixed Country'][:10], x =
netflix_mixed_countries_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mco_ax2 = sns.barplot(y = hulu_mixed_countries_movies['Mixed Country'][:10], x =
hulu_mixed_countries_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mco_ax3 = sns.barplot(y = prime_video_mixed_countries_movies['Mixed Country'][:10], x =
prime_video_mixed_countries_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_mco_ax4 = sns.barplot(y = disney_mixed_countries_movies['Mixed Country'][:10], x =
disney_mixed_countries_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mco_ax1.title.set_text(labels[0])
h_mco_ax2.title.set_text(labels[1])
p_mco_ax3.title.set_text(labels[2])
d_mco_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[111]:
```

```

fig = go.Figure(go.Funnel(y = mixed_countries_data_movies['Mixed Country'][:10], x =
mixed_countries_data_movies['Movies Count'][:10]))
fig.show()

```

ottmovies_director.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

```

```
# In[2]:
```

```

# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

```

```
# In[3]:  
  
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import warnings  
import collections  
import plotly.express as px  
import plotly.graph_objects as go  
import nltk  
import re  
from nltk.corpus import stopwords  
from nltk.tokenize import word_tokenize  
from nltk.probability import FreqDist  
from nltk.util import ngrams  
from plotly.subplots import make_subplots  
from plotly.offline import iplot, init_notebook_mode  
from wordcloud import WordCloud, STOPWORDS  
from pandas_profiling import ProfileReport  
get_ipython().run_line_magic('matplotlib', 'inline')  
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'  
  
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'  
  
df_movies = pd.read_csv(path + 'ottmovies.csv')  
  
df_movies.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_movies)  
# profile
```

```
# In[7]:
```

```
def data_investigate(df):  
    print('No of Rows : ', df.shape[0])  
    print('No of Coloums : ', df.shape[1])  
    print('***'*25)  
    print('Colums Names : \n', df.columns)  
    print('***'*25)  
    print('Datatype of Columns : \n', df.dtypes)  
    print('***'*25)  
    print('Missing Values : ')  
    c = df.isnull().sum()  
    c = c[c > 0]
```

```

print(c)
print('***25)
print('Missing values %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

# In[8]:

data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

```

```

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:

```

```

# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:


# df_netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df_hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df_prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df_disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]


# In[17]:


df_netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df_disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]


# In[18]:


df_movies_directors = df_movies.copy()


# In[19]:


df_movies_directors.drop(df_movies_directors.loc[df_movies_directors['Directors'] == "NA"].index, inplace = True)
# df_movies_directors = df_movies_directors[df_movies_directors.Director != "NA"]
# df_movies_directors['Director'] = df_movies_directors['Director'].astype(str)


# In[20]:


df_movies_count_directors = df_movies_directors.copy()


# In[21]:


df_movies_director = df_movies_directors.copy()


# In[22]:


# Create directors dict where key=name and value = number of directors

```

```

directors = {}

for i in df_movies_count_directors['Directors'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        directors[i] = len(i.split(','))
    else:
        directors[i] = 0

# Add this information to our dataframe as a new column

df_movies_count_directors['Number of Directors'] =
df_movies_count_directors['Directors'].map(directors).astype(int)

# In[23]:


df_movies_mixed_directors = df_movies_count_directors.copy()

# In[24]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_directors_movies =
df_movies_count_directors.loc[df_movies_count_directors['Netflix'] == 1]
hulu_directors_movies = df_movies_count_directors.loc[df_movies_count_directors['Hulu'] ==
1]
prime_video_directors_movies =
df_movies_count_directors.loc[df_movies_count_directors['Prime Video'] == 1]
disney_directors_movies =
df_movies_count_directors.loc[df_movies_count_directors['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_movies_count_directors.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_directors_most_movies = df_movies_count_directors.sort_values(by = 'Number of
Directors', ascending = False).reset_index()
df_directors_most_movies = df_directors_most_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_directors['Number of Directors'] ==
(df_movies_count_directors['Number of Directors'].max()))
# df_directors_most_movies = df_movies_count_directors[filter]

# mostest_rated_movies = df_movies_count_directors.loc[df_movies_count_directors['Number of
Directors'].idxmax()]

```

```

print('\nMovies with Highest Ever Number of Directors are : \n')
df_directors_most_movies.head(5)

# In[27]:


fig = px.bar(y = df_directors_most_movies['Title'][:15],
              x = df_directors_most_movies['Number of Directors'][:15],
              color = df_directors_most_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Highest Number of Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[28]:


df_directors_least_movies = df_movies_count_directors.sort_values(by = 'Number of Directors', ascending = True).reset_index()
df_directors_least_movies = df_directors_least_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_directors['Number of Directors'] ==
# (df_movies_count_directors['Number of Directors'].min()))
# df_directors_least_movies = df_movies_count_directors[filter]

print('\nMovies with Lowest Ever Number of Directors are : \n')
df_directors_least_movies.head(5)

# In[29]:


fig = px.bar(y = df_directors_least_movies['Title'][:15],
              x = df_directors_least_movies['Number of Directors'][:15],
              color = df_directors_least_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Lowest Number of Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
      Total '{df_movies_count_directors['Number of Directors'].unique().shape[0]}' unique
      Number of Directors s were Given, They were Like this,\n

      {df_movies_count_directors.sort_values(by = 'Number of Directors', ascending =
      False)['Number of Directors'].unique()}\n

      The Highest Number of Directors Ever Any Movie Got is
      '{df_directors_most_movies['Title'][0]}' : '{df_directors_most_movies['Number of
      Directors'].max()}'\n

      The Lowest Number of Directors Ever Any Movie Got is
      '{df_directors_least_movies['Title'][0]}' : '{df_directors_least_movies['Number of
      Directors'].min()}'\n
      ''')

```

```

# In[31]:


netflix_directors_most_movies =
df_directors_most_movies.loc[df_directors_most_movies['Netflix']==1].reset_index()
netflix_directors_most_movies = netflix_directors_most_movies.drop(['index'], axis = 1)

netflix_directors_least_movies =
df_directors_least_movies.loc[df_directors_least_movies['Netflix']==1].reset_index()
netflix_directors_least_movies = netflix_directors_least_movies.drop(['index'], axis = 1)

netflix_directors_most_movies.head(5)

# In[32]:


fig = px.bar(y = netflix_directors_most_movies['Title'][:15],
              x = netflix_directors_most_movies['Number of Directors'][:15],
              color = netflix_directors_most_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Highest Number of Directors : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_directors_least_movies['Title'][:15],
              x = netflix_directors_least_movies['Number of Directors'][:15],
              color = netflix_directors_least_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Lowest Number of Directors : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


hulu_directors_most_movies =
df_directors_most_movies.loc[df_directors_most_movies['Hulu']==1].reset_index()
hulu_directors_most_movies = hulu_directors_most_movies.drop(['index'], axis = 1)

hulu_directors_least_movies =
df_directors_least_movies.loc[df_directors_least_movies['Hulu']==1].reset_index()
hulu_directors_least_movies = hulu_directors_least_movies.drop(['index'], axis = 1)

hulu_directors_most_movies.head(5)

# In[35]:


fig = px.bar(y = hulu_directors_most_movies['Title'][:15],
              x = hulu_directors_most_movies['Number of Directors'][:15],
              color = hulu_directors_most_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Highest Number of Directors : Hulu')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_directors_least_movies['Title'][:15],
              x = hulu_directors_least_movies['Number of Directors'][:15],
              color = hulu_directors_least_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Lowest Number of Directors : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_directors_most_movies =
df_directors_most_movies.loc[df_directors_most_movies['Prime Video']==1].reset_index()
prime_video_directors_most_movies = prime_video_directors_most_movies.drop(['index'], axis = 1)

prime_video_directors_least_movies =
df_directors_least_movies.loc[df_directors_least_movies['Prime Video']==1].reset_index()
prime_video_directors_least_movies = prime_video_directors_least_movies.drop(['index'], axis = 1)

prime_video_directors_most_movies.head(5)

# In[38]:


fig = px.bar(y = prime_video_directors_most_movies['Title'][:15],
              x = prime_video_directors_most_movies['Number of Directors'][:15],
              color = prime_video_directors_most_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Highest Number of Directors : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[39]:


fig = px.bar(y = prime_video_directors_least_movies['Title'][:15],
              x = prime_video_directors_least_movies['Number of Directors'][:15],
              color = prime_video_directors_least_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Lowest Number of Directors : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:

```

```

disney_directors_most_movies =
df_directors_most_movies.loc[df_directors_most_movies['Disney+']==1].reset_index()
disney_directors_most_movies = disney_directors_most_movies.drop(['index'], axis = 1)

disney_directors_least_movies =
df_directors_least_movies.loc[df_directors_least_movies['Disney+']==1].reset_index()
disney_directors_least_movies = disney_directors_least_movies.drop(['index'], axis = 1)

disney_directors_most_movies.head(5)

# In[41]:


fig = px.bar(y = disney_directors_most_movies['Title'][:15],
              x = disney_directors_most_movies['Number of Directors'][:15],
              color = disney_directors_most_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Highest Number of Directors : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_directors_least_movies['Title'][:15],
              x = disney_directors_least_movies['Number of Directors'][:15],
              color = disney_directors_least_movies['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Directors'},
              title = 'Movies with Lowest Number of Directors : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''
      The Movie with Highest Number of Directors Ever Got is
'{df_directors_most_movies['Title'][0]}' : '{df_directors_most_movies['Number of Directors'].max()}'\n
      The Movie with Lowest Number of Directors Ever Got is
'{df_directors_least_movies['Title'][0]}' : '{df_directors_least_movies['Number of Directors'].min()}'\n\n
      The Movie with Highest Number of Directors on 'Netflix' is
'{netflix_directors_most_movies['Title'][0]}' : '{netflix_directors_most_movies['Number of Directors'].max()}'\n
      The Movie with Lowest Number of Directors on 'Netflix' is
'{netflix_directors_least_movies['Title'][0]}' : '{netflix_directors_least_movies['Number of Directors'].min()}'\n\n
      The Movie with Highest Number of Directors on 'Hulu' is
'{hulu_directors_most_movies['Title'][0]}' : '{hulu_directors_most_movies['Number of Directors'].max()}'\n
      The Movie with Lowest Number of Directors on 'Hulu' is
'{hulu_directors_least_movies['Title'][0]}' : '{hulu_directors_least_movies['Number of Directors'].min()}'\n

```

```

    The Movie with Highest Number of Directors on 'Prime Video' is
'{prime_video_directors_most_movies['Title'][0]}' :
'{prime_video_directors_most_movies['Number of Directors'].max()}'\n
    The Movie with Lowest Number of Directors on 'Prime Video' is
'{prime_video_directors_least_movies['Title'][0]}' :
'{prime_video_directors_least_movies['Number of Directors'].min()}'\n

    The Movie with Highest Number of Directors on 'Disney+' is
'{disney_directors_most_movies['Title'][0]}' : '{disney_directors_most_movies['Number of
Directors'].max()}'\n
    The Movie with Lowest Number of Directors on 'Disney+' is
'{disney_directors_least_movies['Title'][0]}' : '{disney_directors_least_movies['Number of
Directors'].min()}'\n
    ''')

```

In[44]:

```

print(f'''
    Accross All Platforms the Average Number of Directors is
'{round(df_movies_count_directors['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Netflix' is
'{round(netflix_directors_movies['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Hulu' is '{round(hulu_directors_movies['Number of
Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Prime Video' is
'{round(prime_video_directors_movies['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Disney+' is
'{round(disney_directors_movies['Number of Directors'].mean(), ndigits = 2)}'\n
    ''')

```

In[45]:

```

print(f'''
    Accross All Platforms Total Count of Director is '{df_movies_count_directors['Number
of Directors'].max()}'\n
    Total Count of Director on 'Netflix' is '{netflix_directors_movies['Number of
Directors'].max()}'\n
    Total Count of Director on 'Hulu' is '{hulu_directors_movies['Number of
Directors'].max()}'\n
    Total Count of Director on 'Prime Video' is '{prime_video_directors_movies['Number of
Directors'].max()}'\n
    Total Count of Director on 'Disney+' is '{disney_directors_movies['Number of
Directors'].max()}'\n
    ''')

```

In[46]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_count_directors['Number of Directors'],bins = 20, kde = True, ax =
ax[0])
sns.boxplot(df_movies_count_directors['Number of Directors'], ax = ax[1])
plt.show()

```

In[47]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Number of Directors s Per Platform')

```

```

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_directors_movies['Number of Directors'], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_directors_movies['Number of Directors'], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_directors_movies['Number of Directors'], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_directors_movies['Number of Directors'], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[48]:

```

df_lan = df_movies_director['Directors'].str.split(',').apply(pd.Series).stack()
del df_movies_director['Directors']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Director'
df_movies_director = df_movies_director.join(df_lan)
df_movies_director.drop_duplicates(inplace = True)

```

In[49]:

```
df_movies_director.head(5)
```

In[50]:

```

director_count = df_movies_director.groupby('Director')['Title'].count()
director_movies = df_movies_director.groupby('Director')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
director_data_movies = pd.concat([director_count, director_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
director_data_movies = director_data_movies.sort_values(by = 'Movies Count', ascending =
False)

```

In[51]:

```

# Director with Movies Counts - All Platforms Combined
director_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

```

In[52]:

```

fig = px.bar(x = director_data_movies['Director'][:50],
              y = director_data_movies['Movies Count'][:50],
              color = director_data_movies['Movies Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Director', 'y' : 'Movies Count'},
              title = 'Major Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[53]:


df_director_high_movies = director_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_director_high_movies = df_director_high_movies.drop(['index'], axis = 1)
# filter = (director_data_movies['Movies Count'] == (director_data_movies['Movies Count'].max()))
# df_director_high_movies = director_data_movies[filter]

# highestRatedMovies = director_data_movies.loc[director_data_movies['Movies Count'].idxmax()]

print('\nDirector with Highest Ever Movies Count are : All Platforms Combined\n')
df_director_high_movies.head(5)

# In[54]:


fig = px.bar(y = df_director_high_movies['Director'][:15],
              x = df_director_high_movies['Movies Count'][:15],
              color = df_director_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'Movies Count'},
              title = 'Director with Highest Movies : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[55]:


df_director_low_movies = director_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_director_low_movies = df_director_low_movies.drop(['index'], axis = 1)
# filter = (director_data_movies['Movies Count'] == (director_data_movies['Movies Count'].min()))
# df_director_low_movies = director_data_movies[filter]

print('\nDirector with Lowest Ever Movies Count are : All Platforms Combined\n')
df_director_low_movies.head(5)

# In[56]:


fig = px.bar(y = df_director_low_movies['Director'][:15],
              x = df_director_low_movies['Movies Count'][:15],
              color = df_director_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'Movies Count'},
              title = 'Director with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[57]:


print(f'''
      Total '{director_data_movies['Director'].unique().shape[0]}' unique Director Count s
      were Given, They were Like this,\n
    ''')

```

```

    {director_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Director'].unique()[:5]}\n

    The Highest Ever Movies Count Ever Any Movie Got is
'{df_director_high_movies['Director'][0]} : '{df_director_high_movies['Movies
Count'].max()}'\n

    The Lowest Ever Movies Count Ever Any Movie Got is
'{df_director_low_movies['Director'][0]} : '{df_director_low_movies['Movies
Count'].min()}'\n
    ''')

# In[58]:\n\n

fig = px.pie(director_data_movies[:10], names = 'Director', values = 'Movies Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Director')
fig.show()\n\n

# In[59]:\n\n

# netflix_director_movies = director_data_movies[director_data_movies['Netflix'] !=
0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_director_movies = netflix_director_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

netflix_director_high_movies = df_director_high_movies.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_director_high_movies = netflix_director_high_movies.drop(['index'], axis = 1)

netflix_director_low_movies = df_director_low_movies.sort_values(by = 'Netflix', ascending
= True).reset_index()
netflix_director_low_movies = netflix_director_low_movies.drop(['index'], axis = 1)

netflix_director_high_movies.head(5)\n\n

# In[60]:\n\n

fig = px.bar(x = netflix_director_high_movies['Director'][:15],
y = netflix_director_high_movies['Netflix'][:15],
color = netflix_director_high_movies['Netflix'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Director', 'x' : 'Movies Count'},
title = 'Director with Highest Movies : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()\n\n

# In[61]:\n\n

# hulu_director_movies = director_data_movies[director_data_movies['Hulu'] !=
0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_director_movies = hulu_director_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_director_high_movies = df_director_high_movies.sort_values(by = 'Hulu', ascending =
False).reset_index()

```

```

hulu_director_high_movies = hulu_director_high_movies.drop(['index'], axis = 1)

hulu_director_low_movies = df_director_high_movies.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_director_low_movies = hulu_director_low_movies.drop(['index'], axis = 1)

hulu_director_high_movies.head(5)

# In[62]:


fig = px.bar(x = hulu_director_high_movies['Director'][:15],
              y = hulu_director_high_movies['Hulu'][:15],
              color = hulu_director_high_movies['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'Movies Count'},
              title = 'Director with Highest Movies : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


# prime_video_director_movies = director_data_movies[director_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_director_movies = prime_video_director_movies.drop(['index', 'Netflix',
'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_director_high_movies = df_director_high_movies.sort_values(by = 'Prime Video',
ascending = False).reset_index()
prime_video_director_high_movies = prime_video_director_high_movies.drop(['index'], axis =
1)

prime_video_director_low_movies = df_director_high_movies.sort_values(by = 'Prime Video',
ascending = True).reset_index()
prime_video_director_low_movies = prime_video_director_low_movies.drop(['index'], axis = 1)

prime_video_director_high_movies.head(5)

# In[64]:


fig = px.bar(x = prime_video_director_high_movies['Director'][:15],
              y = prime_video_director_high_movies['Prime Video'][:15],
              color = prime_video_director_high_movies['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'Movies Count'},
              title = 'Director with Highest Movies : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[65]:


# disney_director_movies = director_data_movies[director_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_director_movies = disney_director_movies.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'Movies Count'], axis = 1)

```

```

disney_director_high_movies = df_director_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_director_high_movies = disney_director_high_movies.drop(['index'], axis = 1)

disney_director_low_movies = df_director_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_director_low_movies = disney_director_low_movies.drop(['index'], axis = 1)

disney_director_high_movies.head(5)

# In[66]:


fig = px.bar(x = disney_director_high_movies['Director'][:15],
              y = disney_director_high_movies['Disney+'][:15],
              color = disney_director_high_movies['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'Movies Count'},
              title = 'Director with Highest Movies : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[67]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(director_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(director_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_director_movies = director_data_movies[director_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_director_movies = netflix_director_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_director_movies = director_data_movies[director_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_director_movies = hulu_director_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_director_movies = director_data_movies[director_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_director_movies = prime_video_director_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_director_movies = director_data_movies[director_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_director_movies = disney_director_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Director Movies Count Per Platform')

```

```

# Plotting the information from each dataset into a histogram

sns.histplot(disney_director_movies['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_director_movies['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_director_movies['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_director_movies['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

```

In[70]:

```

print(f'''
    The Director with Highest Movies Count Ever Got is
'{df_director_high_movies['Director'][0]}' : '{df_director_high_movies['Movies
Count'].max()}'\n
    The Director with Lowest Movies Count Ever Got is
'{df_director_low_movies['Director'][0]}' : '{df_director_low_movies['Movies
Count'].min()}'\n

    The Director with Highest Movies Count on 'Netflix' is
'{netflix_director_high_movies['Director'][0]}' :
'{netflix_director_high_movies['Netflix'].max()}'\n
    The Director with Lowest Movies Count on 'Netflix' is
'{netflix_director_low_movies['Director'][0]}' :
'{netflix_director_low_movies['Netflix'].min()}'\n

    The Director with Highest Movies Count on 'Hulu' is
'{hulu_director_high_movies['Director'][0]}' :
'{hulu_director_high_movies['Hulu'].max()}'\n
    The Director with Lowest Movies Count on 'Hulu' is
'{hulu_director_low_movies['Director'][0]}' : '{hulu_director_low_movies['Hulu'].min()}'\n

    The Director with Highest Movies Count on 'Prime Video' is
'{prime_video_director_high_movies['Director'][0]}' :
'{prime_video_director_high_movies['Prime Video'].max()}'\n
    The Director with Lowest Movies Count on 'Prime Video' is
'{prime_video_director_low_movies['Director'][0]}' :
'{prime_video_director_low_movies['Prime Video'].min()}'\n

    The Director with Highest Movies Count on 'Disney+' is
'{disney_director_high_movies['Director'][0]}' :
'{disney_director_high_movies['Disney+'].max()}'\n
    The Director with Lowest Movies Count on 'Disney+' is
'{disney_director_low_movies['Director'][0]}' :
'{disney_director_low_movies['Disney+'].min()}'\n
    ''')

```

In[71]:

```

# Distribution of movies director in each platform
plt.figure(figsize = (20, 5))
plt.title('Director with Movies Count for All Platforms')
sns.violinplot(x = director_data_movies['Movies Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

```

```
# In[72]:
# Distribution of Director Movies Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_director_movies['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_director_movies['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_director_movies['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_director_movies['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()

# In[73]:
print(f'''
    Across All Platforms the Average Movies Count of Director is
'{round(director_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Director on 'Netflix' is
'{round(netflix_director_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Director on 'Hulu' is
'{round(hulu_director_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Director on 'Prime Video' is
'{round(prime_video_director_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Director on 'Disney+' is
'{round(disney_director_movies['Disney+'].mean(), ndigits = 2)}'
'''')

# In[74]:
print(f'''
    Across All Platforms Total Count of Director is
'{director_data_movies['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Netflix' is
'{netflix_director_movies['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Hulu' is
'{hulu_director_movies['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Prime Video' is
'{prime_video_director_movies['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Disney+' is
'{disney_director_movies['Director'].unique().shape[0]}'
'''')

# In[75]:
plt.figure(figsize = (20, 5))
sns.lineplot(x = director_data_movies['Director'][:10], y = director_data_movies['Netflix'][:10], color = 'red')
sns.lineplot(x = director_data_movies['Director'][:10], y = director_data_movies['Hulu'][:10], color = 'lightgreen')
sns.lineplot(x = director_data_movies['Director'][:10], y = director_data_movies['Prime Video'][:10], color = 'lightblue')
sns.lineplot(x = director_data_movies['Director'][:10], y = director_data_movies['Disney+'][:10], color = 'darkblue')
plt.xlabel('Director', fontsize = 20)
plt.ylabel('Movies Count', fontsize = 20)
```

```

plt.show()

# In[76]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_d_ax1 = sns.lineplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_d_ax2 = sns.lineplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_d_ax3 = sns.lineplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_d_ax4 = sns.lineplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

```

```
plt.show()
```

```
# In[77]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(y = netflix_director_movies['Director'][:10], x =
netflix_director_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(y = hulu_director_movies['Director'][:10], x =
hulu_director_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(y = prime_video_director_movies['Director'][:10], x =
prime_video_director_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(y = disney_director_movies['Director'][:10], x =
disney_director_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

```

```
plt.show()
```

```
# In[78]:
```

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Director Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_director_movies['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_director_movies['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_director_movies['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_director_movies['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend

```

```
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[79]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(y = director_data_movies['Director'][:10], x =
director_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[80]:
```

```
df_movies_mixed_directors.drop(df_movies_mixed_directors.loc[df_movies_mixed_directors['Directors'] == "NA"].index, inplace = True)
# df_movies_mixed_directors = df_movies_mixed_directors[df_movies_mixed_directors.Directors != "NA"]
df_movies_mixed_directors.drop(df_movies_mixed_directors.loc[df_movies_mixed_directors['Number of Directors'] == 1].index, inplace = True)
```

```
# In[81]:
```

```
df_movies_mixed_directors.head(5)
```

```
# In[82]:
```

```
mixed_directors_count = df_movies_mixed_directors.groupby('Directors')['Title'].count()
mixed_directors_movies = df_movies_mixed_directors.groupby('Directors')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_directors_data_movies = pd.concat([mixed_directors_count, mixed_directors_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count', 'Directors' : 'Mixed Director'})
mixed_directors_data_movies = mixed_directors_data_movies.sort_values(by = 'Movies Count', ascending = False)
```

```
# In[83]:
```

```
mixed_directors_data_movies.head(5)
```

```
# In[84]:
```

```

# Mixed Director with Movies Counts - All Platforms Combined
mixed_directors_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[85]:


df_mixed_directors_high_movies = mixed_directors_data_movies.sort_values(by = 'Movies
Count', ascending = False).reset_index()
df_mixed_directors_high_movies = df_mixed_directors_high_movies.drop(['index'], axis = 1)
# filter = (mixed_directors_data_movies['Movies Count'] == 
(mixed_directors_data_movies['Movies Count'].max()))
# df_mixed_directors_high_movies = mixed_directors_data_movies[filter]

# highest_rated_movies =
mixed_directors_data_movies.loc[mixed_directors_data_movies['Movies Count'].idxmax()]

print('\nMixed Director with Highest Ever Movies Count are : All Platforms Combined\n')
df_mixed_directors_high_movies.head(5)

# In[86]:


fig = px.bar(y = df_mixed_directors_high_movies['Mixed Director'][:15],
              x = df_mixed_directors_high_movies['Movies Count'][:15],
              color = df_mixed_directors_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Director'},
              title = 'Movies with Highest Number of Mixed Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[87]:


df_mixed_directors_low_movies = mixed_directors_data_movies.sort_values(by = 'Movies
Count', ascending = True).reset_index()
df_mixed_directors_low_movies = df_mixed_directors_low_movies.drop(['index'], axis = 1)
# filter = (mixed_directors_data_movies['Movies Count'] == 
(mixed_directors_data_movies['Movies Count'].min()))
# df_mixed_directors_low_movies = mixed_directors_data_movies[filter]

print('\nMixed Director with Lowest Ever Movies Count are : All Platforms Combined\n')
df_mixed_directors_low_movies.head(5)

# In[88]:


fig = px.bar(y = df_mixed_directors_low_movies['Mixed Director'][:15],
              x = df_mixed_directors_low_movies['Movies Count'][:15],
              color = df_mixed_directors_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Director'},
              title = 'Movies with Lowest Number of Mixed Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[89]:

```

```

print(f'''
    Total '{df_movies_directors['Directors'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see Movies from Total '{mixed_directors_data_movies['Mixed
Director'].unique().shape[0]}' Mixed Director, They were Like this, \n

    {mixed_directors_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Mixed Director'].head(5).unique()} etc. \n

    The Mixed Director with Highest Movies Count have
'{mixed_directors_data_movies['Movies Count'].max()}' Movies Available is
'{df_mixed_directors_high_movies['Mixed Director'][0]}', &\n
    The Mixed Director with Lowest Movies Count have
'{mixed_directors_data_movies['Movies Count'].min()}' Movies Available is
'{df_mixed_directors_low_movies['Mixed Director'][0]}'
    ''')

```

In[90]:

```

fig = px.pie(mixed_directors_data_movies[:4], names = 'Mixed Director', values = 'Movies
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Mixed Director')
fig.show()

```

In[91]:

```

# netflix_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
# netflix_mixed_directors_movies = netflix_mixed_directors_movies.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_mixed_directors_high_movies = df_mixed_directors_high_movies.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_directors_high_movies = netflix_mixed_directors_high_movies.drop(['index'],
axis = 1)

netflix_mixed_directors_low_movies = df_mixed_directors_high_movies.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_mixed_directors_low_movies = netflix_mixed_directors_low_movies.drop(['index'],
axis = 1)

netflix_mixed_directors_high_movies.head(5)

```

In[92]:

```

# hulu_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_mixed_directors_movies = hulu_mixed_directors_movies.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_directors_high_movies = df_mixed_directors_high_movies.sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_directors_high_movies = hulu_mixed_directors_high_movies.drop(['index'], axis =
1)

```

```

hulu_mixed_directors_low_movies = df_mixed_directors_high_movies.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_directors_low_movies = hulu_mixed_directors_low_movies.drop(['index'], axis = 1)

hulu_mixed_directors_high_movies.head(5)

# In[93]:


# prime_video_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_directors_movies = prime_video_mixed_directors_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_directors_high_movies = df_mixed_directors_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_directors_high_movies =
prime_video_mixed_directors_high_movies.drop(['index'], axis = 1)

prime_video_mixed_directors_low_movies = df_mixed_directors_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_mixed_directors_low_movies =
prime_video_mixed_directors_low_movies.drop(['index'], axis = 1)

prime_video_mixed_directors_high_movies.head(5)

# In[94]:


# disney_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_directors_movies = disney_mixed_directors_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_mixed_directors_high_movies = df_mixed_directors_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_directors_high_movies = disney_mixed_directors_high_movies.drop(['index'], axis = 1)

disney_mixed_directors_low_movies = df_mixed_directors_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_directors_low_movies = disney_mixed_directors_low_movies.drop(['index'], axis = 1)

disney_mixed_directors_high_movies.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_directors_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_directors_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms

```

```

netflix_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_directors_movies = netflix_mixed_directors_movies.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_directors_movies = hulu_mixed_directors_movies.drop(['index', 'Netflix', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Prime Video'] != 0].sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_directors_movies = prime_video_mixed_directors_movies.drop(['index',
'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_mixed_directors_movies =
mixed_directors_data_movies[mixed_directors_data_movies['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_directors_movies = disney_mixed_directors_movies.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

```

In[97]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Director Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_directors_movies['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_directors_movies['Netflix'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_mixed_directors_movies['Hulu'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_mixed_directors_movies['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Director with Highest Movies Count Ever Got is
'{df_mixed_directors_high_movies['Mixed Director'][0]}' :
'{df_mixed_directors_high_movies['Movies Count'].max()}'\n
    The Mixed Director with Lowest Movies Count Ever Got is
'{df_mixed_directors_low_movies['Mixed Director'][0]}' :
'{df_mixed_directors_low_movies['Movies Count'].min()}'\n

    The Mixed Director with Highest Movies Count on 'Netflix' is
'{netflix_mixed_directors_high_movies['Mixed Director'][0]}' :
'{netflix_mixed_directors_high_movies['Netflix'].max()}'\n
    The Mixed Director with Lowest Movies Count on 'Netflix' is
'{netflix_mixed_directors_low_movies['Mixed Director'][0]}' :
'{netflix_mixed_directors_low_movies['Netflix'].min()}'\n

```

```

    The Mixed Director with Highest Movies Count on 'Hulu' is
'{hulu_mixed_directors_high_movies['Mixed Director'][0]}' :
'{hulu_mixed_directors_high_movies['Hulu'].max()}'\n
    The Mixed Director with Lowest Movies Count on 'Hulu' is
'{hulu_mixed_directors_low_movies['Mixed Director'][0]}' :
'{hulu_mixed_directors_low_movies['Hulu'].min()}'\n

    The Mixed Director with Highest Movies Count on 'Prime Video' is
'{prime_video_mixed_directors_high_movies['Mixed Director'][0]}' :
'{prime_video_mixed_directors_high_movies['Prime Video'].max()}'\n
    The Mixed Director with Lowest Movies Count on 'Prime Video' is
'{prime_video_mixed_directors_low_movies['Mixed Director'][0]}' :
'{prime_video_mixed_directors_low_movies['Prime Video'].min()}'\n

    The Mixed Director with Highest Movies Count on 'Disney+' is
'{disney_mixed_directors_high_movies['Mixed Director'][0]}' :
'{disney_mixed_directors_high_movies['Disney+'].max()}'\n
    The Mixed Director with Lowest Movies Count on 'Disney+' is
'{disney_mixed_directors_low_movies['Mixed Director'][0]}' :
'{disney_mixed_directors_low_movies['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Accross All Platforms the Average Movies Count of Mixed Director is
'{round(mixed_directors_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Director on 'Netflix' is
'{round(netflix_mixed_directors_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Director on 'Hulu' is
'{round(hulu_mixed_directors_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Director on 'Prime Video' is
'{round(prime_video_mixed_directors_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Director on 'Disney+' is
'{round(disney_mixed_directors_movies['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```

print(f'''
    Accross All Platforms Total Count of Mixed Director is
'{mixed_directors_data_movies['Mixed Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Netflix' is '{netflix_mixed_directors_movies['Mixed
Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Hulu' is '{hulu_mixed_directors_movies['Mixed
Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Prime Video' is
'{prime_video_mixed_directors_movies['Mixed Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Disney+' is '{disney_mixed_directors_movies['Mixed
Director'].unique().shape[0]}'\n
    ''')

```

In[101]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:5], y =
mixed_directors_data_movies['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:5], y =
mixed_directors_data_movies['Hulu'][:5], color = 'lightgreen')

```

```

sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:5], y =
mixed_directors_data_movies['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:5], y =
mixed_directors_data_movies['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Director', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

# In[102]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()

# In[103]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_md_ax1 = sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_md_ax2 = sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_md_ax3 = sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_md_ax4 = sns.lineplot(x = mixed_directors_data_movies['Mixed Director'][:10], y =
mixed_directors_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_md_ax1.title.set_text(labels[0])
h_md_ax2.title.set_text(labels[1])
p_md_ax3.title.set_text(labels[2])
d_md_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Director Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_directors_movies['Netflix'][:50], color = 'red', legend = True)

```

```

sns.kdeplot(hulu_mixed_directors_movies['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_directors_movies['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_directors_movies['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[105]:
```

`fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_md_ax1 = sns.barplot(x = netflix_mixed_directors_movies['Mixed Director'][:10], y =
netflix_mixed_directors_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_md_ax2 = sns.barplot(x = hulu_mixed_directors_movies['Mixed Director'][:10], y =
hulu_mixed_directors_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_md_ax3 = sns.barplot(x = prime_video_mixed_directors_movies['Mixed Director'][:10], y =
prime_video_mixed_directors_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_md_ax4 = sns.barplot(x = disney_mixed_directors_movies['Mixed Director'][:10], y =
disney_mixed_directors_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_md_ax1.title.set_text(labels[0])
h_md_ax2.title.set_text(labels[1])
p_md_ax3.title.set_text(labels[2])
d_md_ax4.title.set_text(labels[3])

plt.show()`

```
# In[106]:
```

`fig = go.Figure(go.Funnel(y = mixed_directors_data_movies['Mixed Director'][:10], x =
mixed_directors_data_movies['Movies Count'][:10]))
fig.show()`

ottmovies_genre.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

```

```
# In[2]:  
  
# Import Dataset  
# Import File from Loacal Drive  
# from google.colab import files  
# data_to_load = files.upload()  
# from google.colab import drive  
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import warnings  
import collections  
import plotly.express as px  
import plotly.graph_objects as go  
import nltk  
import re  
from nltk.corpus import stopwords  
from nltk.tokenize import word_tokenize  
from nltk.probability import FreqDist  
from nltk.util import ngrams  
from plotly.subplots import make_subplots  
from plotly.offline import iplot, init_notebook_mode  
from wordcloud import WordCloud, STOPWORDS  
from pandas_profiling import ProfileReport  
get_ipython().run_line_magic('matplotlib', 'inline')  
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'  
  
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'  
  
df_movies = pd.read_csv(path + 'ottmovies.csv')  
  
df_movies.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_movies)  
# profile
```

```
# In[7]:
```

```
def data_investigate(df):  
    print('No of Rows : ', df.shape[0])
```

```

print('No of Coloums : ', df.shape[1])
print('***25)
print('Colums Names : \n', df.columns)
print('***25)
print('Datatype of Columns : \n', df.dtypes)
print('***25)
print('Missing Values : ')
c = df.isnull().sum()
c = c[c > 0]
print(c)
print('***25)
print('Missing vaules %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

```

In[8]:

```
data_investigate(df_movies)
```

In[9]:

```

# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

```

```

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

```

```

# In[14]:


# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:


# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:


# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]


# In[17]:


df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] ==
0) & (df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]


# In[18]:


df_movies_genres = df_movies.copy()

# In[19]:


df_movies_genres.drop(df_movies_genres.loc[df_movies_genres['Genres'] == "NA"].index,
inplace = True)
# df_movies_genres = df_movies_genres[df_movies_genres.Genre != "NA"]
# df_movies_genres['Genres'] = df_movies_genres['Genres'].astype(str)

# In[20]:


df_movies_count_genres = df_movies_genres.copy()

# In[21]:

```

```

df_movies_genre = df_movies_genres.copy()

# In[22]:

# Create genres dict where key=name and value = number of genres
genres = {}

for i in df_movies_count_genres['Genres'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        genres[i] = len(i.split(','))
    else:
        genres[i] = 0

# Add this information to our dataframe as a new column

df_movies_count_genres['Number of Genres'] =
df_movies_count_genres['Genres'].map(genres).astype(int)

# In[23]:


df_movies_mixed_genres = df_movies_count_genres.copy()

# In[24]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_genres_movies = df_movies_count_genres.loc[df_movies_count_genres['Netflix'] == 1]
hulu_genres_movies = df_movies_count_genres.loc[df_movies_count_genres['Hulu'] == 1]
prime_video_genres_movies = df_movies_count_genres.loc[df_movies_count_genres['Prime
Video'] == 1]
disney_genres_movies = df_movies_count_genres.loc[df_movies_count_genres['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_movies_count_genres.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_genres_most_movies = df_movies_count_genres.sort_values(by = 'Number of Genres',
ascending = False).reset_index()
df_genres_most_movies = df_genres_most_movies.drop(['index'], axis = 1)

```

```

# filter = (df_movies_count_genres['Number of Genres'] == (df_movies_count_genres['Number of Genres'].max()))
# df_genres_most_movies = df_movies_count_genres[filter]

# mostest_rated_movies = df_movies_count_genres.loc[df_movies_count_genres['Number of Genres'].idxmax()]

print('\nMovies with Highest Ever Number of Genres are : \n')
df_genres_most_movies.head(5)

# In[27]:


fig = px.bar(y = df_genres_most_movies['Title'][:15],
              x = df_genres_most_movies['Number of Genres'][:15],
              color = df_genres_most_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Highest Number of Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[28]:


df_genres_least_movies = df_movies_count_genres.sort_values(by = 'Number of Genres',
                                                               ascending = True).reset_index()
df_genres_least_movies = df_genres_least_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_genres['Number of Genres'] == (df_movies_count_genres['Number of Genres'].min()))
# df_genres_least_movies = df_movies_count_genres[filter]

print('\nMovies with Lowest Ever Number of Genres are : \n')
df_genres_least_movies.head(5)

# In[29]:


fig = px.bar(y = df_genres_least_movies['Title'][:15],
              x = df_genres_least_movies['Number of Genres'][:15],
              color = df_genres_least_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Lowest Number of Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
      Total '{df_movies_count_genres['Number of Genres'].unique().shape[0]}' unique Number of Genres s were Given, They were Like this,\n
      {df_movies_count_genres.sort_values(by = 'Number of Genres', ascending = False)[['Number of Genres']].unique()}\n\n
      The Highest Number of Genres Ever Any Movie Got is
      '{df_genres_most_movies['Title'][0]}' : '{df_genres_most_movies['Number of Genres'].max()}'\n
    
```

```
The Lowest Number of Genres Ever Any Movie Got is
'{df_genres_least_movies['Title'][0]} : {df_genres_least_movies['Number of
Genres'].min()}'\n
''')
```

```
# In[31]:
```

```
netflix_genres_most_movies =
df_genres_most_movies.loc[df_genres_most_movies['Netflix']==1].reset_index()
netflix_genres_most_movies = netflix_genres_most_movies.drop(['index'], axis = 1)

netflix_genres_least_movies =
df_genres_least_movies.loc[df_genres_least_movies['Netflix']==1].reset_index()
netflix_genres_least_movies = netflix_genres_least_movies.drop(['index'], axis = 1)

netflix_genres_most_movies.head(5)
```

```
# In[32]:
```

```
fig = px.bar(y = netflix_genres_most_movies['Title'][:15],
              x = netflix_genres_most_movies['Number of Genres'][:15],
              color = netflix_genres_most_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Highest Number of Genres : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:
```

```
fig = px.bar(y = netflix_genres_least_movies['Title'][:15],
              x = netflix_genres_least_movies['Number of Genres'][:15],
              color = netflix_genres_least_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Lowest Number of Genres : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[34]:
```

```
hulu_genres_most_movies =
df_genres_most_movies.loc[df_genres_most_movies['Hulu']==1].reset_index()
hulu_genres_most_movies = hulu_genres_most_movies.drop(['index'], axis = 1)

hulu_genres_least_movies =
df_genres_least_movies.loc[df_genres_least_movies['Hulu']==1].reset_index()
hulu_genres_least_movies = hulu_genres_least_movies.drop(['index'], axis = 1)

hulu_genres_most_movies.head(5)
```

```
# In[35]:
```

```

fig = px.bar(y = hulu_genres_most_movies['Title'][:15],
              x = hulu_genres_most_movies['Number of Genres'][:15],
              color = hulu_genres_most_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Highest Number of Genres : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[36]:

```

fig = px.bar(y = hulu_genres_least_movies['Title'][:15],
              x = hulu_genres_least_movies['Number of Genres'][:15],
              color = hulu_genres_least_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Lowest Number of Genres : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[37]:

```

prime_video_genres_most_movies = df_genres_most_movies.loc[df_genres_most_movies['Prime
Video']==1].reset_index()
prime_video_genres_most_movies = prime_video_genres_most_movies.drop(['index'], axis = 1)

prime_video_genres_least_movies = df_genres_least_movies.loc[df_genres_least_movies['Prime
Video']==1].reset_index()
prime_video_genres_least_movies = prime_video_genres_least_movies.drop(['index'], axis = 1)

prime_video_genres_most_movies.head(5)

```

In[38]:

```

fig = px.bar(y = prime_video_genres_most_movies['Title'][:15],
              x = prime_video_genres_most_movies['Number of Genres'][:15],
              color = prime_video_genres_most_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Highest Number of Genres : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[39]:

```

fig = px.bar(y = prime_video_genres_least_movies['Title'][:15],
              x = prime_video_genres_least_movies['Number of Genres'][:15],
              color = prime_video_genres_least_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Lowest Number of Genres : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[40]:
```

```
disney_genres_most_movies =
df_genres_most_movies.loc[df_genres_most_movies['Disney+']==1].reset_index()
disney_genres_most_movies = disney_genres_most_movies.drop(['index'], axis = 1)

disney_genres_least_movies =
df_genres_least_movies.loc[df_genres_least_movies['Disney+']==1].reset_index()
disney_genres_least_movies = disney_genres_least_movies.drop(['index'], axis = 1)

disney_genres_most_movies.head(5)
```

```
# In[41]:
```

```
fig = px.bar(y = disney_genres_most_movies['Title'][:15],
              x = disney_genres_most_movies['Number of Genres'][:15],
              color = disney_genres_most_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Highest Number of Genres : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[42]:
```

```
fig = px.bar(y = disney_genres_least_movies['Title'][:15],
              x = disney_genres_least_movies['Number of Genres'][:15],
              color = disney_genres_least_movies['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Genres'},
              title = 'Movies with Lowest Number of Genres : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[43]:
```

```
print(f'''
    The Movie with Highest Number of Genres Ever Got is
'{df_genres_most_movies['Title'][0]}' : '{df_genres_most_movies['Number of
Genres'].max()}'\n
    The Movie with Lowest Number of Genres Ever Got is
'{df_genres_least_movies['Title'][0]}' : '{df_genres_least_movies['Number of
Genres'].min()}'\n

    The Movie with Highest Number of Genres on 'Netflix' is
'{netflix_genres_most_movies['Title'][0]}' : '{netflix_genres_most_movies['Number of
Genres'].max()}'\n
    The Movie with Lowest Number of Genres on 'Netflix' is
'{netflix_genres_least_movies['Title'][0]}' : '{netflix_genres_least_movies['Number of
Genres'].min()}'\n

    The Movie with Highest Number of Genres on 'Hulu' is
'{hulu_genres_most_movies['Title'][0]}' : '{hulu_genres_most_movies['Number of
Genres'].max()}'\n
```

```

    The Movie with Lowest Number of Genres on 'Hulu' is
'{hulu_genres_least_movies['Title'][0]} : '{hulu_genres_least_movies['Number of
Genres']].min()}'\n

    The Movie with Highest Number of Genres on 'Prime Video' is
'{prime_video_genres_most_movies['Title'][0]} : '{prime_video_genres_most_movies['Number
of Genres']].max()}'\n
    The Movie with Lowest Number of Genres on 'Prime Video' is
'{prime_video_genres_least_movies['Title'][0]} : '{prime_video_genres_least_movies['Number
of Genres']].min()}'\n

    The Movie with Highest Number of Genres on 'Disney+' is
'{disney_genres_most_movies['Title'][0]} : '{disney_genres_most_movies['Number of
Genres']].max()}'\n
    The Movie with Lowest Number of Genres on 'Disney+' is
'{disney_genres_least_movies['Title'][0]} : '{disney_genres_least_movies['Number of
Genres']].min()}'\n
    ''')

```

In[44]:

```

print(f'''
    Accross All Platforms the Average Number of Genres is
'{round(df_movies_count_genres['Number of Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Netflix' is '{round(netflix_genres_movies['Number of
Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Hulu' is '{round(hulu_genres_movies['Number of
Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Prime Video' is
'{round(prime_video_genres_movies['Number of Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Disney+' is '{round(disney_genres_movies['Number of
Genres'].mean(), ndigits = 2)}'\n
    ''')

```

In[45]:

```

print(f'''
    Accross All Platforms Total Count of Genre is '{df_movies_count_genres['Number of
Genres']].max()}'\n
    Total Count of Genre on 'Netflix' is '{netflix_genres_movies['Number of
Genres']].max()}'\n
    Total Count of Genre on 'Hulu' is '{hulu_genres_movies['Number of Genres']].max()}'\n
    Total Count of Genre on 'Prime Video' is '{prime_video_genres_movies['Number of
Genres']].max()}'\n
    Total Count of Genre on 'Disney+' is '{disney_genres_movies['Number of
Genres']].max()}'\n
    ''')

```

In[46]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_count_genres['Number of Genres'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_movies_count_genres['Number of Genres'], ax = ax[1])
plt.show()

```

In[47]:

```
# Defining plot size and title
```

```

plt.figure(figsize = (20, 5))
plt.title('Number of Genres s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_genres_movies['Number of Genres'], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_genres_movies['Number of Genres'], color = 'red', legend = True, kde = True)
sns.histplot(hulu_genres_movies['Number of Genres'], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_genres_movies['Number of Genres'], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[48]:

```

df_lan = df_movies_genre['Genres'].str.split(',').apply(pd.Series).stack()
del df_movies_genre['Genres']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Genre'
df_movies_genre = df_movies_genre.join(df_lan)
df_movies_genre.drop_duplicates(inplace = True)

```

In[49]:

```
df_movies_genre.head(5)
```

In[50]:

```

genre_count = df_movies_genre.groupby('Genre')['Title'].count()
genre_movies = df_movies_genre.groupby('Genre')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
genre_data_movies = pd.concat([genre_count, genre_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
genre_data_movies = genre_data_movies.sort_values(by = 'Movies Count', ascending = False)

```

In[51]:

```

# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_genre_movies = genre_data_movies[genre_data_movies['Netflix'] != 0].sort_values(by
= 'Netflix', ascending = False).reset_index()
netflix_genre_movies = netflix_genre_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_genre_movies = genre_data_movies[genre_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_genre_movies = hulu_genre_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+',
'Movies Count'], axis = 1)

prime_video_genre_movies = genre_data_movies[genre_data_movies['Prime Video'] !=
0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_genre_movies = prime_video_genre_movies.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'Movies Count'], axis = 1)

```

```

disney_genre_movies = genre_data_movies[genre_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_genre_movies = disney_genre_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

# In[52]:


# Genre with Movies Counts - All Platforms Combined
genre_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[53]:


fig = px.bar(x = genre_data_movies['Genre'][:50],
              y = genre_data_movies['Movies Count'][:50],
              color = genre_data_movies['Movies Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Genre', 'y' : 'Movies Count'},
              title = 'Major Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[54]:


df_genre_high_movies = genre_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_genre_high_movies = df_genre_high_movies.drop(['index'], axis = 1)
# filter = (genre_data_movies['Movies Count'] == (genre_data_movies['Movies Count'].max()))
# df_genre_high_movies = genre_data_movies[filter]

# highestRatedMovies = genre_data_movies.loc[genre_data_movies['Movies Count'].idxmax()]

print('\nGenre with Highest Ever Movies Count are : All Platforms Combined\n')
df_genre_high_movies.head(5)

# In[55]:


fig = px.bar(y = df_genre_high_movies['Genre'][:15],
              x = df_genre_high_movies['Movies Count'][:15],
              color = df_genre_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Highest Movies : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[56]:


df_genre_low_movies = genre_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_genre_low_movies = df_genre_low_movies.drop(['index'], axis = 1)
# filter = (genre_data_movies['Movies Count'] == (genre_data_movies['Movies Count'].min()))
# df_genre_low_movies = genre_data_movies[filter]

print('\nGenre with Lowest Ever Movies Count are : All Platforms Combined\n')

```

```

df_genre_low_movies.head(5)

# In[57]:


fig = px.bar(y = df_genre_low_movies['Genre'][:15],
              x = df_genre_low_movies['Movies Count'][:15],
              color = df_genre_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[58]:


print(f'''Total '{genre_data_movies['Genre'].unique().shape[0]}' unique Genre Count s were Given, They were Like this,\n
{genre_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Genre'].unique()[:5]}\n

The Highest Ever Movies Count Ever Any Movie Got is
'{df_genre_high_movies['Genre'][0]}' : '{df_genre_high_movies['Movies Count'].max()}'\n

The Lowest Ever Movies Count Ever Any Movie Got is
'{df_genre_low_movies['Genre'][0]}' : '{df_genre_low_movies['Movies Count'].min()}'\n
''')

# In[59]:


fig = px.pie(genre_data_movies[:10], names = 'Genre', values = 'Movies Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies Count based on Genre')
fig.show()

# In[60]:


# netflix_genre_movies = genre_data_movies[genre_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_genre_movies = netflix_genre_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_genre_high_movies = df_genre_high_movies.sort_values(by = 'Netflix', ascending =
False).reset_index()
netflix_genre_high_movies = netflix_genre_high_movies.drop(['index'], axis = 1)

netflix_genre_low_movies = df_genre_low_movies.sort_values(by = 'Netflix', ascending =
True).reset_index()
netflix_genre_low_movies = netflix_genre_low_movies.drop(['index'], axis = 1)

netflix_genre_high_movies.head(5)

# In[61]:

```

```

fig = px.bar(x = netflix_genre_high_movies['Genre'][:15],
              y = netflix_genre_high_movies['Netflix'][:15],
              color = netflix_genre_high_movies['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Highest Movies : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[62]:


# hulu_genre_movies = genre_data_movies[genre_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_genre_movies = hulu_genre_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_genre_high_movies = df_genre_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_genre_high_movies = hulu_genre_high_movies.drop(['index'], axis = 1)

hulu_genre_low_movies = df_genre_high_movies.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_genre_low_movies = hulu_genre_low_movies.drop(['index'], axis = 1)

hulu_genre_high_movies.head(5)

# In[63]:


fig = px.bar(x = hulu_genre_high_movies['Genre'][:15],
              y = hulu_genre_high_movies['Hulu'][:15],
              color = hulu_genre_high_movies['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Highest Movies : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[64]:


# prime_video_genre_movies = genre_data_movies[genre_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_genre_movies = prime_video_genre_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_genre_high_movies = df_genre_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_genre_high_movies = prime_video_genre_high_movies.drop(['index'], axis = 1)

prime_video_genre_low_movies = df_genre_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_genre_low_movies = prime_video_genre_low_movies.drop(['index'], axis = 1)

prime_video_genre_high_movies.head(5)

# In[65]:

```

```

fig = px.bar(x = prime_video_genre_high_movies['Genre'][:15],
              y = prime_video_genre_high_movies['Prime Video'][:15],
              color = prime_video_genre_high_movies['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Highest Movies : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[66]:


# disney_genre_movies = genre_data_movies[genre_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_genre_movies = disney_genre_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_genre_high_movies = df_genre_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_genre_high_movies = disney_genre_high_movies.drop(['index'], axis = 1)

disney_genre_low_movies = df_genre_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_genre_low_movies = disney_genre_low_movies.drop(['index'], axis = 1)

disney_genre_high_movies.head(5)

# In[67]:


fig = px.bar(x = disney_genre_high_movies['Genre'][:15],
              y = disney_genre_high_movies['Disney+'][:15],
              color = disney_genre_high_movies['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'Movies Count'},
              title = 'Genre with Highest Movies : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[68]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(genre_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(genre_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Genre Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_genre_movies['Disney+'][:50], color = 'darkblue', legend = True, kde = True)

```

```

sns.histplot(prime_video_genre_movies['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_genre_movies['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_genre_movies['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''
    The Genre with Highest Movies Count Ever Got is '{df_genre_high_movies['Genre'][0]}'
: '{df_genre_high_movies['Movies Count'].max()}'\n
    The Genre with Lowest Movies Count Ever Got is '{df_genre_low_movies['Genre'][0]}'
: '{df_genre_low_movies['Movies Count'].min()}'\n

    The Genre with Highest Movies Count on 'Netflix' is
'{netflix_genre_high_movies['Genre'][0]}'
: '{netflix_genre_high_movies['Netflix'].max()}'\n
    The Genre with Lowest Movies Count on 'Netflix' is
'{netflix_genre_low_movies['Genre'][0]}'
: '{netflix_genre_low_movies['Netflix'].min()}'\n

    The Genre with Highest Movies Count on 'Hulu' is
'{hulu_genre_high_movies['Genre'][0]}'
: '{hulu_genre_high_movies['Hulu'].max()}'\n
    The Genre with Lowest Movies Count on 'Hulu' is
'{hulu_genre_low_movies['Genre'][0]}'
: '{hulu_genre_low_movies['Hulu'].min()}'\n

    The Genre with Highest Movies Count on 'Prime Video' is
'{prime_video_genre_high_movies['Genre'][0]}'
: '{prime_video_genre_high_movies['Prime Video'].max()}'\n
    The Genre with Lowest Movies Count on 'Prime Video' is
'{prime_video_genre_low_movies['Genre'][0]}'
: '{prime_video_genre_low_movies['Prime Video'].min()}'\n

    The Genre with Highest Movies Count on 'Disney+' is
'{disney_genre_high_movies['Genre'][0]}'
: '{disney_genre_high_movies['Disney+'].max()}'\n
    The Genre with Lowest Movies Count on 'Disney+' is
'{disney_genre_low_movies['Genre'][0]}'
: '{disney_genre_low_movies['Disney+'].min()}'\n
''')


# In[71]:


# Distribution of movies genre in each platform
plt.figure(figsize = (20, 5))
plt.title('Genre with Movies Count for All Platforms')
sns.violinplot(x = genre_data_movies['Movies Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

# In[72]:


# Distribution of Genre Movies Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_genre_movies['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_genre_movies['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))

```

```
sns.violinplot(x = prime_video_genre_movies['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_genre_movies['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()
```

In[73]:

```
print(f'''
    Across All Platforms the Average Movies Count of Genre is
'{round(genre_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Genre on 'Netflix' is
'{round(netflix_genre_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Genre on 'Hulu' is
'{round(hulu_genre_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Genre on 'Prime Video' is
'{round(prime_video_genre_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Genre on 'Disney+' is
'{round(disney_genre_movies['Disney+'].mean(), ndigits = 2)}'
''')
```

In[74]:

```
print(f'''
    Across All Platforms Total Count of Genre is
'{genre_data_movies['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Netflix' is
'{netflix_genre_movies['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Hulu' is '{hulu_genre_movies['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Prime Video' is
'{prime_video_genre_movies['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Disney+' is
'{disney_genre_movies['Genre'].unique().shape[0]}'
''')
```

In[75]:

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = genre_data_movies['Genre'][:10], y = genre_data_movies['Netflix'][:10],
color = 'red')
sns.lineplot(x = genre_data_movies['Genre'][:10], y = genre_data_movies['Hulu'][:10], color
= 'lightgreen')
sns.lineplot(x = genre_data_movies['Genre'][:10], y = genre_data_movies['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = genre_data_movies['Genre'][:10], y = genre_data_movies['Disney+'][:10],
color = 'darkblue')
plt.xlabel('Genre', fontsize = 20)
plt.ylabel('Movies Count', fontsize = 20)
plt.show()
```

In[76]:

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_g_ax1 = sns.lineplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_g_ax2 = sns.lineplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
```

```

p_g_ax3 = sns.lineplot(y = genre_data_movies['Genre'][:10], x = genre_data_movies['Prime
Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_g_ax4 = sns.lineplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()

```

In[77]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = netflix_genre_movies['Genre'][:10], x =
netflix_genre_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_g_ax2 = sns.barplot(y = hulu_genre_movies['Genre'][:10], x =
hulu_genre_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = prime_video_genre_movies['Genre'][:10], x =
prime_video_genre_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_g_ax4 = sns.barplot(y = disney_genre_movies['Genre'][:10], x =
disney_genre_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()

```

In[78]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Genre Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_genre_movies['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_genre_movies['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_genre_movies['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_genre_movies['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

```

In[79]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])

```

```

h_g_ax2 = sns.barplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = genre_data_movies['Genre'][:10], x = genre_data_movies['Prime
Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_g_ax4 = sns.barplot(y = genre_data_movies['Genre'][:10], x =
genre_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()

# In[80]:


df_movies_mixed_genres.drop(df_movies_mixed_genres.loc[df_movies_mixed_genres['Genres'] ==
"NA"].index, inplace = True)
# df_movies_mixed_genres = df_movies_mixed_genres[df_movies_mixed_genres.Genre != "NA"]
df_movies_mixed_genres.drop(df_movies_mixed_genres.loc[df_movies_mixed_genres['Number of
Genres'] == 1].index, inplace = True)

# In[81]:


df_movies_mixed_genres.head(5)

# In[82]:


mixed_genres_count = df_movies_mixed_genres.groupby('Genres')['Title'].count()
mixed_genres_movies = df_movies_mixed_genres.groupby('Genres')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
mixed_genres_data_movies = pd.concat([mixed_genres_count, mixed_genres_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count', 'Genres' : 'Mixed Genre'})
mixed_genres_data_movies = mixed_genres_data_movies.sort_values(by = 'Movies Count',
ascending = False)

# In[83]:


mixed_genres_data_movies.head(5)

# In[84]:


# Mixed Genre with Movies Counts - All Platforms Combined
mixed_genres_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[85]:


df_mixed_genres_high_movies = mixed_genres_data_movies.sort_values(by = 'Movies Count',
ascending = False).reset_index()
df_mixed_genres_high_movies = df_mixed_genres_high_movies.drop(['index'], axis = 1)
# filter = (mixed_genres_data_movies['Movies Count'] == (mixed_genres_data_movies['Movies
Count'].max()))

```

```

# df_mixed_genres_high_movies = mixed_genres_data_movies[filter]

# highestRatedMovies = mixed_genres_data_movies.loc[mixed_genres_data_movies['Movies Count'].idxmax()]

print('\nMixed Genre with Highest Ever Movies Count are : All Platforms Combined\n')
df_mixed_genres_high_movies.head(5)

# In[86]:


fig = px.bar(y = df_mixed_genres_high_movies['Mixed Genre'][:15],
              x = df_mixed_genres_high_movies['Movies Count'][:15],
              color = df_mixed_genres_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Genre'},
              title = 'Movies with Highest Number of Mixed Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[87]:


df_mixed_genres_low_movies = mixed_genres_data_movies.sort_values(by = 'Movies Count',
                                                                    ascending = True).reset_index()
df_mixed_genres_low_movies = df_mixed_genres_low_movies.drop(['index'], axis = 1)
# filter = (mixed_genres_data_movies['Movies Count'] == (mixed_genres_data_movies['Movies Count'].min()))
# df_mixed_genres_low_movies = mixed_genres_data_movies[filter]

print('\nMixed Genre with Lowest Ever Movies Count are : All Platforms Combined\n')
df_mixed_genres_low_movies.head(5)

# In[88]:


fig = px.bar(y = df_mixed_genres_low_movies['Mixed Genre'][:15],
              x = df_mixed_genres_low_movies['Movies Count'][:15],
              color = df_mixed_genres_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Genre'},
              title = 'Movies with Lowest Number of Mixed Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[89]:


print(f'''
      Total '{df_movies_genres['Genres'].count()}' Titles are available on All Platforms,
      out of which\n
      You Can Choose to see Movies from Total '{mixed_genres_data_movies['Mixed
      Genre'].unique().shape[0]}' Mixed Genre, They were Like this, \n

      {mixed_genres_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Mixed
      Genre']].head(5).unique()} etc. \n

      The Mixed Genre with Highest Movies Count have '{mixed_genres_data_movies['Movies
      Count'].max()}' Movies Available is '{df_mixed_genres_high_movies['Mixed Genre'][0]}', &\n
      ''')

```

```
The Mixed Genre with Lowest Movies Count have '{mixed_genres_data_movies['Movies Count'].min()}' Movies Available is '{df_mixed_genres_low_movies['Mixed Genre'][0]}'  
''')
```

```
# In[90]:
```

```
fig = px.pie(mixed_genres_data_movies[:10], names = 'Mixed Genre', values = 'Movies Count',  
color_discrete_sequence = px.colors.sequential.Teal)  
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies  
Count based on Mixed Genre')  
fig.show()
```

```
# In[91]:
```

```
# netflix_mixed_genres_movies =  
mixed_genres_data_movies[mixed_genres_data_movies['Netflix'] != 0].sort_values(by =  
'Netflix', ascending = False).reset_index()  
# netflix_mixed_genres_movies = netflix_mixed_genres_movies.drop(['index', 'Hulu', 'Prime  
Video', 'Disney+', 'Movies Count'], axis = 1)  
  
netflix_mixed_genres_high_movies = df_mixed_genres_high_movies.sort_values(by = 'Netflix',  
ascending = False).reset_index()  
netflix_mixed_genres_high_movies = netflix_mixed_genres_high_movies.drop(['index'], axis =  
1)  
  
netflix_mixed_genres_low_movies = df_mixed_genres_high_movies.sort_values(by = 'Netflix',  
ascending = True).reset_index()  
netflix_mixed_genres_low_movies = netflix_mixed_genres_low_movies.drop(['index'], axis = 1)  
  
netflix_mixed_genres_high_movies.head(5)
```

```
# In[92]:
```

```
# hulu_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Hulu'] !=  
0].sort_values(by = 'Hulu', ascending = False).reset_index()  
# hulu_mixed_genres_movies = hulu_mixed_genres_movies.drop(['index', 'Netflix', 'Prime  
Video', 'Disney+', 'Movies Count'], axis = 1)  
  
hulu_mixed_genres_high_movies = df_mixed_genres_high_movies.sort_values(by = 'Hulu',  
ascending = False).reset_index()  
hulu_mixed_genres_high_movies = hulu_mixed_genres_high_movies.drop(['index'], axis = 1)  
  
hulu_mixed_genres_low_movies = df_mixed_genres_high_movies.sort_values(by = 'Hulu',  
ascending = True).reset_index()  
hulu_mixed_genres_low_movies = hulu_mixed_genres_low_movies.drop(['index'], axis = 1)  
  
hulu_mixed_genres_high_movies.head(5)
```

```
# In[93]:
```

```
# prime_video_mixed_genres_movies =  
mixed_genres_data_movies[mixed_genres_data_movies['Prime Video'] != 0].sort_values(by =  
'Prime Video', ascending = False).reset_index()  
# prime_video_mixed_genres_movies = prime_video_mixed_genres_movies.drop(['index',  
'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)  
  
prime_video_mixed_genres_high_movies = df_mixed_genres_high_movies.sort_values(by = 'Prime  
Video', ascending = False).reset_index()
```

```

prime_video_mixed_genres_high_movies = prime_video_mixed_genres_high_movies.drop(['index'],
axis = 1)

prime_video_mixed_genres_low_movies = df_mixed_genres_high_movies.sort_values(by = 'Prime
Video', ascending = True).reset_index()
prime_video_mixed_genres_low_movies = prime_video_mixed_genres_low_movies.drop(['index'],
axis = 1)

prime_video_mixed_genres_high_movies.head(5)

# In[94]:


# disney_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Disney+']
!= 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_genres_movies = disney_mixed_genres_movies.drop(['index', 'Netflix', 'Hulu',
'Prime Video', 'Movies Count'], axis = 1)

disney_mixed_genres_high_movies = df_mixed_genres_high_movies.sort_values(by = 'Disney+', 
ascending = False).reset_index()
disney_mixed_genres_high_movies = disney_mixed_genres_high_movies.drop(['index'], axis = 1)

disney_mixed_genres_low_movies = df_mixed_genres_high_movies.sort_values(by = 'Disney+', 
ascending = True).reset_index()
disney_mixed_genres_low_movies = disney_mixed_genres_low_movies.drop(['index'], axis = 1)

disney_mixed_genres_high_movies.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_genres_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_genres_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Netflix']
!= 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_genres_movies = netflix_mixed_genres_movies.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Hulu'] != 
0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_genres_movies = hulu_mixed_genres_movies.drop(['index', 'Netflix', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Prime
Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_genres_movies = prime_video_mixed_genres_movies.drop(['index', 'Netflix',
'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_mixed_genres_movies = mixed_genres_data_movies[mixed_genres_data_movies['Disney+']
!= 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_genres_movies = disney_mixed_genres_movies.drop(['index', 'Netflix', 'Hulu',
'Prime Video', 'Movies Count'], axis = 1)

# In[97]:

```

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Genre Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_genres_movies['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_genres_movies['Netflix'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_mixed_genres_movies['Hulu'][:100], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_mixed_genres_movies['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Genre with Highest Movies Count Ever Got is
'{df_mixed_genres_high_movies['Mixed Genre'][0]}' : '{df_mixed_genres_high_movies['Movies
Count'].max()}'\n
    The Mixed Genre with Lowest Movies Count Ever Got is
'{df_mixed_genres_low_movies['Mixed Genre'][0]}' : '{df_mixed_genres_low_movies['Movies
Count'].min()}'\n

    The Mixed Genre with Highest Movies Count on 'Netflix' is
'{netflix_mixed_genres_high_movies['Mixed Genre'][0]}' :
'{netflix_mixed_genres_high_movies['Netflix'].max()}'\n
    The Mixed Genre with Lowest Movies Count on 'Netflix' is
'{netflix_mixed_genres_low_movies['Mixed Genre'][0]}' :
'{netflix_mixed_genres_low_movies['Netflix'].min()}'\n

    The Mixed Genre with Highest Movies Count on 'Hulu' is
'{hulu_mixed_genres_high_movies['Mixed Genre'][0]}' :
'{hulu_mixed_genres_high_movies['Hulu'].max()}'\n
    The Mixed Genre with Lowest Movies Count on 'Hulu' is
'{hulu_mixed_genres_low_movies['Mixed Genre'][0]}' :
'{hulu_mixed_genres_low_movies['Hulu'].min()}'\n

    The Mixed Genre with Highest Movies Count on 'Prime Video' is
'{prime_video_mixed_genres_high_movies['Mixed Genre'][0]}' :
'{prime_video_mixed_genres_high_movies['Prime Video'].max()}'\n
    The Mixed Genre with Lowest Movies Count on 'Prime Video' is
'{prime_video_mixed_genres_low_movies['Mixed Genre'][0]}' :
'{prime_video_mixed_genres_low_movies['Prime Video'].min()}'\n

    The Mixed Genre with Highest Movies Count on 'Disney+' is
'{disney_mixed_genres_high_movies['Mixed Genre'][0]}' :
'{disney_mixed_genres_high_movies['Disney+'].max()}'\n
    The Mixed Genre with Lowest Movies Count on 'Disney+' is
'{disney_mixed_genres_low_movies['Mixed Genre'][0]}' :
'{disney_mixed_genres_low_movies['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Across All Platforms the Average Movies Count of Mixed Genre is
'{round(mixed_genres_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Genre on 'Netflix' is
'{round(netflix_mixed_genres_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Genre on 'Hulu' is
'{round(hulu_mixed_genres_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Genre on 'Prime Video' is
'{round(prime_video_mixed_genres_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Genre on 'Disney+' is
'{round(disney_mixed_genres_movies['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```

print(f'''
    Across All Platforms Total Count of Mixed Genre is '{mixed_genres_data_movies['Mixed
Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Netflix' is '{netflix_mixed_genres_movies['Mixed
Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Hulu' is '{hulu_mixed_genres_movies['Mixed
Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Prime Video' is
'{prime_video_mixed_genres_movies['Mixed Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Disney+' is '{disney_mixed_genres_movies['Mixed
Genre'].unique().shape[0]}'\n
    ''')

```

In[101]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_genres_data_movies['Mixed Genre'][:5], y =
mixed_genres_data_movies['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_genres_data_movies['Mixed Genre'][:5], y =
mixed_genres_data_movies['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_genres_data_movies['Mixed Genre'][:5], y =
mixed_genres_data_movies['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_genres_data_movies['Mixed Genre'][:5], y =
mixed_genres_data_movies['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Genre', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

```

In[102]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_g_ax2 = sns.barplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_g_ax4 = sns.barplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])

```

```

p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()

# In[103]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_mg_ax1 = sns.lineplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_mg_ax2 = sns.lineplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_mg_ax3 = sns.lineplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_mg_ax4 = sns.lineplot(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mg_ax1.title.set_text(labels[0])
h_mg_ax2.title.set_text(labels[1])
p_mg_ax3.title.set_text(labels[2])
d_mg_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Genre Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_genres_movies['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_genres_movies['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_genres_movies['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_genres_movies['Disney+'][:50], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_mg_ax1 = sns.barplot(y = netflix_mixed_genres_movies['Mixed Genre'][:10], x =
netflix_mixed_genres_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mg_ax2 = sns.barplot(y = hulu_mixed_genres_movies['Mixed Genre'][:10], x =
hulu_mixed_genres_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mg_ax3 = sns.barplot(y = prime_video_mixed_genres_movies['Mixed Genre'][:10], x =
prime_video_mixed_genres_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_mg_ax4 = sns.barplot(y = disney_mixed_genres_movies['Mixed Genre'][:10], x =
disney_mixed_genres_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mg_ax1.title.set_text(labels[0])

```

```

h_mg_ax2.title.set_text(labels[1])
p_mg_ax3.title.set_text(labels[2])
d_mg_ax4.title.set_text(labels[3])

plt.show()

# In[106]:


fig = go.Figure(go.Funnel(y = mixed_genres_data_movies['Mixed Genre'][:10], x =
mixed_genres_data_movies['Movies Count'][:10]))
fig.show()

```

ottmovies_imdb.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots

```

```

from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_movies = pd.read_csv(path + 'ottmovies.csv')

df_movies.head()

# In[6]:


# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_movies)

# In[9]:

```

```

# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)

```

```

# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix', 'Prime Video', 'Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:
# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

```

```
# In[17]:
df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]
```

```
# In[18]:
df_movies_imdb = df_movies.copy()
```

```
# In[19]:
df_movies_imdb.drop(df_movies_imdb.loc[df_movies_imdb['IMDb'] == "NA"].index, inplace = True)
# df_movies_imdb = df_movies_imdb[df_movies_imdb.IMDb != "NA"]
df_movies_imdb['IMDb'] = df_movies_imdb['IMDb'].astype(int)
```

```
# In[20]:
# Creating distinct dataframes only with the movies present on individual streaming platforms
netflix_imdb_movies = df_movies_imdb.loc[df_movies_imdb['Netflix'] == 1]
hulu_imdb_movies = df_movies_imdb.loc[df_movies_imdb['Hulu'] == 1]
prime_video_imdb_movies = df_movies_imdb.loc[df_movies_imdb['Prime Video'] == 1]
disney_imdb_movies = df_movies_imdb.loc[df_movies_imdb['Disney+'] == 1]
```

```
# In[21]:
df_movies_imdb_group = df_movies_imdb.copy()
```

```
# In[22]:
plt.figure(figsize = (10, 10))
corr = df_movies_imdb.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[23]:
```

```

df_imdb_high_movies = df_movies_imdb.sort_values(by = 'IMDb', ascending =
False).reset_index()
df_imdb_high_movies = df_imdb_high_movies.drop(['index'], axis = 1)
# filter = (df_movies_imdb['IMDb'] == (df_movies_imdb['IMDb'].max()))
# df_imdb_high_movies = df_movies_imdb[filter]

# highestRatedMovies = df_movies_imdb.loc[df_movies_imdb['IMDb'].idxmax()]

print('\nMovies with Highest Ever IMDb are : \n')
df_imdb_high_movies.head(5)

# In[24]:


fig = px.bar(y = df_imdb_high_movies['Title'][:15],
              x = df_imdb_high_movies['IMDb'][:15],
              color = df_imdb_high_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Highest IMDb : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[25]:


df_imdb_low_movies = df_movies_imdb.sort_values(by = 'IMDb', ascending =
True).reset_index()
df_imdb_low_movies = df_imdb_low_movies.drop(['index'], axis = 1)
# filter = (df_movies_imdb['IMDb'] == (df_movies_imdb['IMDb'].min()))
# df_imdb_low_movies = df_movies_imdb[filter]

print('\nMovies with Lowest Ever IMDb are : \n')
df_imdb_low_movies.head(5)

# In[26]:


fig = px.bar(y = df_imdb_low_movies['Title'][:15],
              x = df_imdb_low_movies['IMDb'][:15],
              color = df_imdb_low_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Lowest IMDb : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[27]:


print(f'''
      Total '{df_movies_imdb['IMDb'].unique().shape[0]}' unique IMDb s were Given, They
      were Like this,\n

{df_movies_imdb.sort_values(by = 'IMDb', ascending = False)['IMDb'].unique()}\n

      The Highest Ever IMDb Ever Any Movie Got is '{df_imdb_high_movies['Title'][0]}' :
      '{df_imdb_high_movies['IMDb'].max()}'\n
    
```

```
The Lowest Ever IMDb Ever Any Movie Got is '{df_imdb_low_movies['Title'][0]}' :  
'{df_imdb_low_movies['IMDb'].min()}'\n''')
```

```
# In[28]:
```

```
netflix_imdb_high_movies =  
df_imdb_high_movies.loc[df_imdb_high_movies['Netflix']==1].reset_index()  
netflix_imdb_high_movies = netflix_imdb_high_movies.drop(['index'], axis = 1)  
  
netflix_imdb_low_movies =  
df_imdb_low_movies.loc[df_imdb_low_movies['Netflix']==1].reset_index()  
netflix_imdb_low_movies = netflix_imdb_low_movies.drop(['index'], axis = 1)  
  
netflix_imdb_high_movies.head(5)
```

```
# In[29]:
```

```
fig = px.bar(y = netflix_imdb_high_movies['Title'][:15],  
             x = netflix_imdb_high_movies['IMDb'][:15],  
             color = netflix_imdb_high_movies['IMDb'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},  
             title = 'Movies with Highest IMDb : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[30]:
```

```
fig = px.bar(y = netflix_imdb_low_movies['Title'][:15],  
             x = netflix_imdb_low_movies['IMDb'][:15],  
             color = netflix_imdb_low_movies['IMDb'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},  
             title = 'Movies with Lowest IMDb : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[31]:
```

```
hulu_imdb_high_movies =  
df_imdb_high_movies.loc[df_imdb_high_movies['Hulu']==1].reset_index()  
hulu_imdb_high_movies = hulu_imdb_high_movies.drop(['index'], axis = 1)  
  
hulu_imdb_low_movies = df_imdb_low_movies.loc[df_imdb_low_movies['Hulu']==1].reset_index()  
hulu_imdb_low_movies = hulu_imdb_low_movies.drop(['index'], axis = 1)  
  
hulu_imdb_high_movies.head(5)
```

```
# In[32]:
```

```
fig = px.bar(y = hulu_imdb_high_movies['Title'][:15],  
             x = hulu_imdb_high_movies['IMDb'][:15],  
             color = hulu_imdb_high_movies['IMDb'][:15],
```

```
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
title = 'Movies with Highest IMDb : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[33]:

```
fig = px.bar(y = hulu_imdb_low_movies['Title'][:15],
              x = hulu_imdb_low_movies['IMDb'][:15],
              color = hulu_imdb_low_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Lowest IMDb : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[34]:

```
prime_video_imdb_high_movies = df_imdb_high_movies.loc[df_imdb_high_movies['Prime
Video']==1].reset_index()
prime_video_imdb_high_movies = prime_video_imdb_high_movies.drop(['index'], axis = 1)

prime_video_imdb_low_movies = df_imdb_low_movies.loc[df_imdb_low_movies['Prime
Video']==1].reset_index()
prime_video_imdb_low_movies = prime_video_imdb_low_movies.drop(['index'], axis = 1)

prime_video_imdb_high_movies.head(5)
```

In[35]:

```
fig = px.bar(y = prime_video_imdb_high_movies['Title'][:15],
              x = prime_video_imdb_high_movies['IMDb'][:15],
              color = prime_video_imdb_high_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Highest IMDb : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[36]:

```
fig = px.bar(y = prime_video_imdb_low_movies['Title'][:15],
              x = prime_video_imdb_low_movies['IMDb'][:15],
              color = prime_video_imdb_low_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Lowest IMDb : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[37]:

```

disney_imdb_high_movies =
df_imdb_high_movies.loc[df_imdb_high_movies['Disney+']==1].reset_index()
disney_imdb_high_movies = disney_imdb_high_movies.drop(['index'], axis = 1)

disney_imdb_low_movies =
df_imdb_low_movies.loc[df_imdb_low_movies['Disney+']==1].reset_index()
disney_imdb_low_movies = disney_imdb_low_movies.drop(['index'], axis = 1)

disney_imdb_high_movies.head(5)

# In[38]:


fig = px.bar(y = disney_imdb_high_movies['Title'][:15],
              x = disney_imdb_high_movies['IMDb'][:15],
              color = disney_imdb_high_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Highest IMDb : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[39]:


fig = px.bar(y = disney_imdb_low_movies['Title'][:15],
              x = disney_imdb_low_movies['IMDb'][:15],
              color = disney_imdb_low_movies['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'IMDb : Rating'},
              title = 'Movies with Lowest IMDb : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


print(f'''
    The Movie with Highest IMDb Ever Got is '{df_imdb_high_movies['Title'][0]}' :
'{df_imdb_high_movies['IMDb'].max()}'\n
    The Movie with Lowest IMDb Ever Got is '{df_imdb_low_movies['Title'][0]}' :
'{df_imdb_low_movies['IMDb'].min()}'\n

    The Movie with Highest IMDb on 'Netflix' is '{netflix_imdb_high_movies['Title'][0]}':
'{netflix_imdb_high_movies['IMDb'].max()}'\n
    The Movie with Lowest IMDb on 'Netflix' is '{netflix_imdb_low_movies['Title'][0]}':
'{netflix_imdb_low_movies['IMDb'].min()}'\n

    The Movie with Highest IMDb on 'Hulu' is '{hulu_imdb_high_movies['Title'][0]}':
'{hulu_imdb_high_movies['IMDb'].max()}'\n
    The Movie with Lowest IMDb on 'Hulu' is '{hulu_imdb_low_movies['Title'][0]}':
'{hulu_imdb_low_movies['IMDb'].min()}'\n

    The Movie with Highest IMDb on 'Prime Video' is
'{prime_video_imdb_high_movies['Title'][0]}':
'{prime_video_imdb_high_movies['IMDb'].max()}'\n
    The Movie with Lowest IMDb on 'Prime Video' is
'{prime_video_imdb_low_movies['Title'][0]}':
'{prime_video_imdb_low_movies['IMDb'].min()}'\n

```

```
The Movie with Highest IMDb on 'Disney+' is '{disney_imdb_high_movies['Title'][0]}'\n:{'disney_imdb_high_movies['IMDb'].max()}'\n    The Movie with Lowest IMDb on 'Disney+' is '{disney_imdb_low_movies['Title'][0]}' :'\n{'disney_imdb_low_movies['IMDb'].min()}'\n    ''')
```

```
# In[41]:
```

```
print(f'''\n    Accross All Platforms the Average IMDb is '{round(df_movies_imdb['IMDb'].mean(),\nndigits = 2)}'\n    The Average IMDb on 'Netflix' is '{round(netflix_imdb_movies['IMDb'].mean(), ndigits\n= 2)}'\n    The Average IMDb on 'Hulu' is '{round(hulu_imdb_movies['IMDb'].mean(), ndigits =\n2)}'\n    The Average IMDb on 'Prime Video' is '{round(prime_video_imdb_movies['IMDb'].mean(),\nndigits = 2)}'\n    The Average IMDb on 'Disney+' is '{round(disney_imdb_movies['IMDb'].mean(), ndigits\n= 2)}'\n    ''')
```

```
# In[42]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))\nsns.distplot(df_movies_imdb['IMDb'],bins = 20, kde = True, ax = ax[0])\nsns.boxplot(df_movies_imdb['IMDb'], ax = ax[1])\nplt.show()
```

```
# In[43]:
```

```
# Defining plot size and title\nplt.figure(figsize = (20, 5))\nplt.title('IMDb s Per Platform')\n\n# Plotting the information from each dataset into a histogram\nsns.histplot(prime_video_imdb_movies['IMDb'][:100], color = 'lightblue', legend = True, kde\n= True)\nsns.histplot(netflix_imdb_movies['IMDb'][:100], color = 'red', legend = True, kde = True)\nsns.histplot(hulu_imdb_movies['IMDb'][:100], color = 'lightgreen', legend = True, kde =\nTrue)\nsns.histplot(disney_imdb_movies['IMDb'][:100], color = 'darkblue', legend = True, kde =\nTrue)\n\n# Setting the legend\nplt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])\nplt.show()
```

```
# In[44]:
```

```
def round_val(data):\n    if str(data) != 'nan':\n        return round(data)
```

```
# In[45]:
```

```
df_movies_imdb_group['IMDb Group'] = df_movies_imdb['IMDb'].apply(round_val)
```

```

imdb_values = df_movies_imdb_group['IMDb Group'].value_counts().sort_index(ascending =
False).tolist()
imdb_index = df_movies_imdb_group['IMDb Group'].value_counts().sort_index(ascending =
False).index

# imdb_values, imdb_index

# In[46]:


imdb_group_count = df_movies_imdb_group.groupby('IMDb Group')['Title'].count()
imdb_group_movies = df_movies_imdb_group.groupby('IMDb Group')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
imdb_group_data_movies = pd.concat([imdb_group_count, imdb_group_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
imdb_group_data_movies = imdb_group_data_movies.sort_values(by = 'Movies Count', ascending =
False)

# In[47]:


# IMDb Group with Movies Counts - All Platforms Combined
imdb_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[48]:


imdb_group_data_movies.sort_values(by = 'IMDb Group', ascending = False)

# In[49]:


fig = px.bar(y = imdb_group_data_movies['Movies Count'],
              x = imdb_group_data_movies['IMDb Group'],
              color = imdb_group_data_movies['IMDb Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'IMDb : Rating'},
              title = 'Movies with Group IMDb : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]:


fig = px.pie(imdb_group_data_movies[:10],
             names = imdb_group_data_movies['IMDb Group'],
             values = imdb_group_data_movies['Movies Count'],
             color = imdb_group_data_movies['Movies Count'],
             color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on IMDb Group')
fig.show()

# In[51]:

```

```

df_imdb_group_high_movies = imdb_group_data_movies.sort_values(by = 'Movies Count',
ascending = False).reset_index()
df_imdb_group_high_movies = df_imdb_group_high_movies.drop(['index'], axis = 1)
# filter = (imdb_group_data_movies['Movies Count'] == (imdb_group_data_movies['Movies
Count'].max()))
# df_imdb_group_high_movies = imdb_group_data_movies[filter]

# highestRatedMovies = imdb_group_data_movies.loc[imdb_group_data_movies['Movies
Count'].idxmax()]

# print('\nIMDb with Highest Ever Movies Count are : All Platforms Combined\n')
df_imdb_group_high_movies.head(5)

```

In[52]:

```

df_imdb_group_low_movies = imdb_group_data_movies.sort_values(by = 'Movies Count',
ascending = True).reset_index()
df_imdb_group_low_movies = df_imdb_group_low_movies.drop(['index'], axis = 1)
# filter = (imdb_group_data_movies['Movies Count'] == (imdb_group_data_movies['Movies
Count'].min()))
# df_imdb_group_low_movies = imdb_group_data_movies[filter]

# print('\nIMDb with Lowest Ever Movies Count are : All Platforms Combined\n')
df_imdb_group_low_movies.head(5)

```

In[53]:

```

print(f'''
    Total '{df_movies_imdb['IMDb'].count()}' Titles are available on All Platforms, out
of which\n
    You Can Choose to see Movies from Total '{imdb_group_data_movies['IMDb
Group'].unique().shape[0]}' IMDb Group, They were Like this, \n

    {imdb_group_data_movies.sort_values(by = 'Movies Count', ascending = False)[['IMDb
Group']].unique()} etc. \n

    The IMDb Group with Highest Movies Count have '{imdb_group_data_movies['Movies
Count'].max()}' Movies Available is '{df_imdb_group_high_movies['IMDb Group'][0]}', &\n
    The IMDb Group with Lowest Movies Count have '{imdb_group_data_movies['Movies
Count'].min()}' Movies Available is '{df_imdb_group_low_movies['IMDb Group'][0]}'
    ''')

```

In[54]:

```

netflix_imdb_group_movies = imdb_group_data_movies[imdb_group_data_movies['Netflix'] !=
0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_imdb_group_movies = netflix_imdb_group_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

netflix_imdb_group_high_movies = df_imdb_group_high_movies.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_imdb_group_high_movies = netflix_imdb_group_high_movies.drop(['index'], axis = 1)

netflix_imdb_group_low_movies = df_imdb_group_low_movies.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_imdb_group_low_movies = netflix_imdb_group_low_movies.drop(['index'], axis = 1)

netflix_imdb_group_high_movies.head(5)

```

```
# In[55]:
```

```
hulu_imdb_group_movies = imdb_group_data_movies[imdb_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_imdb_group_movies = hulu_imdb_group_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_imdb_group_high_movies = df_imdb_group_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_imdb_group_high_movies = hulu_imdb_group_high_movies.drop(['index'], axis = 1)

hulu_imdb_group_low_movies = df_imdb_group_high_movies.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_imdb_group_low_movies = hulu_imdb_group_low_movies.drop(['index'], axis = 1)

hulu_imdb_group_high_movies.head(5)
```

```
# In[56]:
```

```
prime_video_imdb_group_movies = imdb_group_data_movies[imdb_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_imdb_group_movies = prime_video_imdb_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_imdb_group_high_movies = df_imdb_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_imdb_group_high_movies = prime_video_imdb_group_high_movies.drop(['index'], axis = 1)

prime_video_imdb_group_low_movies = df_imdb_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_imdb_group_low_movies = prime_video_imdb_group_low_movies.drop(['index'], axis = 1)

prime_video_imdb_group_high_movies.head(5)
```

```
# In[57]:
```

```
disney_imdb_group_movies = imdb_group_data_movies[imdb_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_imdb_group_movies = disney_imdb_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_imdb_group_high_movies = df_imdb_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_imdb_group_high_movies = disney_imdb_group_high_movies.drop(['index'], axis = 1)

disney_imdb_group_low_movies = df_imdb_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_imdb_group_low_movies = disney_imdb_group_low_movies.drop(['index'], axis = 1)

disney_imdb_group_high_movies.head(5)
```

```
# In[58]:
```

```
print(f'''
    The IMDb Group with Highest Movies Count Ever Got is
'{df_imdb_group_high_movies['IMDb Group'][0]}' : '{df_imdb_group_high_movies['Movies Count'].max()}'\n
```

```

The IMDb Group with Lowest Movies Count Ever Got is '{df_imdb_group_low_movies['IMDb Group'][0]}' : '{df_imdb_group_low_movies['Movies Count'].min()}'\n

The IMDb Group with Highest Movies Count on 'Netflix' is
'{netflix_imdb_group_high_movies['IMDb Group'][0]}' :
'{netflix_imdb_group_high_movies['Netflix'].max()}'\n
The IMDb Group with Lowest Movies Count on 'Netflix' is
'{netflix_imdb_group_low_movies['IMDb Group'][0]}' :
'{netflix_imdb_group_low_movies['Netflix'].min()}'\n

The IMDb Group with Highest Movies Count on 'Hulu' is
'{hulu_imdb_group_high_movies['IMDb Group'][0]}' :
'{hulu_imdb_group_high_movies['Hulu'].max()}'\n
The IMDb Group with Lowest Movies Count on 'Hulu' is
'{hulu_imdb_group_low_movies['IMDb Group'][0]}' :
'{hulu_imdb_group_low_movies['Hulu'].min()}'\n

The IMDb Group with Highest Movies Count on 'Prime Video' is
'{prime_video_imdb_group_high_movies['IMDb Group'][0]}' :
'{prime_video_imdb_group_high_movies['Prime Video'].max()}'\n
The IMDb Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_imdb_group_low_movies['IMDb Group'][0]}' :
'{prime_video_imdb_group_low_movies['Prime Video'].min()}'\n

The IMDb Group with Highest Movies Count on 'Disney+' is
'{disney_imdb_group_high_movies['IMDb Group'][0]}' :
'{disney_imdb_group_high_movies['Disney+'].max()}'\n
The IMDb Group with Lowest Movies Count on 'Disney+' is
'{disney_imdb_group_low_movies['IMDb Group'][0]}' :
'{disney_imdb_group_low_movies['Disney+'].min()}'\n
''')

```

In[59]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_i_ax1 = sns.barplot(x = netflix_imdb_group_movies['IMDb Group'][:10], y =
netflix_imdb_group_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_i_ax2 = sns.barplot(x = hulu_imdb_group_movies['IMDb Group'][:10], y =
hulu_imdb_group_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_i_ax3 = sns.barplot(x = prime_video_imdb_group_movies['IMDb Group'][:10], y =
prime_video_imdb_group_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_i_ax4 = sns.barplot(x = disney_imdb_group_movies['IMDb Group'][:10], y =
disney_imdb_group_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()

```

In[60]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = imdb_group_data_movies['IMDb Group'], y =
imdb_group_data_movies['Netflix'], color = 'red')
sns.lineplot(x = imdb_group_data_movies['IMDb Group'], y = imdb_group_data_movies['Hulu'],
color = 'lightgreen')

```

```

sns.lineplot(x = imdb_group_data_movies['IMDb Group'], y = imdb_group_data_movies['Prime Video'], color = 'lightblue')
sns.lineplot(x = imdb_group_data_movies['IMDb Group'], y = imdb_group_data_movies['Disney+'], color = 'darkblue')
plt.xlabel('IMDb Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

```

In[61]:

```

print(f'''
    Accross All Platforms Total Count of IMDb Group is '{imdb_group_data_movies['IMDb Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Netflix' is '{netflix_imdb_group_movies['IMDb Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Hulu' is '{hulu_imdb_group_movies['IMDb Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Prime Video' is '{prime_video_imdb_group_movies['IMDb Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Disney+' is '{disney_imdb_group_movies['IMDb Group'].unique().shape[0]}'\n
    ''')

```

In[62]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_i_ax1 = sns.lineplot(y = imdb_group_data_movies['IMDb Group'], x = imdb_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_i_ax2 = sns.lineplot(y = imdb_group_data_movies['IMDb Group'], x = imdb_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_i_ax3 = sns.lineplot(y = imdb_group_data_movies['IMDb Group'], x = imdb_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_i_ax4 = sns.lineplot(y = imdb_group_data_movies['IMDb Group'], x = imdb_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()

```

In[63]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_i_ax1 = sns.barplot(x = imdb_group_data_movies['IMDb Group'][:10], y = imdb_group_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_i_ax2 = sns.barplot(x = imdb_group_data_movies['IMDb Group'][:10], y = imdb_group_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_i_ax3 = sns.barplot(x = imdb_group_data_movies['IMDb Group'][:10], y = imdb_group_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_i_ax4 = sns.barplot(x = imdb_group_data_movies['IMDb Group'][:10], y = imdb_group_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

```

```
n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()
```

ottmovies_language.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```

nltk.download('all')

# In[5]:

# path = '/content/drive/MyDrive/Files/'
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_movies = pd.read_csv(path + 'ottmovies.csv')
df_movies.head()

# In[6]:

# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)

```

```

df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix', 'Prime Video', 'Disney+', 'Hulu'], axis = 1)

```

```

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:

data_investigate(df_movies)

# In[11]:

df_movies.head()

# In[12]:

df_movies.describe()

# In[13]:

df_movies.corr()

# In[14]:

# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:

# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:

# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:

df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]

```

```

df_prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) & (df_movies['Prime Video'] == 1) & (df_movies['Disney+'] == 0)]
df_disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) & (df_movies['Prime Video'] == 0) & (df_movies['Disney+'] == 1)]

# In[18]:


df_movies_languages = df_movies.copy()

# In[19]:


df_movies_languages.drop(df_movies_languages.loc[df_movies_languages['Language'] == "NA"].index, inplace = True)
# df_movies_languages = df_movies_languages[df_movies_languages.Language != "NA"]
# df_movies_languages['Language'] = df_movies_languages['Language'].astype(str)

# In[20]:


df_movies_count_languages = df_movies_languages.copy()

# In[21]:


df_movies_language = df_movies_languages.copy()

# In[22]:


# Create languages dict where key=name and value = number of languages
languages = {}

for i in df_movies_count_languages['Language'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        languages[i] = len(i.split(','))
    else:
        languages[i] = 0

# Add this information to our dataframe as a new column

df_movies_count_languages['Number of Languages'] =
df_movies_count_languages['Language'].map(languages).astype(int)

# In[23]:


df_movies_mixed_languages = df_movies_count_languages.copy()

# In[24]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_languages_movies =
df_movies_count_languages.loc[df_movies_count_languages['Netflix'] == 1]

```

```
hulu_languages_movies = df_movies_count_languages.loc[df_movies_count_languages['Hulu'] == 1]
prime_video_languages_movies =
df_movies_count_languages.loc[df_movies_count_languages['Prime Video'] == 1]
disney_languages_movies =
df_movies_count_languages.loc[df_movies_count_languages['Disney+'] == 1]
```

In[25]:

```
plt.figure(figsize = (10, 10))
corr = df_movies_count_languages.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

In[26]:

```
df_languages_most_movies = df_movies_count_languages.sort_values(by = 'Number of Languages', ascending = False).reset_index()
df_languages_most_movies = df_languages_most_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_languages['Number of Languages'] ==
(df_movies_count_languages['Number of Languages'].max()))
# df_languages_most_movies = df_movies_count_languages[filter]

# mostest_rated_movies = df_movies_count_languages.loc[df_movies_count_languages['Number of Languages'].idxmax()]

print('\nMovies with Highest Ever Number of Languages are : \n')
df_languages_most_movies.head(5)
```

In[27]:

```
fig = px.bar(y = df_languages_most_movies['Title'][:15],
              x = df_languages_most_movies['Number of Languages'][:15],
              color = df_languages_most_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Highest Number of Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[28]:

```
df_languages_least_movies = df_movies_count_languages.sort_values(by = 'Number of Languages', ascending = True).reset_index()
df_languages_least_movies = df_languages_least_movies.drop(['index'], axis = 1)
# filter = (df_movies_count_languages['Number of Languages'] ==
(df_movies_count_languages['Number of Languages'].min()))
# df_languages_least_movies = df_movies_count_languages[filter]
```

```

print('\nMovies with Lowest Ever Number of Languages are : \n')
df_languages_least_movies.head(5)

# In[29]:


fig = px.bar(y = df_languages_least_movies['Title'][:15],
              x = df_languages_least_movies['Number of Languages'][:15],
              color = df_languages_least_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Lowest Number of Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
      Total '{df_movies_count_languages['Number of Languages'].unique().shape[0]}' unique
      Number of Languages s were Given, They were Like this,\n

      {df_movies_count_languages.sort_values(by = 'Number of Languages', ascending =
      False)['Number of Languages'].unique()}\n

      The Highest Number of Languages Ever Any Movie Got is
      '{df_languages_most_movies['Title'][0]}' : '{df_languages_most_movies['Number of
      Languages'].max()}'\n

      The Lowest Number of Languages Ever Any Movie Got is
      '{df_languages_least_movies['Title'][0]}' : '{df_languages_least_movies['Number of
      Languages'].min()}'\n
      ''')

# In[31]:


netflix_languages_most_movies =
df_languages_most_movies.loc[df_languages_most_movies['Netflix']==1].reset_index()
netflix_languages_most_movies = netflix_languages_most_movies.drop(['index'], axis = 1)

netflix_languages_least_movies =
df_languages_least_movies.loc[df_languages_least_movies['Netflix']==1].reset_index()
netflix_languages_least_movies = netflix_languages_least_movies.drop(['index'], axis = 1)

netflix_languages_most_movies.head(5)

# In[32]:


fig = px.bar(y = netflix_languages_most_movies['Title'][:15],
              x = netflix_languages_most_movies['Number of Languages'][:15],
              color = netflix_languages_most_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Highest Number of Languages : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[33]:


fig = px.bar(y = netflix_languages_least_movies['Title'][:15],
              x = netflix_languages_least_movies['Number of Languages'][:15],
              color = netflix_languages_least_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Lowest Number of Languages : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


hulu_languages_most_movies =
df_languages_most_movies.loc[df_languages_most_movies['Hulu']==1].reset_index()
hulu_languages_most_movies = hulu_languages_most_movies.drop(['index'], axis = 1)

hulu_languages_least_movies =
df_languages_least_movies.loc[df_languages_least_movies['Hulu']==1].reset_index()
hulu_languages_least_movies = hulu_languages_least_movies.drop(['index'], axis = 1)

hulu_languages_most_movies.head(5)

# In[35]:


fig = px.bar(y = hulu_languages_most_movies['Title'][:15],
              x = hulu_languages_most_movies['Number of Languages'][:15],
              color = hulu_languages_most_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Highest Number of Languages : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_languages_least_movies['Title'][:15],
              x = hulu_languages_least_movies['Number of Languages'][:15],
              color = hulu_languages_least_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Lowest Number of Languages : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_languages_most_movies =
df_languages_most_movies.loc[df_languages_most_movies['Prime Video']==1].reset_index()
prime_video_languages_most_movies = prime_video_languages_most_movies.drop(['index'], axis = 1)

```

```

prime_video_languages_least_movies =
df_languages_least_movies.loc[df_languages_least_movies['Prime Video']==1].reset_index()
prime_video_languages_least_movies = prime_video_languages_least_movies.drop(['index'],
axis = 1)

prime_video_languages_most_movies.head(5)

# In[38]:


fig = px.bar(y = prime_video_languages_most_movies['Title'][:15],
              x = prime_video_languages_most_movies['Number of Languages'][:15],
              color = prime_video_languages_most_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Highest Number of Languages : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[39]:


fig = px.bar(y = prime_video_languages_least_movies['Title'][:15],
              x = prime_video_languages_least_movies['Number of Languages'][:15],
              color = prime_video_languages_least_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Lowest Number of Languages : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


disney_languages_most_movies =
df_languages_most_movies.loc[df_languages_most_movies['Disney+']==1].reset_index()
disney_languages_most_movies = disney_languages_most_movies.drop(['index'], axis = 1)

disney_languages_least_movies =
df_languages_least_movies.loc[df_languages_least_movies['Disney+']==1].reset_index()
disney_languages_least_movies = disney_languages_least_movies.drop(['index'], axis = 1)

disney_languages_most_movies.head(5)

# In[41]:


fig = px.bar(y = disney_languages_most_movies['Title'][:15],
              x = disney_languages_most_movies['Number of Languages'][:15],
              color = disney_languages_most_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Highest Number of Languages : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:

```

```

fig = px.bar(y = disney_languages_least_movies['Title'][:15],
              x = disney_languages_least_movies['Number of Languages'][:15],
              color = disney_languages_least_movies['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Languages'},
              title = 'Movies with Lowest Number of Languages : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''
    The Movie with Highest Number of Languages Ever Got is
'{df_languages_most_movies['Title'][0]}' : '{df_languages_most_movies['Number of
Languages'].max()}'\n
    The Movie with Lowest Number of Languages Ever Got is
'{df_languages_least_movies['Title'][0]}' : '{df_languages_least_movies['Number of
Languages'].min()}'\n

    The Movie with Highest Number of Languages on 'Netflix' is
'{netflix_languages_most_movies['Title'][0]}' : '{netflix_languages_most_movies['Number of
Languages'].max()}'\n
    The Movie with Lowest Number of Languages on 'Netflix' is
'{netflix_languages_least_movies['Title'][0]}' : '{netflix_languages_least_movies['Number of
Languages'].min()}'\n

    The Movie with Highest Number of Languages on 'Hulu' is
'{hulu_languages_most_movies['Title'][0]}' : '{hulu_languages_most_movies['Number of
Languages'].max()}'\n
    The Movie with Lowest Number of Languages on 'Hulu' is
'{hulu_languages_least_movies['Title'][0]}' : '{hulu_languages_least_movies['Number of
Languages'].min()}'\n

    The Movie with Highest Number of Languages on 'Prime Video' is
'{prime_video_languages_most_movies['Title'][0]}' :
'{prime_video_languages_most_movies['Number of Languages'].max()}'\n
    The Movie with Lowest Number of Languages on 'Prime Video' is
'{prime_video_languages_least_movies['Title'][0]}' :
'{prime_video_languages_least_movies['Number of Languages'].min()}'\n

    The Movie with Highest Number of Languages on 'Disney+' is
'{disney_languages_most_movies['Title'][0]}' : '{disney_languages_most_movies['Number of
Languages'].max()}'\n
    The Movie with Lowest Number of Languages on 'Disney+' is
'{disney_languages_least_movies['Title'][0]}' : '{disney_languages_least_movies['Number of
Languages'].min()}'\n
''')


# In[44]:


print(f'''
    Accross All Platforms the Average Number of Languages is
'{round(df_movies_count_languages['Number of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Netflix' is
'{round(netflix_languages_movies['Number of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Hulu' is '{round(hulu_languages_movies['Number of
Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Prime Video' is
'{round(prime_video_languages_movies['Number of Languages'].mean(), ndigits = 2)}'\n
''')



```

```

The Average Number of Languages on 'Disney+' is
'{round(disney_languages_movies['Number of Languages'].mean(), ndigits = 2)}'\n
''')

# In[45]:


print(f'''
    Across All Platforms Total Count of Language is '{df_movies_count_languages['Number
of Languages'].max()}'\n
    Total Count of Language on 'Netflix' is '{netflix_languages_movies['Number of
Languages'].max()}'\n
    Total Count of Language on 'Hulu' is '{hulu_languages_movies['Number of
Languages'].max()}'\n
    Total Count of Language on 'Prime Video' is '{prime_video_languages_movies['Number of
Languages'].max()}'\n
    Total Count of Language on 'Disney+' is '{disney_languages_movies['Number of
Languages'].max()}'\n
'''')

```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_count_languages['Number of Languages'],bins = 20, kde = True, ax =
ax[0])
sns.boxplot(df_movies_count_languages['Number of Languages'], ax = ax[1])
plt.show()
```

```
# In[47]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Number of Languages s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_languages_movies['Number of Languages'], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_languages_movies['Number of Languages'], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_languages_movies['Number of Languages'], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_languages_movies['Number of Languages'], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

```
# In[48]:
```

```
df_lan = df_movies_language['Language'].str.split(',').apply(pd.Series).stack()
del df_movies_language['Language']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Language'
df_movies_language = df_movies_language.join(df_lan)
df_movies_language.drop_duplicates(inplace = True)
```

```
# In[49]:
```

```

df_movies_language.head(5)

# In[50]:


language_count = df_movies_language.groupby('Language')['Title'].count()
language_movies = df_movies_language.groupby('Language')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
language_data_movies = pd.concat([language_count, language_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
language_data_movies = language_data_movies.sort_values(by = 'Movies Count', ascending =
False)

# In[51]:


# Language with Movies Counts - All Platforms Combined
language_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[52]:


fig = px.bar(x = language_data_movies['Language'][:50],
              y = language_data_movies['Movies Count'][:50],
              color = language_data_movies['Movies Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Language', 'y' : 'Movies Count'},
              title = 'Major Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[53]:


df_language_high_movies = language_data_movies.sort_values(by = 'Movies Count', ascending =
False).reset_index()
df_language_high_movies = df_language_high_movies.drop(['index'], axis = 1)
# filter = (language_data_movies['Movies Count'] == (language_data_movies['Movies
Count'].max()))
# df_language_high_movies = language_data_movies[filter]

# highestRatedMovies = language_data_movies.loc[language_data_movies['Movies
Count'].idxmax()]

print('\nLanguage with Highest Ever Movies Count are : All Platforms Combined\n')
df_language_high_movies.head(5)

# In[54]:


fig = px.bar(y = df_language_high_movies['Language'][:15],
              x = df_language_high_movies['Movies Count'][:15],
              color = df_language_high_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'Movies Count'},
              title = 'Language with Highest Movies : All Platforms')

fig.update_layout(plot_bgcolor = 'white')

```

```

fig.show()

# In[55]:


df_language_low_movies = language_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_language_low_movies = df_language_low_movies.drop(['index'], axis = 1)
# filter = (language_data_movies['Movies Count'] == (language_data_movies['Movies Count'].min()))
# df_language_low_movies = language_data_movies[filter]

print('\nLanguage with Lowest Ever Movies Count are : All Platforms Combined\n')
df_language_low_movies.head(5)

# In[56]:


fig = px.bar(y = df_language_low_movies['Language'][:15],
              x = df_language_low_movies['Movies Count'][:15],
              color = df_language_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'Movies Count'},
              title = 'Language with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[57]:


print(f'''
    Total '{language_data_movies['Language'].unique().shape[0]}' unique Language Count s
    were Given, They were Like this,\n

    {language_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Language']].unique()[:5]}\n

    The Highest Ever Movies Count Ever Any Movie Got is
    '{df_language_high_movies['Language'][0]}' : '{df_language_high_movies['Movies Count'].max()}'\n

    The Lowest Ever Movies Count Ever Any Movie Got is
    '{df_language_low_movies['Language'][0]}' : '{df_language_low_movies['Movies Count'].min()}'\n
    ''')

# In[58]:


fig = px.pie(language_data_movies[:10], names = 'Language', values = 'Movies Count',
             color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies Count based on Language')
fig.show()

# In[59]:


# netflix_language_movies = language_data_movies[language_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()

```

```
# netflix_language_movies = netflix_language_movies.drop(['index', 'Hulu', 'Prime Video',  
'Disney+', 'Movies Count'], axis = 1)  
  
netflix_language_high_movies = df_language_high_movies.sort_values(by = 'Netflix',  
ascending = False).reset_index()  
netflix_language_high_movies = netflix_language_high_movies.drop(['index'], axis = 1)  
  
netflix_language_low_movies = df_language_high_movies.sort_values(by = 'Netflix', ascending  
= True).reset_index()  
netflix_language_low_movies = netflix_language_low_movies.drop(['index'], axis = 1)  
  
netflix_language_high_movies.head(5)
```

In[60]:

```
fig = px.bar(x = netflix_language_high_movies['Language'][:15],  
             y = netflix_language_high_movies['Netflix'][:15],  
             color = netflix_language_high_movies['Netflix'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Language', 'x' : 'Movies Count'},  
             title = 'Language with Highest Movies : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

In[61]:

```
# hulu_language_movies = language_data_movies[language_data_movies['Hulu'] !=  
0].sort_values(by = 'Hulu', ascending = False).reset_index()  
# hulu_language_movies = hulu_language_movies.drop(['index', 'Netflix', 'Prime Video',  
'Disney+', 'Movies Count'], axis = 1)  
  
hulu_language_high_movies = df_language_high_movies.sort_values(by = 'Hulu', ascending =  
False).reset_index()  
hulu_language_high_movies = hulu_language_high_movies.drop(['index'], axis = 1)  
  
hulu_language_low_movies = df_language_high_movies.sort_values(by = 'Hulu', ascending =  
True).reset_index()  
hulu_language_low_movies = hulu_language_low_movies.drop(['index'], axis = 1)  
  
hulu_language_high_movies.head(5)
```

In[62]:

```
fig = px.bar(x = hulu_language_high_movies['Language'][:15],  
             y = hulu_language_high_movies['Hulu'][:15],  
             color = hulu_language_high_movies['Hulu'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Language', 'x' : 'Movies Count'},  
             title = 'Language with Highest Movies : Hulu')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

In[63]:

```
# prime_video_language_movies = language_data_movies[language_data_movies['Prime Video'] !=  
0].sort_values(by = 'Prime Video', ascending = False).reset_index()
```

```

# prime_video_language_movies = prime_video_language_movies.drop(['index', 'Netflix',
'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_language_high_movies = df_language_high_movies.sort_values(by = 'Prime Video',
ascending = False).reset_index()
prime_video_language_high_movies = prime_video_language_high_movies.drop(['index'], axis =
1)

prime_video_language_low_movies = df_language_high_movies.sort_values(by = 'Prime Video',
ascending = True).reset_index()
prime_video_language_low_movies = prime_video_language_low_movies.drop(['index'], axis = 1)

prime_video_language_high_movies.head(5)

# In[64]:


fig = px.bar(x = prime_video_language_high_movies['Language'][:15],
              y = prime_video_language_high_movies['Prime Video'][:15],
              color = prime_video_language_high_movies['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'Movies Count'},
              title = 'Language with Highest Movies : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[65]:


# disney_language_movies = language_data_movies[language_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_language_movies = disney_language_movies.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'Movies Count'], axis = 1)

disney_language_high_movies = df_language_high_movies.sort_values(by = 'Disney+', ascending
= False).reset_index()
disney_language_high_movies = disney_language_high_movies.drop(['index'], axis = 1)

disney_language_low_movies = df_language_high_movies.sort_values(by = 'Disney+', ascending
= True).reset_index()
disney_language_low_movies = disney_language_low_movies.drop(['index'], axis = 1)

disney_language_high_movies.head(5)

# In[66]:


fig = px.bar(x = disney_language_high_movies['Language'][:15],
              y = disney_language_high_movies['Disney+'][:15],
              color = disney_language_high_movies['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'Movies Count'},
              title = 'Language with Highest Movies : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[67]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))

```

```

sns.distplot(language_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(language_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_language_movies = language_data_movies[language_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_language_movies = netflix_language_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_language_movies = language_data_movies[language_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_language_movies = hulu_language_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

prime_video_language_movies = language_data_movies[language_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_language_movies = prime_video_language_movies.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'Movies Count'], axis = 1)

disney_language_movies = language_data_movies[language_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_language_movies = disney_language_movies.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'Movies Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Language Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_language_movies['Disney+'][:50], color = 'darkblue', legend = True, kde =
True)
sns.histplot(prime_video_language_movies['Prime Video'][:50], color = 'lightblue', legend =
True, kde = True)
sns.histplot(netflix_language_movies['Netflix'][:50], color = 'red', legend = True, kde =
True)
sns.histplot(hulu_language_movies['Hulu'][:50], color = 'lightgreen', legend = True, kde =
True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''The Language with Highest Movies Count Ever Got is
'{df_language_high_movies['Language'][0]}' : '{df_language_high_movies['Movies
Count'].max()}'\n
The Language with Lowest Movies Count Ever Got is
'{df_language_low_movies['Language'][0]}' : '{df_language_low_movies['Movies
Count'].min()}'\n
''')

```

```

    The Language with Highest Movies Count on 'Netflix' is
'{netflix_language_high_movies['Language'][0]}' :
'{netflix_language_high_movies['Netflix'].max()}'\n
    The Language with Lowest Movies Count on 'Netflix' is
'{netflix_language_low_movies['Language'][0]}' :
'{netflix_language_low_movies['Netflix'].min()}'\n

    The Language with Highest Movies Count on 'Hulu' is
'{hulu_language_high_movies['Language'][0]}' :
'{hulu_language_high_movies['Hulu'].max()}'\n
    The Language with Lowest Movies Count on 'Hulu' is
'{hulu_language_low_movies['Language'][0]}' : '{hulu_language_low_movies['Hulu'].min()}'\n

    The Language with Highest Movies Count on 'Prime Video' is
'{prime_video_language_high_movies['Language'][0]}' :
'{prime_video_language_high_movies['Prime Video'].max()}'\n
    The Language with Lowest Movies Count on 'Prime Video' is
'{prime_video_language_low_movies['Language'][0]}' :
'{prime_video_language_low_movies['Prime Video'].min()}'\n

    The Language with Highest Movies Count on 'Disney+' is
'{disney_language_high_movies['Language'][0]}' :
'{disney_language_high_movies['Disney+'].max()}'\n
    The Language with Lowest Movies Count on 'Disney+' is
'{disney_language_low_movies['Language'][0]}' :
'{disney_language_low_movies['Disney+'].min()}'\n
    ''')

```

In[71]:

```

# Distribution of movies language in each platform
plt.figure(figsize = (20, 5))
plt.title('Language with Movies Count for All Platforms')
sns.violinplot(x = language_data_movies['Movies Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

```

In[72]:

```

# Distribution of Language Movies Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_language_movies['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_language_movies['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_language_movies['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_language_movies['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()

```

In[73]:

```

print(f'''
    Accross All Platforms the Average Movies Count of Language is
'{round(language_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Language on 'Netflix' is
'{round(netflix_language_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Language on 'Hulu' is
'{round(hulu_language_movies['Hulu'].mean(), ndigits = 2)}'\n

```

```
The Average Movies Count of Language on 'Prime Video' is
'{round(prime_video_language_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Language on 'Disney+' is
'{round(disney_language_movies['Disney+'].mean(), ndigits = 2)}'\n
    ''')
```

```
# In[74]:
```

```
print(f'''
    Across All Platforms Total Count of Language is
'{language_data_movies['Language'].unique().shape[0]}'\n
    Total Count of Language on 'Netflix' is
'{netflix_language_movies['Language'].unique().shape[0]}'\n
    Total Count of Language on 'Hulu' is
'{hulu_language_movies['Language'].unique().shape[0]}'\n
    Total Count of Language on 'Prime Video' is
'{prime_video_language_movies['Language'].unique().shape[0]}'\n
    Total Count of Language on 'Disney+' is
'{disney_language_movies['Language'].unique().shape[0]}'
    ''')
```

```
# In[75]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = language_data_movies['Language'][:10], y =
language_data_movies['Netflix'][:10], color = 'red')
sns.lineplot(x = language_data_movies['Language'][:10], y =
language_data_movies['Hulu'][:10], color = 'lightgreen')
sns.lineplot(x = language_data_movies['Language'][:10], y = language_data_movies['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = language_data_movies['Language'][:10], y =
language_data_movies['Disney+'][:10], color = 'darkblue')
plt.xlabel('Language', fontsize = 20)
plt.ylabel('Movies Count', fontsize = 20)
plt.show()
```

```
# In[76]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_l_ax1 = sns.lineplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_l_ax2 = sns.lineplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_l_ax3 = sns.lineplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_l_ax4 = sns.lineplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[77]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = netflix_language_movies['Language'][:10], x =
netflix_language_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = hulu_language_movies['Language'][:10], x =
hulu_language_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = prime_video_language_movies['Language'][:10], x =
prime_video_language_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = disney_language_movies['Language'][:10], x =
disney_language_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()

```

In[78]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Language Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_language_movies['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_language_movies['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_language_movies['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_language_movies['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

```

In[79]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = language_data_movies['Language'][:10], x =
language_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()

```

```

# In[80]:
df_movies_mixed_languages.drop(df_movies_mixed_languages.loc[df_movies_mixed_languages['Language'] == "NA"].index, inplace = True)
# df_movies_mixed_languages = df_movies_mixed_languages[df_movies_mixed_languages.Language != "NA"]
df_movies_mixed_languages.drop(df_movies_mixed_languages.loc[df_movies_mixed_languages['Number of Languages'] == 1].index, inplace = True)

# In[81]:
df_movies_mixed_languages.head(5)

# In[82]:
mixed_languages_count = df_movies_mixed_languages.groupby('Language')['Title'].count()
mixed_languages_movies = df_movies_mixed_languages.groupby('Language')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_languages_data_movies = pd.concat([mixed_languages_count, mixed_languages_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count', 'Language' : 'Mixed Language'})
mixed_languages_data_movies = mixed_languages_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[83]:
mixed_languages_data_movies.head(5)

# In[84]:
# Mixed Language with Movies Counts - All Platforms Combined
mixed_languages_data_movies.sort_values(by = 'Movies Count', ascending = False)[:10]

# In[85]:
df_mixed_languages_high_movies = mixed_languages_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_mixed_languages_high_movies = df_mixed_languages_high_movies.drop(['index'], axis = 1)
# filter = (mixed_languages_data_movies['Movies Count'] == (mixed_languages_data_movies['Movies Count'].max()))
# df_mixed_languages_high_movies = mixed_languages_data_movies[filter]

# highest_rated_movies =
mixed_languages_data_movies.loc[mixed_languages_data_movies['Movies Count'].idxmax()]

print('\nMixed Language with Highest Ever Movies Count are : All Platforms Combined\n')
df_mixed_languages_high_movies.head(5)

# In[86]:
fig = px.bar(y = df_mixed_languages_high_movies['Mixed Language'][:15],
              x = df_mixed_languages_high_movies['Movies Count'][:15],
              color = df_mixed_languages_high_movies['Movies Count'][:15],

```

```

    color_continuous_scale = 'Teal_r',
    labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Language'},
    title  = 'Movies with Highest Number of Mixed Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[87]:

```

df_mixed_languages_low_movies = mixed_languages_data_movies.sort_values(by = 'Movies
Count', ascending = True).reset_index()
df_mixed_languages_low_movies = df_mixed_languages_low_movies.drop(['index'], axis = 1)
# filter = (mixed_languages_data_movies['Movies Count'] == 
(mixed_languages_data_movies['Movies Count'].min()))
# df_mixed_languages_low_movies = mixed_languages_data_movies[filter]

print('\nMixed Language with Lowest Ever Movies Count are : All Platforms Combined\n')
df_mixed_languages_low_movies.head(5)

```

In[88]:

```

fig = px.bar(y = df_mixed_languages_low_movies['Mixed Language'][:15],
              x = df_mixed_languages_low_movies['Movies Count'][:15],
              color = df_mixed_languages_low_movies['Movies Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Number of Mixed Language'},
              title  = 'Movies with Lowest Number of Mixed Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[89]:

```

print(f'''
    Total '{df_movies_languages['Language'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see Movies from Total '{mixed_languages_data_movies['Mixed
Language'].unique().shape[0]}' Mixed Language, They were Like this, \n

    {mixed_languages_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Mixed Language'].head(5).unique()} etc. \n

    The Mixed Language with Highest Movies Count have
'{mixed_languages_data_movies['Movies Count'].max()}' Movies Available is
'{df_mixed_languages_high_movies['Mixed Language'][0]}', &\n
    The Mixed Language with Lowest Movies Count have
'{mixed_languages_data_movies['Movies Count'].min()}' Movies Available is
'{df_mixed_languages_low_movies['Mixed Language'][0]}'
    ''')

```

In[90]:

```

fig = px.pie(mixed_languages_data_movies[:10], names = 'Mixed Language', values = 'Movies
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'Movies
Count based on Mixed Language')
fig.show()

```

```

# In[91]:


# netflix_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
# netflix_mixed_languages_movies = netflix_mixed_languages_movies.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_mixed_languages_high_movies = df_mixed_languages_high_movies.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_languages_high_movies = netflix_mixed_languages_high_movies.drop(['index'],
axis = 1)

netflix_mixed_languages_low_movies = df_mixed_languages_high_movies.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_mixed_languages_low_movies = netflix_mixed_languages_low_movies.drop(['index'],
axis = 1)

netflix_mixed_languages_high_movies.head(5)

# In[92]:


# hulu_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_mixed_languages_movies = hulu_mixed_languages_movies.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_languages_high_movies = df_mixed_languages_high_movies.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_mixed_languages_high_movies = hulu_mixed_languages_high_movies.drop(['index'], axis =
1)

hulu_mixed_languages_low_movies = df_mixed_languages_high_movies.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_languages_low_movies = hulu_mixed_languages_low_movies.drop(['index'], axis = 1)

hulu_mixed_languages_high_movies.head(5)

# In[93]:


# prime_video_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Prime Video'] !=
0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_languages_movies = prime_video_mixed_languages_movies.drop(['index',
'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_languages_high_movies = df_mixed_languages_high_movies.sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_languages_high_movies =
prime_video_mixed_languages_high_movies.drop(['index'], axis = 1)

prime_video_mixed_languages_low_movies = df_mixed_languages_high_movies.sort_values(by =
'Prime Video', ascending = True).reset_index()
prime_video_mixed_languages_low_movies =
prime_video_mixed_languages_low_movies.drop(['index'], axis = 1)

prime_video_mixed_languages_high_movies.head(5)

```

```

# In[94]:


# disney_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_languages_movies = disney_mixed_languages_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_mixed_languages_high_movies = df_mixed_languages_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_languages_high_movies = disney_mixed_languages_high_movies.drop(['index'], axis = 1)

disney_mixed_languages_low_movies = df_mixed_languages_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_languages_low_movies = disney_mixed_languages_low_movies.drop(['index'], axis = 1)

disney_mixed_languages_high_movies.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_languages_data_movies['Movies Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_languages_data_movies['Movies Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the movies present on individual streaming platforms
netflix_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_languages_movies = netflix_mixed_languages_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_languages_movies = hulu_mixed_languages_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

prime_video_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_languages_movies = prime_video_mixed_languages_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

disney_mixed_languages_movies =
mixed_languages_data_movies[mixed_languages_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_languages_movies = disney_mixed_languages_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

# In[97]:


# Defining plot size and title

```

```

plt.figure(figsize = (20, 5))
plt.title('Mixed Language Movies Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_languages_movies['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_languages_movies['Netflix'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_mixed_languages_movies['Hulu'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_mixed_languages_movies['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Language with Highest Movies Count Ever Got is
'{df_mixed_languages_high_movies['Mixed Language'][0]}' :
'{df_mixed_languages_high_movies['Movies Count'].max()}'\n
    The Mixed Language with Lowest Movies Count Ever Got is
'{df_mixed_languages_low_movies['Mixed Language'][0]}' :
'{df_mixed_languages_low_movies['Movies Count'].min()}'\n

    The Mixed Language with Highest Movies Count on 'Netflix' is
'{netflix_mixed_languages_high_movies['Mixed Language'][0]}' :
'{netflix_mixed_languages_high_movies['Netflix'].max()}'\n
    The Mixed Language with Lowest Movies Count on 'Netflix' is
'{netflix_mixed_languages_low_movies['Mixed Language'][0]}' :
'{netflix_mixed_languages_low_movies['Netflix'].min()}'\n

    The Mixed Language with Highest Movies Count on 'Hulu' is
'{hulu_mixed_languages_high_movies['Mixed Language'][0]}' :
'{hulu_mixed_languages_high_movies['Hulu'].max()}'\n
    The Mixed Language with Lowest Movies Count on 'Hulu' is
'{hulu_mixed_languages_low_movies['Mixed Language'][0]}' :
'{hulu_mixed_languages_low_movies['Hulu'].min()}'\n

    The Mixed Language with Highest Movies Count on 'Prime Video' is
'{prime_video_mixed_languages_high_movies['Mixed Language'][0]}' :
'{prime_video_mixed_languages_high_movies['Prime Video'].max()}'\n
    The Mixed Language with Lowest Movies Count on 'Prime Video' is
'{prime_video_mixed_languages_low_movies['Mixed Language'][0]}' :
'{prime_video_mixed_languages_low_movies['Prime Video'].min()}'\n

    The Mixed Language with Highest Movies Count on 'Disney+' is
'{disney_mixed_languages_high_movies['Mixed Language'][0]}' :
'{disney_mixed_languages_high_movies['Disney+'].max()}'\n
    The Mixed Language with Lowest Movies Count on 'Disney+' is
'{disney_mixed_languages_low_movies['Mixed Language'][0]}' :
'{disney_mixed_languages_low_movies['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Accross All Platforms the Average Movies Count of Mixed Language is
'{round(mixed_languages_data_movies['Movies Count'].mean(), ndigits = 2)}'\n

```

```

    The Average Movies Count of Mixed Language on 'Netflix' is
'{round(netflix_mixed_languages_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Language on 'Hulu' is
'{round(hulu_mixed_languages_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Language on 'Prime Video' is
'{round(prime_video_mixed_languages_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Mixed Language on 'Disney+' is
'{round(disney_mixed_languages_movies['Disney+'].mean(), ndigits = 2)}'\n
    ''')

# In[100]:


print(f'''
    Accross All Platforms Total Count of Mixed Language is
'{mixed_languages_data_movies['Mixed Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Netflix' is '{netflix_mixed_languages_movies['Mixed
Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Hulu' is '{hulu_mixed_languages_movies['Mixed
Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Prime Video' is
'{prime_video_mixed_languages_movies['Mixed Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Disney+' is '{disney_mixed_languages_movies['Mixed
Language'].unique().shape[0]}'
    ''')

# In[101]:


plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_languages_data_movies['Mixed Language'][:5], y =
mixed_languages_data_movies['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_languages_data_movies['Mixed Language'][:5], y =
mixed_languages_data_movies['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_languages_data_movies['Mixed Language'][:5], y =
mixed_languages_data_movies['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_languages_data_movies['Mixed Language'][:5], y =
mixed_languages_data_movies['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Language', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

# In[102]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

```

```

plt.show()

# In[103]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_ml_ax1 = sns.lineplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_ml_ax2 = sns.lineplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_ml_ax3 = sns.lineplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_ml_ax4 = sns.lineplot(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ml_ax1.title.set_text(labels[0])
h_ml_ax2.title.set_text(labels[1])
p_ml_ax3.title.set_text(labels[2])
d_ml_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Language Movies Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_languages_movies['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_languages_movies['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_languages_movies['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_languages_movies['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ml_ax1 = sns.barplot(y = netflix_mixed_languages_movies['Mixed Language'][:10], x =
netflix_mixed_languages_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ml_ax2 = sns.barplot(y = hulu_mixed_languages_movies['Mixed Language'][:10], x =
hulu_mixed_languages_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ml_ax3 = sns.barplot(y = prime_video_mixed_languages_movies['Mixed Language'][:10], x =
prime_video_mixed_languages_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_ml_ax4 = sns.barplot(y = disney_mixed_languages_movies['Mixed Language'][:10], x =
disney_mixed_languages_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ml_ax1.title.set_text(labels[0])
h_ml_ax2.title.set_text(labels[1])

```

```

p_ml_ax3.title.set_text(labels[2])
d_ml_ax4.title.set_text(labels[3])

plt.show()

# In[106]:


fig = go.Figure(go.Funnel(y = mixed_languages_data_movies['Mixed Language'][:10], x =
mixed_languages_data_movies['Movies Count'][:10]))
fig.show()

```

ottmovies_ott.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# ! pip install wordcloud
# ! pip install Flask

```

```
# In[2]:
```

```

# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

```

```
# In[3]:
```

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS

```

```

from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_movies = pd.read_csv(path + 'ottmovies.csv')
df_movies.head()

# In[6]:


# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

```

```

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)

```

```

# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix', 'Prime Video', 'Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:
# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:

```

```
df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]
```

```
# In[18]:
```

```
df.movies_ott = df_movies.copy()
```

```
# In[19]:
```

```
df.movies_ott.drop(df.movies_ott.loc[df.movies_ott['Title'] == "NA"].index, inplace = True)
# df.movies_ott = df.movies_ott[df.movies_ott.Title != "NA"]
```

```
# In[20]:
```

```
# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_ott_movies = df.movies_ott.loc[df.movies_ott['Netflix'] == 1]
hulu_ott_movies = df.movies_ott.loc[df.movies_ott['Hulu'] == 1]
prime_video_ott_movies = df.movies_ott.loc[df.movies_ott['Prime Video'] == 1]
disney_ott_movies = df.movies_ott.loc[df.movies_ott['Disney+'] == 1]
```

```
# In[21]:
```

```
df.movies_ott_group = df.movies_ott.copy()
```

```
# In[22]:
```

```
plt.figure(figsize = (10, 10))
corr = df.movies_ott.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[23]:
```

```
print('\nMovies Available on All Platfroms Are : \n')
df.movies_ott.head(5)
```

```

# In[24]:
print('\nMovies Available on Netflix Are : \n')
netflix_ott_movies.head(5)

# In[25]:
print('\nMovies Available on Hulu Are : \n')
hulu_ott_movies.head(5)

# In[26]:
print('\nMovies Available on Prime Video Are : \n')
prime_video_ott_movies.head(5)

# In[27]:
print('\nMovies Available on Disney+ Are : \n')
disney_ott_movies.head(5)

# In[28]:
print(f'''
    Total '{df_movies_ott['Title'].count()}' Titles are available on All Platforms, out
    of Which,\n
        Total '{netflix_ott_movies['Title'].count()}' Titles are available on 'Netflix'
        Total '{hulu_ott_movies['Title'].count()}' Titles are available on 'Hulu'
        Total '{prime_video_ott_movies['Title'].count()}' Titles are available on 'Prime
        video'
        Total '{disney_ott_movies['Title'].count()}' Titles are available on 'Disney+'
    ''')

# In[29]:
Platform = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

Count = [netflix_ott_movies['Title'].count(),
         hulu_ott_movies['Title'].count(),
         prime_video_ott_movies['Title'].count(),
         disney_ott_movies['Title'].count()]

fig = px.pie(names = Platform,
              values = Count,
              title = 'Movies Count Of Different Platforms',
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textposition = 'inside',
                  textinfo = 'percent + label')
fig.show()

# In[30]:

```

```

df_movies_ott['OTT Count'] = df_movies_ott['Netflix'] + df_movies_ott['Hulu'] +
df_movies_ott['Prime Video'] + df_movies_ott['Disney+']

# In[31]:
(df_movies_ott['OTT Count'].value_counts()/df_movies_ott.shape[0])*100

# In[32]:
print(f'''
    Total '{df_movies_ott[df_movies_ott['OTT Count'] == 4].shape[0]}' Titles are
available on All Platforms
    Total '{df_movies_ott[df_movies_ott['OTT Count'] == 3].shape[0]}' Titles are
available on at least 3 Platforms
    Total '{df_movies_ott[df_movies_ott['OTT Count'] == 2].shape[0]}' Titles are
available on at least 2 Platforms
    Total '{df_movies_ott[df_movies_ott['OTT Count'] == 1].shape[0]}' Titles are
available on at least 1 Platforms
    ''')

# In[33]:
# Movies Available on All Platforms

# df_movies_ott[(df_movies_ott['Netflix'] == 1) & (df_movies_ott['Hulu'] == 1) &
# (df_movies_ott['Prime Video'] == 1) & (df_movies_ott['Disney+'] == 1)]
print('\nTotal ', df_movies_ott[df_movies_ott['OTT Count'] == 4].shape[0], ' Titles are
available on All Platforms\n')
movies_on_4_platforms = df_movies_ott[df_movies_ott['OTT Count'] == 4]
movies_on_4_platforms

# In[34]:
# Movies Available on at least 3 Platforms

print('\nTotal ', df_movies_ott[df_movies_ott['OTT Count'] == 3].shape[0], ' Titles are
available on at least 3 Platforms\n')
movies_on_3_platforms = df_movies_ott[df_movies_ott['OTT Count'] == 3]
movies_on_3_platforms

# In[35]:
# Movies Available on at least 2 Platforms

print('\nTotal ', df_movies_ott[df_movies_ott['OTT Count'] == 2].shape[0], ' Titles are
available on at least 2 Platforms\n')
movies_on_2_platforms = df_movies_ott[df_movies_ott['OTT Count'] == 2]
movies_on_2_platforms

# In[36]:
# Movies Available on at least 1 Platform

print('\nTotal ', df_movies_ott[df_movies_ott['OTT Count'] == 1].shape[0], ' Titles are
available on at least 1 Platforms\n')
movies_on_1_platforms = df_movies_ott[df_movies_ott['OTT Count'] == 1]

```

```
movies_on_1_platforms
```

```
# In[37]:
```

```
df_netflix_only_movies_ott = df_movies_ott[(df_movies_ott['Netflix'] == 1) &
(df_movies_ott['Hulu'] == 0) & (df_movies_ott['Prime Video'] == 0 ) &
(df_movies_ott['Disney+'] == 0)]
df_hulu_only_movies_ott = df_movies_ott[(df_movies_ott['Netflix'] == 0) &
(df_movies_ott['Hulu'] == 1) & (df_movies_ott['Prime Video'] == 0 ) &
(df_movies_ott['Disney+'] == 0)]
df_prime_video_only_movies_ott = df_movies_ott[(df_movies_ott['Netflix'] == 0) &
(df_movies_ott['Hulu'] == 0) & (df_movies_ott['Prime Video'] == 1 ) &
(df_movies_ott['Disney+'] == 0)]
df_disney_only_movies_ott = df_movies_ott[(df_movies_ott['Netflix'] == 0) &
(df_movies_ott['Hulu'] == 0) & (df_movies_ott['Prime Video'] == 0 ) &
(df_movies_ott['Disney+'] == 1)]
```

```
# In[38]:
```

```
# Movies Available only on Netflix
```

```
print('\nTotal ', df_netflix_only_movies_ott['Title'].shape[0], ' Titles are available only  
on Netflix\n')
```

```
df_netflix_only_movies_ott
```

```
# In[39]:
```

```
# Movies Available only on Hulu
```

```
print('\nTotal ', df_hulu_only_movies_ott['Title'].shape[0], ' Titles are available only on  
Hulu\n')
```

```
df_hulu_only_movies_ott
```

```
# In[40]:
```

```
# Movies Available only on Prime Video
```

```
print('\nTotal ', df_prime_video_only_movies_ott['Title'].shape[0], ' Titles are available  
only on Prime Video\n')
```

```
df_prime_video_only_movies_ott
```

```
# In[41]:
```

```
# Movies Available only on Disney+
```

```
print('\nTotal ', df_disney_only_movies_ott['Title'].shape[0], ' Titles are available only  
on Disney+\n')
```

```
df_disney_only_movies_ott
```

```
# In[42]:
```

```

ott_group_count = df_movies_ott.groupby('OTT Count')['Title'].count()
ott_group_movies = df_movies_ott.groupby('OTT Count')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
ott_group_data_movies = pd.concat([ott_group_count, ott_group_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
ott_group_data_movies = ott_group_data_movies.sort_values(by = 'Movies Count', ascending =
False)

# In[43]: 

ott_group_data_movies

# In[44]: 

# Title Group with Movies Counts - All Platforms Combined
ott_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[45]: 

ott_group_data_movies.sort_values(by = 'OTT Count', ascending = False)

# In[46]: 

fig = px.bar(y = ott_group_data_movies['Movies Count'],
x = ott_group_data_movies['OTT Count'],
color = ott_group_data_movies['OTT Count'],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies Count', 'x' : 'OTT Count'},
title = 'Movies with Group Title : All Platforms')
fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[47]: 

fig = px.pie(ott_group_data_movies,
names = ott_group_data_movies['OTT Count'],
values = ott_group_data_movies['Movies Count'],
color = ott_group_data_movies['Movies Count'],
color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
title = 'Movies Count based on OTT Count')
fig.show()

```

ottmovies_plotline.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]: 

```

```
# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_movies = pd.read_csv(path + 'ottmovies.csv')
df_movies.head()
```

```
# In[6]:
```

```

# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('*'*25)
    print('Colums Names : \n', df.columns)
    print('*'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('*'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('*'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('*'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_movies)

# In[9]:


# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)

```

```

# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

```

```

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:
# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:
df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] ==
0) & (df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]

# In[18]:
df_movies_plotline = df_movies.copy()

# In[19]:
df_movies_plotline.drop(df_movies_plotline.loc[df_movies_plotline['Plotline'] ==
"NA"].index, inplace = True)
# df_movies_plotline = df_movies_plotline[df_movies_plotline.Plotline != "NA"]

```

```
# In[20]:
# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_plotline_movies = df_movies_plotline.loc[df_movies_plotline['Netflix'] == 1]
hulu_plotline_movies = df_movies_plotline.loc[df_movies_plotline['Hulu'] == 1]
prime_video_plotline_movies = df_movies_plotline.loc[df_movies_plotline['Prime Video'] ==
1]
disney_plotline_movies = df_movies_plotline.loc[df_movies_plotline['Disney+'] == 1]

# In[21]:
plt.figure(figsize = (10, 10))
corr = df_movies_plotline.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[22]:
df_movies_plotline = df_movies_plotline['Plotline']
movies_plotline_w = ' '.join(df_movies_plotline)

# In[23]:
stopwords = set(STOPWORDS)

wordcloud_all_plotline_movies = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10).generate(movies_plotline_w)

# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_all_plotline_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

# In[24]:
movies_plotline_w = movies_plotline_w.lower()

stop_words_english_movies = set(STOPWORDS)

word_tokens_english_movies = word_tokenize(movies_plotline_w)
```

```

filtered_movie_plotline = [w for w in word_tokens_english_movies if not w in
stop_words_english_movies]

filtered_movie_plotline = " ".join(filtered_movie_plotline)

filtered_movie_plotline = re.sub("'s", ' ', filtered_movie_plotline)

filtered_movie_plotline = re.sub(r'[0-9]+', ' ', filtered_movie_plotline)

final_movie_plotline = re.sub(r'^\w\s]', ' ', filtered_movie_plotline)

plotline_movies_corpus_len = len(filtered_movie_plotline.split())
plotline_movies_corpus_len

# In[25]:


def extract_ngrams(data, num):
    n_grams = ngrams(nltk.word_tokenize(data), num)
    return [ ' '.join(grams) for grams in n_grams]

# In[26]:


plotline_ngram1_movies = FreqDist()

plotline_ngram1 = extract_ngrams(final_movie_plotline[:plotline_movies_corpus_len], 1)

for word in plotline_ngram1:
    plotline_ngram1_movies[word.lower()] += 1

# In[27]:


plotline_ngram1_movies.most_common(10)

# In[28]:


plotline_ngram2_movies = FreqDist()

plotline_ngram2 = extract_ngrams(final_movie_plotline[:plotline_movies_corpus_len], 2)

for word in plotline_ngram2:
    plotline_ngram2_movies[word.lower()] += 1

# In[29]:


plotline_ngram2_movies.most_common(10)

# In[30]:


plotline_ngram3_movies = FreqDist()

plotline_ngram3 = extract_ngrams(final_movie_plotline[:plotline_movies_corpus_len], 3)

for word in plotline_ngram3:

```

```

plotline_ngram3_movies[word.lower()] += 1

# In[31]:

plotline_ngram3_movies.most_common(10)

# In[32]:

plotline_ngram4_movies = FreqDist()
plotline_ngram4 = extract_ngrams(final_movie_plotline[:plotline_movies_corpus_len], 4)

for word in plotline_ngram4:
    plotline_ngram4_movies[word.lower()] += 1

# In[33]:

plotline_ngram4_movies.most_common(10)

# In[34]:

plotline_ngram5_movies = FreqDist()
plotline_ngram5 = extract_ngrams(final_movie_plotline[:plotline_movies_corpus_len], 5)

for word in plotline_ngram5:
    plotline_ngram5_movies[word.lower()] += 1

# In[35]:

plotline_ngram5_movies.most_common(10)

# In[36]:

# Netflix Wordcloud
netflix_plotline_movies_t = netflix_plotline_movies['Plotline']
netflix_movies_plotline_w = ' '.join(netflix_plotline_movies_t)

# In[37]:

stopwords = set(STOPWORDS)

wordcloud.netflix_plotline_movies = WordCloud(width = 1000, height = 500,
                                              background_color ='white',
                                              stopwords = stopwords,
                                              min_font_size = 10
                                              ).generate(netflix_movies_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Netflix is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud.netflix_plotline_movies)

```

```
plt.axis("off")
plt.tight_layout(pad = 0)
```

```
plt.show()
```

```
# In[38]:
```

```
# Hulu Wordcloud
hulu_plotline_movies_t = hulu_plotline_movies['Plotline']
hulu_movies_plotline_w = ' '.join(hulu_plotline_movies_t)
```

```
# In[39]:
```

```
stopwords = set(STOPWORDS)

wordcloud_hulu_plotline_movies = WordCloud(width = 1000, height = 500,
                                             background_color ='white',
                                             stopwords = stopwords,
                                             min_font_size = 10
                                           ).generate(hulu_movies_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Hulu is : \n')
```

```
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_hulu_plotline_movies)
plt.axis("off")
plt.tight_layout(pad = 0)
```

```
plt.show()
```

```
# In[40]:
```

```
# Prime Video Wordcloud
prime_video_plotline_movies_t = prime_video_plotline_movies['Plotline']
prime_video_movies_plotline_w = ' '.join(prime_video_plotline_movies_t)
```

```
# In[41]:
```

```
stopwords = set(STOPWORDS)

wordcloud_prime_video_plotline_movies = WordCloud(width = 1000, height = 500,
                                                 background_color ='white',
                                                 stopwords = stopwords,
                                                 min_font_size = 10
                                               ).generate(prime_video_movies_plotline_w)
```

```
print('\nThe Wordcloud Generated from Plotlines of Prime Video is : \n')
```

```
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_prime_video_plotline_movies)
plt.axis("off")
plt.tight_layout(pad = 0)
```

```
plt.show()
```

```
# In[42]:
```

```

# Disney+ Wordcloud
disney_plotline_movies_t = disney_plotline_movies['Plotline']
disney_movies_plotline_w = ' '.join(disney_plotline_movies_t)

# In[43]:


stopwords = set(STOPWORDS)

wordcloud_disney_plotline_movies = WordCloud(width = 1000, height = 500,
                                              background_color ='white',
                                              stopwords = stopwords,
                                              min_font_size = 10
                                              ).generate(disney_movies_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Disney+ is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_disney_plotline_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

```

ottmovies_roto.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/rotopy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

```

```

import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_movies = pd.read_csv(path + 'ottmovies.csv')

df_movies.head()

# In[6]:


# profile = ProfileReport(df_movies)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing values %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)

```

```

plt.show()

# In[8]:
data_investigate(df_movies)

# In[9]:

# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

```

```

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('Rotten Tomatoes', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

```

```

# udf_movies

# In[16]:


# df_netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df_hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df_prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df_disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]


# In[17]:


df_netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
                                    (df_movies['Prime Video'] == 0) & (df_movies['Disney+'] == 0)]
df_hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
                                 (df_movies['Prime Video'] == 0) & (df_movies['Disney+'] == 0)]
df_prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
                                         (df_movies['Prime Video'] == 1) & (df_movies['Disney+'] == 0)]
df_disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
                                   (df_movies['Prime Video'] == 0) & (df_movies['Disney+'] == 1)]


# In[18]:


df_movies_roto = df_movies.copy()


# In[19]:


df_movies_roto.drop(df_movies_roto.loc[df_movies_roto['Rotten Tomatoes'] == "NA"].index,
inplace = True)
# df_movies_roto = df_movies_roto[df_movies_roto.Rotten Tomatoes != "NA"]
df_movies_roto['Rotten Tomatoes'] = df_movies_roto['Rotten Tomatoes'].astype(int)


# In[20]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_roto_movies = df_movies_roto.loc[df_movies_roto['Netflix'] == 1]
hulu_roto_movies = df_movies_roto.loc[df_movies_roto['Hulu'] == 1]
prime_video_roto_movies = df_movies_roto.loc[df_movies_roto['Prime Video'] == 1]
disney_roto_movies = df_movies_roto.loc[df_movies_roto['Disney+'] == 1]


# In[21]:


df_movies_roto_group = df_movies_roto.copy()


# In[22]:


plt.figure(figsize = (10, 10))
corr = df_movies_roto.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map

```

```
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# Show plot
plt.show()
fig.show()
```

In[23]:

```
df_roto_high_movies = df_movies_roto.sort_values(by = 'Rotten Tomatoes', ascending =
False).reset_index()
df_roto_high_movies = df_roto_high_movies.drop(['index'], axis = 1)
# filter = (df_movies_roto['Rotten Tomatoes'] == (df_movies_roto['Rotten Tomatoes'].max()))
# df_roto_high_movies = df_movies_roto[filter]

# highestRatedMovies = df_movies_roto.loc[df_movies_roto['Rotten Tomatoes'].idxmax()]

print('\nMovies with Highest Ever Rotten Tomatoes are : \n')
df_roto_high_movies.head(5)
```

In[24]:

```
fig = px.bar(y = df_roto_high_movies['Title'][:15],
              x = df_roto_high_movies['Rotten Tomatoes'][:15],
              color = df_roto_high_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Highest Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[25]:

```
df_roto_low_movies = df_movies_roto.sort_values(by = 'Rotten Tomatoes', ascending =
True).reset_index()
df_roto_low_movies = df_roto_low_movies.drop(['index'], axis = 1)
# filter = (df_movies_roto['Rotten Tomatoes'] == (df_movies_roto['Rotten Tomatoes'].min()))
# df_roto_low_movies = df_movies_roto[filter]

print('\nMovies with Lowest Ever Rotten Tomatoes are : \n')
df_roto_low_movies.head(5)
```

In[26]:

```
fig = px.bar(y = df_roto_low_movies['Title'][:15],
              x = df_roto_low_movies['Rotten Tomatoes'][:15],
              color = df_roto_low_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Lowest Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[27]:
```

```
print(f'''  
    Total '{df_movies_roto['Rotten Tomatoes'].unique().shape[0]}' unique Rotten Tomatoes  
s were Given, They were Like this,\n  
{df_movies_roto.sort_values(by = 'Rotten Tomatoes', ascending = False)['Rotten  
Tomatoes'].unique()}\n  
  
    The Highest Ever Rotten Tomatoes Ever Any Movie Got is  
'{df_roto_high_movies['Title'][0]}' : '{df_roto_high_movies['Rotten Tomatoes'].max()}'\n  
  
    The Lowest Ever Rotten Tomatoes Ever Any Movie Got is  
'{df_roto_low_movies['Title'][0]}' : '{df_roto_low_movies['Rotten Tomatoes'].min()}'\n''')
```

```
# In[28]:
```

```
netflix_roto_high_movies =  
df_roto_high_movies.loc[df_roto_high_movies['Netflix']==1].reset_index()  
netflix_roto_high_movies = netflix_roto_high_movies.drop(['index'], axis = 1)  
  
netflix_roto_low_movies =  
df_roto_low_movies.loc[df_roto_low_movies['Netflix']==1].reset_index()  
netflix_roto_low_movies = netflix_roto_low_movies.drop(['index'], axis = 1)  
  
netflix_roto_high_movies.head(5)
```

```
# In[29]:
```

```
fig = px.bar(y = netflix_roto_high_movies['Title'][:15],  
             x = netflix_roto_high_movies['Rotten Tomatoes'][:15],  
             color = netflix_roto_high_movies['Rotten Tomatoes'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating' },  
             title = 'Movies with Highest Rotten Tomatoes : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[30]:
```

```
fig = px.bar(y = netflix_roto_low_movies['Title'][:15],  
             x = netflix_roto_low_movies['Rotten Tomatoes'][:15],  
             color = netflix_roto_low_movies['Rotten Tomatoes'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating' },  
             title = 'Movies with Lowest Rotten Tomatoes : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[31]:
```

```
hulu_roto_high_movies =  
df_roto_high_movies.loc[df_roto_high_movies['Hulu']==1].reset_index()  
hulu_roto_high_movies = hulu_roto_high_movies.drop(['index'], axis = 1)
```

```
hulu_roto_low_movies = df_roto_low_movies.loc[df_roto_low_movies['Hulu']==1].reset_index()
hulu_roto_low_movies = hulu_roto_low_movies.drop(['index'], axis = 1)
```

```
hulu_roto_high_movies.head(5)
```

```
# In[32]:
```

```
fig = px.bar(y = hulu_roto_high_movies['Title'][:15],
              x = hulu_roto_high_movies['Rotten Tomatoes'][:15],
              color = hulu_roto_high_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Highest Rotten Tomatoes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:
```

```
fig = px.bar(y = hulu_roto_low_movies['Title'][:15],
              x = hulu_roto_low_movies['Rotten Tomatoes'][:15],
              color = hulu_roto_low_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Lowest Rotten Tomatoes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[34]:
```

```
prime_video_roto_high_movies = df_roto_high_movies.loc[df_roto_high_movies['Prime
Video']==1].reset_index()
prime_video_roto_high_movies = prime_video_roto_high_movies.drop(['index'], axis = 1)

prime_video_roto_low_movies = df_roto_low_movies.loc[df_roto_low_movies['Prime
Video']==1].reset_index()
prime_video_roto_low_movies = prime_video_roto_low_movies.drop(['index'], axis = 1)

prime_video_roto_high_movies.head(5)
```

```
# In[35]:
```

```
fig = px.bar(y = prime_video_roto_high_movies['Title'][:15],
              x = prime_video_roto_high_movies['Rotten Tomatoes'][:15],
              color = prime_video_roto_high_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Highest Rotten Tomatoes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[36]:
```

```

fig = px.bar(y = prime_video_roto_low_movies['Title'][:15],
              x = prime_video_roto_low_movies['Rotten Tomatoes'][:15],
              color = prime_video_roto_low_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Lowest Rotten Tomatoes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[37]:

```

disney_roto_high_movies =
df_roto_high_movies.loc[df_roto_high_movies['Disney+']==1].reset_index()
disney_roto_high_movies = disney_roto_high_movies.drop(['index'], axis = 1)

disney_roto_low_movies =
df_roto_low_movies.loc[df_roto_low_movies['Disney+']==1].reset_index()
disney_roto_low_movies = disney_roto_low_movies.drop(['index'], axis = 1)

disney_roto_high_movies.head(5)

```

In[38]:

```

fig = px.bar(y = disney_roto_high_movies['Title'][:15],
              x = disney_roto_high_movies['Rotten Tomatoes'][:15],
              color = disney_roto_high_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Highest Rotten Tomatoes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[39]:

```

fig = px.bar(y = disney_roto_low_movies['Title'][:15],
              x = disney_roto_low_movies['Rotten Tomatoes'][:15],
              color = disney_roto_low_movies['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Lowest Rotten Tomatoes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[40]:

```

print(f'''
    The Movie with Highest Rotten Tomatoes Ever Got is
'{df_roto_high_movies['Title'][0]}' : '{df_roto_high_movies['Rotten Tomatoes'].max()}'\n
    The Movie with Lowest Rotten Tomatoes Ever Got is '{df_roto_low_movies['Title'][0]}'
: '{df_roto_low_movies['Rotten Tomatoes'].min()}'\n

    The Movie with Highest Rotten Tomatoes on 'Netflix' is
'{netflix_roto_high_movies['Title'][0]}' : '{netflix_roto_high_movies['Rotten Tomatoes'].max()}'\n

```

```

    The Movie with Lowest Rotten Tomatoes on 'Netflix' is
'{netflix_roto_low_movies['Title'][0]} : '{netflix_roto_low_movies['Rotten
Tomatoes'].min()}'\n

    The Movie with Highest Rotten Tomatoes on 'Hulu' is
'{hulu_roto_high_movies['Title'][0]} : '{hulu_roto_high_movies['Rotten
Tomatoes'].max()}'\n
    The Movie with Lowest Rotten Tomatoes on 'Hulu' is
'{hulu_roto_low_movies['Title'][0]} : '{hulu_roto_low_movies['Rotten Tomatoes'].min()}'\n

    The Movie with Highest Rotten Tomatoes on 'Prime Video' is
'{prime_video_roto_high_movies['Title'][0]} : '{prime_video_roto_high_movies['Rotten
Tomatoes'].max()}'\n
    The Movie with Lowest Rotten Tomatoes on 'Prime Video' is
'{prime_video_roto_low_movies['Title'][0]} : '{prime_video_roto_low_movies['Rotten
Tomatoes'].min()}'\n

    The Movie with Highest Rotten Tomatoes on 'Disney+' is
'{disney_roto_high_movies['Title'][0]} : '{disney_roto_high_movies['Rotten
Tomatoes'].max()}'\n
    The Movie with Lowest Rotten Tomatoes on 'Disney+' is
'{disney_roto_low_movies['Title'][0]} : '{disney_roto_low_movies['Rotten
Tomatoes'].min()}'\n
    ''')

```

In[41]:

```

print(f'''
    Across All Platforms the Average Rotten Tomatoes is '{round(df_movies_roto['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
    The Average Rotten Tomatoes on 'Netflix' is '{round(netflix_roto_movies['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
    The Average Rotten Tomatoes on 'Hulu' is '{round(hulu_roto_movies['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
    The Average Rotten Tomatoes on 'Prime Video' is
'{round(prime_video_roto_movies['Rotten Tomatoes'].mean(), ndigits = 2)}'\n
    The Average Rotten Tomatoes on 'Disney+' is '{round(disney_roto_movies['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
    ''')

```

In[42]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_roto['Rotten Tomatoes'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_movies_roto['Rotten Tomatoes'], ax = ax[1])
plt.show()

```

In[43]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Rotten Tomatoes s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_roto_movies['Rotten Tomatoes'][:100], color = 'lightblue', legend
= True, kde = True)
sns.histplot(netflix_roto_movies['Rotten Tomatoes'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_roto_movies['Rotten Tomatoes'][:100], color = 'lightgreen', legend =
True, kde = True)

```

```
sns.histplot(disney_roto_movies['Rotten Tomatoes'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

In[44]:

```
def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(0,11):
        # print(data)
        return 10
    if data in range(11,21):
        return 20
    if data in range(21,31):
        return 30
    if data in range(31,41):
        return 40
    if data in range(41,51):
        return 50
    if data in range(51,61):
        return 60
    if data in range(61,71):
        return 70
    if data in range(71,81):
        return 80
    if data in range(81,91):
        return 90
    if data in range(91,101):
        return 100
```

In[45]:

```
df_movies_roto_group['Rotten Tomatoes Group'] = df_movies_roto['Rotten
Tomatoes'].apply(round_fix)

roto_values = df_movies_roto_group['Rotten Tomatoes
Group'].value_counts().sort_index(ascending = False).tolist()
roto_index = df_movies_roto_group['Rotten Tomatoes
Group'].value_counts().sort_index(ascending = False).index

# rotov_values, rotov_index
```

In[46]:

```
rotogroup_count = df_movies_roto_group.groupby('Rotten Tomatoes Group')['Title'].count()
rotogroup_movies = df_movies_roto_group.groupby('Rotten Tomatoes Group')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
rotogroup_data_movies = pd.concat([rotogroup_count, rotogroup_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
rotogroup_data_movies = rotogroup_data_movies.sort_values(by = 'Movies Count', ascending
= False)
```

In[47]:

```

# Rotten Tomatoes Group with Movies Counts - All Platforms Combined
roto_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[48]: 

roto_group_data_movies.sort_values(by = 'Rotten Tomatoes Group', ascending = False)

# In[49]: 

fig = px.bar(y = rotogroup_data_movies['Movies Count'],
              x = rotogroup_data_movies['Rotten Tomatoes Group'],
              color = rotogroup_data_movies['Rotten Tomatoes Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'Movies with Group Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]: 

fig = px.pie(rotogroup_data_movies[:10],
              names = rotogroup_data_movies['Rotten Tomatoes Group'],
              values = rotogroup_data_movies['Movies Count'],
              color = rotogroup_data_movies['Movies Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on Rotten Tomatoes Group')
fig.show()

# In[51]: 

df_rotogroup_high_movies = rotogroup_data_movies.sort_values(by = 'Movies Count',
                                                             ascending = False).reset_index()
df_rotogroup_high_movies = df_rotogroup_high_movies.drop(['index'], axis = 1)
# filter = (rotogroup_data_movies['Movies Count'] == (rotogroup_data_movies['Movies Count'].max()))
# df_rotogroup_high_movies = rotogroup_data_movies[filter]

# highestRatedMovies = rotogroup_data_movies.loc[rotogroup_data_movies['Movies Count'].idxmax()]

# print('\nRotten Tomatoes with Highest Ever Movies Count are : All Platforms Combined\n')
df_rotogroup_high_movies.head(5)

# In[52]: 

df_rotogroup_low_movies = rotogroup_data_movies.sort_values(by = 'Movies Count',
                                                             ascending = True).reset_index()
df_rotogroup_low_movies = df_rotogroup_low_movies.drop(['index'], axis = 1)
# filter = (rotogroup_data_movies['Movies Count'] == (rotogroup_data_movies['Movies Count'].min()))
# df_rotogroup_low_movies = rotogroup_data_movies[filter]

```

```
# print('\nRotten Tomatoes with Lowest Ever Movies Count are : All Platforms Combined\n')
df_roto_group_low_movies.head(5)
```

```
# In[53]:
```

```
print(f'''
    Total '{df_movies_roto['Rotten Tomatoes'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see Movies from Total '{roto_group_data_movies['Rotten Tomatoes
Group'].unique().shape[0]}' Rotten Tomatoes Group, They were Like this, \n
        {roto_group_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Rotten
Tomatoes Group']].unique()} etc. \n
        The Rotten Tomatoes Group with Highest Movies Count have
'{roto_group_data_movies['Movies Count'].max()}' Movies Available is
'{df_roto_group_high_movies['Rotten Tomatoes Group'][0]}', &\n
        The Rotten Tomatoes Group with Lowest Movies Count have
'{roto_group_data_movies['Movies Count'].min()}' Movies Available is
'{df_roto_group_low_movies['Rotten Tomatoes Group'][0]}'
        ''')
```

```
# In[54]:
```

```
netflix_roto_group_movies = roto_group_data_movies[roto_group_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_roto_group_movies = netflix_roto_group_movies.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

netflix_roto_group_high_movies = df_roto_group_high_movies.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_roto_group_high_movies = netflix_roto_group_high_movies.drop(['index'], axis = 1)

netflix_roto_group_low_movies = df_roto_group_low_movies.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_roto_group_low_movies = netflix_roto_group_low_movies.drop(['index'], axis = 1)

netflix_roto_group_high_movies.head(5)
```

```
# In[55]:
```

```
hulu_roto_group_movies = roto_group_data_movies[roto_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_roto_group_movies = hulu_roto_group_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_roto_group_high_movies = df_roto_group_high_movies.sort_values(by = 'Hulu', ascending =
False).reset_index()
hulu_roto_group_high_movies = hulu_roto_group_high_movies.drop(['index'], axis = 1)

hulu_roto_group_low_movies = df_roto_group_low_movies.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_roto_group_low_movies = hulu_roto_group_low_movies.drop(['index'], axis = 1)

hulu_roto_group_high_movies.head(5)
```

```
# In[56]:
```

```

prime_video_roto_group_movies = roto_group_data_movies[roto_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_roto_group_movies = prime_video_roto_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_roto_group_high_movies = df_roto_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_roto_group_high_movies = prime_video_roto_group_high_movies.drop(['index'], axis = 1)

prime_video_roto_group_low_movies = df_roto_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_roto_group_low_movies = prime_video_roto_group_low_movies.drop(['index'], axis = 1)

prime_video_roto_group_high_movies.head(5)

```

In[57]:

```

disney_roto_group_movies = roto_group_data_movies[roto_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_roto_group_movies = disney_roto_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_roto_group_high_movies = df_roto_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_roto_group_high_movies = disney_roto_group_high_movies.drop(['index'], axis = 1)

disney_roto_group_low_movies = df_roto_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_roto_group_low_movies = disney_roto_group_low_movies.drop(['index'], axis = 1)

disney_roto_group_high_movies.head(5)

```

In[58]:

```

print(f'''
    The Rotten Tomatoes Group with Highest Movies Count Ever Got is
'{df_roto_group_high_movies['Rotten Tomatoes Group'][0]}' :
'{df_roto_group_high_movies['Movies Count'].max()}'\n
    The Rotten Tomatoes Group with Lowest Movies Count Ever Got is
'{df_roto_group_low_movies['Rotten Tomatoes Group'][0]}' :
'{df_roto_group_low_movies['Movies Count'].min()}'\n

    The Rotten Tomatoes Group with Highest Movies Count on 'Netflix' is
'{netflix_roto_group_high_movies['Rotten Tomatoes Group'][0]}' :
'{netflix_roto_group_high_movies['Netflix'].max()}'\n
    The Rotten Tomatoes Group with Lowest Movies Count on 'Netflix' is
'{netflix_roto_group_low_movies['Rotten Tomatoes Group'][0]}' :
'{netflix_roto_group_low_movies['Netflix'].min()}'\n

    The Rotten Tomatoes Group with Highest Movies Count on 'Hulu' is
'{hulu_roto_group_high_movies['Rotten Tomatoes Group'][0]}' :
'{hulu_roto_group_high_movies['Hulu'].max()}'\n
    The Rotten Tomatoes Group with Lowest Movies Count on 'Hulu' is
'{hulu_roto_group_low_movies['Rotten Tomatoes Group'][0]}' :
'{hulu_roto_group_low_movies['Hulu'].min()}'\n

    The Rotten Tomatoes Group with Highest Movies Count on 'Prime Video' is
'{prime_video_roto_group_high_movies['Rotten Tomatoes Group'][0]}' :
'{prime_video_roto_group_high_movies['Prime Video'].max()}'\n

```

```

The Rotten Tomatoes Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_roto_group_low_movies['Rotten Tomatoes Group'][0]} :
'{prime_video_roto_group_low_movies['Prime Video'].min()}'\n

The Rotten Tomatoes Group with Highest Movies Count on 'Disney+' is
'{disney_roto_group_high_movies['Rotten Tomatoes Group'][0]} :
'{disney_roto_group_high_movies['Disney+'].max()}'\n
The Rotten Tomatoes Group with Lowest Movies Count on 'Disney+' is
'{disney_roto_group_low_movies['Rotten Tomatoes Group'][0]} :
'{disney_roto_group_low_movies['Disney+'].min()}'\n
''')

# In[59]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.barplot(x = netflix_roto_group_movies['Rotten Tomatoes Group'][:10], y =
netflix_roto_group_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ro_ax2 = sns.barplot(x = hulu_roto_group_movies['Rotten Tomatoes Group'][:10], y =
hulu_roto_group_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ro_ax3 = sns.barplot(x = prime_video_roto_group_movies['Rotten Tomatoes Group'][:10], y =
prime_video_roto_group_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ro_ax4 = sns.barplot(x = disney_roto_group_movies['Rotten Tomatoes Group'][:10], y =
disney_roto_group_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()

# In[60]:


plt.figure(figsize = (20, 5))
sns.lineplot(x = roto_group_data_movies['Rotten Tomatoes Group'], y =
roto_group_data_movies['Netflix'], color = 'red')
sns.lineplot(x = roto_group_data_movies['Rotten Tomatoes Group'], y =
roto_group_data_movies['Hulu'], color = 'lightgreen')
sns.lineplot(x = roto_group_data_movies['Rotten Tomatoes Group'], y =
roto_group_data_movies['Prime Video'], color = 'lightblue')
sns.lineplot(x = roto_group_data_movies['Rotten Tomatoes Group'], y =
roto_group_data_movies['Disney+'], color = 'darkblue')
plt.xlabel('Rotten Tomatoes Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

# In[61]:


print(f'''
    Accross All Platforms Total Count of Rotten Tomatoes Group is
'{roto_group_data_movies['Rotten Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Netflix' is
'{netflix_roto_group_movies['Rotten Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Hulu' is '{hulu_roto_group_movies['Rotten
Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Prime Video' is
'{prime_video_roto_group_movies['Rotten Tomatoes Group'].unique().shape[0]}'\n

```

```
Total Count of Rotten Tomatoes Group on 'Disney+' is
'{disney_roto_group_movies['Rotten Tomatoes Group'].unique().shape[0]}'\n
''')
```

```
# In[62]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.lineplot(y = roto_group_data_movies['Rotten Tomatoes Group'], x =
roto_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_ro_ax2 = sns.lineplot(y = roto_group_data_movies['Rotten Tomatoes Group'], x =
roto_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ro_ax3 = sns.lineplot(y = roto_group_data_movies['Rotten Tomatoes Group'], x =
roto_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ro_ax4 = sns.lineplot(y = roto_group_data_movies['Rotten Tomatoes Group'], x =
roto_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[63]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.barplot(x = roto_group_data_movies['Rotten Tomatoes Group'][:10], y =
roto_group_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ro_ax2 = sns.barplot(x = roto_group_data_movies['Rotten Tomatoes Group'][:10], y =
roto_group_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ro_ax3 = sns.barplot(x = roto_group_data_movies['Rotten Tomatoes Group'][:10], y =
roto_group_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ro_ax4 = sns.barplot(x = roto_group_data_movies['Rotten Tomatoes Group'][:10], y =
roto_group_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()
```

ottmovies_runtim.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
```

```
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_movies = pd.read_csv(path + 'ottmovies.csv')

df_movies.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_movies)
```

```

# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

```

```
# In[8]:
```

```
data_investigate(df_movies)
```

```
# In[9]:
```

```

# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)

```

```

df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:

```

```

df_movies.describe()

# In[13]:

df_movies.corr()

# In[14]:

# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:

# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:

# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:

df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]

# In[18]:

df_movies_runtimes = df_movies.copy()

# In[19]:

df_movies_runtimes.drop(df_movies_runtimes.loc[df_movies_runtimes['Runtime'] ==
"NA"].index, inplace = True)
# df_movies_runtimes = df_movies_runtimes[df_movies_runtimes.Runtime != "NA"]
df_movies_runtimes['Runtime'] = df_movies_runtimes['Runtime'].astype(int)

```

```

# In[20]:
# Creating distinct dataframes only with the movies present on individual streaming
# platforms
netflix_runtimes_movies = df_movies_runtimes.loc[df_movies_runtimes['Netflix'] == 1]
hulu_runtimes_movies = df_movies_runtimes.loc[df_movies_runtimes['Hulu'] == 1]
prime_video_runtimes_movies = df_movies_runtimes.loc[df_movies_runtimes['Prime Video'] ==
1]
disney_runtimes_movies = df_movies_runtimes.loc[df_movies_runtimes['Disney+'] == 1]

# In[21]:
df_movies_runtimes_group = df_movies_runtimes.copy()

# In[22]:
df_movies_screentimes = df_movies_runtimes.copy()
df_movies_screentimes['Screentime'] = round(df_movies_runtimes['Runtime']/60, ndigits = 2)

# In[23]:
# Creating distinct dataframes only with the movies present on individual streaming
# platforms
netflix_screentimes_movies = df_movies_screentimes.loc[df_movies_screentimes['Netflix'] == 1]
hulu_screentimes_movies = df_movies_screentimes.loc[df_movies_screentimes['Hulu'] == 1]
prime_video_screentimes_movies = df_movies_screentimes.loc[df_movies_screentimes['Prime
Video'] == 1]
disney_screentimes_movies = df_movies_screentimes.loc[df_movies_screentimes['Disney+'] == 1]

# In[24]:
plt.figure(figsize = (10, 10))
corr = df_movies_runtimes.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[25]:
df_runtimes_high_movies = df_movies_runtimes.sort_values(by = 'Runtime', ascending =
False).reset_index()
df_runtimes_high_movies = df_runtimes_high_movies.drop(['index'], axis = 1)
# filter = (df_movies_runtimes['Runtime'] == (df_movies_runtimes['Runtime'].max()))
# df_runtimes_high_movies = df_movies_runtimes[filter]

```

```
# highestRatedMovies = df_movies_runtimes.loc[df_movies_runtimes['Runtime'].idxmax()]

print('\nMovies with Highest Ever Runtime are : \n')
df_runtimes_high_movies.head(5)
```

```
# In[26]:
```

```
fig = px.bar(y = df_runtimes_high_movies['Title'][:15],
              x = df_runtimes_high_movies['Runtime'][:15],
              color = df_runtimes_high_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Highest Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[27]:
```

```
df_runtimes_low_movies = df_movies_runtimes.sort_values(by = 'Runtime', ascending = True).reset_index()
df_runtimes_low_movies = df_runtimes_low_movies.drop(['index'], axis = 1)
# filter = (df_movies_runtimes['Runtime'] == (df_movies_runtimes['Runtime'].min()))
# df_runtimes_low_movies = df_movies_runtimes[filter]

print('\nMovies with Lowest Ever Runtime are : \n')
df_runtimes_low_movies.head(5)
```

```
# In[28]:
```

```
fig = px.bar(y = df_runtimes_low_movies['Title'][:15],
              x = df_runtimes_low_movies['Runtime'][:15],
              color = df_runtimes_low_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Lowest Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[29]:
```

```
print(f'''
    Total '{df_movies_runtimes['Runtime'].unique().shape[0]}' unique Runtime s were
Given, They were Like this,\n

{df_movies_runtimes.sort_values(by = 'Runtime', ascending = False)['Runtime'].unique()}\n

    The Highest Ever Runtime Ever Any Movie Got is
'{df_runtimes_high_movies['Title'][0]} : '{df_runtimes_high_movies['Runtime'].max()}'\n

    The Lowest Ever Runtime Ever Any Movie Got is '{df_runtimes_low_movies['Title'][0]}'
: '{df_runtimes_low_movies['Runtime'].min()}'\n'''')
```

```
# In[30]:
```

```

netflix_runtimes_high_movies =
df_runtimes_high_movies.loc[df_runtimes_high_movies['Netflix']==1].reset_index()
netflix_runtimes_high_movies = netflix_runtimes_high_movies.drop(['index'], axis = 1)

netflix_runtimes_low_movies =
df_runtimes_low_movies.loc[df_runtimes_low_movies['Netflix']==1].reset_index()
netflix_runtimes_low_movies = netflix_runtimes_low_movies.drop(['index'], axis = 1)

netflix_runtimes_high_movies.head(5)

# In[31]:


fig = px.bar(y = netflix_runtimes_high_movies['Title'][:15],
              x = netflix_runtimes_high_movies['Runtime'][:15],
              color = netflix_runtimes_high_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Highest Runtime in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[32]:


fig = px.bar(y = netflix_runtimes_low_movies['Title'][:15],
              x = netflix_runtimes_low_movies['Runtime'][:15],
              color = netflix_runtimes_low_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Lowest Runtime in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


hulu_runtimes_high_movies =
df_runtimes_high_movies.loc[df_runtimes_high_movies['Hulu']==1].reset_index()
hulu_runtimes_high_movies = hulu_runtimes_high_movies.drop(['index'], axis = 1)

hulu_runtimes_low_movies =
df_runtimes_low_movies.loc[df_runtimes_low_movies['Hulu']==1].reset_index()
hulu_runtimes_low_movies = hulu_runtimes_low_movies.drop(['index'], axis = 1)

hulu_runtimes_high_movies.head(5)

# In[34]:


fig = px.bar(y = hulu_runtimes_high_movies['Title'][:15],
              x = hulu_runtimes_high_movies['Runtime'][:15],
              color = hulu_runtimes_high_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Highest Runtime in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[35]:
```

```
fig = px.bar(y = hulu_runtimes_low_movies['Title'][:15],
              x = hulu_runtimes_low_movies['Runtime'][:15],
              color = hulu_runtimes_low_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Lowest Runtime in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[36]:
```

```
prime_video_runtimes_high_movies =
df_runtimes_high_movies.loc[df_runtimes_high_movies['Prime Video']==1].reset_index()
prime_video_runtimes_high_movies = prime_video_runtimes_high_movies.drop(['index'], axis = 1)

prime_video_runtimes_low_movies = df_runtimes_low_movies.loc[df_runtimes_low_movies['Prime Video']==1].reset_index()
prime_video_runtimes_low_movies = prime_video_runtimes_low_movies.drop(['index'], axis = 1)

prime_video_runtimes_high_movies.head(5)
```

```
# In[37]:
```

```
fig = px.bar(y = prime_video_runtimes_high_movies['Title'][:15],
              x = prime_video_runtimes_high_movies['Runtime'][:15],
              color = prime_video_runtimes_high_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Highest Runtime in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[38]:
```

```
fig = px.bar(y = prime_video_runtimes_low_movies['Title'][:15],
              x = prime_video_runtimes_low_movies['Runtime'][:15],
              color = prime_video_runtimes_low_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Lowest Runtime in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[39]:
```

```
disney_runtimes_high_movies =
df_runtimes_high_movies.loc[df_runtimes_high_movies['Disney+']==1].reset_index()
disney_runtimes_high_movies = disney_runtimes_high_movies.drop(['index'], axis = 1)
```

```

disney_runtimes_low_movies =
df_runtimes_low_movies.loc[df_runtimes_low_movies['Disney+']==1].reset_index()
disney_runtimes_low_movies = disney_runtimes_low_movies.drop(['index'], axis = 1)

disney_runtimes_high_movies.head(5)

# In[40]:


fig = px.bar(y = disney_runtimes_high_movies['Title'][:15],
              x = disney_runtimes_high_movies['Runtime'][:15],
              color = disney_runtimes_high_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Highest Runtime in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[41]:


fig = px.bar(y = disney_runtimes_low_movies['Title'][:15],
              x = disney_runtimes_low_movies['Runtime'][:15],
              color = disney_runtimes_low_movies['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Lowest Runtime in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


print(f'''The Movie with Highest Runtime Ever Got is '{df_runtimes_high_movies['Title'][0]}':\n'{df_runtimes_high_movies['Runtime'].max()}'\nThe Movie with Lowest Runtime Ever Got is '{df_runtimes_low_movies['Title'][0]}':\n'{df_runtimes_low_movies['Runtime'].min()}'\n\nThe Movie with Highest Runtime on 'Netflix' is\n'{netflix_runtimes_high_movies['Title'][0]}':\n'{netflix_runtimes_high_movies['Runtime'].max()}'\nThe Movie with Lowest Runtime on 'Netflix' is\n'{netflix_runtimes_low_movies['Title'][0]}':\n'{netflix_runtimes_low_movies['Runtime'].min()}'\n\nThe Movie with Highest Runtime on 'Hulu' is\n'{hulu_runtimes_high_movies['Title'][0]}':\n'{hulu_runtimes_high_movies['Runtime'].max()}'\nThe Movie with Lowest Runtime on 'Hulu' is '{hulu_runtimes_low_movies['Title'][0]}':\n'{hulu_runtimes_low_movies['Runtime'].min()}'\n\nThe Movie with Highest Runtime on 'Prime Video' is\n'{prime_video_runtimes_high_movies['Title'][0]}':\n'{prime_video_runtimes_high_movies['Runtime'].max()}'\nThe Movie with Lowest Runtime on 'Prime Video' is\n'{prime_video_runtimes_low_movies['Title'][0]}':\n'{prime_video_runtimes_low_movies['Runtime'].min()}'\n

```

```

        The Movie with Highest Runtime on 'Disney+' is
'{disney_runtimes_high_movies['Title'][0]}' :
'{disney_runtimes_high_movies['Runtime'].max()}'\n
        The Movie with Lowest Runtime on 'Disney+' is
'{disney_runtimes_low_movies['Title'][0]}' :
'{disney_runtimes_low_movies['Runtime'].min()}'\n
        ''')

# In[43]:
```

```

print(f'''
    Across All Platforms the Average Runtime is
'{round(df_movies_runtimes['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Netflix' is
'{round(netflix_runtimes_movies['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Hulu' is '{round(hulu_runtimes_movies['Runtime'].mean(),
ndigits = 2)}'\n
    The Average Runtime on 'Prime Video' is
'{round(prime_video_runtimes_movies['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Disney+' is
'{round(disney_runtimes_movies['Runtime'].mean(), ndigits = 2)}'\n
        ''')
```

```
# In[44]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_runtimes['Runtime'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_movies_runtimes['Runtime'], ax = ax[1])
plt.show()
```

```
# In[45]:
```

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Runtime s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_runtimes_movies['Runtime'][:100], color = 'lightblue', legend =
True, kde = True)
sns.histplot(netflix_runtimes_movies['Runtime'][:100], color = 'red', legend = True, kde =
True)
sns.histplot(hulu_runtimes_movies['Runtime'][:100], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_runtimes_movies['Runtime'][:100], color = 'darkblue', legend = True,
kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

```
# In[46]:
```

```

def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(0,51):
```

```

# print(data)
return 50
if data in range(51,101):
    return 100
if data in range(101,151):
    return 150
if data in range(151,201):
    return 200
if data in range(201,251):
    return 250
if data in range(251,301):
    return 300
if data in range(301,351):
    return 350
if data in range(351,401):
    return 400
if data in range(401,451):
    return 450
if data in range(451,501):
    return 500
if data in range(501,551):
    return 550
if data in range(551,601):
    return 600
if data in range(601,651):
    return 650
if data in range(651,701):
    return 700
if data in range(701,751):
    return 750
if data in range(751,801):
    return 800
if data in range(801,851):
    return 850
if data in range(851,901):
    return 900
if data in range(901,951):
    return 950
if data in range(951,1001):
    return 1000
if data in range(1001,1051):
    return 1050
if data in range(1051,1101):
    return 1100
if data in range(1101,1151):
    return 1150
if data in range(1151,1201):
    return 1200
if data in range(1201,1251):
    return 1250
if data in range(1251,1301):
    return 1300
if data in range(1301,1351):
    return 1350
if data in range(1351,2001):
    return 2000

```

In[47]:

```

df_movies_runtimes_group['Runtime Group'] = df_movies_runtimes['Runtime'].apply(round_fix)

runtimes_values = df_movies_runtimes_group['Runtime Group'].value_counts().sort_index(ascending = False).tolist()

```

```

runtimes_index = df_movies_runtimes_group['Runtime Group'].value_counts().sort_index(ascending = False).index

# runtimes_values, runtimes_index

# In[48]:


runtimes_group_count = df_movies_runtimes_group.groupby('Runtime Group')['Title'].count()
runtimes_group_movies = df_movies_runtimes_group.groupby('Runtime Group')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
runtimes_group_data_movies = pd.concat([runtimes_group_count, runtimes_group_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies Count'})
runtimes_group_data_movies = runtimes_group_data_movies.sort_values(by = 'Movies Count',
ascending = False)

# In[49]:


# Runtime Group with Movies Counts - All Platforms Combined
runtimes_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[50]:


runtimes_group_data_movies.sort_values(by = 'Runtime Group', ascending = False)

# In[51]:


fig = px.bar(y = runtimes_group_data_movies['Movies Count'],
              x = runtimes_group_data_movies['Runtime Group'],
              color = runtimes_group_data_movies['Runtime Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Runtime : In Minutes'},
              title = 'Movies with Group Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[52]:


fig = px.pie(runtimes_group_data_movies[:10],
              names = runtimes_group_data_movies['Runtime Group'],
              values = runtimes_group_data_movies['Movies Count'],
              color = runtimes_group_data_movies['Movies Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on Runtime Group')
fig.show()

# In[53]:


df_runtimes_group_high_movies = runtimes_group_data_movies.sort_values(by = 'Movies Count',
ascending = False).reset_index()
df_runtimes_group_high_movies = df_runtimes_group_high_movies.drop(['index'], axis = 1)

```

```

# filter = (runtimes_group_data_movies['Movies Count'] ==
(runtimes_group_data_movies['Movies Count'].max()))
# df_runtimes_group_high_movies = runtimes_group_data_movies[filter]

# highestRatedMovies = runtimes_group_data_movies.loc[runtimes_group_data_movies['Movies
Count'].idxmax()]

# print('\nRuntime with Highest Ever Movies Count are : All Platforms Combined\n')
df_runtimes_group_high_movies.head(5)

# In[54]:


df_runtimes_group_low_movies = runtimes_group_data_movies.sort_values(by = 'Movies Count',
ascending = True).reset_index()
df_runtimes_group_low_movies = df_runtimes_group_low_movies.drop(['index'], axis = 1)
# filter = (runtimes_group_data_movies['Movies Count'] == 
(runtimes_group_data_movies['Movies Count'].min()))
# df_runtimes_group_low_movies = runtimes_group_data_movies[filter]

# print('\nRuntime with Lowest Ever Movies Count are : All Platforms Combined\n')
df_runtimes_group_low_movies.head(5)

# In[55]:


print(f'''
    Total '{df_movies_runtimes['Runtime'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see Movies from Total '{runtimes_group_data_movies['Runtime
Group'].unique().shape[0]}' Runtime Group, They were Like this, \n
    {runtimes_group_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Runtime Group'].unique()} etc. \n
    The Runtime Group with Highest Movies Count have '{runtimes_group_data_movies['Movies
Count'].max()}' Movies Available is '{df_runtimes_group_high_movies['Runtime Group'][0]}',
&\n
    The Runtime Group with Lowest Movies Count have '{runtimes_group_data_movies['Movies
Count'].min()}' Movies Available is '{df_runtimes_group_low_movies['Runtime Group'][0]}'
''')


# In[56]:


netflix_runtimes_group_movies =
runtimes_group_data_movies[runtimes_group_data_movies['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_runtimes_group_movies = netflix_runtimes_group_movies.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_runtimes_group_high_movies = df_runtimes_group_high_movies.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_runtimes_group_high_movies = netflix_runtimes_group_high_movies.drop(['index'],
axis = 1)

netflix_runtimes_group_low_movies = df_runtimes_group_low_movies.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_runtimes_group_low_movies = netflix_runtimes_group_low_movies.drop(['index'], axis
= 1)

netflix_runtimes_group_high_movies.head(5)

```

```
# In[57]:
```

```

hulu_runtimes_group_movies = runtimes_group_data_movies[runtimes_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_runtimes_group_movies = hulu_runtimes_group_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_runtimes_group_high_movies = df_runtimes_group_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_runtimes_group_high_movies = hulu_runtimes_group_high_movies.drop(['index'], axis = 1)

hulu_runtimes_group_low_movies = df_runtimes_group_high_movies.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_runtimes_group_low_movies = hulu_runtimes_group_low_movies.drop(['index'], axis = 1)

hulu_runtimes_group_high_movies.head(5)

```

```
# In[58]:
```

```

prime_video_runtimes_group_movies =
runtimes_group_data_movies[runtimes_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_runtimes_group_movies = prime_video_runtimes_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_runtimes_group_high_movies = df_runtimes_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_runtimes_group_high_movies =
prime_video_runtimes_group_high_movies.drop(['index'], axis = 1)

prime_video_runtimes_group_low_movies = df_runtimes_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_runtimes_group_low_movies =
prime_video_runtimes_group_low_movies.drop(['index'], axis = 1)

prime_video_runtimes_group_high_movies.head(5)

```

```
# In[59]:
```

```

disney_runtimes_group_movies =
runtimes_group_data_movies[runtimes_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_runtimes_group_movies = disney_runtimes_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_runtimes_group_high_movies = df_runtimes_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_runtimes_group_high_movies = disney_runtimes_group_high_movies.drop(['index'], axis = 1)

disney_runtimes_group_low_movies = df_runtimes_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_runtimes_group_low_movies = disney_runtimes_group_low_movies.drop(['index'], axis = 1)

disney_runtimes_group_high_movies.head(5)

```

```
# In[60]:
```

```

print(f'''
    The Runtime Group with Highest Movies Count Ever Got is
'{df_runtimes_group_high_movies['Runtime Group'][0]}' :
'{df_runtimes_group_high_movies['Movies Count'].max()}'\n
    The Runtime Group with Lowest Movies Count Ever Got is
'{df_runtimes_group_low_movies['Runtime Group'][0]}' :
'{df_runtimes_group_low_movies['Movies Count'].min()}'\n

    The Runtime Group with Highest Movies Count on 'Netflix' is
'{netflix_runtimes_group_high_movies['Runtime Group'][0]}' :
'{netflix_runtimes_group_high_movies['Netflix'].max()}'\n
    The Runtime Group with Lowest Movies Count on 'Netflix' is
'{netflix_runtimes_group_low_movies['Runtime Group'][0]}' :
'{netflix_runtimes_group_low_movies['Netflix'].min()}'\n

    The Runtime Group with Highest Movies Count on 'Hulu' is
'{hulu_runtimes_group_high_movies['Runtime Group'][0]}' :
'{hulu_runtimes_group_high_movies['Hulu'].max()}'\n
    The Runtime Group with Lowest Movies Count on 'Hulu' is
'{hulu_runtimes_group_low_movies['Runtime Group'][0]}' :
'{hulu_runtimes_group_low_movies['Hulu'].min()}'\n

    The Runtime Group with Highest Movies Count on 'Prime Video' is
'{prime_video_runtimes_group_high_movies['Runtime Group'][0]}' :
'{prime_video_runtimes_group_high_movies['Prime Video'].max()}'\n
    The Runtime Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_runtimes_group_low_movies['Runtime Group'][0]}' :
'{prime_video_runtimes_group_low_movies['Prime Video'].min()}'\n

    The Runtime Group with Highest Movies Count on 'Disney+' is
'{disney_runtimes_group_high_movies['Runtime Group'][0]}' :
'{disney_runtimes_group_high_movies['Disney+'].max()}'\n
    The Runtime Group with Lowest Movies Count on 'Disney+' is
'{disney_runtimes_group_low_movies['Runtime Group'][0]}' :
'{disney_runtimes_group_low_movies['Disney+'].min()}'\n
    ''')

```

In[61]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = netflix_runtimes_group_movies['Runtime Group'][:10], y =
netflix_runtimes_group_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = hulu_runtimes_group_movies['Runtime Group'][:10], y =
hulu_runtimes_group_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = prime_video_runtimes_group_movies['Runtime Group'][:10], y =
prime_video_runtimes_group_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_ru_ax4 = sns.barplot(x = disney_runtimes_group_movies['Runtime Group'][:10], y =
disney_runtimes_group_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[62]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = runtimes_group_data_movies['Runtime Group'], y =
runtimes_group_data_movies['Netflix'], color = 'red')
sns.lineplot(x = runtimes_group_data_movies['Runtime Group'], y =
runtimes_group_data_movies['Hulu'], color = 'lightgreen')
sns.lineplot(x = runtimes_group_data_movies['Runtime Group'], y =
runtimes_group_data_movies['Prime Video'], color = 'lightblue')
sns.lineplot(x = runtimes_group_data_movies['Runtime Group'], y =
runtimes_group_data_movies['Disney+'], color = 'darkblue')
plt.xlabel('Runtime Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()

```

In[63]:

```

print(f'''
    Accross All Platforms Total Count of Runtime Group is
'{runtimes_group_data_movies['Runtime Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Netflix' is '{netflix_runtimes_group_movies['Runtime
Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Hulu' is '{hulu_runtimes_group_movies['Runtime
Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Prime Video' is
'{prime_video_runtimes_group_movies['Runtime Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Disney+' is '{disney_runtimes_group_movies['Runtime
Group'].unique().shape[0]}'\n
    ''')

```

In[64]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.lineplot(y = runtimes_group_data_movies['Runtime Group'], x =
runtimes_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_ru_ax2 = sns.lineplot(y = runtimes_group_data_movies['Runtime Group'], x =
runtimes_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ru_ax3 = sns.lineplot(y = runtimes_group_data_movies['Runtime Group'], x =
runtimes_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ru_ax4 = sns.lineplot(y = runtimes_group_data_movies['Runtime Group'], x =
runtimes_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[65]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = runtimes_group_data_movies['Runtime Group'][:10], y =
runtimes_group_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = runtimes_group_data_movies['Runtime Group'][:10], y =
runtimes_group_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])

```

```

p_ru_ax3 = sns.barplot(x = runtimes_group_data_movies['Runtime Group'][:10], y =
runtimes_group_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = runtimes_group_data_movies['Runtime Group'][:10], y =
runtimes_group_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

# In[66]:


df_screentimes_high_movies = df_movies_screentimes.sort_values(by = 'Screentime', ascending
= False).reset_index()
df_screentimes_high_movies = df_screentimes_high_movies.drop(['index'], axis = 1)
# filter = (df_movies_screentimes['Screentime'] ==
(df_movies_screentimes['Screentime'].max()))
# df_screentimes_high_movies = df_movies_screentimes[filter]

# highestRatedMovies =
df_movies_screentimes.loc[df_movies_screentimes['Screentime'].idxmax()]

print('\nMovies with Highest Ever Screentime are : \n')
df_screentimes_high_movies.head(5)

# In[67]:


fig = px.bar(y = df_screentimes_high_movies['Title'][:15],
              x = df_screentimes_high_movies['Screentime'][:15],
              color = df_screentimes_high_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Highest Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[68]:


df_screentimes_low_movies = df_movies_screentimes.sort_values(by = 'Screentime', ascending
= True).reset_index()
df_screentimes_low_movies = df_screentimes_low_movies.drop(['index'], axis = 1)
# filter = (df_movies_screentimes['Screentime'] ==
(df_movies_screentimes['Screentime'].min()))
# df_screentimes_low_movies = df_movies_screentimes[filter]

print('\nMovies with Lowest Ever Screentime are : \n')
df_screentimes_low_movies.head(5)

# In[69]:


fig = px.bar(y = df_screentimes_low_movies['Title'][:15],
              x = df_screentimes_low_movies['Screentime'][:15],
              color = df_screentimes_low_movies['Screentime'][:15],

```

```

color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
title  = 'Movies with Lowest Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[70]:


print(f'''
    Total '{df_movies_screentimes['Screentime'].unique().shape[0]}' unique Screentime s
were Given, They were Like this,\n

{df_movies_screentimes.sort_values(by = 'Screentime', ascending =
False)['Screentime'].unique()}\n

    The Highest Ever Screentime Ever Any Movie Got is
'{df_screentimes_high_movies['Title'][0]}' :
'{df_screentimes_high_movies['Screentime'].max()}'\n

    The Lowest Ever Screentime Ever Any Movie Got is
'{df_screentimes_low_movies['Title'][0]}' :
'{df_screentimes_low_movies['Screentime'].min()}'\n
    ''')

# In[71]:


netflix_screentimes_high_movies =
df_screentimes_high_movies.loc[df_screentimes_high_movies['Netflix']==1].reset_index()
netflix_screentimes_high_movies = netflix_screentimes_high_movies.drop(['index'], axis = 1)

netflix_screentimes_low_movies =
df_screentimes_low_movies.loc[df_screentimes_low_movies['Netflix']==1].reset_index()
netflix_screentimes_low_movies = netflix_screentimes_low_movies.drop(['index'], axis = 1)

netflix_screentimes_high_movies.head(5)

# In[72]:


fig = px.bar(y = netflix_screentimes_high_movies['Title'][:15],
              x = netflix_screentimes_high_movies['Screentime'][:15],
              color = netflix_screentimes_high_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title  = 'Movies with Highest Screentime in Hours : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[73]:


fig = px.bar(y = netflix_screentimes_low_movies['Title'][:15],
              x = netflix_screentimes_low_movies['Screentime'][:15],
              color = netflix_screentimes_low_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title  = 'Movies with Lowest Screentime in Hours : Netflix')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[74]:


hulu_screentimes_high_movies =
df_screentimes_high_movies.loc[df_screentimes_high_movies['Hulu']==1].reset_index()
hulu_screentimes_high_movies = hulu_screentimes_high_movies.drop(['index'], axis = 1)

hulu_screentimes_low_movies =
df_screentimes_low_movies.loc[df_screentimes_low_movies['Hulu']==1].reset_index()
hulu_screentimes_low_movies = hulu_screentimes_low_movies.drop(['index'], axis = 1)

hulu_screentimes_high_movies.head(5)

# In[75]:


fig = px.bar(y = hulu_screentimes_high_movies['Title'][:15],
              x = hulu_screentimes_high_movies['Screentime'][:15],
              color = hulu_screentimes_high_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Highest Screentime in Hours : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[76]:


fig = px.bar(y = hulu_screentimes_low_movies['Title'][:15],
              x = hulu_screentimes_low_movies['Screentime'][:15],
              color = hulu_screentimes_low_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Lowest Screentime in Hours : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[77]:


prime_video_screentimes_high_movies =
df_screentimes_high_movies.loc[df_screentimes_high_movies['Prime Video']==1].reset_index()
prime_video_screentimes_high_movies = prime_video_screentimes_high_movies.drop(['index'],
axis = 1)

prime_video_screentimes_low_movies =
df_screentimes_low_movies.loc[df_screentimes_low_movies['Prime Video']==1].reset_index()
prime_video_screentimes_low_movies = prime_video_screentimes_low_movies.drop(['index'],
axis = 1)

prime_video_screentimes_high_movies.head(5)

# In[78]:


fig = px.bar(y = prime_video_screentimes_high_movies['Title'][:15],

```

```

x = prime_video_screentimes_high_movies['Screentime'][:15],
color = prime_video_screentimes_high_movies['Screentime'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
title = 'Movies with Highest Screentime in Hours : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[79]:

```

fig = px.bar(y = prime_video_screentimes_low_movies['Title'][:15],
              x = prime_video_screentimes_low_movies['Screentime'][:15],
              color = prime_video_screentimes_low_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Lowest Screentime in Hours : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[80]:

```

disney_screentimes_high_movies =
df_screentimes_high_movies.loc[df_screentimes_high_movies['Disney+']==1].reset_index()
disney_screentimes_high_movies = disney_screentimes_high_movies.drop(['index'], axis = 1)

disney_screentimes_low_movies =
df_screentimes_low_movies.loc[df_screentimes_low_movies['Disney+']==1].reset_index()
disney_screentimes_low_movies = disney_screentimes_low_movies.drop(['index'], axis = 1)

disney_screentimes_high_movies.head(5)

```

In[81]:

```

fig = px.bar(y = disney_screentimes_high_movies['Title'][:15],
              x = disney_screentimes_high_movies['Screentime'][:15],
              color = disney_screentimes_high_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Highest Screentime in Hours : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[82]:

```

fig = px.bar(y = disney_screentimes_low_movies['Title'][:15],
              x = disney_screentimes_low_movies['Screentime'][:15],
              color = disney_screentimes_low_movies['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Lowest Screentime in Hours : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[83]:
```

```
print(f'''  
    The Movie with Highest Screentime Ever Got is  
'{df_screentimes_high_movies['Title'][0]}' :  
'{df_screentimes_high_movies['Screentime'].max()}'\n  
    The Movie with Lowest Screentime Ever Got is  
'{df_screentimes_low_movies['Title'][0]}' :  
'{df_screentimes_low_movies['Screentime'].min()}'\n  
  
    The Movie with Highest Screentime on 'Netflix' is  
'{netflix_screentimes_high_movies['Title'][0]}' :  
'{netflix_screentimes_high_movies['Screentime'].max()}'\n  
    The Movie with Lowest Screentime on 'Netflix' is  
'{netflix_screentimes_low_movies['Title'][0]}' :  
'{netflix_screentimes_low_movies['Screentime'].min()}'\n  
  
    The Movie with Highest Screentime on 'Hulu' is  
'{hulu_screentimes_high_movies['Title'][0]}' :  
'{hulu_screentimes_high_movies['Screentime'].max()}'\n  
    The Movie with Lowest Screentime on 'Hulu' is  
'{hulu_screentimes_low_movies['Title'][0]}' :  
'{hulu_screentimes_low_movies['Screentime'].min()}'\n  
  
    The Movie with Highest Screentime on 'Prime Video' is  
'{prime_video_screentimes_high_movies['Title'][0]}' :  
'{prime_video_screentimes_high_movies['Screentime'].max()}'\n  
    The Movie with Lowest Screentime on 'Prime Video' is  
'{prime_video_screentimes_low_movies['Title'][0]}' :  
'{prime_video_screentimes_low_movies['Screentime'].min()}'\n  
  
    The Movie with Highest Screentime on 'Disney+' is  
'{disney_screentimes_high_movies['Title'][0]}' :  
'{disney_screentimes_high_movies['Screentime'].max()}'\n  
    The Movie with Lowest Screentime on 'Disney+' is  
'{disney_screentimes_low_movies['Title'][0]}' :  
'{disney_screentimes_low_movies['Screentime'].min()}'\n  
    ''')
```

```
# In[84]:
```

```
print(f'''  
    Across All Platforms the Average Screentime is  
'{round(df_movies_screentimes['Screentime'].mean(), ndigits = 2)}'\n  
    The Average Screentime on 'Netflix' is  
'{round(netflix_screentimes_movies['Screentime'].mean(), ndigits = 2)}'\n  
    The Average Screentime on 'Hulu' is  
'{round(hulu_screentimes_movies['Screentime'].mean(), ndigits = 2)}'\n  
    The Average Screentime on 'Prime Video' is  
'{round(prime_video_screentimes_movies['Screentime'].mean(), ndigits = 2)}'\n  
    The Average Screentime on 'Disney+' is  
'{round(disney_screentimes_movies['Screentime'].mean(), ndigits = 2)}'\n  
    ''')
```

```
# In[85]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))  
sns.distplot(df_movies_screentimes['Screentime'], bins = 20, kde = True, ax = ax[0])  
sns.boxplot(df_movies_screentimes['Screentime'], ax = ax[1])  
plt.show()
```

```

# In[86]:


# Defining plot size and title
plt.figure(figsize = (20, 10))
plt.title('Screentime s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_screentimes_movies['Screentime'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_screentimes_movies['Screentime'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_screentimes_movies['Screentime'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_screentimes_movies['Screentime'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[87]:


def round_val(data):
    if str(data) != 'nan':
        return round(data)

# In[88]:


df_movies_screentimes_group = df_movies_screentimes.copy()

# In[89]:


df_movies_screentimes_group['Screentime Group'] =
df_movies_screentimes['Screentime'].apply(round_val)

screentimes_values = df_movies_screentimes_group['Screentime
Group'].value_counts().sort_index(ascending = False).tolist()
screentimes_index = df_movies_screentimes_group['Screentime
Group'].value_counts().sort_index(ascending = False).index

# screentimes_values, screentimes_index

# In[90]:


screentimes_group_count = df_movies_screentimes_group.groupby('Screentime
Group')['Title'].count()
screentimes_group_movies = df_movies_screentimes_group.groupby('Screentime
Group')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
screentimes_group_data_movies = pd.concat([screentimes_group_count,
screentimes_group_movies], axis = 1).reset_index().rename(columns = {'Title' : 'Movies
Count'})
screentimes_group_data_movies = screentimes_group_data_movies.sort_values(by = 'Movies
Count', ascending = False)

# In[91]:

```

```

# Screentime Group with Movies Counts - All Platforms Combined
screentimes_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[92]: 

screentimes_group_data_movies.sort_values(by = 'Screentime Group', ascending = False)

# In[93]: 

fig = px.bar(y = screentimes_group_data_movies['Movies Count'],
              x = screentimes_group_data_movies['Screentime Group'],
              color = screentimes_group_data_movies['Screentime Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Screentime : In Hours'},
              title = 'Movies with Group Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[94]: 

fig = px.pie(screentimes_group_data_movies[:10],
              names = screentimes_group_data_movies['Screentime Group'],
              values = screentimes_group_data_movies['Movies Count'],
              color = screentimes_group_data_movies['Movies Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on Screentime Group')
fig.show()

# In[95]: 

df_screentimes_group_high_movies = screentimes_group_data_movies.sort_values(by = 'Movies Count', ascending = False).reset_index()
df_screentimes_group_high_movies = df_screentimes_group_high_movies.drop(['index'], axis = 1)
# filter = (screentimes_group_data_movies['Movies Count'] ==
# (screentimes_group_data_movies['Movies Count'].max()))
# df_screentimes_group_high_movies = screentimes_group_data_movies[filter]

# highest_rated_movies =
screentimes_group_data_movies.loc[screentimes_group_data_movies['Movies Count'].idxmax()]

# print('\nScreentime with Highest Ever Movies Count are : All Platforms Combined\n')
df_screentimes_group_high_movies.head(5)

# In[96]: 

df_screentimes_group_low_movies = screentimes_group_data_movies.sort_values(by = 'Movies Count', ascending = True).reset_index()
df_screentimes_group_low_movies = df_screentimes_group_low_movies.drop(['index'], axis = 1)
# filter = (screentimes_group_data_movies['Movies Count'] ==
# (screentimes_group_data_movies['Movies Count'].min()))

```

```

# df_screentimes_group_low_movies = screentimes_group_data_movies[filter]

# print('\nScreentime with Lowest Ever Movies Count are : All Platforms Combined\n')
df_screentimes_group_low_movies.head(5)

# In[97]:


print(f'''
    Total '{df_movies_screentimes['Screentime'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see Movies from Total '{screentimes_group_data_movies['Screentime
Group'].unique().shape[0]}' Screentime Group, They were Like this, \n

    {screentimes_group_data_movies.sort_values(by = 'Movies Count', ascending =
False)['Screentime Group'].unique()} etc. \n

    The Screentime Group with Highest Movies Count have
'{screentimes_group_data_movies['Movies Count'].max()}' Movies Available is
'{df_screentimes_group_high_movies['Screentime Group'][0]}', &\n
    The Screentime Group with Lowest Movies Count have
'{screentimes_group_data_movies['Movies Count'].min()}' Movies Available is
'{df_screentimes_group_low_movies['Screentime Group'][0]}'
    ''')

```

```

# In[98]:


netflix_screentimes_group_movies =
screentimes_group_data_movies[screentimes_group_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_screentimes_group_movies = netflix_screentimes_group_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_screentimes_group_high_movies = df_screentimes_group_high_movies.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_screentimes_group_high_movies =
netflix_screentimes_group_high_movies.drop(['index'], axis = 1)

netflix_screentimes_group_low_movies = df_screentimes_group_low_movies.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_screentimes_group_low_movies = netflix_screentimes_group_low_movies.drop(['index'], axis = 1)

netflix_screentimes_group_high_movies.head(5)

```

```

# In[99]:


hulu_screentimes_group_movies =
screentimes_group_data_movies[screentimes_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_screentimes_group_movies = hulu_screentimes_group_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'Movies Count'], axis = 1)

hulu_screentimes_group_high_movies = df_screentimes_group_high_movies.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_screentimes_group_high_movies = hulu_screentimes_group_high_movies.drop(['index'], axis = 1)

hulu_screentimes_group_low_movies = df_screentimes_group_low_movies.sort_values(by = 'Hulu', ascending = True).reset_index()

```

```

hulu_screentimes_group_low_movies = hulu_screentimes_group_low_movies.drop(['index'], axis = 1)

hulu_screentimes_group_high_movies.head(5)

# In[100]:


prime_video_screentimes_group_movies =
screentimes_group_data_movies[screentimes_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_screentimes_group_movies = prime_video_screentimes_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_screentimes_group_high_movies = df_screentimes_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_screentimes_group_high_movies =
prime_video_screentimes_group_high_movies.drop(['index'], axis = 1)

prime_video_screentimes_group_low_movies = df_screentimes_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_screentimes_group_low_movies =
prime_video_screentimes_group_low_movies.drop(['index'], axis = 1)

prime_video_screentimes_group_high_movies.head(5)

# In[101]:


disney_screentimes_group_movies =
screentimes_group_data_movies[screentimes_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_screentimes_group_movies = disney_screentimes_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_screentimes_group_high_movies = df_screentimes_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_screentimes_group_high_movies = disney_screentimes_group_high_movies.drop(['index'], axis = 1)

disney_screentimes_group_low_movies = df_screentimes_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_screentimes_group_low_movies = disney_screentimes_group_low_movies.drop(['index'], axis = 1)

disney_screentimes_group_high_movies.head(5)

# In[102]:


print(f'''The Screentime Group with Highest Movies Count Ever Got is
'{df_screentimes_group_high_movies['Screentime Group'][0]}' :
'{df_screentimes_group_high_movies['Movies Count'].max()}'\n
The Screentime Group with Lowest Movies Count Ever Got is
'{df_screentimes_group_low_movies['Screentime Group'][0]}' :
'{df_screentimes_group_low_movies['Movies Count'].min()}'\n

The Screentime Group with Highest Movies Count on 'Netflix' is
'{netflix_screentimes_group_high_movies['Screentime Group'][0]}' :
'{netflix_screentimes_group_high_movies['Netflix'].max()}'\n

```

```

The Screentime Group with Lowest Movies Count on 'Netflix' is
'{netflix_screentimes_group_low_movies['Screentime Group'][0]} :
'{netflix_screentimes_group_low_movies['Netflix'].min()}'\n

The Screentime Group with Highest Movies Count on 'Hulu' is
'{hulu_screentimes_group_high_movies['Screentime Group'][0]} :
'{hulu_screentimes_group_high_movies['Hulu'].max()}'\n
The Screentime Group with Lowest Movies Count on 'Hulu' is
'{hulu_screentimes_group_low_movies['Screentime Group'][0]} :
'{hulu_screentimes_group_low_movies['Hulu'].min()}'\n

The Screentime Group with Highest Movies Count on 'Prime Video' is
'{prime_video_screentimes_group_high_movies['Screentime Group'][0]} :
'{prime_video_screentimes_group_high_movies['Prime Video'].max()}'\n
The Screentime Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_screentimes_group_low_movies['Screentime Group'][0]} :
'{prime_video_screentimes_group_low_movies['Prime Video'].min()}'\n

The Screentime Group with Highest Movies Count on 'Disney+' is
'{disney_screentimes_group_high_movies['Screentime Group'][0]} :
'{disney_screentimes_group_high_movies['Disney+'].max()}'\n
The Screentime Group with Lowest Movies Count on 'Disney+' is
'{disney_screentimes_group_low_movies['Screentime Group'][0]} :
'{disney_screentimes_group_low_movies['Disney+'].min()}'\n
''')

```

In[103]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_scr_ax1 = sns.barplot(x = netflix_screentimes_group_movies['Screentime Group'][:10], y =
netflix_screentimes_group_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_scr_ax2 = sns.barplot(x = hulu_screentimes_group_movies['Screentime Group'][:10], y =
hulu_screentimes_group_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_scr_ax3 = sns.barplot(x = prime_video_screentimes_group_movies['Screentime Group'][:10],
y = prime_video_screentimes_group_movies['Prime Video'][:10], palette = 'Blues_r', ax =
axes[1, 0])
d_scr_ax4 = sns.barplot(x = disney_screentimes_group_movies['Screentime Group'][:10], y =
disney_screentimes_group_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_scr_ax1.title.set_text(labels[0])
h_scr_ax2.title.set_text(labels[1])
p_scr_ax3.title.set_text(labels[2])
d_scr_ax4.title.set_text(labels[3])

plt.show()

```

In[104]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = screentimes_group_data_movies['Screentime Group'], y =
screentimes_group_data_movies['Netflix'], color = 'red')
sns.lineplot(x = screentimes_group_data_movies['Screentime Group'], y =
screentimes_group_data_movies['Hulu'], color = 'lightgreen')
sns.lineplot(x = screentimes_group_data_movies['Screentime Group'], y =
screentimes_group_data_movies['Prime Video'], color = 'lightblue')
sns.lineplot(x = screentimes_group_data_movies['Screentime Group'], y =
screentimes_group_data_movies['Disney+'], color = 'darkblue')
plt.xlabel('Screentime Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)

```

```

plt.show()

# In[105]:


print(f'''
    Across All Platforms Total Count of Screenshot Group is
'{screentimes_group_data_movies['Screenshot Group'].unique().shape[0]}'\n
    Total Count of Screenshot Group on 'Netflix' is
'{netflix_screentimes_group_movies['Screenshot Group'].unique().shape[0]}'\n
    Total Count of Screenshot Group on 'Hulu' is
'{hulu_screentimes_group_movies['Screenshot Group'].unique().shape[0]}'\n
    Total Count of Screenshot Group on 'Prime Video' is
'{prime_video_screentimes_group_movies['Screenshot Group'].unique().shape[0]}'\n
    Total Count of Screenshot Group on 'Disney+' is
'{disney_screentimes_group_movies['Screenshot Group'].unique().shape[0]}'
    ''')

```

```
# In[106]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_scr_ax1 = sns.lineplot(y = screentimes_group_data_movies['Screenshot Group'], x =
screentimes_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_scr_ax2 = sns.lineplot(y = screentimes_group_data_movies['Screenshot Group'], x =
screentimes_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_scr_ax3 = sns.lineplot(y = screentimes_group_data_movies['Screenshot Group'], x =
screentimes_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_scr_ax4 = sns.lineplot(y = screentimes_group_data_movies['Screenshot Group'], x =
screentimes_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_scr_ax1.title.set_text(labels[0])
h_scr_ax2.title.set_text(labels[1])
p_scr_ax3.title.set_text(labels[2])
d_scr_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[107]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = screentimes_group_data_movies['Screenshot Group'][:10], y =
screentimes_group_data_movies['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = screentimes_group_data_movies['Screenshot Group'][:10], y =
screentimes_group_data_movies['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = screentimes_group_data_movies['Screenshot Group'][:10], y =
screentimes_group_data_movies['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = screentimes_group_data_movies['Screenshot Group'][:10], y =
screentimes_group_data_movies['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

ottmovies_title.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")


# In[4]:


nltk.download('all')
```

```

# In[5]:
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\Users\\pawan\\OneDrive\\Desktop\\ott\\Data\\'
df_movies = pd.read_csv(path + 'ottmovies.csv')
df_movies.head()

# In[6]:
# profile = ProfileReport(df_movies)
# profile

# In[7]:
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:
data_investigate(df_movies)

# In[9]:
# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

```

```

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country
df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

```

```

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)
# df_movies.sort_values('IMDb', ascending = False)

# In[15]:
# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)
# path = '/content/drive/MyDrive/Files/'
# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')
# udf_movies

# In[16]:
# df_netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df_hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df_prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df_disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]

# In[17]:
df_netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df_prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df_disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]

```

```

# In[18]:
df_movies_title = df_movies.copy()

# In[19]:
df_movies_title.drop(df_movies_title.loc[df_movies_title['Title'] == "NA"].index, inplace = True)
# df_movies_title = df_movies_title[df_movies_title.Title != "NA"]

# In[20]:
# Creating distinct dataframes only with the movies present on individual streaming
# platforms
netflix_title_movies = df_movies_title.loc[df_movies_title['Netflix'] == 1]
hulu_title_movies = df_movies_title.loc[df_movies_title['Hulu'] == 1]
prime_video_title_movies = df_movies_title.loc[df_movies_title['Prime Video'] == 1]
disney_title_movies = df_movies_title.loc[df_movies_title['Disney+'] == 1]

# In[21]:
plt.figure(figsize = (10, 10))
corr = df_movies_title.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[22]:
df_movies_title = df_movies_title['Title']
movies_title_w = ' '.join(df_movies_title)

# In[23]:
stopwords = set(STOPWORDS)

wordcloud_all_title_movies = WordCloud(width = 1000, height = 500,
                                       background_color ='white',
                                       stopwords = stopwords,
                                       min_font_size = 10).generate(movies_title_w)

# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_all_title_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

```

```

plt.show()

# In[24]:


movies_title_w = movies_title_w.lower()
stop_words_english_movies = set(STOPWORDS)
word_tokens_english_movies = word_tokenize(movies_title_w)
filtered_movie_title = [w for w in word_tokens_english_movies if not w in stop_words_english_movies]
filtered_movie_title = " ".join(filtered_movie_title)
filtered_movie_title = re.sub("'s", ' ', filtered_movie_title)
filtered_movie_title = re.sub(r'[0-9]+', ' ', filtered_movie_title)
final_movie_title = re.sub(r'[^w\s]', ' ', filtered_movie_title)
title_movies_corpus_len = len(filtered_movie_title.split())
title_movies_corpus_len

# In[25]:


def extract_ngrams(data, num):
    n_grams = ngrams(nltk.word_tokenize(data), num)
    return [ ' '.join(grams) for grams in n_grams]

# In[26]:


title_ngram1_movies = FreqDist()
title_ngram1 = extract_ngrams(final_movie_title[:title_movies_corpus_len], 1)
for word in title_ngram1:
    title_ngram1_movies[word.lower()] += 1

# In[27]:


title_ngram1_movies.most_common(10)

# In[28]:


title_ngram2_movies = FreqDist()
title_ngram2 = extract_ngrams(final_movie_title[:title_movies_corpus_len], 2)
for word in title_ngram2:
    title_ngram2_movies[word.lower()] += 1

# In[29]:

```

```

title_ngram2_movies.most_common(10)

# In[30]:


title_ngram3_movies = FreqDist()

title_ngram3 = extract_ngrams(final_movie_title[:title_movies_corpus_len], 3)

for word in title_ngram3:
    title_ngram3_movies[word.lower()] += 1

# In[31]:


title_ngram3_movies.most_common(10)

# In[32]:


title_ngram4_movies = FreqDist()

title_ngram4 = extract_ngrams(final_movie_title[:title_movies_corpus_len], 4)

for word in title_ngram4:
    title_ngram4_movies[word.lower()] += 1

# In[33]:


title_ngram4_movies.most_common(10)

# In[34]:


title_ngram5_movies = FreqDist()

title_ngram5 = extract_ngrams(final_movie_title[:title_movies_corpus_len], 5)

for word in title_ngram5:
    title_ngram5_movies[word.lower()] += 1

# In[35]:


title_ngram5_movies.most_common(10)

# In[36]:


# Netflix Wordcloud
netflix_title_movies_t = netflix_title_movies['Title']
netflix_movies_title_w = ' '.join(netflix_title_movies_t)

# In[37]:

```

```

stopwords = set(STOPWORDS)

wordcloud.netflix_title_movies = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10
                                         ).generate(netflix_movies_title_w)

print('\nThe Wordcloud Generated from Titles of Netflix is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud.netflix_title_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

# In[38]:


# Hulu Wordcloud
hulu_title_movies_t = hulu_title_movies['Title']
hulu_movies_title_w = ' '.join(hulu_title_movies_t)

# In[39]:


stopwords = set(STOPWORDS)

wordcloud.hulu_title_movies = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10
                                         ).generate(hulu_movies_title_w)

print('\nThe Wordcloud Generated from Titles of Hulu is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud.hulu_title_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

# In[40]:


# Prime Video Wordcloud
prime_video_title_movies_t = prime_video_title_movies['Title']
prime_video_movies_title_w = ' '.join(prime_video_title_movies_t)

# In[41]:


stopwords = set(STOPWORDS)

wordcloud.prime_video_title_movies = WordCloud(width = 1000, height = 500,
                                               background_color ='white',
                                               stopwords = stopwords,
                                               min_font_size = 10
                                               ).generate(prime_video_movies_title_w)

```

```

print('\nThe Wordcloud Generated from Titles of Prime Video is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_prime_video_title_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

# In[42]:


# Disney+ Wordcloud
disney_title_movies_t = disney_title_movies['Title']
disney_movies_title_w = ' '.join(disney_title_movies_t)

# In[43]:


stopwords = set(STOPWORDS)

wordcloud_disney_title_movies = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10
                                         ).generate(disney_movies_title_w)

print('\nThe Wordcloud Generated from Titles of Disney+ is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_disney_title_movies)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

```

ottmovies_year.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

# In[2]:


# Import Dataset
# Import File from Local Drive

```

```
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_movies = pd.read_csv(path + 'ottmovies.csv')

df_movies.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_movies)
# profile
```

```
# In[7]:
```

```
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
```

```

print('***25)
print('Missing Values : ')
c = df.isnull().sum()
c = c[c > 0]
print(c)
print('***25)
print('Missing values %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

```

In[8]:

```
data_investigate(df_movies)
```

In[9]:

```

# ID
# df_movies = df_movies.drop(['ID'], axis = 1)

# Age
df_movies.loc[df_movies['Age'].isnull() & df_movies['Disney+'] == 1, "Age"] = '13'
# df_movies.fillna({'Age' : 18}, inplace = True)
df_movies.fillna({'Age' : 'NR'}, inplace = True)
df_movies['Age'].replace({'all': '0'}, inplace = True)
df_movies['Age'].replace({'7+': '7'}, inplace = True)
df_movies['Age'].replace({'13+': '13'}, inplace = True)
df_movies['Age'].replace({'16+': '16'}, inplace = True)
df_movies['Age'].replace({'18+': '18'}, inplace = True)
# df_movies['Age'] = df_movies['Age'].astype(int)

# IMDb
# df_movies.fillna({'IMDb' : df_movies['IMDb'].mean()}, inplace = True)
# df_movies.fillna({'IMDb' : df_movies['IMDb'].median()}, inplace = True)
df_movies.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'][df_movies['Rotten Tomatoes'].notnull()].astype(int)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].mean()}, inplace = True)
# df_movies.fillna({'Rotten Tomatoes' : df_movies['Rotten Tomatoes'].median()}, inplace = True)
# df_movies['Rotten Tomatoes'] = df_movies['Rotten Tomatoes'].astype(int)
df_movies.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_movies = df_movies.drop(['Directors'], axis = 1)
df_movies.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_movies.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_movies.fillna({'Genres': "NA"}, inplace = True)

# Country

```

```

df_movies.fillna({'Country': "NA"}, inplace = True)

# Language
df_movies.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_movies.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_movies.fillna({'Runtime' : df_movies['Runtime'].mean()}, inplace = True)
# df_movies['Runtime'] = df_movies['Runtime'].astype(int)
df_movies.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_movies.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_movies.fillna({'Type': "NA"}, inplace = True)
# df_movies = df_movies.drop(['Type'], axis = 1)

# Seasons
# df_movies.fillna({'Seasons': 1}, inplace = True)
# df_movies.fillna({'Seasons': "NA"}, inplace = True)
df_movies = df_movies.drop(['Seasons'], axis = 1)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)
# df_movies.fillna({'Seasons' : df_movies['Seasons'].mean()}, inplace = True)
# df_movies['Seasons'] = df_movies['Seasons'].astype(int)

# Service Provider
df_movies['Service Provider'] = df_movies.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_movies.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_movies.dropna(how = 'any', inplace = True)
df_movies.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_movies)

# In[11]:
df_movies.head()

# In[12]:
df_movies.describe()

# In[13]:
df_movies.corr()

# In[14]:
# df_movies.sort_values('Year', ascending = True)

```

```

# df_movies.sort_values('IMDb', ascending = False)

# In[15]:


# df_movies.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_ottmovies.csv', index
# = False)

# path = '/content/drive/MyDrive/Files/'

# udf_movies = pd.read_csv(path + 'updated_ottmovies.csv')

# udf_movies

# In[16]:


# df.netflix_movies = df_movies.loc[(df_movies['Netflix'] > 0)]
# df.hulu_movies = df_movies.loc[(df_movies['Hulu'] > 0)]
# df.prime_video_movies = df_movies.loc[(df_movies['Prime Video'] > 0)]
# df.disney_movies = df_movies.loc[(df_movies['Disney+'] > 0)]


# In[17]:


df.netflix_only_movies = df_movies[(df_movies['Netflix'] == 1) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.hulu_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 1) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 0)]
df.prime_video_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] ==
0) & (df_movies['Prime Video'] == 1 ) & (df_movies['Disney+'] == 0)]
df.disney_only_movies = df_movies[(df_movies['Netflix'] == 0) & (df_movies['Hulu'] == 0) &
(df_movies['Prime Video'] == 0 ) & (df_movies['Disney+'] == 1)]


# In[18]:


df_movies_years = df_movies.copy()

# In[19]:


df_movies_years.drop(df_movies_years.loc[df_movies_years['Year'] == "NA"].index, inplace =
True)
# df_movies_years = df_movies_years[df_movies_years.Year != "NA"]
df_movies_years['Year'] = df_movies_years['Year'].astype(int)

# In[20]:


# Creating distinct dataframes only with the movies present on individual streaming
platforms
netflix_years_movies = df_movies_years.loc[df_movies_years['Netflix'] == 1]
hulu_years_movies = df_movies_years.loc[df_movies_years['Hulu'] == 1]
prime_video_years_movies = df_movies_years.loc[df_movies_years['Prime Video'] == 1]
disney_years_movies = df_movies_years.loc[df_movies_years['Disney+'] == 1]

# In[21]:

```

```
df_movies_years_group = df_movies_years.copy()
```

```
# In[22]:
```

```
plt.figure(figsize = (10, 10))
corr = df_movies_years.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# Show plot
plt.show()
fig.show()
```

```
# In[23]:
```

```
df_years_high_movies = df_movies_years.sort_values(by = 'Year', ascending =
False).reset_index()
df_years_high_movies = df_years_high_movies.drop(['index'], axis = 1)
# filter = (df_movies_years['Year'] == (df_movies_years['Year'].max()))
# df_years_high_movies = df_movies_years[filter]

# highestRatedMovies = df_movies_years.loc[df_movies_years['Year'].idxmax()]

print('\nMovies with Highest Ever Year are : \n')
df_years_high_movies.head(5)
```

```
# In[24]:
```

```
fig = px.bar(y = df_years_high_movies['Title'][:15],
              x = df_years_high_movies['Year'][:15],
              color = df_years_high_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Highest Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[25]:
```

```
df_years_low_movies = df_movies_years.sort_values(by = 'Year', ascending =
True).reset_index()
df_years_low_movies = df_years_low_movies.drop(['index'], axis = 1)
# filter = (df_movies_years['Year'] == (df_movies_years['Year'].min()))
# df_years_low_movies = df_movies_years[filter]

print('\nMovies with Lowest Ever Year are : \n')
df_years_low_movies.head(5)
```

```
# In[26]:
```

```

fig = px.bar(y = df_years_low_movies['Title'][:15],
              x = df_years_low_movies['Year'][:15],
              color = df_years_low_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Lowest Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[27]:


print(f'''
    Total '{df_movies_years['Year'].unique().shape[0]}' unique Year s were Given, They
    were Like this,\n

{df_movies_years.sort_values(by = 'Year', ascending = False)['Year'].unique()}\n

    The Highest Ever Year Ever Any Movie Got is '{df_years_high_movies['Title'][0]}' :
    '{df_years_high_movies['Year'].max()}'\n

    The Lowest Ever Year Ever Any Movie Got is '{df_years_low_movies['Title'][0]}' :
    '{df_years_low_movies['Year'].min()}'\n
    ''')

# In[28]:


netflix_years_high_movies =
df_years_high_movies.loc[df_years_high_movies['Netflix']==1].reset_index()
netflix_years_high_movies = netflix_years_high_movies.drop(['index'], axis = 1)

netflix_years_low_movies =
df_years_low_movies.loc[df_years_low_movies['Netflix']==1].reset_index()
netflix_years_low_movies = netflix_years_low_movies.drop(['index'], axis = 1)

netflix_years_high_movies.head(5)

# In[29]:


fig = px.bar(y = netflix_years_high_movies['Title'][:15],
              x = netflix_years_high_movies['Year'][:15],
              color = netflix_years_high_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Highest Year in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


fig = px.bar(y = netflix_years_low_movies['Title'][:15],
              x = netflix_years_low_movies['Year'][:15],
              color = netflix_years_low_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Lowest Year in Minutes : Netflix')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[31]:


hulu_years_high_movies =
df_years_high_movies.loc[df_years_high_movies['Hulu']==1].reset_index()
hulu_years_high_movies = hulu_years_high_movies.drop(['index'], axis = 1)

hulu_years_low_movies =
df_years_low_movies.loc[df_years_low_movies['Hulu']==1].reset_index()
hulu_years_low_movies = hulu_years_low_movies.drop(['index'], axis = 1)

hulu_years_high_movies.head(5)

# In[32]:


fig = px.bar(y = hulu_years_high_movies['Title'][:15],
              x = hulu_years_high_movies['Year'][:15],
              color = hulu_years_high_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Highest Year in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = hulu_years_low_movies['Title'][:15],
              x = hulu_years_low_movies['Year'][:15],
              color = hulu_years_low_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Lowest Year in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


prime_video_years_high_movies = df_years_high_movies.loc[df_years_high_movies['Prime
Video']==1].reset_index()
prime_video_years_high_movies = prime_video_years_high_movies.drop(['index'], axis = 1)

prime_video_years_low_movies = df_years_low_movies.loc[df_years_low_movies['Prime
Video']==1].reset_index()
prime_video_years_low_movies = prime_video_years_low_movies.drop(['index'], axis = 1)

prime_video_years_high_movies.head(5)

# In[35]:


fig = px.bar(y = prime_video_years_high_movies['Title'][:15],
              x = prime_video_years_high_movies['Year'][:15],

```

```
color = prime_video_years_high_movies['Year'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
title = 'Movies with Highest Year in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[36]:

```
fig = px.bar(y = prime_video_years_low_movies['Title'][:15],
              x = prime_video_years_low_movies['Year'][:15],
              color = prime_video_years_low_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Lowest Year in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[37]:

```
disney_years_high_movies =
df_years_high_movies.loc[df_years_high_movies['Disney+']==1].reset_index()
disney_years_high_movies = disney_years_high_movies.drop(['index'], axis = 1)

disney_years_low_movies =
df_years_low_movies.loc[df_years_low_movies['Disney+']==1].reset_index()
disney_years_low_movies = disney_years_low_movies.drop(['index'], axis = 1)

disney_years_high_movies.head(5)
```

In[38]:

```
fig = px.bar(y = disney_years_high_movies['Title'][:15],
              x = disney_years_high_movies['Year'][:15],
              color = disney_years_high_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Highest Year in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[39]:

```
fig = px.bar(y = disney_years_low_movies['Title'][:15],
              x = disney_years_low_movies['Year'][:15],
              color = disney_years_low_movies['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies', 'x' : 'Year : In Minutes'},
              title = 'Movies with Lowest Year in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[40]:

```

print(f'''
    The Movie with Highest Year Ever Got is '{df_years_high_movies['Title'][0]}' :
'{df_years_high_movies['Year'].max()}'\n
    The Movie with Lowest Year Ever Got is '{df_years_low_movies['Title'][0]}' :
'{df_years_low_movies['Year'].min()}'\n

    The Movie with Highest Year on 'Netflix' is
'{netflix_years_high_movies['Title'][0]}' : '{netflix_years_high_movies['Year'].max()}'\n
    The Movie with Lowest Year on 'Netflix' is '{netflix_years_low_movies['Title'][0]}' :
'{netflix_years_low_movies['Year'].min()}'\n

    The Movie with Highest Year on 'Hulu' is '{hulu_years_high_movies['Title'][0]}' :
'{hulu_years_high_movies['Year'].max()}'\n
    The Movie with Lowest Year on 'Hulu' is '{hulu_years_low_movies['Title'][0]}' :
'{hulu_years_low_movies['Year'].min()}'\n

    The Movie with Highest Year on 'Prime Video' is
'{prime_video_years_high_movies['Title'][0]}' :
'{prime_video_years_high_movies['Year'].max()}'\n
    The Movie with Lowest Year on 'Prime Video' is
'{prime_video_years_low_movies['Title'][0]}' :
'{prime_video_years_low_movies['Year'].min()}'\n

    The Movie with Highest Year on 'Disney+' is '{disney_years_high_movies['Title'][0]}' :
'{disney_years_high_movies['Year'].max()}'\n
    The Movie with Lowest Year on 'Disney+' is '{disney_years_low_movies['Title'][0]}' :
'{disney_years_low_movies['Year'].min()}'\n
    ''')

```

In[41]:

```

print(f'''
    Accross All Platforms the Average Year is '{round(df_movies_years['Year'].mean(),
ndigits = 2)}'\n
    The Average Year on 'Netflix' is '{round(netflix_years_movies['Year'].mean(),
ndigits = 2)}'\n
    The Average Year on 'Hulu' is '{round(hulu_years_movies['Year'].mean(), ndigits =
2)}'\n
    The Average Year on 'Prime Video' is
'{round(prime_video_years_movies['Year'].mean(), ndigits = 2)}'\n
    The Average Year on 'Disney+' is '{round(disney_years_movies['Year'].mean(), ndigits
= 2)}'\n
    ''')

```

In[42]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_movies_years['Year'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_movies_years['Year'], ax = ax[1])
plt.show()

```

In[43]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Year s Per Platform')

# Plotting the information from each dataset into a histogram

```

```

sns.histplot(prime_video_years_movies['Year'][:100], color = 'lightblue', legend = True,
kde = True)
sns.histplot(netflix_years_movies['Year'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_years_movies['Year'][:100], color = 'lightgreen', legend = True, kde =
True)
sns.histplot(disney_years_movies['Year'][:100], color = 'darkblue', legend = True, kde =
True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[44]:


year_count = df_movies_years.groupby('Year')['Title'].count()
year_movies = df_movies_years.groupby('Year')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
year_data_movies = pd.concat([year_count, year_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
year_data_movies = year_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[45]:


# Movies Count per Year - All Platforms Combined
year_data_movies.head()

# In[46]:


fig = px.bar(y = year_data_movies['Movies Count'],
x = year_data_movies['Year'],
color = year_data_movies['Year'],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Movies Count', 'x' : 'Year : In Minutes'},
title = 'Movies with Year : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[47]:


fig = px.pie(year_data_movies[:10],
names = year_data_movies['Year'][:10],
values = year_data_movies['Movies Count'][:10],
color = year_data_movies['Movies Count'][:10],
color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
title = 'Movies Count based on Year Group')
fig.show()

# In[48]:


# Highest Movies Count per Year - All Platforms Combined
df_year_high_movies = year_data_movies.sort_values(by = 'Movies Count', ascending =
False).reset_index()
df_year_high_movies = df_year_high_movies.drop(['index'], axis = 1)

```

```

# filter = (year_data_movies['Movies Count'] == (year_data_movies['Movies Count'].max()))
# df_year_high_movies = year_data_movies[filter]

# highestRatedMovies = year_data_movies.loc[year_data_movies['Movies Count'].idxmax()]

print('\nYear with Highest Ever Movies Count are : All Platforms Combined\n')
df_year_high_movies.head(5)

# In[49]:


fig = px.bar(y = df_year_high_movies['Movies Count'][:10],
              x = df_year_high_movies['Year'][:10],
              color = df_year_high_movies['Movies Count'][:10],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Year : In Minutes'},
              title = 'Year with Highest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]:


# Lowest Movies Count per Year - All Platforms Combined
df_year_low_movies = year_data_movies.sort_values(by = 'Movies Count', ascending =
True).reset_index()
df_year_low_movies = df_year_low_movies.drop(['index'], axis = 1)
# filter = (year_data_movies['Movies Count'] == (year_data_movies['Movies Count'].min()))
# df_year_low_movies = year_data_movies[filter]

print('\nYear with Lowest Ever Movies Count are : All Platforms Combined\n')
df_year_low_movies.head(5)

# In[51]:


fig = px.bar(y = df_year_low_movies['Movies Count'][:10],
              x = df_year_low_movies['Year'][:10],
              color = df_year_low_movies['Movies Count'][:10],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Year : In Minutes'},
              title = 'Year with Lowest Movies Count : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[52]:


print(f'''
      Total '{df_movies_years['Year'].count()}' Titles are available on All Platforms, out
      of which\n
      You Can Choose to see Movies from Total
      '{year_data_movies['Year'].unique().shape[0]}' Year, They were Like this, \n
      {year_data_movies.sort_values(by = 'Movies Count', ascending =
      False)['Year'].head(5).unique()} etc. \n
      The Year with Highest Movies Count have '{year_data_movies['Movies Count'].max()}' 
      Movies Available is '{df_year_high_movies['Year'][0]}', &\n
      ''')

```

```
The Year with Lowest Movies Count have '{year_data_movies['Movies Count'].min()}'  
Movies Available is '{df_year_low_movies['Year'][0]}'  
'''
```

```
# In[53]:
```

```
# Highest Movies Count per Year - Netflix  
netflix_year_movies = year_data_movies[year_data_movies['Netflix'] != 0].sort_values(by =  
'Netflix', ascending = False).reset_index()  
netflix_year_movies = netflix_year_movies.drop(['index', 'Hulu', 'Prime Video', 'Disney+',  
'Movies Count'], axis = 1)  
  
netflix_year_high_movies = df_year_high_movies.sort_values(by = 'Netflix', ascending =  
False).reset_index()  
netflix_year_high_movies = netflix_year_high_movies.drop(['index'], axis = 1)  
  
netflix_year_low_movies = df_year_high_movies.sort_values(by = 'Netflix', ascending =  
True).reset_index()  
netflix_year_low_movies = netflix_year_low_movies.drop(['index'], axis = 1)  
  
netflix_year_high_movies.head(5)
```

```
# In[54]:
```

```
# Highest Movies Count per Year - Hulu  
hulu_year_movies = year_data_movies[year_data_movies['Hulu'] != 0].sort_values(by =  
'Hulu', ascending = False).reset_index()  
hulu_year_movies = hulu_year_movies.drop(['index', 'Netflix', 'Prime Video', 'Disney+',  
'Movies Count'], axis = 1)  
  
hulu_year_high_movies = df_year_high_movies.sort_values(by = 'Hulu', ascending =  
False).reset_index()  
hulu_year_high_movies = hulu_year_high_movies.drop(['index'], axis = 1)  
  
hulu_year_low_movies = df_year_high_movies.sort_values(by = 'Hulu', ascending =  
True).reset_index()  
hulu_year_low_movies = hulu_year_low_movies.drop(['index'], axis = 1)  
  
hulu_year_high_movies.head(5)
```

```
# In[55]:
```

```
# Highest Movies Count per Year - Prime Video  
prime_video_year_movies = year_data_movies[year_data_movies['Prime Video'] !=  
0].sort_values(by = 'Prime Video', ascending = False).reset_index()  
prime_video_year_movies = prime_video_year_movies.drop(['index', 'Netflix', 'Hulu',  
'Disney+', 'Movies Count'], axis = 1)  
  
prime_video_year_high_movies = df_year_high_movies.sort_values(by = 'Prime Video',  
ascending = False).reset_index()  
prime_video_year_high_movies = prime_video_year_high_movies.drop(['index'], axis = 1)  
  
prime_video_year_low_movies = df_year_high_movies.sort_values(by = 'Prime Video', ascending  
= True).reset_index()  
prime_video_year_low_movies = prime_video_year_low_movies.drop(['index'], axis = 1)  
  
prime_video_year_high_movies.head(5)
```

```
# In[56]:
```

```

# Highest Movies Count per Year - Disney+
disney_year_movies = year_data_movies[year_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_year_movies = disney_year_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_year_high_movies = df_year_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_year_high_movies = disney_year_high_movies.drop(['index'], axis = 1)

disney_year_low_movies = df_year_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_year_low_movies = disney_year_low_movies.drop(['index'], axis = 1)

disney_year_high_movies.head(5)

```

In[57]:

```

print(f'''
    The Year with Highest Movies Count Ever Got is '{df_year_high_movies['Year'][0]}' :
'{df_year_high_movies['Movies Count'].max()}'\n
    The Year with Lowest Movies Count Ever Got is '{df_year_low_movies['Year'][0]}' :
'{df_year_low_movies['Movies Count'].min()}'\n

    The Year with Highest Movies Count on 'Netflix' is
'{netflix_year_high_movies['Year'][0]}' : '{netflix_year_high_movies['Netflix'].max()}'\n
    The Year with Lowest Movies Count on 'Netflix' is
'{netflix_year_low_movies['Year'][0]}' : '{netflix_year_low_movies['Netflix'].min()}'\n

    The Year with Highest Movies Count on 'Hulu' is '{hulu_year_high_movies['Year'][0]}'
: '{hulu_year_high_movies['Hulu'].max()}'\n
    The Year with Lowest Movies Count on 'Hulu' is '{hulu_year_low_movies['Year'][0]}'
: '{hulu_year_low_movies['Hulu'].min()}'\n

    The Year with Highest Movies Count on 'Prime Video' is
'{prime_video_year_high_movies['Year'][0]}' : '{prime_video_year_high_movies['Prime
Video'].max()}'\n
    The Year with Lowest Movies Count on 'Prime Video' is
'{prime_video_year_low_movies['Year'][0]}' : '{prime_video_year_low_movies['Prime
Video'].min()}'\n

    The Year with Highest Movies Count on 'Disney+' is
'{disney_year_high_movies['Year'][0]}' : '{disney_year_high_movies['Disney+'].max()}'\n
    The Year with Lowest Movies Count on 'Disney+' is
'{disney_year_low_movies['Year'][0]}' : '{disney_year_low_movies['Disney+'].min()}'\n
    ''')

```

In[58]:

```

print(f'''
    Accross All Platforms the Average Movies Count of Year is
'{round(year_data_movies['Movies Count'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Year on 'Netflix' is
'{round(netflix_year_movies['Netflix'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Year on 'Hulu' is
'{round(hulu_year_movies['Hulu'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Year on 'Prime Video' is
'{round(prime_video_year_movies['Prime Video'].mean(), ndigits = 2)}'\n
    The Average Movies Count of Year on 'Disney+' is
'{round(disney_year_movies['Disney+'].mean(), ndigits = 2)}'\n

```

```
'''
```

```
# In[59]:
```

```
print(f'''  
    Accross All Platforms Total Count of Year is  
'{year_data_movies['Year'].unique().shape[0]}'\n  
    Total Count of Year on 'Netflix' is  
'{netflix_year_movies['Year'].unique().shape[0]}'\n  
    Total Count of Year on 'Hulu' is '{hulu_year_movies['Year'].unique().shape[0]}'\n  
    Total Count of Year on 'Prime Video' is  
'{prime_video_year_movies['Year'].unique().shape[0]}'\n  
    Total Count of Year on 'Disney+' is  
'{disney_year_movies['Year'].unique().shape[0]}'\n'''')
```

```
# In[60]:
```

```
fig = plt.figure(figsize = (20, 10))  
sns.lineplot(data = year_data_movies, x = 'Year', y = 'Movies Count')  
plt.show()
```

```
# In[61]:
```

```
plt.figure(figsize = (20, 10))  
sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Netflix'], color = 'red')  
sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Hulu'], color =  
'lightgreen')  
sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Prime Video'], color =  
'lightblue')  
sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Disney+'], color =  
'darkblue')  
plt.xlabel('Release Year', fontsize = 15)  
plt.ylabel('Movies Count', fontsize = 15)  
plt.show()
```

```
# In[62]:
```

```
fig, axes = plt.subplots(2, 2, figsize=(20, 20))  
  
n_y_ax1 = sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Netflix'], color =  
'red', ax = axes[0, 0])  
h_y_ax2 = sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Hulu'], color =  
'lightgreen', ax = axes[0, 1])  
p_y_ax3 = sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Prime Video'],  
color = 'lightblue', ax = axes[1, 0])  
d_y_ax4 = sns.lineplot(x = year_data_movies['Year'], y = year_data_movies['Disney+'], color =  
'darkblue', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_y_ax1.title.set_text(labels[0])  
h_y_ax2.title.set_text(labels[1])  
p_y_ax3.title.set_text(labels[2])  
d_y_ax4.title.set_text(labels[3])  
  
plt.show()
```

```
# In[63]:
```

```
def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(1801,1901):
        # print(data)
        return 1900
    if data in range(1901,1911):
        return 1910
    if data in range(1911,1921):
        return 1920
    if data in range(1921,1931):
        return 1930
    if data in range(1931,1941):
        return 1940
    if data in range(1941,1951):
        return 1950
    if data in range(1951,1961):
        return 1960
    if data in range(1961,1971):
        return 1970
    if data in range(1971,1981):
        return 1980
    if data in range(1981,1991):
        return 1990
    if data in range(1991,2001):
        return 2000
    if data in range(2000,2011):
        return 2010
    if data in range(2010,2021):
        return 2020
    if data in range(2020,2031):
        return 2030
    else:
        return 2100
```

```
# In[64]:
```

```
df_movies_years_group['Year Group'] =
df_movies_years_group['Year'].apply(round_fix).astype(int)

years_values = df_movies_years_group['Year Group'].value_counts().sort_index(ascending =
False).tolist()
years_index = df_movies_years_group['Year Group'].value_counts().sort_index(ascending =
False).index

# years_values, years_index
```

```
# In[65]:
```

```
years_group_count = df_movies_years_group.groupby('Year Group')['Title'].count()
years_group_movies = df_movies_years_group.groupby('Year Group')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
years_group_data_movies = pd.concat([years_group_count, years_group_movies], axis =
1).reset_index().rename(columns = {'Title' : 'Movies Count'})
years_group_data_movies = years_group_data_movies.sort_values(by = 'Movies Count',
ascending = False)
```

```

# In[66]:


# Year Group with Movies Counts - All Platforms Combined
years_group_data_movies.sort_values(by = 'Movies Count', ascending = False)

# In[67]:


years_group_data_movies.sort_values(by = 'Year Group', ascending = False)

# In[68]:


fig = px.bar(y = years_group_data_movies['Movies Count'],
              x = years_group_data_movies['Year Group'],
              color = years_group_data_movies['Year Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Movies Count', 'x' : 'Year : In Minutes'},
              title = 'Movies with Group Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[69]:


fig = px.pie(years_group_data_movies[:10],
              names = years_group_data_movies['Year Group'],
              values = years_group_data_movies['Movies Count'],
              color = years_group_data_movies['Movies Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'Movies Count based on Year Group')
fig.show()

# In[70]:


df_years_group_high_movies = years_group_data_movies.sort_values(by = 'Movies Count',
                                                               ascending = False).reset_index()
df_years_group_high_movies = df_years_group_high_movies.drop(['index'], axis = 1)
# filter = (years_group_data_movies['Movies Count'] == (years_group_data_movies['Movies Count'].max()))
# df_years_group_high_movies = years_group_data_movies[filter]

# highestRatedMovies = years_group_data_movies.loc[years_group_data_movies['Movies Count'].idxmax()]

# print('\nYear with Highest Ever Movies Count are : All Platforms Combined\n')
df_years_group_high_movies.head(5)

# In[71]:


df_years_group_low_movies = years_group_data_movies.sort_values(by = 'Movies Count',
                                                               ascending = True).reset_index()
df_years_group_low_movies = df_years_group_low_movies.drop(['index'], axis = 1)

```

```
# filter = (years_group_data_movies['Movies Count'] == (years_group_data_movies['Movies Count'].min()))
# df_years_group_low_movies = years_group_data_movies[filter]

# print('\nYear with Lowest Ever Movies Count are : All Platforms Combined\n')
df_years_group_low_movies.head(5)
```

In[72]:

```
print(f'''
    Total '{df_movies_years['Year'].count()}' Titles are available on All Platforms, out
    of which\n
        You Can Choose to see Movies from Total '{years_group_data_movies['Year
    Group'].unique().shape[0]}' Year Group, They were Like this, \n
        {years_group_data_movies.sort_values(by = 'Movies Count', ascending = False)[['Year
    Group']].unique()} etc. \n
        The Year Group with Highest Movies Count have '{years_group_data_movies['Movies
    Count'].max()}' Movies Available is '{df_years_group_high_movies['Year Group'][0]}', &\n
        The Year Group with Lowest Movies Count have '{years_group_data_movies['Movies
    Count'].min()}' Movies Available is '{df_years_group_low_movies['Year Group'][0]}'
    ''')
```

In[73]:

```
netflix_years_group_movies = years_group_data_movies[years_group_data_movies['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_years_group_movies = netflix_years_group_movies.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'Movies Count'], axis = 1)

netflix_years_group_high_movies = df_years_group_high_movies.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_years_group_high_movies = netflix_years_group_high_movies.drop(['index'], axis = 1)

netflix_years_group_low_movies = df_years_group_low_movies.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_years_group_low_movies = netflix_years_group_low_movies.drop(['index'], axis = 1)

netflix_years_group_high_movies.head(5)
```

In[74]:

```
hulu_years_group_movies = years_group_data_movies[years_group_data_movies['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_years_group_movies = hulu_years_group_movies.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'Movies Count'], axis = 1)

hulu_years_group_high_movies = df_years_group_high_movies.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_years_group_high_movies = hulu_years_group_high_movies.drop(['index'], axis = 1)

hulu_years_group_low_movies = df_years_group_low_movies.sort_values(by = 'Hulu', ascending
= True).reset_index()
hulu_years_group_low_movies = hulu_years_group_low_movies.drop(['index'], axis = 1)

hulu_years_group_high_movies.head(5)
```

In[75]:

```

prime_video_years_group_movies = years_group_data_movies[years_group_data_movies['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_years_group_movies = prime_video_years_group_movies.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'Movies Count'], axis = 1)

prime_video_years_group_high_movies = df_years_group_high_movies.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_years_group_high_movies = prime_video_years_group_high_movies.drop(['index'], axis = 1)

prime_video_years_group_low_movies = df_years_group_high_movies.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_years_group_low_movies = prime_video_years_group_low_movies.drop(['index'], axis = 1)

prime_video_years_group_high_movies.head(5)

```

In[76]:

```

disney_years_group_movies = years_group_data_movies[years_group_data_movies['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_years_group_movies = disney_years_group_movies.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'Movies Count'], axis = 1)

disney_years_group_high_movies = df_years_group_high_movies.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_years_group_high_movies = disney_years_group_high_movies.drop(['index'], axis = 1)

disney_years_group_low_movies = df_years_group_high_movies.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_years_group_low_movies = disney_years_group_low_movies.drop(['index'], axis = 1)

disney_years_group_high_movies.head(5)

```

In[77]:

```

print(f'''
    The Year Group with Highest Movies Count Ever Got is
'{df_years_group_high_movies['Year Group'][0]}' : '{df_years_group_high_movies['Movies Count'].max()}'\n
    The Year Group with Lowest Movies Count Ever Got is '{df_years_group_low_movies['Year Group'][0]}' : '{df_years_group_low_movies['Movies Count'].min()}'\n

    The Year Group with Highest Movies Count on 'Netflix' is
'{netflix_years_group_high_movies['Year Group'][0]}' :
'{netflix_years_group_high_movies['Netflix'].max()}'\n
    The Year Group with Lowest Movies Count on 'Netflix' is
'{netflix_years_group_low_movies['Year Group'][0]}' :
'{netflix_years_group_low_movies['Netflix'].min()}'\n

    The Year Group with Highest Movies Count on 'Hulu' is
'{hulu_years_group_high_movies['Year Group'][0]}' :
'{hulu_years_group_high_movies['Hulu'].max()}'\n
    The Year Group with Lowest Movies Count on 'Hulu' is
'{hulu_years_group_low_movies['Year Group'][0]}' :
'{hulu_years_group_low_movies['Hulu'].min()}'\n

    The Year Group with Highest Movies Count on 'Prime Video' is
'{prime_video_years_group_high_movies['Year Group'][0]}' :
'{prime_video_years_group_high_movies['Prime Video'].max()}'\n

```

```
The Year Group with Lowest Movies Count on 'Prime Video' is
'{prime_video_years_group_low_movies['Year Group'][0]}' :
'{prime_video_years_group_low_movies['Prime Video'].min()}'\n
```

```
The Year Group with Highest Movies Count on 'Disney+' is
'{disney_years_group_high_movies['Year Group'][0]}' :
'{disney_years_group_high_movies['Disney+'].max()}'\n
The Year Group with Lowest Movies Count on 'Disney+' is
'{disney_years_group_low_movies['Year Group'][0]}' :
'{disney_years_group_low_movies['Disney+'].min()}'\n
''')
```

```
# In[78]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = netflix_years_group_movies['Year Group'], y =
netflix_years_group_movies['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = hulu_years_group_movies['Year Group'], y =
hulu_years_group_movies['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = prime_video_years_group_movies['Year Group'], y =
prime_video_years_group_movies['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = disney_years_group_movies['Year Group'], y =
disney_years_group_movies['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[79]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Netflix'], color = 'red')
sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Hulu'], color = 'lightgreen')
sns.lineplot(x = years_group_data_movies['Year Group'], y = years_group_data_movies['Prime
Video'], color = 'lightblue')
sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Disney+'], color = 'darkblue')
plt.xlabel('Year Group', fontsize = 15)
plt.ylabel('Movies Count', fontsize = 15)
plt.show()
```

```
# In[80]:
```

```
print(f'''
    Accross All Platforms Total Count of Year Group is '{years_group_data_movies['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Netflix' is '{netflix_years_group_movies['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Hulu' is '{hulu_years_group_movies['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Prime Video' is '{prime_video_years_group_movies['Year
Group'].unique().shape[0]}'\n
```

```
Total Count of Year Group on 'Disney+' is '{disney_years_group_movies['Year Group'].unique().shape[0]}'\n'''
```

```
# In[81]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.lineplot(y = years_group_data_movies['Year Group'], x =
years_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_ru_ax2 = sns.lineplot(y = years_group_data_movies['Year Group'], x =
years_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ru_ax3 = sns.lineplot(y = years_group_data_movies['Year Group'], x =
years_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ru_ax4 = sns.lineplot(y = years_group_data_movies['Year Group'], x =
years_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[82]:
```

```
fig, axes = plt.subplots(2, 2, figsize=(20 ,20))

n_yg_ax1 = sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Netflix'], color = 'red', ax = axes[0, 0])
h_yg_ax2 = sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_yg_ax3 = sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_yg_ax4 = sns.lineplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_yg_ax1.title.set_text(labels[0])
h_yg_ax2.title.set_text(labels[1])
p_yg_ax3.title.set_text(labels[2])
d_yg_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[83]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = years_group_data_movies['Year Group'], y =
years_group_data_movies['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])
```

```

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

TV SHOWS

otttvshows_age.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re

```

```

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\Users\\pawan\\OneDrive\\Desktop\\ott\\Data\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()

# In[6]:


# profile = ProfileReport(df_tvshows)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

```

```

# In[8]:

data_investigate(df_tvshows)

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime

```

```

# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_tvshows)

# In[11]:
df_tvshows.head()

# In[12]:
df_tvshows.describe()

# In[13]:
df_tvshows.corr()

# In[14]:
# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:

```

```

# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df_netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df_hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df_prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df_disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df_netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df_hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df_prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df_disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_age = df_tvshows.copy()


# In[19]:


df_tvshows_age.drop(df_tvshows_age.loc[df_tvshows_age['Age'] == "NA"].index, inplace = True)
df_tvshows_age.drop(df_tvshows_age.loc[df_tvshows_age['Age'] == "NR"].index, inplace = True)
# df_tvshows_age = df_tvshows_age[df_tvshows_age.Age != "NA"]
df_tvshows_age['Age'] = df_tvshows_age['Age'].astype(int)


# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_age_tvshows = df_tvshows_age.loc[df_tvshows_age['Netflix'] == 1]
hulu_age_tvshows = df_tvshows_age.loc[df_tvshows_age['Hulu'] == 1]
prime_video_age_tvshows = df_tvshows_age.loc[df_tvshows_age['Prime Video'] == 1]
disney_age_tvshows = df_tvshows_age.loc[df_tvshows_age['Disney+'] == 1]

```

```
# In[21]:
```

```
df_tvshows_age_group = df_tvshows_age.copy()
```

```
# In[22]:
```

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_age.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[23]:
```

```
df_age_all_tvshows = df_tvshows_age
```

```
print('\nTV Shows with Age Rating are : \n')
df_age_all_tvshows.head(5)
```

```
# In[24]:
```

```
df_age_0_tvshows = df_tvshows_age.loc[df_tvshows_age['Age'] == 0]

print('\nTV Shows with All Age Rating are : \n')
df_age_0_tvshows.head(5)
```

```
# In[25]:
```

```
df_age_7_tvshows = df_tvshows_age.loc[df_tvshows_age['Age'] == 7]

print('\nTV Shows with 7+ Age Rating are : \n')
df_age_7_tvshows.head(5)
```

```
# In[26]:
```

```
df_age_13_tvshows = df_tvshows_age.loc[df_tvshows_age['Age'] == 13]

print('\nTV Shows with 13+ Age Rating are : \n')
df_age_13_tvshows.head(5)
```

```

# In[27]:


df_age_16_tvshows = df_tvshows_age.loc[df_tvshows_age['Age'] == 16]

print('\nTV Shows with 16+ Age Rating are : \n')
df_age_16_tvshows.head(5)

# In[28]:


df_age_18_tvshows = df_tvshows_age.loc[df_tvshows_age['Age'] == 18]

print('\nTV Shows with 18+ Age Rating are : \n')
df_age_18_tvshows.head(5)

# In[29]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_age['Age'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_age['Age'], ax = ax[1])
plt.show()

# In[30]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Age s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_age_tvshows['Age'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_age_tvshows['Age'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_age_tvshows['Age'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_age_tvshows['Age'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[31]:


def round_val(data):
    if str(data) != 'nan':
        return round(data)

# In[32]:


df_tvshows_age_group['Age Group'] = df_tvshows_age['Age'].apply(round_val)

```

```

age_values = df_tvshows_age_group['Age Group'].value_counts().sort_index(ascending = False).tolist()
age_index = df_tvshows_age_group['Age Group'].value_counts().sort_index(ascending = False).index

# age_values, age_index

# In[33]:


age_group_count = df_tvshows_age_group.groupby('Age Group')['Title'].count()
age_group_tvshows = df_tvshows_age_group.groupby('Age Group')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
age_group_data_tvshows = pd.concat([age_group_count, age_group_tvshows], axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
age_group_data_tvshows = age_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[34]:


# Age Group with TV Shows Counts - All Platforms Combined
age_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[35]:


age_group_data_tvshows.sort_values(by = 'Age Group', ascending = False)

# In[36]:


fig = px.bar(y = age_group_data_tvshows['TV Shows Count'],
              x = age_group_data_tvshows['Age Group'],
              color = age_group_data_tvshows['Age Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Age : '},
              title = 'TV Shows with Group Age : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[37]:


fig = px.pie(age_group_data_tvshows,
              names = age_group_data_tvshows['Age Group'],
              values = age_group_data_tvshows['TV Shows Count'],
              color = age_group_data_tvshows['TV Shows Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'TV Shows Count based on Age Group')
fig.show()

```

```

# In[38]:


df_age_group_high_tvshows = age_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_age_group_high_tvshows = df_age_group_high_tvshows.drop(['index'], axis = 1)
# filter = (age_group_data_tvshows['TV Shows Count'] == (age_group_data_tvshows['TV Shows
Count'].max()))
# df_age_group_high_tvshows = age_group_data_tvshows[filter]

# highestRated_tvshows = age_group_data_tvshows.loc[age_group_data_tvshows['TV Shows
Count'].idxmax()]

# print('\nAge with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_age_group_high_tvshows.head(5)

# In[39]:


df_age_group_low_tvshows = age_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_age_group_low_tvshows = df_age_group_low_tvshows.drop(['index'], axis = 1)
# filter = (age_group_data_tvshows['TV Shows Count'] == (age_group_data_tvshows['TV Shows
Count'].min()))
# df_age_group_low_tvshows = age_group_data_tvshows[filter]

# print('\nAge with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_age_group_low_tvshows.head(5)

# In[40]:


print(f'''
    Total '{df_tvshows_age['Age'].count()}' Titles are available on All Platforms, out of
which\n
    You Can Choose to see TV Shows from Total '{age_group_data_tvshows['Age
Group'].unique().shape[0]}' Age Group, They were Like this, \n

    {age_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[['Age
Group']].unique()} etc. \n

    The Age Group with Highest TV Shows Count have '{age_group_data_tvshows['TV Shows
Count'].max()}' TV Shows Available is '{df_age_group_high_tvshows['Age Group'][0]}', &\n
    The Age Group with Lowest TV Shows Count have '{age_group_data_tvshows['TV Shows
Count'].min()}' TV Shows Available is '{df_age_group_low_tvshows['Age Group'][0]}'
''')


# In[41]:


netflix_age_group_tvshows = age_group_data_tvshows[age_group_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_age_group_tvshows = netflix_age_group_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

netflix_age_group_high_tvshows = df_age_group_high_tvshows.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_age_group_high_tvshows = netflix_age_group_high_tvshows.drop(['index'], axis = 1)

```

```

netflix_age_group_low_tvshows = df_age_group_high_tvshows.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_age_group_low_tvshows = netflix_age_group_low_tvshows.drop(['index'], axis = 1)

netflix_age_group_high_tvshows.head(5)

# In[42]:


hulu_age_group_tvshows = age_group_data_tvshows[age_group_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_age_group_tvshows = hulu_age_group_tvshows.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

hulu_age_group_high_tvshows = df_age_group_high_tvshows.sort_values(by = 'Hulu', ascending =
False).reset_index()
hulu_age_group_high_tvshows = hulu_age_group_high_tvshows.drop(['index'], axis = 1)

hulu_age_group_low_tvshows = df_age_group_low_tvshows.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_age_group_low_tvshows = hulu_age_group_low_tvshows.drop(['index'], axis = 1)

hulu_age_group_high_tvshows.head(5)

# In[43]:


prime_video_age_group_tvshows = age_group_data_tvshows[age_group_data_tvshows['Prime
Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_age_group_tvshows = prime_video_age_group_tvshows.drop(['index', 'Netflix',
'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_age_group_high_tvshows = df_age_group_high_tvshows.sort_values(by = 'Prime
Video', ascending = False).reset_index()
prime_video_age_group_high_tvshows = prime_video_age_group_high_tvshows.drop(['index'],
axis = 1)

prime_video_age_group_low_tvshows = df_age_group_low_tvshows.sort_values(by = 'Prime
Video', ascending = True).reset_index()
prime_video_age_group_low_tvshows = prime_video_age_group_low_tvshows.drop(['index'],
axis = 1)

prime_video_age_group_high_tvshows.head(5)

# In[44]:


disney_age_group_tvshows = age_group_data_tvshows[age_group_data_tvshows['Disney+'] !=
0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_age_group_tvshows = disney_age_group_tvshows.drop(['index', 'Netflix', 'Hulu',
'Prime Video', 'TV Shows Count'], axis = 1)

disney_age_group_high_tvshows = df_age_group_high_tvshows.sort_values(by = 'Disney+', ascending =
False).reset_index()
disney_age_group_high_tvshows = disney_age_group_high_tvshows.drop(['index'], axis = 1)

```

```

disney_age_group_low_tvshows = df_age_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_age_group_low_tvshows = disney_age_group_low_tvshows.drop(['index'], axis = 1)

disney_age_group_high_tvshows.head(5)

# In[45]:


print(f'''
    The Age Group with Highest TV Shows Count Ever Got is
'{df_age_group_high_tvshows['Age Group'][0]}' : '{df_age_group_high_tvshows['TV Shows Count']].max()'\n
    The Age Group with Lowest TV Shows Count Ever Got is '{df_age_group_low_tvshows['Age Group'][0]}' : '{df_age_group_low_tvshows['TV Shows Count']].min()'\n

    The Age Group with Highest TV Shows Count on 'Netflix' is
'{netflix_age_group_high_tvshows['Age Group'][0]}' :
'{netflix_age_group_high_tvshows['Netflix']].max()'\n
    The Age Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_age_group_low_tvshows['Age Group'][0]}' :
'{netflix_age_group_low_tvshows['Netflix']].min()'\n

    The Age Group with Highest TV Shows Count on 'Hulu' is
'{hulu_age_group_high_tvshows['Age Group'][0]}' :
'{hulu_age_group_high_tvshows['Hulu']].max()'\n
    The Age Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_age_group_low_tvshows['Age Group'][0]}' :
'{hulu_age_group_low_tvshows['Hulu']].min()'\n

    The Age Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_age_group_high_tvshows['Age Group'][0]}' :
'{prime_video_age_group_high_tvshows['Prime Video']].max()'\n
    The Age Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_age_group_low_tvshows['Age Group'][0]}' :
'{prime_video_age_group_low_tvshows['Prime Video']].min()'\n

    The Age Group with Highest TV Shows Count on 'Disney+' is
'{disney_age_group_high_tvshows['Age Group'][0]}' :
'{disney_age_group_high_tvshows['Disney+']].max()'\n
    The Age Group with Lowest TV Shows Count on 'Disney+' is
'{disney_age_group_low_tvshows['Age Group'][0]}' :
'{disney_age_group_low_tvshows['Disney+']].min()'
''')


# In[46]:

```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.barplot(x = netflix_age_group_tvshows['Age Group'], y = netflix_age_group_tvshows['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_a_ax2 = sns.barplot(x = hulu_age_group_tvshows['Age Group'], y = hulu_age_group_tvshows['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_a_ax3 = sns.barplot(x = prime_video_age_group_tvshows['Age Group'], y = prime_video_age_group_tvshows['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_a_ax4 = sns.barplot(x = disney_age_group_tvshows['Age Group'], y = disney_age_group_tvshows['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

```

```

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])
h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])
d_a_ax4.title.set_text(labels[3])

plt.show()

# In[47]:


plt.figure(figsize = (20, 5))
sns.lineplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = age_group_data_tvshows['Age Group'], y = age_group_data_tvshows['Hulu'],
color = 'lightgreen')
sns.lineplot(x = age_group_data_tvshows['Age Group'], y = age_group_data_tvshows['Prime
Video'], color = 'lightblue')
sns.lineplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Age Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

# In[48]:


print(f'''
    Accross All Platforms Total Count of Age Group is '{age_group_data_tvshows['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Netflix' is '{netflix_age_group_tvshows['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Hulu' is '{hulu_age_group_tvshows['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Prime Video' is '{prime_video_age_group_tvshows['Age
Group'].unique().shape[0]}'\n
    Total Count of Age Group on 'Disney+' is '{disney_age_group_tvshows['Age
Group'].unique().shape[0]}'
''')


# In[49]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.lineplot(y = age_group_data_tvshows['Age Group'], x =
age_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_a_ax2 = sns.lineplot(y = age_group_data_tvshows['Age Group'], x =
age_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_a_ax3 = sns.lineplot(y = age_group_data_tvshows['Age Group'], x =
age_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_a_ax4 = sns.lineplot(y = age_group_data_tvshows['Age Group'], x =
age_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])

```

```

h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])
d_a_ax4.title.set_text(labels[3])

plt.show()

# In[50]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_a_ax1 = sns.barplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_a_ax2 = sns.barplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_a_ax3 = sns.barplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_a_ax4 = sns.barplot(x = age_group_data_tvshows['Age Group'], y =
age_group_data_tvshows['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_a_ax1.title.set_text(labels[0])
h_a_ax2.title.set_text(labels[1])
p_a_ax3.title.set_text(labels[2])
d_a_ax4.title.set_text(labels[3])

plt.show()

```

otttvshows_cast.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

```

```
# In[3]:\n\nimport pandas as pd\nimport numpy as np\nimport matplotlib.pyplot as plt\nimport seaborn as sns\nimport warnings\nimport collections\nimport plotly.express as px\nimport plotly.graph_objects as go\nimport nltk\nimport re\nfrom nltk.corpus import stopwords\nfrom nltk.tokenize import word_tokenize\nfrom nltk.probability import FreqDist\nfrom nltk.util import ngrams\nfrom plotly.subplots import make_subplots\nfrom plotly.offline import iplot, init_notebook_mode\nfrom wordcloud import WordCloud, STOPWORDS\nfrom pandas_profiling import ProfileReport\nget_ipython().run_line_magic('matplotlib', 'inline')\nwarnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'\n\npath = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'\n\ndf_tvshows = pd.read_csv(path + 'otttvshows.csv')\n\ndf_tvshows.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_tvshows)\n# profile
```

```
# In[7]:
```

```
def data_investigate(df):\n    print('No of Rows : ', df.shape[0])\n    print('No of Coloums : ', df.shape[1])\n    print('***'*25)\n    print('Colums Names : \n', df.columns)\n    print('***'*25)
```

```

print('Datatype of Columns : \n', df.dtypes)
print('*'*25)
print('Missing Values : ')
c = df.isnull().sum()
c = c[c > 0]
print(c)
print('*'*25)
print('Missing values %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('*'*25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

```

In[8]:

```
data_investigate(df_tvshows)
```

In[9]:

```

# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Casts
# df_tvshows = df_tvshows.drop(['Casts'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast

```

```

df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
```

`data_investigate(df_tvshows)`

```
# In[11]:
```

`df_tvshows.head()`

```
# In[12]:
```

`df_tvshows.describe()`

```

# In[13]: df_tvshows.corr()

# In[14]: # df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]: # df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]: # df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]: df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]

# In[18]: df_tvshows_casts = df_tvshows.copy()

# In[19]: df_tvshows_casts.drop(df_tvshows_casts.loc[df_tvshows_casts['Cast'] == "NA"].index, inplace = True)
# df_tvshows_casts = df_tvshows_casts[df_tvshows_casts.Cast != "NA"]

```

```

# df_tvshows_casts['Cast'] = df_tvshows_casts['Cast'].astype(str)

# In[20]:


df_tvshows_count_casts = df_tvshows_casts.copy()

# In[21]:


df_tvshows_cast = df_tvshows_casts.copy()

# In[22]:


# Create casts dict where key=name and value = number of casts

casts = {}

for i in df_tvshows_count_casts['Cast'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        casts[i] = len(i.split(','))
    else:
        casts[i] = 0

# Add this information to our dataframe as a new column

df_tvshows_count_casts['Number of Casts'] =
df_tvshows_count_casts['Cast'].map(casts).astype(int)

# In[23]:


df_tvshows_mixed_casts = df_tvshows_count_casts.copy()

# In[24]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_casts_tvshows = df_tvshows_count_casts.loc[df_tvshows_count_casts['Netflix'] == 1]
hulu_casts_tvshows = df_tvshows_count_casts.loc[df_tvshows_count_casts['Hulu'] == 1]
prime_video_casts_tvshows = df_tvshows_count_casts.loc[df_tvshows_count_casts['Prime
Video'] == 1]
disney_casts_tvshows = df_tvshows_count_casts.loc[df_tvshows_count_casts['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_count_casts.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map

```

```

sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

```

In[26]:

```

df_casts_most_tvshows = df_tvshows_count_casts.sort_values(by = 'Number of Casts',
ascending = False).reset_index()
df_casts_most_tvshows = df_casts_most_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_casts['Number of Casts'] == (df_tvshows_count_casts['Number of
Casts'].max()))
# df_casts_most_tvshows = df_tvshows_count_casts[filter]

# mostest_rated_tvshows = df_tvshows_count_casts.loc[df_tvshows_count_casts['Number of
Casts'].idxmax()]

print('\nTV Shows with Highest Ever Number of Casts are : \n')
df_casts_most_tvshows.head(5)

```

In[27]:

```

fig = px.bar(y = df_casts_most_tvshows['Title'][:15],
              x = df_casts_most_tvshows['Number of Casts'][:15],
              color = df_casts_most_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title = 'TV Shows with Highest Number of Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[28]:

```

df_casts_least_tvshows = df_tvshows_count_casts.sort_values(by = 'Number of Casts',
ascending = True).reset_index()
df_casts_least_tvshows = df_casts_least_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_casts['Number of Casts'] == (df_tvshows_count_casts['Number of
Casts'].min()))
# df_casts_least_tvshows = df_tvshows_count_casts[filter]

print('\nTV Shows with Lowest Ever Number of Casts are : \n')
df_casts_least_tvshows.head(5)

```

In[29]:

```

fig = px.bar(y = df_casts_least_tvshows['Title'][:15],
              x = df_casts_least_tvshows['Number of Casts'][:15],
              color = df_casts_least_tvshows['Number of Casts'][:15],

```

```

color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
title  = 'TV Shows with Lowest Number of Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
    Total '{df_tvshows_count_casts['Number of Casts'].unique().shape[0]}' unique Number
of Casts s were Given, They were Like this,\n

    {df_tvshows_count_casts.sort_values(by = 'Number of Casts', ascending =
False)['Number of Casts'].unique()}\n

    The Highest Number of Casts Ever Any TV Show Got is
'{df_casts_most_tvshows['Title'][0]}' : '{df_casts_most_tvshows['Number of
Casts'].max()}'\n

    The Lowest Number of Casts Ever Any TV Show Got is
'{df_casts_least_tvshows['Title'][0]}' : '{df_casts_least_tvshows['Number of
Casts'].min()}'\n
''')


# In[31]:


netflix_casts_most_tvshows =
df_casts_most_tvshows.loc[df_casts_most_tvshows['Netflix']==1].reset_index()
netflix_casts_most_tvshows = netflix_casts_most_tvshows.drop(['index'], axis = 1)

netflix_casts_least_tvshows =
df_casts_least_tvshows.loc[df_casts_least_tvshows['Netflix']==1].reset_index()
netflix_casts_least_tvshows = netflix_casts_least_tvshows.drop(['index'], axis = 1)

netflix_casts_most_tvshows.head(5)

# In[32]:


fig = px.bar(y = netflix_casts_most_tvshows['Title'][:15],
              x = netflix_casts_most_tvshows['Number of Casts'][:15],
              color = netflix_casts_most_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title  = 'TV Shows with Highest Number of Casts : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_casts_least_tvshows['Title'][:15],
              x = netflix_casts_least_tvshows['Number of Casts'][:15],

```

```

color = netflix_casts_least_tvshows['Number of Casts'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
title  = 'TV Shows with Lowest Number of Casts : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[34]:

```

hulu_casts_most_tvshows =
df_casts_most_tvshows.loc[df_casts_most_tvshows['Hulu']==1].reset_index()
hulu_casts_most_tvshows = hulu_casts_most_tvshows.drop(['index'], axis = 1)

hulu_casts_least_tvshows =
df_casts_least_tvshows.loc[df_casts_least_tvshows['Hulu']==1].reset_index()
hulu_casts_least_tvshows = hulu_casts_least_tvshows.drop(['index'], axis = 1)

hulu_casts_most_tvshows.head(5)

```

In[35]:

```

fig = px.bar(y = hulu_casts_most_tvshows['Title'][:15],
              x = hulu_casts_most_tvshows['Number of Casts'][:15],
              color = hulu_casts_most_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title  = 'TV Shows with Highest Number of Casts : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[36]:

```

fig = px.bar(y = hulu_casts_least_tvshows['Title'][:15],
              x = hulu_casts_least_tvshows['Number of Casts'][:15],
              color = hulu_casts_least_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title  = 'TV Shows with Lowest Number of Casts : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[37]:

```

prime_video_casts_most_tvshows = df_casts_most_tvshows.loc[df_casts_most_tvshows['Prime
Video']==1].reset_index()
prime_video_casts_most_tvshows = prime_video_casts_most_tvshows.drop(['index'], axis = 1)

prime_video_casts_least_tvshows = df_casts_least_tvshows.loc[df_casts_least_tvshows['Prime
Video']==1].reset_index()
prime_video_casts_least_tvshows = prime_video_casts_least_tvshows.drop(['index'], axis = 1)

```

```

prime_video_casts_most_tvshows.head(5)

# In[38]:


fig = px.bar(y = prime_video_casts_most_tvshows['Title'][:15],
              x = prime_video_casts_most_tvshows['Number of Casts'][:15],
              color = prime_video_casts_most_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title = 'TV Shows with Highest Number of Casts : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[39]:


fig = px.bar(y = prime_video_casts_least_tvshows['Title'][:15],
              x = prime_video_casts_least_tvshows['Number of Casts'][:15],
              color = prime_video_casts_least_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title = 'TV Shows with Lowest Number of Casts : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[40]:


disney_casts_most_tvshows =
df_casts_most_tvshows.loc[df_casts_most_tvshows['Disney+']==1].reset_index()
disney_casts_most_tvshows = disney_casts_most_tvshows.drop(['index'], axis = 1)

disney_casts_least_tvshows =
df_casts_least_tvshows.loc[df_casts_least_tvshows['Disney+']==1].reset_index()
disney_casts_least_tvshows = disney_casts_least_tvshows.drop(['index'], axis = 1)

disney_casts_most_tvshows.head(5)

```

```

# In[41]:


fig = px.bar(y = disney_casts_most_tvshows['Title'][:15],
              x = disney_casts_most_tvshows['Number of Casts'][:15],
              color = disney_casts_most_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title = 'TV Shows with Highest Number of Casts : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[42]:
```

```

fig = px.bar(y = disney_casts_least_tvshows['Title'][:15],
              x = disney_casts_least_tvshows['Number of Casts'][:15],
              color = disney_casts_least_tvshows['Number of Casts'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Casts'},
              title = 'TV Shows with Lowest Number of Casts : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''
    The TV Show with Highest Number of Casts Ever Got is
'{df_casts_most_tvshows['Title'][0]}' : '{df_casts_most_tvshows['Number of
Casts']].max()}'\n
    The TV Show with Lowest Number of Casts Ever Got is
'{df_casts_least_tvshows['Title'][0]}' : '{df_casts_least_tvshows['Number of
Casts']].min()}'\n

    The TV Show with Highest Number of Casts on 'Netflix' is
'{netflix_casts_most_tvshows['Title'][0]}' : '{netflix_casts_most_tvshows['Number of
Casts']].max()}'\n
    The TV Show with Lowest Number of Casts on 'Netflix' is
'{netflix_casts_least_tvshows['Title'][0]}' : '{netflix_casts_least_tvshows['Number of
Casts']].min()}'\n

    The TV Show with Highest Number of Casts on 'Hulu' is
'{hulu_casts_most_tvshows['Title'][0]}' : '{hulu_casts_most_tvshows['Number of
Casts']].max()}'\n
    The TV Show with Lowest Number of Casts on 'Hulu' is
'{hulu_casts_least_tvshows['Title'][0]}' : '{hulu_casts_least_tvshows['Number of
Casts']].min()}'\n

    The TV Show with Highest Number of Casts on 'Prime Video' is
'{prime_video_casts_most_tvshows['Title'][0]}' : '{prime_video_casts_most_tvshows['Number
of Casts']].max()}'\n
    The TV Show with Lowest Number of Casts on 'Prime Video' is
'{prime_video_casts_least_tvshows['Title'][0]}' : '{prime_video_casts_least_tvshows['Number
of Casts']].min()}'\n

    The TV Show with Highest Number of Casts on 'Disney+' is
'{disney_casts_most_tvshows['Title'][0]}' : '{disney_casts_most_tvshows['Number of
Casts']].max()}'\n
    The TV Show with Lowest Number of Casts on 'Disney+' is
'{disney_casts_least_tvshows['Title'][0]}' : '{disney_casts_least_tvshows['Number of
Casts']].min()}'\n
''')


# In[44]:


print(f'''
    Across All Platforms the Average Number of Casts is
'{round(df_tvshows_count_casts['Number of Casts'].mean(), ndigits = 2)}'\n

```

```
The Average Number of Casts on 'Netflix' is '{round(netflix_casts_tvshows['Number of
Casts'].mean(), ndigits = 2)}'\n
The Average Number of Casts on 'Hulu' is '{round(hulu_casts_tvshows['Number of
Casts'].mean(), ndigits = 2)}'\n
The Average Number of Casts on 'Prime Video' is
'{round(prime_video_casts_tvshows['Number of Casts'].mean(), ndigits = 2)}'\n
The Average Number of Casts on 'Disney+' is '{round(disney_casts_tvshows['Number of
Casts'].mean(), ndigits = 2)}'\n
'''')
```

```
# In[45]:
```

```
print(f'''
    Accross All Platforms Total Count of Cast is '{df_tvshows_count_casts['Number of
Casts'].max()}'\n
    Total Count of Cast on 'Netflix' is '{netflix_casts_tvshows['Number of
Casts'].max()}'\n
    Total Count of Cast on 'Hulu' is '{hulu_casts_tvshows['Number of Casts'].max()}'\n
    Total Count of Cast on 'Prime Video' is '{prime_video_casts_tvshows['Number of
Casts'].max()}'\n
    Total Count of Cast on 'Disney+' is '{disney_casts_tvshows['Number of
Casts'].max()}'\n
''')
```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_count_casts['Number of Casts'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_count_casts['Number of Casts'], ax = ax[1])
plt.show()
```

```
# In[47]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Number of Casts s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_casts_tvshows['Number of Casts'], color = 'lightblue', legend =
True, kde = True)
sns.histplot(netflix_casts_tvshows['Number of Casts'], color = 'red', legend = True, kde =
True)
sns.histplot(hulu_casts_tvshows['Number of Casts'], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_casts_tvshows['Number of Casts'], color = 'darkblue', legend = True,
kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

```
# In[48]:
```

```

df_lan = df_tvshows_cast['Cast'].str.split(',').apply(pd.Series).stack()
del df_tvshows_cast['Cast']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Cast'
df_tvshows_cast = df_tvshows_cast.join(df_lan)
df_tvshows_cast.drop_duplicates(inplace = True)

# In[49]:
```

```

df_tvshows_cast.head(5)
```

```

# In[50]:
```

```

cast_count = df_tvshows_cast.groupby('Cast')['Title'].count()
cast_tvshows = df_tvshows_cast.groupby('Cast')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
cast_data_tvshows = pd.concat([cast_count, cast_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
cast_data_tvshows = cast_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[51]:
```

```

# Cast with TV Shows Counts - All Platforms Combined
cast_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]
```

```

# In[52]:
```

```

fig = px.bar(x = cast_data_tvshows['Cast'][:50],
              y = cast_data_tvshows['TV Shows Count'][:50],
              color = cast_data_tvshows['TV Shows Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Cast', 'y' : 'TV Shows Count'},
              title = 'Major Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```

# In[53]:
```

```

df_cast_high_tvshows = cast_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False).reset_index()
df_cast_high_tvshows = df_cast_high_tvshows.drop(['index'], axis = 1)
# filter = (cast_data_tvshows['TV Shows Count'] == (cast_data_tvshows['TV Shows
Count'].max()))
# df_cast_high_tvshows = cast_data_tvshows[filter]

# highestRated_tvshows = cast_data_tvshows.loc[cast_data_tvshows['TV Shows
Count'].idxmax()]

print('\nCast with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_cast_high_tvshows.head(5)
```

```

# In[54]:


fig = px.bar(y = df_cast_high_tvshows['Cast'][:15],
              x = df_cast_high_tvshows['TV Shows Count'][:15],
              color = df_cast_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
              title = 'Cast with Highest TV Shows : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[55]:


df_cast_low_tvshows = cast_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
True).reset_index()
df_cast_low_tvshows = df_cast_low_tvshows.drop(['index'], axis = 1)
# filter = (cast_data_tvshows['TV Shows Count'] == (cast_data_tvshows['TV Shows
Count'].min()))
# df_cast_low_tvshows = cast_data_tvshows[filter]

print('\nCast with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_cast_low_tvshows.head(5)

# In[56]:


fig = px.bar(y = df_cast_low_tvshows['Cast'][:15],
              x = df_cast_low_tvshows['TV Shows Count'][:15],
              color = df_cast_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
              title = 'Cast with Lowest TV Shows Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[57]:


print(f'''
      Total '{cast_data_tvshows['Cast'].unique().shape[0]}' unique Cast Count s were Given,
      They were Like this,\n

      {cast_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Cast'].unique()[:5]}\n

      The Highest Ever TV Shows Count Ever Any TV Show Got is
      '{df_cast_high_tvshows['Cast'][0]}' : '{df_cast_high_tvshows['TV Shows Count'].max()}'\n

      The Lowest Ever TV Shows Count Ever Any TV Show Got is
      '{df_cast_low_tvshows['Cast'][0]}' : '{df_cast_low_tvshows['TV Shows Count'].min()}'\n
      ''')

```

```
# In[58]:
```

```
fig = px.pie(cast_data_tvshows[:10], names = 'Cast', values = 'TV Shows Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Cast')
fig.show()
```



```
# In[59]:
```

```
# netflix_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_cast_tvshows = netflix_cast_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_cast_high_tvshows = df_cast_high_tvshows.sort_values(by = 'Netflix', ascending =
False).reset_index()
netflix_cast_high_tvshows = netflix_cast_high_tvshows.drop(['index'], axis = 1)

netflix_cast_low_tvshows = df_cast_high_tvshows.sort_values(by = 'Netflix', ascending =
True).reset_index()
netflix_cast_low_tvshows = netflix_cast_low_tvshows.drop(['index'], axis = 1)

netflix_cast_high_tvshows.head(5)
```



```
# In[60]:
```

```
fig = px.bar(x = netflix_cast_high_tvshows['Cast'][:15],
y = netflix_cast_high_tvshows['Netflix'][:15],
color = netflix_cast_high_tvshows['Netflix'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
title = 'Cast with Highest TV Shows : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[61]:
```

```
# hulu_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_cast_tvshows = hulu_cast_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_cast_high_tvshows = df_cast_high_tvshows.sort_values(by = 'Hulu', ascending =
False).reset_index()
hulu_cast_high_tvshows = hulu_cast_high_tvshows.drop(['index'], axis = 1)

hulu_cast_low_tvshows = df_cast_high_tvshows.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_cast_low_tvshows = hulu_cast_low_tvshows.drop(['index'], axis = 1)

hulu_cast_high_tvshows.head(5)
```

```

# In[62]:


fig = px.bar(x = hulu_cast_high_tvshows['Cast'][:15],
              y = hulu_cast_high_tvshows['Hulu'][:15],
              color = hulu_cast_high_tvshows['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
              title  = 'Cast with Highest TV Shows : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


# prime_video_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_cast_tvshows = prime_video_cast_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_cast_high_tvshows = df_cast_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_cast_high_tvshows = prime_video_cast_high_tvshows.drop(['index'], axis = 1)

prime_video_cast_low_tvshows = df_cast_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_cast_low_tvshows = prime_video_cast_low_tvshows.drop(['index'], axis = 1)

prime_video_cast_high_tvshows.head(5)

# In[64]:


fig = px.bar(x = prime_video_cast_high_tvshows['Cast'][:15],
              y = prime_video_cast_high_tvshows['Prime Video'][:15],
              color = prime_video_cast_high_tvshows['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
              title  = 'Cast with Highest TV Shows : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[65]:


# disney_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_cast_tvshows = disney_cast_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_cast_high_tvshows = df_cast_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_cast_high_tvshows = disney_cast_high_tvshows.drop(['index'], axis = 1)

```

```

disney_cast_low_tvshows = df_cast_high_tvshows.sort_values(by = 'Disney+', ascending =
True).reset_index()
disney_cast_low_tvshows = disney_cast_low_tvshows.drop(['index'], axis = 1)

disney_cast_high_tvshows.head(5)

# In[66]:


fig = px.bar(x = disney_cast_high_tvshows['Cast'][:15],
              y = disney_cast_high_tvshows['Disney+'][:15],
              color = disney_cast_high_tvshows['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Cast', 'x' : 'TV Shows Count'},
              title = 'Cast with Highest TV Shows : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[67]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(cast_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(cast_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Netflix'] != 0].sort_values(by =
='Netflix', ascending = False).reset_index()
netflix_cast_tvshows = netflix_cast_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

hulu_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Hulu'] != 0].sort_values(by =
='Hulu', ascending = False).reset_index()
hulu_cast_tvshows = hulu_cast_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+',
'TV Shows Count'], axis = 1)

prime_video_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Prime Video'] !=
0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_cast_tvshows = prime_video_cast_tvshows.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'TV Shows Count'], axis = 1)

disney_cast_tvshows = cast_data_tvshows[cast_data_tvshows['Disney+'] != 0].sort_values(by =
='Disney+', ascending = False).reset_index()
disney_cast_tvshows = disney_cast_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video',
'TV Shows Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))

```

```

plt.title('Cast TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_cast_tvshows['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_cast_tvshows['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_cast_tvshows['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_cast_tvshows['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''
    The Cast with Highest TV Shows Count Ever Got is '{df_cast_high_tvshows['Cast'][0]}'
: '{df_cast_high_tvshows['TV Shows Count'].max()}'\n
    The Cast with Lowest TV Shows Count Ever Got is '{df_cast_low_tvshows['Cast'][0]}'
: '{df_cast_low_tvshows['TV Shows Count'].min()}'\n

    The Cast with Highest TV Shows Count on 'Netflix' is
'{netflix_cast_high_tvshows['Cast'][0]} : '{netflix_cast_high_tvshows['Netflix'].max()}'\n
    The Cast with Lowest TV Shows Count on 'Netflix' is
'{netflix_cast_low_tvshows['Cast'][0]} : '{netflix_cast_low_tvshows['Netflix'].min()}'\n

    The Cast with Highest TV Shows Count on 'Hulu' is
'{hulu_cast_high_tvshows['Cast'][0]} : '{hulu_cast_high_tvshows['Hulu'].max()}'\n
    The Cast with Lowest TV Shows Count on 'Hulu' is '{hulu_cast_low_tvshows['Cast'][0]}'
: '{hulu_cast_low_tvshows['Hulu'].min()}'\n

    The Cast with Highest TV Shows Count on 'Prime Video' is
'{prime_video_cast_high_tvshows['Cast'][0]} : '{prime_video_cast_high_tvshows['Prime
Video'].max()}'\n
    The Cast with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_cast_low_tvshows['Cast'][0]} : '{prime_video_cast_low_tvshows['Prime
Video'].min()}'\n

    The Cast with Highest TV Shows Count on 'Disney+' is
'{disney_cast_high_tvshows['Cast'][0]} : '{disney_cast_high_tvshows['Disney+'].max()}'\n
    The Cast with Lowest TV Shows Count on 'Disney+' is
'{disney_cast_low_tvshows['Cast'][0]} : '{disney_cast_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[71]:

```

# Distribution of tvshows cast in each platform
plt.figure(figsize = (20, 5))
plt.title('Cast with TV Shows Count for All Platforms')
sns.violinplot(x = cast_data_tvshows['TV Shows Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

```

```
# In[72]:
```

```
# Distribution of Cast TV Shows Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_cast_tvshows['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_cast_tvshows['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_cast_tvshows['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_cast_tvshows['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()
```

```
# In[73]:
```

```
print(f'''
    Accross All Platforms the Average TV Shows Count of Cast is
'{round(cast_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Cast on 'Netflix' is
'{round(netflix_cast_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Cast on 'Hulu' is
'{round(hulu_cast_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Cast on 'Prime Video' is
'{round(prime_video_cast_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Cast on 'Disney+' is
'{round(disney_cast_tvshows['Disney+'].mean(), ndigits = 2)}'
    ''')
```

```
# In[74]:
```

```
print(f'''
    Accross All Platforms Total Count of Cast is
'{cast_data_tvshows['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Netflix' is
'{netflix_cast_tvshows['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Hulu' is '{hulu_cast_tvshows['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Prime Video' is
'{prime_video_cast_tvshows['Cast'].unique().shape[0]}'\n
    Total Count of Cast on 'Disney+' is
'{disney_cast_tvshows['Cast'].unique().shape[0]}'
    ''')
```

```
# In[75]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = cast_data_tvshows['Cast'][:10], y = cast_data_tvshows['Netflix'][:10],
color = 'red')
sns.lineplot(x = cast_data_tvshows['Cast'][:10], y = cast_data_tvshows['Hulu'][:10], color =
'lightgreen')
sns.lineplot(x = cast_data_tvshows['Cast'][:10], y = cast_data_tvshows['Prime Video'][:10],
color = 'lightblue')
sns.lineplot(x = cast_data_tvshows['Cast'][:10], y = cast_data_tvshows['Disney+'][:10],
color = 'darkblue')
```

```

plt.xlabel('Cast', fontsize = 20)
plt.ylabel('TV Shows Count', fontsize = 20)
plt.show()

# In[76]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_c_ax1 = sns.lineplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_c_ax2 = sns.lineplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_c_ax3 = sns.lineplot(y = cast_data_tvshows['Cast'][:10], x = cast_data_tvshows['Prime
Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_c_ax4 = sns.lineplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

# In[77]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_c_ax1 = sns.barplot(y = netflix_cast_tvshows['Cast'][:10], x =
netflix_cast_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_c_ax2 = sns.barplot(y = hulu_cast_tvshows['Cast'][:10], x =
hulu_cast_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_c_ax3 = sns.barplot(y = prime_video_cast_tvshows['Cast'][:10], x =
prime_video_cast_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_c_ax4 = sns.barplot(y = disney_cast_tvshows['Cast'][:10], x =
disney_cast_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

# In[78]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Cast TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

```

```
sns.kdeplot(netflix_cast_tvshows['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_cast_tvshows['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_cast_tvshows['Prime Video'][:10], color = 'lightblue', legend = True)
sns.kdeplot(disney_cast_tvshows['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()
```

```
# In[79]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_c_ax1 = sns.barplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_c_ax2 = sns.barplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_c_ax3 = sns.barplot(y = cast_data_tvshows['Cast'][:10], x = cast_data_tvshows['Prime
Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_c_ax4 = sns.barplot(y = cast_data_tvshows['Cast'][:10], x =
cast_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[80]:
```

```
df_tvshows_mixed_casts.drop(df_tvshows_mixed_casts.loc[df_tvshows_mixed_casts['Cast'] ==
"NA"].index, inplace = True)
# df_tvshows_mixed_casts = df_tvshows_mixed_casts[df_tvshows_mixed_casts.Cast != "NA"]
df_tvshows_mixed_casts.drop(df_tvshows_mixed_casts.loc[df_tvshows_mixed_casts['Number of
Casts'] == 1].index, inplace = True)
```

```
# In[81]:
```

```
df_tvshows_mixed_casts.head(5)
```

```
# In[82]:
```

```
mixed_casts_count = df_tvshows_mixed_casts.groupby('Cast')['Title'].count()
mixed_casts_tvshows = df_tvshows_mixed_casts.groupby('Cast')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
mixed_casts_data_tvshows = pd.concat([mixed_casts_count, mixed_casts_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count', 'Cast' : 'Mixed Cast'})
mixed_casts_data_tvshows = mixed_casts_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)
```

```

# In[83]:


mixed_casts_data_tvshows.head(5)

# In[84]:


# Mixed Cast with TV Shows Counts - All Platforms Combined
mixed_casts_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[85]:


df_mixed_casts_high_tvshows = mixed_casts_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_mixed_casts_high_tvshows = df_mixed_casts_high_tvshows.drop(['index'], axis = 1)
# filter = (mixed_casts_data_tvshows['TV Shows Count'] == (mixed_casts_data_tvshows['TV
Shows Count'].max()))
# df_mixed_casts_high_tvshows = mixed_casts_data_tvshows[filter]

# highestRated_tvshows = mixed_casts_data_tvshows.loc[mixed_casts_data_tvshows['TV Shows
Count'].idxmax()]

print('\nMixed Cast with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_casts_high_tvshows.head(5)

# In[86]:


fig = px.bar(y = df_mixed_casts_high_tvshows['Mixed Cast'][:15],
              x = df_mixed_casts_high_tvshows['TV Shows Count'][:15],
              color = df_mixed_casts_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Cast'},
              title = 'TV Shows with Highest Number of Mixed Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[87]:


df_mixed_casts_low_tvshows = mixed_casts_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_mixed_casts_low_tvshows = df_mixed_casts_low_tvshows.drop(['index'], axis = 1)
# filter = (mixed_casts_data_tvshows['TV Shows Count'] == (mixed_casts_data_tvshows['TV
Shows Count'].min()))
# df_mixed_casts_low_tvshows = mixed_casts_data_tvshows[filter]

print('\nMixed Cast with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_casts_low_tvshows.head(5)

# In[88]:

```

```

fig = px.bar(y = df_mixed_casts_low_tvshows['Mixed Cast'][:15],
              x = df_mixed_casts_low_tvshows['TV Shows Count'][:15],
              color = df_mixed_casts_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Cast'},
              title  = 'TV Shows with Lowest Number of Mixed Casts : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[89]:


print(f'''
    Total '{df_tvshows_casts['Cast'].count()}' Titles are available on All Platforms, out
of which\n
    You Can Choose to see TV Shows from Total '{mixed_casts_data_tvshows['Mixed
Cast'].unique().shape[0]}' Mixed Cast, They were Like this, \n
        {mixed_casts_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Mixed Cast'].head(5).unique()} etc. \n
    The Mixed Cast with Highest TV Shows Count have '{mixed_casts_data_tvshows['TV Shows
Count'].max()}' TV Shows Available is '{df_mixed_casts_high_tvshows['Mixed Cast'][0]}', &\n
    The Mixed Cast with Lowest TV Shows Count have '{mixed_casts_data_tvshows['TV Shows
Count'].min()}' TV Shows Available is '{df_mixed_casts_low_tvshows['Mixed Cast'][0]}'
''')


# In[90]:


fig = px.pie(mixed_casts_data_tvshows[:10], names = 'Mixed Cast', values = 'TV Shows
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Mixed Cast')
fig.show()

# In[91]:


# netflix_mixed_casts_tvshows =
mixed_casts_data_tvshows[mixed_casts_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
# netflix_mixed_casts_tvshows = netflix_mixed_casts_tvshows.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_mixed_casts_high_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_mixed_casts_high_tvshows = netflix_mixed_casts_high_tvshows.drop(['index'], axis =
1)

netflix_mixed_casts_low_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_mixed_casts_low_tvshows = netflix_mixed_casts_low_tvshows.drop(['index'], axis =
1)

netflix_mixed_casts_high_tvshows.head(5)

```

```
# In[92]:
```

```
# hulu_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_mixed_casts_tvshows = hulu_mixed_casts_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_casts_high_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_casts_high_tvshows = hulu_mixed_casts_high_tvshows.drop(['index'], axis = 1)

hulu_mixed_casts_low_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_mixed_casts_low_tvshows = hulu_mixed_casts_low_tvshows.drop(['index'], axis = 1)

hulu_mixed_casts_high_tvshows.head(5)
```



```
# In[93]:
```

```
# prime_video_mixed_casts_tvshows =
mixed_casts_data_tvshows[mixed_casts_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_casts_tvshows = prime_video_mixed_casts_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_casts_high_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_casts_high_tvshows = prime_video_mixed_casts_high_tvshows.drop(['index'], axis = 1)

prime_video_mixed_casts_low_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_mixed_casts_low_tvshows = prime_video_mixed_casts_low_tvshows.drop(['index'], axis = 1)

prime_video_mixed_casts_high_tvshows.head(5)
```



```
# In[94]:
```

```
# disney_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_casts_tvshows = disney_mixed_casts_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_mixed_casts_high_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_casts_high_tvshows = disney_mixed_casts_high_tvshows.drop(['index'], axis = 1)

disney_mixed_casts_low_tvshows = df_mixed_casts_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_casts_low_tvshows = disney_mixed_casts_low_tvshows.drop(['index'], axis = 1)

disney_mixed_casts_high_tvshows.head(5)
```

```
# In[95]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_casts_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_casts_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()
```

```
# In[96]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_casts_tvshows = netflix_mixed_casts_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_casts_tvshows = hulu_mixed_casts_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_casts_tvshows = prime_video_mixed_casts_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_mixed_casts_tvshows = mixed_casts_data_tvshows[mixed_casts_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_casts_tvshows = disney_mixed_casts_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)
```

```
# In[97]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Cast TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_casts_tvshows['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_casts_tvshows['Netflix'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_mixed_casts_tvshows['Hulu'][:100], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_mixed_casts_tvshows['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

```
# In[98]:
```

```

print(f'''
    The Mixed Cast with Highest TV Shows Count Ever Got is
'{df_mixed_casts_high_tvshows['Mixed Cast'][0]} : '{df_mixed_casts_high_tvshows['TV Shows
Count'].max()}'\n
    The Mixed Cast with Lowest TV Shows Count Ever Got is
'{df_mixed_casts_low_tvshows['Mixed Cast'][0]} : '{df_mixed_casts_low_tvshows['TV Shows
Count'].min()}'\n

    The Mixed Cast with Highest TV Shows Count on 'Netflix' is
'{netflix_mixed_casts_high_tvshows['Mixed Cast'][0]} :
'{netflix_mixed_casts_high_tvshows['Netflix'].max()}'\n
    The Mixed Cast with Lowest TV Shows Count on 'Netflix' is
'{netflix_mixed_casts_low_tvshows['Mixed Cast'][0]} :
'{netflix_mixed_casts_low_tvshows['Netflix'].min()}'\n

    The Mixed Cast with Highest TV Shows Count on 'Hulu' is
'{hulu_mixed_casts_high_tvshows['Mixed Cast'][0]} :
'{hulu_mixed_casts_high_tvshows['Hulu'].max()}'\n
    The Mixed Cast with Lowest TV Shows Count on 'Hulu' is
'{hulu_mixed_casts_low_tvshows['Mixed Cast'][0]} :
'{hulu_mixed_casts_low_tvshows['Hulu'].min()}'\n

    The Mixed Cast with Highest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_casts_high_tvshows['Mixed Cast'][0]} :
'{prime_video_mixed_casts_high_tvshows['Prime Video'].max()}'\n
    The Mixed Cast with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_casts_low_tvshows['Mixed Cast'][0]} :
'{prime_video_mixed_casts_low_tvshows['Prime Video'].min()}'\n

    The Mixed Cast with Highest TV Shows Count on 'Disney+' is
'{disney_mixed_casts_high_tvshows['Mixed Cast'][0]} :
'{disney_mixed_casts_high_tvshows['Disney+'].max()}'\n
    The Mixed Cast with Lowest TV Shows Count on 'Disney+' is
'{disney_mixed_casts_low_tvshows['Mixed Cast'][0]} :
'{disney_mixed_casts_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Accross All Platforms the Average TV Shows Count of Mixed Cast is
'{round(mixed_casts_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Cast on 'Netflix' is
'{round(netflix_mixed_casts_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Cast on 'Hulu' is
'{round(hulu_mixed_casts_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Cast on 'Prime Video' is
'{round(prime_video_mixed_casts_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Cast on 'Disney+' is
'{round(disney_mixed_casts_tvshows['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```
print(f'''
```

```

    Accross All Platforms Total Count of Mixed Cast is '{mixed_casts_data_tvshows['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Netflix' is '{netflix_mixed_casts_tvshows['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Hulu' is '{hulu_mixed_casts_tvshows['Mixed
Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Prime Video' is
'{prime_video_mixed_casts_tvshows['Mixed Cast'].unique().shape[0]}'\n
    Total Count of Mixed Cast on 'Disney+' is '{disney_mixed_casts_tvshows['Mixed
Cast'].unique().shape[0]}'\n
    ''')

```

```
# In[101]:
```

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:5], y =
mixed_casts_data_tvshows['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:5], y =
mixed_casts_data_tvshows['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:5], y =
mixed_casts_data_tvshows['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:5], y =
mixed_casts_data_tvshows['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Cast', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

```
# In[102]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_c_ax1 = sns.barplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_c_ax2 = sns.barplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_c_ax3 = sns.barplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_c_ax4 = sns.barplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_c_ax1.title.set_text(labels[0])
h_c_ax2.title.set_text(labels[1])
p_c_ax3.title.set_text(labels[2])
d_c_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[103]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_mc_ax1 = sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])

```

```

h_mc_ax2 = sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_mc_ax3 = sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_mc_ax4 = sns.lineplot(x = mixed_casts_data_tvshows['Mixed Cast'][:10], y =
mixed_casts_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mc_ax1.title.set_text(labels[0])
h_mc_ax2.title.set_text(labels[1])
p_mc_ax3.title.set_text(labels[2])
d_mc_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Cast TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_casts_tvshows['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_casts_tvshows['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_casts_tvshows['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_casts_tvshows['Disney+'][:50], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()


# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_mc_ax1 = sns.barplot(x = netflix_mixed_casts_tvshows['Mixed Cast'][:10], y =
netflix_mixed_casts_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mc_ax2 = sns.barplot(x = hulu_mixed_casts_tvshows['Mixed Cast'][:10], y =
hulu_mixed_casts_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mc_ax3 = sns.barplot(x = prime_video_mixed_casts_tvshows['Mixed Cast'][:10], y =
prime_video_mixed_casts_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_mc_ax4 = sns.barplot(x = disney_mixed_casts_tvshows['Mixed Cast'][:10], y =
disney_mixed_casts_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mc_ax1.title.set_text(labels[0])
h_mc_ax2.title.set_text(labels[1])
p_mc_ax3.title.set_text(labels[2])
d_mc_ax4.title.set_text(labels[3])

plt.show()

```

In[106]:

```
fig = go.Figure(go.Funnel(y = mixed_casts_data_tvshows['Mixed Cast'][:10], x =  
mixed_casts_data_tvshows['TV Shows Count'][:10]))  
fig.show()
```

otttvshows_country.ipynb

```
#!/usr/bin/env python  
# coding: utf-8  
  
# In[1]:  
  
# !pip install git+https://github.com/alberanid/imdbpy  
# !pip install pandas  
# !pip install numpy  
# !pip install matplotlib  
# !pip install seaborn  
# !pip install pandas_profiling --upgrade  
# !pip install plotly  
# !pip install wordcloud  
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset  
# Import File from Loacal Drive  
# from google.colab import files  
# data_to_load = files.upload()  
# from google.colab import drive  
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import warnings  
import collections  
import plotly.express as px  
import plotly.graph_objects as go  
import nltk  
import re  
from nltk.corpus import stopwords  
from nltk.tokenize import word_tokenize  
from nltk.probability import FreqDist  
from nltk.util import ngrams  
from plotly.subplots import make_subplots  
from plotly.offline import iplot, init_notebook_mode  
from wordcloud import WordCloud, STOPWORDS  
from pandas_profiling import ProfileReport  
get_ipython().run_line_magic('matplotlib', 'inline')
```

```

warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()

# In[6]:


# profile = ProfileReport(df_tvshows)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_tvshows)

# In[9]:

```

```

# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)

```

```

# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]: 

data_investigate(df_tvshows)

# In[11]: 

df_tvshows.head()

# In[12]: 

df_tvshows.describe()

# In[13]: 

df_tvshows.corr()

# In[14]: 

# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]: 

# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

```

```

# udf_tvshows

# In[16]:


# df_netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df_hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df_prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df_disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df_netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df_hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df_prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df_disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_countries = df_tvshows.copy()


# In[19]:


df_tvshows_countries.drop(df_tvshows_countries.loc[df_tvshows_countries['Country'] == "NA"].index, inplace = True)
# df_tvshows_countries = df_tvshows_countries[df_tvshows_countries.Country != "NA"]
# df_tvshows_countries['Country'] = df_tvshows_countries['Country'].astype(str)


# In[20]:


df_tvshows_count_countries = df_tvshows_countries.copy()


# In[21]:


df_tvshows_country = df_tvshows_countries.copy()


# In[22]:


# Create countries dict where key=name and value = number of countries

countries = {}

for i in df_tvshows_count_countries['Country'].dropna():
    if i != "NA":

```

```

        #print(i,len(i.split(',')))
        countries[i] = len(i.split(','))
    else:
        countries[i] = 0

# Add this information to our dataframe as a new column

df_tvshows_count_countries['Number of Countries'] =
df_tvshows_count_countries['Country'].map(countries).astype(int)

# In[23]:


df_tvshows_mixed_countries = df_tvshows_count_countries.copy()

# In[24]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_countries_tvshows =
df_tvshows_count_countries.loc[df_tvshows_count_countries['Netflix'] == 1]
hulu_countries_tvshows = df_tvshows_count_countries.loc[df_tvshows_count_countries['Hulu']
== 1]
prime_video_countries_tvshows =
df_tvshows_count_countries.loc[df_tvshows_count_countries['Prime Video'] == 1]
disney_countries_tvshows =
df_tvshows_count_countries.loc[df_tvshows_count_countries['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_count_countries.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_countries_most_tvshows = df_tvshows_count_countries.sort_values(by = 'Number of
Countries', ascending = False).reset_index()
df_countries_most_tvshows = df_countries_most_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_countries['Number of Countries'] ==
(df_tvshows_count_countries['Number of Countries'].max()))
# df_countries_most_tvshows = df_tvshows_count_countries[filter]

```

```

# mostest_rated_tvshows = df_tvshows_count_countries.loc[df_tvshows_count_countries['Number of Countries'].idxmax()]

print('\nTV Shows with Highest Ever Number of Countries are : \n')
df_countries_most_tvshows.head(5)

# In[27]:


fig = px.bar(y = df_countries_most_tvshows['Title'][:15],
              x = df_countries_most_tvshows['Number of Countries'][:15],
              color = df_countries_most_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Highest Number of Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[28]:


df_countries_least_tvshows = df_tvshows_count_countries.sort_values(by = 'Number of Countries', ascending = True).reset_index()
df_countries_least_tvshows = df_countries_least_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_countries['Number of Countries'] ==
# (df_tvshows_count_countries['Number of Countries'].min()))
# df_countries_least_tvshows = df_tvshows_count_countries[filter]

print('\nTV Shows with Lowest Ever Number of Countries are : \n')
df_countries_least_tvshows.head(5)

# In[29]:


fig = px.bar(y = df_countries_least_tvshows['Title'][:15],
              x = df_countries_least_tvshows['Number of Countries'][:15],
              color = df_countries_least_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Lowest Number of Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
      Total '{df_tvshows_count_countries['Number of Countries'].unique().shape[0]}' unique
      Number of Countries s were Given, They were Like this,\n

      {df_tvshows_count_countries.sort_values(by = 'Number of Countries', ascending =
      False)['Number of Countries'].unique()}\n
    
```

```

The Highest Number of Countries Ever Any TV Show Got is
'{df_countries_most_tvshows['Title'][0]} : '{df_countries_most_tvshows['Number of
Countries'].max()}'\n

The Lowest Number of Countries Ever Any TV Show Got is
'{df_countries_least_tvshows['Title'][0]} : '{df_countries_least_tvshows['Number of
Countries'].min()}'\n
    ''')

# In[31]:


netflix_countries_most_tvshows =
df_countries_most_tvshows.loc[df_countries_most_tvshows['Netflix']==1].reset_index()
netflix_countries_most_tvshows = netflix_countries_most_tvshows.drop(['index'], axis = 1)

netflix_countries_least_tvshows =
df_countries_least_tvshows.loc[df_countries_least_tvshows['Netflix']==1].reset_index()
netflix_countries_least_tvshows = netflix_countries_least_tvshows.drop(['index'], axis = 1)

netflix_countries_most_tvshows.head(5)

# In[32]:


fig = px.bar(y = netflix_countries_most_tvshows['Title'][:15],
              x = netflix_countries_most_tvshows['Number of Countries'][:15],
              color = netflix_countries_most_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title  = 'TV Shows with Highest Number of Countries : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_countries_least_tvshows['Title'][:15],
              x = netflix_countries_least_tvshows['Number of Countries'][:15],
              color = netflix_countries_least_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title  = 'TV Shows with Lowest Number of Countries : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


hulu_countries_most_tvshows =
df_countries_most_tvshows.loc[df_countries_most_tvshows['Hulu']==1].reset_index()
hulu_countries_most_tvshows = hulu_countries_most_tvshows.drop(['index'], axis = 1)

hulu_countries_least_tvshows =
df_countries_least_tvshows.loc[df_countries_least_tvshows['Hulu']==1].reset_index()

```

```

hulu_countries_least_tvshows = hulu_countries_least_tvshows.drop(['index'], axis = 1)

hulu_countries_most_tvshows.head(5)

# In[35]:


fig = px.bar(y = hulu_countries_most_tvshows['Title'][:15],
              x = hulu_countries_most_tvshows['Number of Countries'][:15],
              color = hulu_countries_most_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Highest Number of Countries : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_countries_least_tvshows['Title'][:15],
              x = hulu_countries_least_tvshows['Number of Countries'][:15],
              color = hulu_countries_least_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Lowest Number of Countries : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_countries_most_tvshows =
df_countries_most_tvshows.loc[df_countries_most_tvshows['Prime Video']==1].reset_index()
prime_video_countries_most_tvshows = prime_video_countries_most_tvshows.drop(['index'],
axis = 1)

prime_video_countries_least_tvshows =
df_countries_least_tvshows.loc[df_countries_least_tvshows['Prime Video']==1].reset_index()
prime_video_countries_least_tvshows = prime_video_countries_least_tvshows.drop(['index'],
axis = 1)

prime_video_countries_most_tvshows.head(5)

# In[38]:


fig = px.bar(y = prime_video_countries_most_tvshows['Title'][:15],
              x = prime_video_countries_most_tvshows['Number of Countries'][:15],
              color = prime_video_countries_most_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Highest Number of Countries : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[39]:
```

```
fig = px.bar(y = prime_video_countries_least_tvshows['Title'][:15],
              x = prime_video_countries_least_tvshows['Number of Countries'][:15],
              color = prime_video_countries_least_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Lowest Number of Countries : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[40]:
```

```
disney_countries_most_tvshows =
df_countries_most_tvshows.loc[df_countries_most_tvshows['Disney+']==1].reset_index()
disney_countries_most_tvshows = disney_countries_most_tvshows.drop(['index'], axis = 1)

disney_countries_least_tvshows =
df_countries_least_tvshows.loc[df_countries_least_tvshows['Disney+']==1].reset_index()
disney_countries_least_tvshows = disney_countries_least_tvshows.drop(['index'], axis = 1)

disney_countries_most_tvshows.head(5)
```



```
# In[41]:
```

```
fig = px.bar(y = disney_countries_most_tvshows['Title'][:15],
              x = disney_countries_most_tvshows['Number of Countries'][:15],
              color = disney_countries_most_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Highest Number of Countries : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[42]:
```

```
fig = px.bar(y = disney_countries_least_tvshows['Title'][:15],
              x = disney_countries_least_tvshows['Number of Countries'][:15],
              color = disney_countries_least_tvshows['Number of Countries'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Countries'},
              title = 'TV Shows with Lowest Number of Countries : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[43]:
```

```

print(f'''
    The TV Show with Highest Number of Countries Ever Got is
'{df_countries_most_tvshows['Title'][0]} : '{df_countries_most_tvshows['Number of
Countries'].max()}'\n
    The TV Show with Lowest Number of Countries Ever Got is
'{df_countries_least_tvshows['Title'][0]} : '{df_countries_least_tvshows['Number of
Countries'].min()}'\n

    The TV Show with Highest Number of Countries on 'Netflix' is
'{netflix_countries_most_tvshows['Title'][0]} : '{netflix_countries_most_tvshows['Number
of Countries'].max()}'\n
    The TV Show with Lowest Number of Countries on 'Netflix' is
'{netflix_countries_least_tvshows['Title'][0]} : '{netflix_countries_least_tvshows['Number
of Countries'].min()}'\n

    The TV Show with Highest Number of Countries on 'Hulu' is
'{hulu_countries_most_tvshows['Title'][0]} : '{hulu_countries_most_tvshows['Number of
Countries'].max()}'\n
    The TV Show with Lowest Number of Countries on 'Hulu' is
'{hulu_countries_least_tvshows['Title'][0]} : '{hulu_countries_least_tvshows['Number of
Countries'].min()}'\n

    The TV Show with Highest Number of Countries on 'Prime Video' is
'{prime_video_countries_most_tvshows['Title'][0]} :
'{prime_video_countries_most_tvshows['Number of Countries'].max()}'\n
    The TV Show with Lowest Number of Countries on 'Prime Video' is
'{prime_video_countries_least_tvshows['Title'][0]} :
'{prime_video_countries_least_tvshows['Number of Countries'].min()}'\n

    The TV Show with Highest Number of Countries on 'Disney+' is
'{disney_countries_most_tvshows['Title'][0]} : '{disney_countries_most_tvshows['Number of
Countries'].max()}'\n
    The TV Show with Lowest Number of Countries on 'Disney+' is
'{disney_countries_least_tvshows['Title'][0]} : '{disney_countries_least_tvshows['Number
of Countries'].min()}'\n
    ''')

```

In[44]:

```

print(f'''
    Accross All Platforms the Average Number of Countries is
'{round(df_tvshows_count_countries['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Netflix' is
'{round(netflix_countries_tvshows['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Hulu' is '{round(hulu_countries_tvshows['Number
of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Prime Video' is
'{round(prime_video_countries_tvshows['Number of Countries'].mean(), ndigits = 2)}'\n
    The Average Number of Countries on 'Disney+' is
'{round(disney_countries_tvshows['Number of Countries'].mean(), ndigits = 2)}'\n
    ''')

```

In[45]:

```

print(f'''
    Accross All Platforms Total Count of Country is '{df_tvshows_count_countries['Number
of Countries'].max()}'\n

```

```
Total Count of Country on 'Netflix' is '{netflix_countries_tvshows['Number of Countries']].max()\nTotal Count of Country on 'Hulu' is '{hulu_countries_tvshows['Number of Countries']].max()\nTotal Count of Country on 'Prime Video' is '{prime_video_countries_tvshows['Number of Countries']].max()\nTotal Count of Country on 'Disney+' is '{disney_countries_tvshows['Number of Countries']].max()\n'''
```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))\nsns.distplot(df_tvshows_count_countries['Number of Countries'], bins = 20, kde = True, ax = ax[0])\nsns.boxplot(df_tvshows_count_countries['Number of Countries'], ax = ax[1])\nplt.show()
```

```
# In[47]:
```

```
# Defining plot size and title\nplt.figure(figsize = (20, 5))\nplt.title('Number of Countries s Per Platform')\n\n# Plotting the information from each dataset into a histogram\nsns.histplot(prime_video_countries_tvshows['Number of Countries'], color = 'lightblue', legend = True, kde = True)\nsns.histplot(netflix_countries_tvshows['Number of Countries'], color = 'red', legend = True, kde = True)\nsns.histplot(hulu_countries_tvshows['Number of Countries'], color = 'lightgreen', legend = True, kde = True)\nsns.histplot(disney_countries_tvshows['Number of Countries'], color = 'darkblue', legend = True, kde = True)\n\n# Setting the legend\nplt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])\nplt.show()
```

```
# In[48]:
```

```
df_lan = df_tvshows_country['Country'].str.split(',').apply(pd.Series).stack()\ndel df_tvshows_country['Country']\n df_lan.index = df_lan.index.droplevel(-1)\n df_lan.name = 'Country'\n df_tvshows_country = df_tvshows_country.join(df_lan)\n df_tvshows_country.drop_duplicates(inplace = True)
```

```
# In[49]:
```

```
df_tvshows_country.head(5)
```

```
# In[50]:
```

```
country_count = df_tvshows_country.groupby('Country')['Title'].count()
country_tvshows = df_tvshows_country.groupby('Country')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
country_data_tvshows = pd.concat([country_count, country_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
country_data_tvshows = country_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)
```

```
# In[51]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_country_tvshows = country_data_tvshows[country_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_country_tvshows = netflix_country_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

hulu_country_tvshows = country_data_tvshows[country_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_country_tvshows = hulu_country_tvshows.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

prime_video_country_tvshows = country_data_tvshows[country_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_country_tvshows = prime_video_country_tvshows.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'TV Shows Count'], axis = 1)

disney_country_tvshows = country_data_tvshows[country_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_country_tvshows = disney_country_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'TV Shows Count'], axis = 1)
```

```
# In[52]:
```

```
# Country with TV Shows Counts - All Platforms Combined
country_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]
```

```
# In[53]:
```

```
fig = px.bar(x = country_data_tvshows['Country'][:50],
              y = country_data_tvshows['TV Shows Count'][:50],
              color = country_data_tvshows['TV Shows Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Country', 'y' : 'TV Shows Count'},
              title = 'Major Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[54]:
```

```
fig = px.choropleth(data_frame = country_data_tvshows, locations = 'Country', locationmode = 'country names', color = 'TV Shows Count', color_continuous_scale = 'deep')

fig.show()
```

In[55]:

```
df_country_high_tvshows = country_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_country_high_tvshows = df_country_high_tvshows.drop(['index'], axis = 1)
# filter = (country_data_tvshows['TV Shows Count'] == (country_data_tvshows['TV Shows Count'].max()))
# df_country_high_tvshows = country_data_tvshows[filter]

# highestRated_tvshows = country_data_tvshows.loc[country_data_tvshows['TV Shows Count'].idxmax()]

print('\nCountry with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_country_high_tvshows.head(5)
```

In[56]:

```
fig = px.bar(y = df_country_high_tvshows['Country'][:15],
              x = df_country_high_tvshows['TV Shows Count'][:15],
              color = df_country_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},
              title = 'Country with Highest TV Shows : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[57]:

```
df_country_low_tvshows = country_data_tvshows.sort_values(by = 'TV Shows Count', ascending = True).reset_index()
df_country_low_tvshows = df_country_low_tvshows.drop(['index'], axis = 1)
# filter = (country_data_tvshows['TV Shows Count'] == (country_data_tvshows['TV Shows Count'].min()))
# df_country_low_tvshows = country_data_tvshows[filter]

print('\nCountry with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_country_low_tvshows.head(5)
```

In[58]:

```
fig = px.bar(y = df_country_low_tvshows['Country'][:15],
              x = df_country_low_tvshows['TV Shows Count'][:15],
              color = df_country_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},
              title = 'Country with Lowest TV Shows Count : All Platforms')
```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[59]:


print(f'''
    Total '{country_data_tvshows['Country'].unique().shape[0]}' unique Country Count s
were Given, They were Like this,\n

    {country_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Country'].unique()[:5]}\n

    The Highest Ever TV Shows Count Ever Any TV Show Got is
'{df_country_high_tvshows['Country'][0]}' : '{df_country_high_tvshows['TV Shows
Count'].max()}'\n

    The Lowest Ever TV Shows Count Ever Any TV Show Got is
'{df_country_low_tvshows['Country'][0]}' : '{df_country_low_tvshows['TV Shows
Count'].min()}'\n
    ''')

```

```

# In[60]:


fig = px.pie(country_data_tvshows[:10], names = 'Country', values = 'TV Shows Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Country')
fig.show()

```

```

# In[61]:


# netflix_country_tvshows = country_data_tvshows[country_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_country_tvshows = netflix_country_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

netflix_country_high_tvshows = df_country_high_tvshows.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_country_high_tvshows = netflix_country_high_tvshows.drop(['index'], axis = 1)

netflix_country_low_tvshows = df_country_low_tvshows.sort_values(by = 'Netflix', ascending
= True).reset_index()
netflix_country_low_tvshows = netflix_country_low_tvshows.drop(['index'], axis = 1)

netflix_country_high_tvshows.head(5)

```

```

# In[62]:


fig = px.bar(x = netflix_country_high_tvshows['Country'][:15],
y = netflix_country_high_tvshows['Netflix'][:15],
color = netflix_country_high_tvshows['Netflix'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},

```

```

        title  = 'Country with Highest TV Shows : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


fig = px.choropleth(data_frame = netflix_country_tvshows, locations = 'Country',
locationmode = 'country names', color = 'Netflix', color_continuous_scale = 'Reds')

fig.show()

# In[64]:


# hulu_country_tvshows = country_data_tvshows[country_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_country_tvshows = hulu_country_tvshows.drop(['index', 'Netflix', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

hulu_country_high_tvshows = df_country_high_tvshows.sort_values(by = 'Hulu', ascending =
False).reset_index()
hulu_country_high_tvshows = hulu_country_high_tvshows.drop(['index'], axis = 1)

hulu_country_low_tvshows = df_country_low_tvshows.sort_values(by = 'Hulu', ascending =
True).reset_index()
hulu_country_low_tvshows = hulu_country_low_tvshows.drop(['index'], axis = 1)

hulu_country_high_tvshows.head(5)

# In[65]:


fig = px.bar(x = hulu_country_high_tvshows['Country'][:15],
              y = hulu_country_high_tvshows['Hulu'][:15],
              color = hulu_country_high_tvshows['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},
              title  = 'Country with Highest TV Shows : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[66]:


fig = px.choropleth(data_frame = hulu_country_tvshows, locations = 'Country', locationmode
= 'country names', color = 'Hulu', color_continuous_scale = 'Greens')

fig.show()

# In[67]:

```

```

# prime_video_country_tvshows = country_data_tvshows[country_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_country_tvshows = prime_video_country_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_country_high_tvshows = df_country_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_country_high_tvshows = prime_video_country_high_tvshows.drop(['index'], axis = 1)

prime_video_country_low_tvshows = df_country_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_country_low_tvshows = prime_video_country_low_tvshows.drop(['index'], axis = 1)

prime_video_country_high_tvshows.head(5)

```

In[68]:

```

fig = px.bar(x = prime_video_country_high_tvshows['Country'][:15],
              y = prime_video_country_high_tvshows['Prime Video'][:15],
              color = prime_video_country_high_tvshows['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},
              title = 'Country with Highest TV Shows : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[69]:

```

fig = px.choropleth(data_frame = prime_video_country_tvshows, locations = 'Country',
                     locationmode = 'country names', color = 'Prime Video', color_continuous_scale = 'Blues')

fig.show()

```

In[70]:

```

# disney_country_tvshows = country_data_tvshows[country_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_country_tvshows = disney_country_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_country_high_tvshows = df_country_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_country_high_tvshows = disney_country_high_tvshows.drop(['index'], axis = 1)

disney_country_low_tvshows = df_country_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_country_low_tvshows = disney_country_low_tvshows.drop(['index'], axis = 1)

disney_country_high_tvshows.head(5)

```

In[71]:

```

fig = px.bar(x = disney_country_high_tvshows['Country'][:15],
              y = disney_country_high_tvshows['Disney+'][:15],
              color = disney_country_high_tvshows['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Country', 'x' : 'TV Shows Count'},
              title = 'Country with Highest TV Shows : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[72]:


fig = px.choropleth(data_frame = disney_country_tvshows, locations = 'Country',
                     locationmode = 'country names', color = 'Disney+', color_continuous_scale = 'BuPu')

fig.show()

# In[73]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(country_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(country_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[74]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Country TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_country_tvshows['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_country_tvshows['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_country_tvshows['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_country_tvshows['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[75]:


print(f'''The Country with Highest TV Shows Count Ever Got is
'{df_country_high_tvshows['Country'][0]}' : '{df_country_high_tvshows['TV Shows Count'].max()}'\n

```

```

        The Country with Lowest TV Shows Count Ever Got is
'{df_country_low_tvshows['Country'][0]} : '{df_country_low_tvshows['TV Shows
Count'].min()}'\n

        The Country with Highest TV Shows Count on 'Netflix' is
'{netflix_country_high_tvshows['Country'][0]} :
'{netflix_country_high_tvshows['Netflix'].max()}'\n
        The Country with Lowest TV Shows Count on 'Netflix' is
'{netflix_country_low_tvshows['Country'][0]} :
'{netflix_country_low_tvshows['Netflix'].min()}'\n

        The Country with Highest TV Shows Count on 'Hulu' is
'{hulu_country_high_tvshows['Country'][0]} : '{hulu_country_high_tvshows['Hulu'].max()}'\n
        The Country with Lowest TV Shows Count on 'Hulu' is
'{hulu_country_low_tvshows['Country'][0]} : '{hulu_country_low_tvshows['Hulu'].min()}'\n

        The Country with Highest TV Shows Count on 'Prime Video' is
'{prime_video_country_high_tvshows['Country'][0]} :
'{prime_video_country_high_tvshows['Prime Video'].max()}'\n
        The Country with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_country_low_tvshows['Country'][0]} :
'{prime_video_country_low_tvshows['Prime Video'].min()}'\n

        The Country with Highest TV Shows Count on 'Disney+' is
'{disney_country_high_tvshows['Country'][0]} :
'{disney_country_high_tvshows['Disney+'].max()}'\n
        The Country with Lowest TV Shows Count on 'Disney+' is
'{disney_country_low_tvshows['Country'][0]} :
'{disney_country_low_tvshows['Disney+'].min()}'\n
        ''')

```

```
# In[76]:
```

```

# Distribution of tvshows country in each platform
plt.figure(figsize = (20, 5))
plt.title('Country with TV Shows Count for All Platforms')
sns.violinplot(x = country_data_tvshows['TV Shows Count'][:100], color = 'gold', legend =
True, kde = True, shade = False)
plt.show()

```

```
# In[77]:
```

```

# Distribution of Country TV Shows Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_country_tvshows['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_country_tvshows['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_country_tvshows['Prime Video'][:100], color = 'lightblue',
ax = ax2[0])
sns.violinplot(x = disney_country_tvshows['Disney+'][:100], color = 'darkblue', ax =
ax2[1])
plt.show()

```

```
# In[78]:
```

```
print(f'''
    Accross All Platforms the Average TV Shows Count of Country is
'{round(country_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Country on 'Netflix' is
'{round(netflix_country_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Country on 'Hulu' is
'{round(hulu_country_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Country on 'Prime Video' is
'{round(prime_video_country_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Country on 'Disney+' is
'{round(disney_country_tvshows['Disney+'].mean(), ndigits = 2)}'
    ''')
```

```
# In[79]:
```

```
print(f'''
    Accross All Platforms Total Count of Country is
'{country_data_tvshows['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Netflix' is
'{netflix_country_tvshows['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Hulu' is
'{hulu_country_tvshows['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Prime Video' is
'{prime_video_country_tvshows['Country'].unique().shape[0]}'\n
    Total Count of Country on 'Disney+' is
'{disney_country_tvshows['Country'].unique().shape[0]}'
    ''')
```

```
# In[80]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = country_data_tvshows['Country'][:10], y =
country_data_tvshows['Netflix'][:10], color = 'red')
sns.lineplot(x = country_data_tvshows['Country'][:10], y =
country_data_tvshows['Hulu'][:10], color = 'lightgreen')
sns.lineplot(x = country_data_tvshows['Country'][:10], y = country_data_tvshows['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = country_data_tvshows['Country'][:10], y =
country_data_tvshows['Disney+'][:10], color = 'darkblue')
plt.xlabel('Country', fontsize = 20)
plt.ylabel('TV Shows Count', fontsize = 20)
plt.show()
```

```
# In[81]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_co_ax1 = sns.lineplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_co_ax2 = sns.lineplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_co_ax3 = sns.lineplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
```

```
d_co_ax4 = sns.lineplot(y = country_data_tvshows['Country'][:10], x = country_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])
```

```
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']
```

```
n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])
```

```
plt.show()
```

```
# In[82]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))
```

```
n_co_ax1 = sns.barplot(y = netflix_country_tvshows['Country'][:10], x = netflix_country_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
```

```
h_co_ax2 = sns.barplot(y = hulu_country_tvshows['Country'][:10], x = hulu_country_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
```

```
p_co_ax3 = sns.barplot(y = prime_video_country_tvshows['Country'][:10], x = prime_video_country_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
```

```
d_co_ax4 = sns.barplot(y = disney_country_tvshows['Country'][:10], x = disney_country_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])
```

```
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']
```

```
n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])
```

```
plt.show()
```

```
# In[83]:
```

```
# Defining plot size and title
```

```
plt.figure(figsize = (20, 5))
```

```
plt.title('Country TV Shows Count Per Platform')
```

```
# Plotting the information from each dataset into a histogram
```

```
sns.kdeplot(netflix_country_tvshows['Netflix'][:10], color = 'red', legend = True)
```

```
sns.kdeplot(hulu_country_tvshows['Hulu'][:10], color = 'green', legend = True)
```

```
sns.kdeplot(prime_video_country_tvshows['Prime Video'][:10], color = 'lightblue', legend = True)
```

```
sns.kdeplot(disney_country_tvshows['Disney+'][:10], color = 'darkblue', legend = True)
```

```
# Setting the legend
```

```
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
```

```
plt.show()
```

```
# In[84]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))
```

```

n_co_ax1 = sns.barplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_co_ax2 = sns.barplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_co_ax3 = sns.barplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_co_ax4 = sns.barplot(y = country_data_tvshows['Country'][:10], x =
country_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()

```

In[85]:

```

df_tvshows_mixed_countries.drop(df_tvshows_mixed_countries.loc[df_tvshows_mixed_countries['Country'] == "NA"].index, inplace = True)
# df_tvshows_mixed_countries =
df_tvshows_mixed_countries[df_tvshows_mixed_countries.Country != "NA"]
df_tvshows_mixed_countries.drop(df_tvshows_mixed_countries.loc[df_tvshows_mixed_countries['Number of Countries'] == 1].index, inplace = True)

```

In[86]:

```
df_tvshows_mixed_countries.head(5)
```

In[87]:

```

mixed_countries_count = df_tvshows_mixed_countries.groupby('Country')['Title'].count()
mixed_countries_tvshows = df_tvshows_mixed_countries.groupby('Country')[['Netflix', 'Hulu',
'Prime Video', 'Disney+']].sum()
mixed_countries_data_tvshows = pd.concat([mixed_countries_count, mixed_countries_tvshows],
axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count', 'Country' : 'Mixed
Country'})
mixed_countries_data_tvshows = mixed_countries_data_tvshows.sort_values(by = 'TV Shows
Count', ascending = False)

```

In[88]:

```
mixed_countries_data_tvshows.head(5)
```

In[89]:

```

# Mixed Country with TV Shows Counts - All Platforms Combined
mixed_countries_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

```

```

# In[90]:


df_mixed_countries_high_tvshows = mixed_countries_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_mixed_countries_high_tvshows = df_mixed_countries_high_tvshows.drop(['index'], axis = 1)
# filter = (mixed_countries_data_tvshows['TV Shows Count'] == (mixed_countries_data_tvshows['TV Shows Count'].max()))
# df_mixed_countries_high_tvshows = mixed_countries_data_tvshows[filter]

# highestRated_tvshows = mixed_countries_data_tvshows.loc[mixed_countries_data_tvshows['TV Shows Count'].idxmax()]

print('\nMixed Country with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_countries_high_tvshows.head(5)

# In[91]:


fig = px.bar(y = df_mixed_countries_high_tvshows['Mixed Country'][:15],
              x = df_mixed_countries_high_tvshows['TV Shows Count'][:15],
              color = df_mixed_countries_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Country'},
              title = 'TV Shows with Highest Number of Mixed Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[92]:


df_mixed_countries_low_tvshows = mixed_countries_data_tvshows.sort_values(by = 'TV Shows Count', ascending = True).reset_index()
df_mixed_countries_low_tvshows = df_mixed_countries_low_tvshows.drop(['index'], axis = 1)
# filter = (mixed_countries_data_tvshows['TV Shows Count'] == (mixed_countries_data_tvshows['TV Shows Count'].min()))
# df_mixed_countries_low_tvshows = mixed_countries_data_tvshows[filter]

print('\nMixed Country with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_countries_low_tvshows.head(5)

# In[93]:


fig = px.bar(y = df_mixed_countries_low_tvshows['Mixed Country'][:15],
              x = df_mixed_countries_low_tvshows['TV Shows Count'][:15],
              color = df_mixed_countries_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Country'},
              title = 'TV Shows with Lowest Number of Mixed Countries : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[94]:

```

```

print(f'''
    Total '{df_tvshows_countries['Country'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see TV Shows from Total '{mixed_countries_data_tvshows['Mixed
Country'].unique().shape[0]}' Mixed Country, They were Like this, \n

    {mixed_countries_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Mixed Country'].head(5).unique()} etc. \n

    The Mixed Country with Highest TV Shows Count have '{mixed_countries_data_tvshows['TV
Shows Count'].max()}' TV Shows Available is '{df_mixed_countries_high_tvshows['Mixed
Country'][0]}', &\n
    The Mixed Country with Lowest TV Shows Count have '{mixed_countries_data_tvshows['TV
Shows Count'].min()}' TV Shows Available is '{df_mixed_countries_low_tvshows['Mixed
Country'][0]}'
    ''')

```

In[95]:

```

fig = px.pie(mixed_countries_data_tvshows[:10], names = 'Mixed Country', values = 'TV Shows
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Mixed Country')
fig.show()

```

In[96]:

```

# netflix_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
# netflix_mixed_countries_tvshows = netflix_mixed_countries_tvshows.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_mixed_countries_high_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_countries_high_tvshows = netflix_mixed_countries_high_tvshows.drop(['index'],
axis = 1)

netflix_mixed_countries_low_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_mixed_countries_low_tvshows = netflix_mixed_countries_low_tvshows.drop(['index'],
axis = 1)

netflix_mixed_countries_high_tvshows.head(5)

```

In[97]:

```

# hulu_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_mixed_countries_tvshows = hulu_mixed_countries_tvshows.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

```

```

hulu_mixed_countries_high_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_countries_high_tvshows = hulu_mixed_countries_high_tvshows.drop(['index'], axis
= 1)

hulu_mixed_countries_low_tvshows = df_mixed_countries_high_tvshows.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_countries_low_tvshows = hulu_mixed_countries_low_tvshows.drop(['index'], axis =
1)

hulu_mixed_countries_high_tvshows.head(5)

# In[98]:


# prime_video_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_countries_tvshows = prime_video_mixed_countries_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_countries_high_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_countries_high_tvshows =
prime_video_mixed_countries_high_tvshows.drop(['index'], axis = 1)

prime_video_mixed_countries_low_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Prime Video', ascending = True).reset_index()
prime_video_mixed_countries_low_tvshows =
prime_video_mixed_countries_low_tvshows.drop(['index'], axis = 1)

prime_video_mixed_countries_high_tvshows.head(5)

# In[99]:


# disney_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Disney+'] != 0].sort_values(by
= 'Disney+', ascending = False).reset_index()
# disney_mixed_countries_tvshows = disney_mixed_countries_tvshows.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_mixed_countries_high_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_countries_high_tvshows = disney_mixed_countries_high_tvshows.drop(['index'],
axis = 1)

disney_mixed_countries_low_tvshows = df_mixed_countries_high_tvshows.sort_values(by =
'Disney+', ascending = True).reset_index()
disney_mixed_countries_low_tvshows = disney_mixed_countries_low_tvshows.drop(['index'],
axis = 1)

disney_mixed_countries_high_tvshows.head(5)

# In[100]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))

```

```

sns.distplot(mixed_countries_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax =
ax[0])
sns.boxplot(mixed_countries_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[101]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_countries_tvshows = netflix_mixed_countries_tvshows.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_countries_tvshows = hulu_mixed_countries_tvshows.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Prime Video'] != 0].sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_countries_tvshows = prime_video_mixed_countries_tvshows.drop(['index',
'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_mixed_countries_tvshows =
mixed_countries_data_tvshows[mixed_countries_data_tvshows['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_countries_tvshows = disney_mixed_countries_tvshows.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

# In[102]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Country TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_countries_tvshows['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_countries_tvshows['Netflix'][:100], color = 'red', legend =
True, kde = True)
sns.histplot(hulu_mixed_countries_tvshows['Hulu'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_mixed_countries_tvshows['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[103]:

```

```

print(f'''
    The Mixed Country with Highest TV Shows Count Ever Got is
'{df_mixed_countries_high_tvshows['Mixed Country'][0]}' :
'{df_mixed_countries_high_tvshows['TV Shows Count'].max()}'\n
    The Mixed Country with Lowest TV Shows Count Ever Got is
'{df_mixed_countries_low_tvshows['Mixed Country'][0]}' :
'{df_mixed_countries_low_tvshows['TV Shows Count'].min()}'\n

    The Mixed Country with Highest TV Shows Count on 'Netflix' is
'{netflix_mixed_countries_high_tvshows['Mixed Country'][0]}' :
'{netflix_mixed_countries_high_tvshows['Netflix'].max()}'\n
    The Mixed Country with Lowest TV Shows Count on 'Netflix' is
'{netflix_mixed_countries_low_tvshows['Mixed Country'][0]}' :
'{netflix_mixed_countries_low_tvshows['Netflix'].min()}'\n

    The Mixed Country with Highest TV Shows Count on 'Hulu' is
'{hulu_mixed_countries_high_tvshows['Mixed Country'][0]}' :
'{hulu_mixed_countries_high_tvshows['Hulu'].max()}'\n
    The Mixed Country with Lowest TV Shows Count on 'Hulu' is
'{hulu_mixed_countries_low_tvshows['Mixed Country'][0]}' :
'{hulu_mixed_countries_low_tvshows['Hulu'].min()}'\n

    The Mixed Country with Highest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_countries_high_tvshows['Mixed Country'][0]}' :
'{prime_video_mixed_countries_high_tvshows['Prime Video'].max()}'\n
    The Mixed Country with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_countries_low_tvshows['Mixed Country'][0]}' :
'{prime_video_mixed_countries_low_tvshows['Prime Video'].min()}'\n

    The Mixed Country with Highest TV Shows Count on 'Disney+' is
'{disney_mixed_countries_high_tvshows['Mixed Country'][0]}' :
'{disney_mixed_countries_high_tvshows['Disney+'].max()}'\n
    The Mixed Country with Lowest TV Shows Count on 'Disney+' is
'{disney_mixed_countries_low_tvshows['Mixed Country'][0]}' :
'{disney_mixed_countries_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[104]:

```

print(f'''
    Accross All Platforms the Average TV Shows Count of Mixed Country is
'{round(mixed_countries_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Country on 'Netflix' is
'{round(netflix_mixed_countries_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Country on 'Hulu' is
'{round(hulu_mixed_countries_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Country on 'Prime Video' is
'{round(prime_video_mixed_countries_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Country on 'Disney+' is
'{round(disney_mixed_countries_tvshows['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[105]:

```
print(f'''
```

```

    Accross All Platforms Total Count of Mixed Country is
'{mixed_countries_data_tvshows['Mixed Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Netflix' is '{netflix_mixed_countries_tvshows['Mixed
Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Hulu' is '{hulu_mixed_countries_tvshows['Mixed
Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Prime Video' is
'{prime_video_mixed_countries_tvshows['Mixed Country'].unique().shape[0]}'\n
    Total Count of Mixed Country on 'Disney+' is '{disney_mixed_countries_tvshows['Mixed
Country'].unique().shape[0]}'\n
    ''')

```

```
# In[106]:
```

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_countries_data_tvshows['Mixed Country'][:5], y =
mixed_countries_data_tvshows['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_countries_data_tvshows['Mixed Country'][:5], y =
mixed_countries_data_tvshows['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_countries_data_tvshows['Mixed Country'][:5], y =
mixed_countries_data_tvshows['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_countries_data_tvshows['Mixed Country'][:5], y =
mixed_countries_data_tvshows['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Country', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

```
# In[107]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_co_ax1 = sns.barplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_co_ax2 = sns.barplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_co_ax3 = sns.barplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_co_ax4 = sns.barplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_co_ax1.title.set_text(labels[0])
h_co_ax2.title.set_text(labels[1])
p_co_ax3.title.set_text(labels[2])
d_co_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[108]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_mco_ax1 = sns.lineplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])

```

```

h_mco_ax2 = sns.lineplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_mco_ax3 = sns.lineplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_mco_ax4 = sns.lineplot(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =
mixed_countries_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mco_ax1.title.set_text(labels[0])
h_mco_ax2.title.set_text(labels[1])
p_mco_ax3.title.set_text(labels[2])
d_mco_ax4.title.set_text(labels[3])

plt.show()

# In[109]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Country TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_countries_tvshows['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_countries_tvshows['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_countries_tvshows['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_countries_tvshows['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()


# In[110]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_mco_ax1 = sns.barplot(y = netflix_mixed_countries_tvshows['Mixed Country'][:10], x =
netflix_mixed_countries_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mco_ax2 = sns.barplot(y = hulu_mixed_countries_tvshows['Mixed Country'][:10], x =
hulu_mixed_countries_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mco_ax3 = sns.barplot(y = prime_video_mixed_countries_tvshows['Mixed Country'][:10], x =
prime_video_mixed_countries_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_mco_ax4 = sns.barplot(y = disney_mixed_countries_tvshows['Mixed Country'][:10], x =
disney_mixed_countries_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mco_ax1.title.set_text(labels[0])
h_mco_ax2.title.set_text(labels[1])
p_mco_ax3.title.set_text(labels[2])
d_mco_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[111]:  
  
fig = go.Funnel(y = mixed_countries_data_tvshows['Mixed Country'][:10], x =  
mixed_countries_data_tvshows['TV Shows Count'][:10])  
fig.show()
```

otttvshows_director.ipynb

```
#!/usr/bin/env python  
# coding: utf-8  
  
# In[1]:  
  
# !pip install git+https://github.com/alberanid/imdbpy  
# !pip install pandas  
# !pip install numpy  
# !pip install matplotlib  
# !pip install seaborn  
# !pip install pandas_profiling --upgrade  
# !pip install plotly  
# !pip install wordcloud  
# !pip install Flask  
  
# In[2]:  
  
# Import Dataset  
# Import File from Loacal Drive  
# from google.colab import files  
# data_to_load = files.upload()  
# from google.colab import drive  
# drive.mount('/content/drive')  
  
# In[3]:  
  
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import warnings  
import collections  
import plotly.express as px  
import plotly.graph_objects as go  
import nltk  
import re  
from nltk.corpus import stopwords  
from nltk.tokenize import word_tokenize  
from nltk.probability import FreqDist  
from nltk.util import ngrams  
from plotly.subplots import make_subplots  
from plotly.offline import iplot, init_notebook_mode  
from wordcloud import WordCloud, STOPWORDS
```

```

from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()

# In[6]:


# profile = ProfileReport(df_tvshows)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_tvshows)

```

```

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

```

```

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_tvshows)

# In[11]:
df_tvshows.head()

# In[12]:
df_tvshows.describe()

# In[13]:
df_tvshows.corr()

# In[14]:
# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:
# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

```

```

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')
# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_directors = df_tvshows.copy()


# In[19]:


df_tvshows_directors.drop(df_tvshows_directors.loc[df_tvshows_directors['Directors'] == "NA"].index, inplace = True)
# df_tvshows_directors = df_tvshows_directors[df_tvshows_directors.Director != "NA"]
# df_tvshows_directors['Director'] = df_tvshows_directors['Director'].astype(str)


# In[20]:


df_tvshows_count_directors = df_tvshows_directors.copy()


# In[21]:


df_tvshows_director = df_tvshows_directors.copy()


# In[22]:


# Create directors dict where key=name and value = number of directors
directors = {}

```

```

for i in df_tvshows_count_directors['Directors'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        directors[i] = len(i.split(','))
    else:
        directors[i] = 0

# Add this information to our dataframe as a new column

df_tvshows_count_directors['Number of Directors'] =
df_tvshows_count_directors['Directors'].map(directors).astype(int)

# In[23]:


df_tvshows_mixed_directors = df_tvshows_count_directors.copy()

# In[24]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_directors_tvshows =
df_tvshows_count_directors.loc[df_tvshows_count_directors['Netflix'] == 1]
hulu_directors_tvshows = df_tvshows_count_directors.loc[df_tvshows_count_directors['Hulu']
== 1]
prime_video_directors_tvshows =
df_tvshows_count_directors.loc[df_tvshows_count_directors['Prime Video'] == 1]
disney_directors_tvshows =
df_tvshows_count_directors.loc[df_tvshows_count_directors['Disney+'] == 1]

# In[25]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_count_directors.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[26]:


df_directors_most_tvshows = df_tvshows_count_directors.sort_values(by = 'Number of
Directors', ascending = False).reset_index()
df_directors_most_tvshows = df_directors_most_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_directors['Number of Directors'] ==
(df_tvshows_count_directors['Number of Directors'].max()))
# df_directors_most_tvshows = df_tvshows_count_directors[filter]

```

```
# mostest_rated_tvshows = df_tvshows_count_directors.loc[df_tvshows_count_directors['Number of Directors'].idxmax()]

print('\nTV Shows with Highest Ever Number of Directors are : \n')
df_directors_most_tvshows.head(5)
```

In[27]:

```
fig = px.bar(y = df_directors_most_tvshows['Title'][:15],
              x = df_directors_most_tvshows['Number of Directors'][:15],
              color = df_directors_most_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Highest Number of Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[28]:

```
df_directors_least_tvshows = df_tvshows_count_directors.sort_values(by = 'Number of Directors', ascending = True).reset_index()
df_directors_least_tvshows = df_directors_least_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_directors['Number of Directors'] ==
# (df_tvshows_count_directors['Number of Directors'].min()))
# df_directors_least_tvshows = df_tvshows_count_directors[filter]

print('\nTV Shows with Lowest Ever Number of Directors are : \n')
df_directors_least_tvshows.head(5)
```

In[29]:

```
fig = px.bar(y = df_directors_least_tvshows['Title'][:15],
              x = df_directors_least_tvshows['Number of Directors'][:15],
              color = df_directors_least_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Lowest Number of Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[30]:

```
print(f'''
    Total '{df_tvshows_count_directors['Number of Directors'].unique().shape[0]}' unique
    Number of Directors s were Given, They were Like this,\n

    {df_tvshows_count_directors.sort_values(by = 'Number of Directors', ascending =
    False)['Number of Directors'].unique()}\n
```

```

    The Highest Number of Directors Ever Any TV Show Got is
'{df_directors_most_tvshows['Title'][0]} : '{df_directors_most_tvshows['Number of
Directors'].max()}'\n

    The Lowest Number of Directors Ever Any TV Show Got is
'{df_directors_least_tvshows['Title'][0]} : '{df_directors_least_tvshows['Number of
Directors'].min()}'\n
    ''')

# In[31]:


netflix_directors_most_tvshows =
df_directors_most_tvshows.loc[df_directors_most_tvshows['Netflix']==1].reset_index()
netflix_directors_most_tvshows = netflix_directors_most_tvshows.drop(['index'], axis = 1)

netflix_directors_least_tvshows =
df_directors_least_tvshows.loc[df_directors_least_tvshows['Netflix']==1].reset_index()
netflix_directors_least_tvshows = netflix_directors_least_tvshows.drop(['index'], axis = 1)

netflix_directors_most_tvshows.head(5)

# In[32]:


fig = px.bar(y = netflix_directors_most_tvshows['Title'][:15],
              x = netflix_directors_most_tvshows['Number of Directors'][:15],
              color = netflix_directors_most_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title  = 'TV Shows with Highest Number of Directors : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[33]:


fig = px.bar(y = netflix_directors_least_tvshows['Title'][:15],
              x = netflix_directors_least_tvshows['Number of Directors'][:15],
              color = netflix_directors_least_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title  = 'TV Shows with Lowest Number of Directors : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


hulu_directors_most_tvshows =
df_directors_most_tvshows.loc[df_directors_most_tvshows['Hulu']==1].reset_index()
hulu_directors_most_tvshows = hulu_directors_most_tvshows.drop(['index'], axis = 1)

hulu_directors_least_tvshows =
df_directors_least_tvshows.loc[df_directors_least_tvshows['Hulu']==1].reset_index()

```

```

hulu_directors_least_tvshows = hulu_directors_least_tvshows.drop(['index'], axis = 1)

hulu_directors_most_tvshows.head(5)

# In[35]:


fig = px.bar(y = hulu_directors_most_tvshows['Title'][:15],
              x = hulu_directors_most_tvshows['Number of Directors'][:15],
              color = hulu_directors_most_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Highest Number of Directors : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = hulu_directors_least_tvshows['Title'][:15],
              x = hulu_directors_least_tvshows['Number of Directors'][:15],
              color = hulu_directors_least_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Lowest Number of Directors : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_directors_most_tvshows =
df_directors_most_tvshows.loc[df_directors_most_tvshows['Prime Video']==1].reset_index()
prime_video_directors_most_tvshows = prime_video_directors_most_tvshows.drop(['index'],
axis = 1)

prime_video_directors_least_tvshows =
df_directors_least_tvshows.loc[df_directors_least_tvshows['Prime Video']==1].reset_index()
prime_video_directors_least_tvshows = prime_video_directors_least_tvshows.drop(['index'],
axis = 1)

prime_video_directors_most_tvshows.head(5)

# In[38]:


fig = px.bar(y = prime_video_directors_most_tvshows['Title'][:15],
              x = prime_video_directors_most_tvshows['Number of Directors'][:15],
              color = prime_video_directors_most_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Highest Number of Directors : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[39]:


fig = px.bar(y = prime_video_directors_least_tvshows['Title'][:15],
              x = prime_video_directors_least_tvshows['Number of Directors'][:15],
              color = prime_video_directors_least_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Lowest Number of Directors : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


disney_directors_most_tvshows =
df_directors_most_tvshows.loc[df_directors_most_tvshows['Disney+']==1].reset_index()
disney_directors_most_tvshows = disney_directors_most_tvshows.drop(['index'], axis = 1)

disney_directors_least_tvshows =
df_directors_least_tvshows.loc[df_directors_least_tvshows['Disney+']==1].reset_index()
disney_directors_least_tvshows = disney_directors_least_tvshows.drop(['index'], axis = 1)

disney_directors_most_tvshows.head(5)

# In[41]:


fig = px.bar(y = disney_directors_most_tvshows['Title'][:15],
              x = disney_directors_most_tvshows['Number of Directors'][:15],
              color = disney_directors_most_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Highest Number of Directors : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_directors_least_tvshows['Title'][:15],
              x = disney_directors_least_tvshows['Number of Directors'][:15],
              color = disney_directors_least_tvshows['Number of Directors'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Directors'},
              title = 'TV Shows with Lowest Number of Directors : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:

```

```

print(f'''
    The TV Show with Highest Number of Directors Ever Got is
'{df_directors_most_tvshows['Title'][0]} : '{df_directors_most_tvshows['Number of
Directors'].max()}'\n
    The TV Show with Lowest Number of Directors Ever Got is
'{df_directors_least_tvshows['Title'][0]} : '{df_directors_least_tvshows['Number of
Directors'].min()}'\n

    The TV Show with Highest Number of Directors on 'Netflix' is
'{netflix_directors_most_tvshows['Title'][0]} : '{netflix_directors_most_tvshows['Number
of Directors'].max()}'\n
    The TV Show with Lowest Number of Directors on 'Netflix' is
'{netflix_directors_least_tvshows['Title'][0]} : '{netflix_directors_least_tvshows['Number
of Directors'].min()}'\n

    The TV Show with Highest Number of Directors on 'Hulu' is
'{hulu_directors_most_tvshows['Title'][0]} : '{hulu_directors_most_tvshows['Number of
Directors'].max()}'\n
    The TV Show with Lowest Number of Directors on 'Hulu' is
'{hulu_directors_least_tvshows['Title'][0]} : '{hulu_directors_least_tvshows['Number of
Directors'].min()}'\n

    The TV Show with Highest Number of Directors on 'Prime Video' is
'{prime_video_directors_most_tvshows['Title'][0]} :
'{prime_video_directors_most_tvshows['Number of Directors'].max()}'\n
    The TV Show with Lowest Number of Directors on 'Prime Video' is
'{prime_video_directors_least_tvshows['Title'][0]} :
'{prime_video_directors_least_tvshows['Number of Directors'].min()}'\n

    The TV Show with Highest Number of Directors on 'Disney+' is
'{disney_directors_most_tvshows['Title'][0]} : '{disney_directors_most_tvshows['Number of
Directors'].max()}'\n
    The TV Show with Lowest Number of Directors on 'Disney+' is
'{disney_directors_least_tvshows['Title'][0]} : '{disney_directors_least_tvshows['Number
of Directors'].min()}'\n
    ''')

```

In[44]:

```

print(f'''
    Accross All Platforms the Average Number of Directors is
'{round(df_tvshows_count_directors['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Netflix' is
'{round(netflix_directors_tvshows['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Hulu' is '{round(hulu_directors_tvshows['Number
of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Prime Video' is
'{round(prime_video_directors_tvshows['Number of Directors'].mean(), ndigits = 2)}'\n
    The Average Number of Directors on 'Disney+' is
'{round(disney_directors_tvshows['Number of Directors'].mean(), ndigits = 2)}'\n
    ''')

```

In[45]:

```

print(f'''
    Accross All Platforms Total Count of Director is '{df_tvshows_count_directors['Number
of Directors'].max()}'\n

```

```
Total Count of Director on 'Netflix' is '{netflix_directors_tvshows['Number of Directors']].max()}'\nTotal Count of Director on 'Hulu' is '{hulu_directors_tvshows['Number of Directors']].max()}'\nTotal Count of Director on 'Prime Video' is '{prime_video_directors_tvshows['Number of Directors']].max()}'\nTotal Count of Director on 'Disney+' is '{disney_directors_tvshows['Number of Directors']].max()}'\n'''
```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))\nsns.distplot(df_tvshows_count_directors['Number of Directors'], bins = 20, kde = True, ax = ax[0])\nsns.boxplot(df_tvshows_count_directors['Number of Directors'], ax = ax[1])\nplt.show()
```

```
# In[47]:
```

```
# Defining plot size and title\nplt.figure(figsize = (20, 5))\nplt.title('Number of Directors s Per Platform')\n\n# Plotting the information from each dataset into a histogram\nsns.histplot(prime_video_directors_tvshows['Number of Directors'], color = 'lightblue', legend = True, kde = True)\nsns.histplot(netflix_directors_tvshows['Number of Directors'], color = 'red', legend = True, kde = True)\nsns.histplot(hulu_directors_tvshows['Number of Directors'], color = 'lightgreen', legend = True, kde = True)\nsns.histplot(disney_directors_tvshows['Number of Directors'], color = 'darkblue', legend = True, kde = True)\n\n# Setting the legend\nplt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])\nplt.show()
```

```
# In[48]:
```

```
df_lan = df_tvshows_director['Directors'].str.split(',').apply(pd.Series).stack()\ndel df_tvshows_director['Directors']\n df_lan.index = df_lan.index.droplevel(-1)\n df_lan.name = 'Director'\n df_tvshows_director = df_tvshows_director.join(df_lan)\n df_tvshows_director.drop_duplicates(inplace = True)
```

```
# In[49]:
```

```
df_tvshows_director.head(5)
```

```
# In[50]:
```

```

director_count = df_tvshows_director.groupby('Director')['Title'].count()
director_tvshows = df_tvshows_director.groupby('Director')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
director_data_tvshows = pd.concat([director_count, director_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
director_data_tvshows = director_data_tvshows.sort_values(by = 'TV Shows Count', ascending
= False)

# In[51]:


# Director with TV Shows Counts - All Platforms Combined
director_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[52]:


fig = px.bar(x = director_data_tvshows['Director'][:50],
              y = director_data_tvshows['TV Shows Count'][:50],
              color = director_data_tvshows['TV Shows Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Director', 'y' : 'TV Shows Count'},
              title = 'Major Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[53]:


df_director_high_tvshows = director_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_director_high_tvshows = df_director_high_tvshows.drop(['index'], axis = 1)
# filter = (director_data_tvshows['TV Shows Count'] == (director_data_tvshows['TV Shows
Count'].max()))
# df_director_high_tvshows = director_data_tvshows[filter]

# highestRated_tvshows = director_data_tvshows.loc[director_data_tvshows['TV Shows
Count'].idxmax()]

print('\nDirector with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_director_high_tvshows.head(5)

# In[54]:


fig = px.bar(y = df_director_high_tvshows['Director'][:15],
              x = df_director_high_tvshows['TV Shows Count'][:15],
              color = df_director_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Highest TV Shows : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```
# In[55]:
```

```
df_director_low_tvshows = director_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_director_low_tvshows = df_director_low_tvshows.drop(['index'], axis = 1)
# filter = (director_data_tvshows['TV Shows Count'] == (director_data_tvshows['TV Shows Count'].min()))
# df_director_low_tvshows = director_data_tvshows[filter]

print('\nDirector with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_director_low_tvshows.head(5)
```



```
# In[56]:
```

```
fig = px.bar(y = df_director_low_tvshows['Director'][:15],
              x = df_director_low_tvshows['TV Shows Count'][:15],
              color = df_director_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Lowest TV Shows Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[57]:
```

```
print(f'''
    Total '{director_data_tvshows['Director'].unique().shape[0]}' unique Director Count s
    were Given, They were Like this,\n

        {director_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Director'].unique()[:5]}\n

    The Highest Ever TV Shows Count Ever Any TV Show Got is
'{df_director_high_tvshows['Director'][0]}' : '{df_director_high_tvshows['TV Shows
Count'].max()}'\n

    The Lowest Ever TV Shows Count Ever Any TV Show Got is
'{df_director_low_tvshows['Director'][0]}' : '{df_director_low_tvshows['TV Shows
Count'].min()}'\n
'''')
```



```
# In[58]:
```

```
fig = px.pie(director_data_tvshows[:10], names = 'Director', values = 'TV Shows Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Director')
fig.show()
```



```
# In[59]:
```

```

# netflix_director_tvshows = director_data_tvshows[director_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_director_tvshows = netflix_director_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_director_high_tvshows = df_director_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_director_high_tvshows = netflix_director_high_tvshows.drop(['index'], axis = 1)

netflix_director_low_tvshows = df_director_high_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_director_low_tvshows = netflix_director_low_tvshows.drop(['index'], axis = 1)

netflix_director_high_tvshows.head(5)

```

In[60]:

```

fig = px.bar(x = netflix_director_high_tvshows['Director'][:15],
              y = netflix_director_high_tvshows['Netflix'][:15],
              color = netflix_director_high_tvshows['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Highest TV Shows : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[61]:

```

# hulu_director_tvshows = director_data_tvshows[director_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_director_tvshows = hulu_director_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_director_high_tvshows = df_director_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_director_high_tvshows = hulu_director_high_tvshows.drop(['index'], axis = 1)

hulu_director_low_tvshows = df_director_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_director_low_tvshows = hulu_director_low_tvshows.drop(['index'], axis = 1)

hulu_director_high_tvshows.head(5)

```

In[62]:

```

fig = px.bar(x = hulu_director_high_tvshows['Director'][:15],
              y = hulu_director_high_tvshows['Hulu'][:15],
              color = hulu_director_high_tvshows['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Highest TV Shows : Hulu')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


# prime_video_director_tvshows = director_data_tvshows[director_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_director_tvshows = prime_video_director_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_director_high_tvshows = df_director_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_director_high_tvshows = prime_video_director_high_tvshows.drop(['index'], axis = 1)

prime_video_director_low_tvshows = df_director_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_director_low_tvshows = prime_video_director_low_tvshows.drop(['index'], axis = 1)

prime_video_director_high_tvshows.head(5)

```

In[64]:

```

fig = px.bar(x = prime_video_director_high_tvshows['Director'][:15],
              y = prime_video_director_high_tvshows['Prime Video'][:15],
              color = prime_video_director_high_tvshows['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Highest TV Shows : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[65]:

```

# disney_director_tvshows = director_data_tvshows[director_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_director_tvshows = disney_director_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_director_high_tvshows = df_director_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_director_high_tvshows = disney_director_high_tvshows.drop(['index'], axis = 1)

disney_director_low_tvshows = df_director_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_director_low_tvshows = disney_director_low_tvshows.drop(['index'], axis = 1)

disney_director_high_tvshows.head(5)

```

In[66]:

```

fig = px.bar(x = disney_director_high_tvshows['Director'][:15],
              y = disney_director_high_tvshows['Disney+'][:15],
              color = disney_director_high_tvshows['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Director', 'x' : 'TV Shows Count'},
              title = 'Director with Highest TV Shows : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[67]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(director_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(director_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_director_tvshows = director_data_tvshows[director_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_director_tvshows = netflix_director_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_director_tvshows = director_data_tvshows[director_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_director_tvshows = hulu_director_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_director_tvshows = director_data_tvshows[director_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_director_tvshows = prime_video_director_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_director_tvshows = director_data_tvshows[director_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_director_tvshows = disney_director_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Director TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_director_tvshows['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_director_tvshows['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_director_tvshows['Netflix'][:50], color = 'red', legend = True, kde = True)

```

```

sns.histplot(hulu_director_tvshows['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''
    The Director with Highest TV Shows Count Ever Got is
'{df_director_high_tvshows['Director'][0]}' : '{df_director_high_tvshows['TV Shows Count'].max()}'\n
    The Director with Lowest TV Shows Count Ever Got is
'{df_director_low_tvshows['Director'][0]}' : '{df_director_low_tvshows['TV Shows Count'].min()}'\n

    The Director with Highest TV Shows Count on 'Netflix' is
'{netflix_director_high_tvshows['Director'][0]}' :
'{netflix_director_high_tvshows['Netflix'].max()}'\n
    The Director with Lowest TV Shows Count on 'Netflix' is
'{netflix_director_low_tvshows['Director'][0]}' :
'{netflix_director_low_tvshows['Netflix'].min()}'\n

    The Director with Highest TV Shows Count on 'Hulu' is
'{hulu_director_high_tvshows['Director'][0]}' :
'{hulu_director_high_tvshows['Hulu'].max()}'\n
    The Director with Lowest TV Shows Count on 'Hulu' is
'{hulu_director_low_tvshows['Director'][0]}' :
'{hulu_director_low_tvshows['Hulu'].min()}'\n

    The Director with Highest TV Shows Count on 'Prime Video' is
'{prime_video_director_high_tvshows['Director'][0]}' :
'{prime_video_director_high_tvshows['Prime Video'].max()}'\n
    The Director with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_director_low_tvshows['Director'][0]}' :
'{prime_video_director_low_tvshows['Prime Video'].min()}'\n

    The Director with Highest TV Shows Count on 'Disney+' is
'{disney_director_high_tvshows['Director'][0]}' :
'{disney_director_high_tvshows['Disney+'].max()}'\n
    The Director with Lowest TV Shows Count on 'Disney+' is
'{disney_director_low_tvshows['Director'][0]}' :
'{disney_director_low_tvshows['Disney+'].min()}'\n
''')


# In[71]:


# Distribution of tvshows director in each platform
plt.figure(figsize = (20, 5))
plt.title('Director with TV Shows Count for All Platforms')
sns.violinplot(x = director_data_tvshows['TV Shows Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

# In[72]:

```

```

# Distribution of Director TV Shows Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_director_tvshows['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_director_tvshows['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_director_tvshows['Prime Video'][:100], color = 'lightblue',
ax = ax2[0])
sns.violinplot(x = disney_director_tvshows['Disney+'][:100], color = 'darkblue', ax =
ax2[1])
plt.show()

```

In[73]:

```

print(f'''
    Accross All Platforms the Average TV Shows Count of Director is
'{round(director_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Director on 'Netflix' is
'{round(netflix_director_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Director on 'Hulu' is
'{round(hulu_director_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Director on 'Prime Video' is
'{round(prime_video_director_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Director on 'Disney+' is
'{round(disney_director_tvshows['Disney+'].mean(), ndigits = 2)}'
    ''')

```

In[74]:

```

print(f'''
    Accross All Platforms Total Count of Director is
'{director_data_tvshows['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Netflix' is
'{netflix_director_tvshows['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Hulu' is
'{hulu_director_tvshows['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Prime Video' is
'{prime_video_director_tvshows['Director'].unique().shape[0]}'\n
    Total Count of Director on 'Disney+' is
'{disney_director_tvshows['Director'].unique().shape[0]}'
    ''')

```

In[75]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = director_data_tvshows['Director'][:10], y =
director_data_tvshows['Netflix'][:10], color = 'red')
sns.lineplot(x = director_data_tvshows['Director'][:10], y =
director_data_tvshows['Hulu'][:10], color = 'lightgreen')
sns.lineplot(x = director_data_tvshows['Director'][:10], y = director_data_tvshows['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = director_data_tvshows['Director'][:10], y =
director_data_tvshows['Disney+'][:10], color = 'darkblue')

```

```

plt.xlabel('Director', fontsize = 20)
plt.ylabel('TV Shows Count', fontsize = 20)
plt.show()

# In[76]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_d_ax1 = sns.lineplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_d_ax2 = sns.lineplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_d_ax3 = sns.lineplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_d_ax4 = sns.lineplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[77]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(y = netflix_director_tvshows['Director'][:10], x =
netflix_director_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(y = hulu_director_tvshows['Director'][:10], x =
hulu_director_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(y = prime_video_director_tvshows['Director'][:10], x =
prime_video_director_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(y = disney_director_tvshows['Director'][:10], x =
disney_director_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[78]:
```

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Director TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
```

```

sns.kdeplot(netflix_director_tvshows['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_director_tvshows['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_director_tvshows['Prime Video'][:10], color = 'lightblue', legend = True)
sns.kdeplot(disney_director_tvshows['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

```

In[79]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(y = director_data_tvshows['Director'][:10], x =
director_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()

```

In[80]:

```

df_tvshows_mixed_directors.drop(df_tvshows_mixed_directors.loc[df_tvshows_mixed_directors['
Directors'] == "NA"].index, inplace = True)
# df_tvshows_mixed_directors =
df_tvshows_mixed_directors[df_tvshows_mixed_directors.Director != "NA"]
df_tvshows_mixed_directors.drop(df_tvshows_mixed_directors.loc[df_tvshows_mixed_directors['
Number of Directors'] == 1].index, inplace = True)

```

In[81]:

```
df_tvshows_mixed_directors.head(5)
```

In[82]:

```

mixed_directors_count = df_tvshows_mixed_directors.groupby('Directors')['Title'].count()
mixed_directors_tvshows = df_tvshows_mixed_directors.groupby('Directors')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_directors_data_tvshows = pd.concat([mixed_directors_count, mixed_directors_tvshows],
axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count', 'Directors' : 'Mixed
Director'})

```

```

mixed_directors_data_tvshows = mixed_directors_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[83]:


mixed_directors_data_tvshows.head(5)

# In[84]:


# Mixed Director with TV Shows Counts - All Platforms Combined
mixed_directors_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[85]:


df_mixed_directors_high_tvshows = mixed_directors_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_mixed_directors_high_tvshows = df_mixed_directors_high_tvshows.drop(['index'], axis = 1)
# filter = (mixed_directors_data_tvshows['TV Shows Count'] == (mixed_directors_data_tvshows['TV Shows Count'].max()))
# df_mixed_directors_high_tvshows = mixed_directors_data_tvshows[filter]

# highest_rated_tvshows = mixed_directors_data_tvshows.loc[mixed_directors_data_tvshows['TV Shows Count'].idxmax()]

print('\nMixed Director with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_directors_high_tvshows.head(5)

# In[86]:


fig = px.bar(y = df_mixed_directors_high_tvshows['Mixed Director'][:15],
              x = df_mixed_directors_high_tvshows['TV Shows Count'][:15],
              color = df_mixed_directors_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Director'},
              title = 'TV Shows with Highest Number of Mixed Directors : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[87]:


df_mixed_directors_low_tvshows = mixed_directors_data_tvshows.sort_values(by = 'TV Shows Count', ascending = True).reset_index()
df_mixed_directors_low_tvshows = df_mixed_directors_low_tvshows.drop(['index'], axis = 1)
# filter = (mixed_directors_data_tvshows['TV Shows Count'] == (mixed_directors_data_tvshows['TV Shows Count'].min()))
# df_mixed_directors_low_tvshows = mixed_directors_data_tvshows[filter]

print('\nMixed Director with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_directors_low_tvshows.head(5)

```

```
# In[88]:  
  
fig = px.bar(y = df_mixed_directors_low_tvshows['Mixed Director'][:15],  
             x = df_mixed_directors_low_tvshows['TV Shows Count'][:15],  
             color = df_mixed_directors_low_tvshows['TV Shows Count'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Director'},  
             title = 'TV Shows with Lowest Number of Mixed Directors : All Platforms')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[89]:
```

```
print(f'''  
Total '{df_tvshows_directors['Directors'].count()}' Titles are available on All  
Platforms, out of which\n  
You Can Choose to see TV Shows from Total '{mixed_directors_data_tvshows['Mixed  
Director'].unique().shape[0]}' Mixed Director, They were Like this, \n  
  
{mixed_directors_data_tvshows.sort_values(by = 'TV Shows Count', ascending =  
False)['Mixed Director'].head(5).unique()} etc. \n  
  
The Mixed Director with Highest TV Shows Count have  
'{mixed_directors_data_tvshows['TV Shows Count'].max()}' TV Shows Available is  
'{df_mixed_directors_high_tvshows['Mixed Director'][0]}', &\n  
The Mixed Director with Lowest TV Shows Count have '{mixed_directors_data_tvshows['TV  
Shows Count'].min()}' TV Shows Available is '{df_mixed_directors_low_tvshows['Mixed  
Director'][0]}'  
'''')
```

```
# In[90]:
```

```
fig = px.pie(mixed_directors_data_tvshows[:4], names = 'Mixed Director', values = 'TV Shows  
Count', color_discrete_sequence = px.colors.sequential.Teal)  
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows  
Count based on Mixed Director')  
fig.show()
```

```
# In[91]:
```

```
# netflix_mixed_directors_tvshows =  
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Netflix'] != 0].sort_values(by  
= 'Netflix', ascending = False).reset_index()  
# netflix_mixed_directors_tvshows = netflix_mixed_directors_tvshows.drop(['index', 'Hulu',  
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)  
  
netflix_mixed_directors_high_tvshows = df_mixed_directors_high_tvshows.sort_values(by =  
'Netflix', ascending = False).reset_index()  
netflix_mixed_directors_high_tvshows = netflix_mixed_directors_high_tvshows.drop(['index'],  
axis = 1)
```

```

netflix_mixed_directors_low_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_mixed_directors_low_tvshows = netflix_mixed_directors_low_tvshows.drop(['index'],
axis = 1)

netflix_mixed_directors_high_tvshows.head(5)

# In[92]:


# hulu_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
# hulu_mixed_directors_tvshows = hulu_mixed_directors_tvshows.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_directors_high_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_directors_high_tvshows = hulu_mixed_directors_high_tvshows.drop(['index'], axis =
1)

hulu_mixed_directors_low_tvshows = df_mixed_directors_high_tvshows.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_directors_low_tvshows = hulu_mixed_directors_low_tvshows.drop(['index'], axis =
1)

hulu_mixed_directors_high_tvshows.head(5)

# In[93]:


# prime_video_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Prime Video'] != 0].sort_values(by =
'Prime Video', ascending = False).reset_index()
# prime_video_mixed_directors_tvshows = prime_video_mixed_directors_tvshows.drop(['index',
'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_directors_high_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_directors_high_tvshows =
prime_video_mixed_directors_high_tvshows.drop(['index'], axis = 1)

prime_video_mixed_directors_low_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Prime Video', ascending = True).reset_index()
prime_video_mixed_directors_low_tvshows =
prime_video_mixed_directors_low_tvshows.drop(['index'], axis = 1)

prime_video_mixed_directors_high_tvshows.head(5)

# In[94]:


# disney_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
# disney_mixed_directors_tvshows = disney_mixed_directors_tvshows.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

```

```

disney_mixed_directors_high_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_directors_high_tvshows = disney_mixed_directors_high_tvshows.drop(['index'],
axis = 1)

disney_mixed_directors_low_tvshows = df_mixed_directors_high_tvshows.sort_values(by =
'Disney+', ascending = True).reset_index()
disney_mixed_directors_low_tvshows = disney_mixed_directors_low_tvshows.drop(['index'],
axis = 1)

disney_mixed_directors_high_tvshows.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_directors_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax =
ax[0])
sns.boxplot(mixed_directors_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms

netflix_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_directors_tvshows = netflix_mixed_directors_tvshows.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_mixed_directors_tvshows = hulu_mixed_directors_tvshows.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Prime Video'] !=
0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_directors_tvshows = prime_video_mixed_directors_tvshows.drop(['index',
'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_mixed_directors_tvshows =
mixed_directors_data_tvshows[mixed_directors_data_tvshows['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_directors_tvshows = disney_mixed_directors_tvshows.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

# In[97]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Director TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

```

```

sns.histplot(prime_video_mixed_directors_tvshows['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_directors_tvshows['Netflix'][:100], color = 'red', legend =
True, kde = True)
sns.histplot(hulu_mixed_directors_tvshows['Hulu'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_mixed_directors_tvshows['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Director with Highest TV Shows Count Ever Got is
'{df_mixed_directors_high_tvshows['Mixed Director'][0]}' :
'{df_mixed_directors_high_tvshows['TV Shows Count'].max()}'\n
    The Mixed Director with Lowest TV Shows Count Ever Got is
'{df_mixed_directors_low_tvshows['Mixed Director'][0]}' :
'{df_mixed_directors_low_tvshows['TV Shows Count'].min()}'\n

    The Mixed Director with Highest TV Shows Count on 'Netflix' is
'{netflix_mixed_directors_high_tvshows['Mixed Director'][0]}' :
'{netflix_mixed_directors_high_tvshows['Netflix'].max()}'\n
    The Mixed Director with Lowest TV Shows Count on 'Netflix' is
'{netflix_mixed_directors_low_tvshows['Mixed Director'][0]}' :
'{netflix_mixed_directors_low_tvshows['Netflix'].min()}'\n

    The Mixed Director with Highest TV Shows Count on 'Hulu' is
'{hulu_mixed_directors_high_tvshows['Mixed Director'][0]}' :
'{hulu_mixed_directors_high_tvshows['Hulu'].max()}'\n
    The Mixed Director with Lowest TV Shows Count on 'Hulu' is
'{hulu_mixed_directors_low_tvshows['Mixed Director'][0]}' :
'{hulu_mixed_directors_low_tvshows['Hulu'].min()}'\n

    The Mixed Director with Highest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_directors_high_tvshows['Mixed Director'][0]}' :
'{prime_video_mixed_directors_high_tvshows['Prime Video'].max()}'\n
    The Mixed Director with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_directors_low_tvshows['Mixed Director'][0]}' :
'{prime_video_mixed_directors_low_tvshows['Prime Video'].min()}'\n

    The Mixed Director with Highest TV Shows Count on 'Disney+' is
'{disney_mixed_directors_high_tvshows['Mixed Director'][0]}' :
'{disney_mixed_directors_high_tvshows['Disney+'].max()}'\n
    The Mixed Director with Lowest TV Shows Count on 'Disney+' is
'{disney_mixed_directors_low_tvshows['Mixed Director'][0]}' :
'{disney_mixed_directors_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[99]:

```
print(f'''
```

```

    Accross All Platforms the Average TV Shows Count of Mixed Director is
'{round(mixed_directors_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Director on 'Netflix' is
'{round(netflix_mixed_directors_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Director on 'Hulu' is
'{round(hulu_mixed_directors_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Director on 'Prime Video' is
'{round(prime_video_mixed_directors_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Director on 'Disney+' is
'{round(disney_mixed_directors_tvshows['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```

print(f'''
    Accross All Platforms Total Count of Mixed Director is
'{mixed_directors_data_tvshows['Mixed Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Netflix' is
'{netflix_mixed_directors_tvshows['Mixed Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Hulu' is '{hulu_mixed_directors_tvshows['Mixed
Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Prime Video' is
'{prime_video_mixed_directors_tvshows['Mixed Director'].unique().shape[0]}'\n
    Total Count of Mixed Director on 'Disney+' is '{disney_mixed_directors_tvshows['Mixed
Director'].unique().shape[0]}'
    ''')

```

In[101]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:5], y =
mixed_directors_data_tvshows['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:5], y =
mixed_directors_data_tvshows['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:5], y =
mixed_directors_data_tvshows['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:5], y =
mixed_directors_data_tvshows['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Director', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

In[102]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_d_ax1 = sns.barplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_d_ax2 = sns.barplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_d_ax3 = sns.barplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_d_ax4 = sns.barplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

```

```

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_d_ax1.title.set_text(labels[0])
h_d_ax2.title.set_text(labels[1])
p_d_ax3.title.set_text(labels[2])
d_d_ax4.title.set_text(labels[3])

plt.show()

# In[103]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_md_ax1 = sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_md_ax2 = sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_md_ax3 = sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_md_ax4 = sns.lineplot(x = mixed_directors_data_tvshows['Mixed Director'][:10], y =
mixed_directors_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_md_ax1.title.set_text(labels[0])
h_md_ax2.title.set_text(labels[1])
p_md_ax3.title.set_text(labels[2])
d_md_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Director TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_directors_tvshows['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_directors_tvshows['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_directors_tvshows['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_directors_tvshows['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()

# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_md_ax1 = sns.barplot(x = netflix_mixed_directors_tvshows['Mixed Director'][:10], y =
netflix_mixed_directors_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])

```

```

h_md_ax2 = sns.barplot(x = hulu_mixed_directors_tvshows['Mixed Director'][:10], y =
hulu_mixed_directors_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_md_ax3 = sns.barplot(x = prime_video_mixed_directors_tvshows['Mixed Director'][:10], y =
prime_video_mixed_directors_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_md_ax4 = sns.barplot(x = disney_mixed_directors_tvshows['Mixed Director'][:10], y =
disney_mixed_directors_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_md_ax1.title.set_text(labels[0])
h_md_ax2.title.set_text(labels[1])
p_md_ax3.title.set_text(labels[2])
d_md_ax4.title.set_text(labels[3])

plt.show()

# In[106]:
```

```

fig = go.Figure(go.Funnel(y = mixed_directors_data_tvshows['Mixed Director'][:10], x =
mixed_directors_data_tvshows['TV Shows Count'][:10]))
fig.show()
```

otttvshows_genre.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[116]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
from sklearn.manifold import TSNE
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_tvshows)
# profile
```

```
# In[7]:
```

```
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('*****')
    print('Colums Names : \n', df.columns)
    print('*****')
    print('Datatype of Columns : \n', df.dtypes)
    print('*****')
```

```

print('Missing Values : ')
c = df.isnull().sum()
c = c[c > 0]
print(c)
print('***'*25)
print('Missing values %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***'*25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

```

In[8]:

```
data_investigate(df_tvshows)
```

In[9]:

```

# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

```

```

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
```

data_investigate(df_tvshows)

```
# In[11]:
```

df_tvshows.head()

```
# In[12]:
```

df_tvshows.describe()

```
# In[13]:
```

```

df_tvshows.corr()

# In[14]:


# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]

# In[107]:


df_tvshows_genres = df_tvshows.copy()

# In[108]:


df_tvshows_genres.drop(df_tvshows_genres.loc[df_tvshows_genres['Genres'] == "NA"].index,
inplace = True)
# df_tvshows_genres = df_tvshows_genres[df_tvshows_genres.Genre != "NA"]
# df_tvshows_genres['Genres'] = df_tvshows_genres['Genres'].astype(str)

```

```

# In[109]:


df_tvshows_count_genres = df_tvshows_genres.copy()

# In[110]:


df_tvshows_genre = df_tvshows_genres.copy()

# In[125]:


df_tvshows_genre_all = df_tvshows_genres.copy()

# In[22]:


# Create genres dict where key=name and value = number of genres

genres = {}

for i in df_tvshows_count_genres['Genres'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        genres[i] = len(i.split(','))
    else:
        genres[i] = 0

# Add this information to our dataframe as a new column

df_tvshows_count_genres['Number of Genres'] =
df_tvshows_count_genres['Genres'].map(genres).astype(int)

# In[23]:


df_tvshows_mixed_genres = df_tvshows_count_genres.copy()

# In[24]:


# Creating distinct dataframes only with the tvshows present on individual streaming
# platforms
netflix_genres_tvshows = df_tvshows_count_genres.loc[df_tvshows_count_genres['Netflix'] ==
1]
hulu_genres_tvshows = df_tvshows_count_genres.loc[df_tvshows_count_genres['Hulu'] == 1]
prime_video_genres_tvshows = df_tvshows_count_genres.loc[df_tvshows_count_genres['Prime
Video'] == 1]
disney_genres_tvshows = df_tvshows_count_genres.loc[df_tvshows_count_genres['Disney+'] ==
1]

# In[25]:

```

```

plt.figure(figsize = (10, 10))
corr = df_tvshows_count_genres.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

```

In[26]:

```

df_genres_most_tvshows = df_tvshows_count_genres.sort_values(by = 'Number of Genres',
ascending = False).reset_index()
df_genres_most_tvshows = df_genres_most_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_genres['Number of Genres'] == (df_tvshows_count_genres['Number of Genres'].max()))
# df_genres_most_tvshows = df_tvshows_count_genres[filter]

# mostest_rated_tvshows = df_tvshows_count_genres.loc[df_tvshows_count_genres['Number of Genres'].idxmax()]

print('\nTV Shows with Highest Ever Number of Genres are : \n')
df_genres_most_tvshows.head(5)

```

In[27]:

```

fig = px.bar(y = df_genres_most_tvshows['Title'][:15],
              x = df_genres_most_tvshows['Number of Genres'][:15],
              color = df_genres_most_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title = 'TV Shows with Highest Number of Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[28]:

```

df_genres_least_tvshows = df_tvshows_count_genres.sort_values(by = 'Number of Genres',
ascending = True).reset_index()
df_genres_least_tvshows = df_genres_least_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_genres['Number of Genres'] == (df_tvshows_count_genres['Number of Genres'].min()))
# df_genres_least_tvshows = df_tvshows_count_genres[filter]

print('\nTV Shows with Lowest Ever Number of Genres are : \n')
df_genres_least_tvshows.head(5)

```

```
# In[29]:
```

```
fig = px.bar(y = df_genres_least_tvshows['Title'][:15],
              x = df_genres_least_tvshows['Number of Genres'][:15],
              color = df_genres_least_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title = 'TV Shows with Lowest Number of Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[30]:
```

```
print(f'''  
Total '{df_tvshows_count_genres['Number of Genres'].unique().shape[0]}' unique Number  
of Genres s were Given, They were Like this,\n  
{df_tvshows_count_genres.sort_values(by = 'Number of Genres', ascending =  
False)['Number of Genres'].unique()}\n  
  
The Highest Number of Genres Ever Any TV Show Got is  
'{df_genres_most_tvshows['Title'][0]}' : '{df_genres_most_tvshows['Number of  
Genres']].max()}'\n  
  
The Lowest Number of Genres Ever Any TV Show Got is  
'{df_genres_least_tvshows['Title'][0]}' : '{df_genres_least_tvshows['Number of  
Genres']].min()}'\n''' )
```

```
# In[31]:
```

```
netflix_genres_most_tvshows =  
df_genres_most_tvshows.loc[df_genres_most_tvshows['Netflix']==1].reset_index()  
netflix_genres_most_tvshows = netflix_genres_most_tvshows.drop(['index'], axis = 1)  
  
netflix_genres_least_tvshows =  
df_genres_least_tvshows.loc[df_genres_least_tvshows['Netflix']==1].reset_index()  
netflix_genres_least_tvshows = netflix_genres_least_tvshows.drop(['index'], axis = 1)  
  
netflix_genres_most_tvshows.head(5)
```

```
# In[32]:
```

```
fig = px.bar(y = netflix_genres_most_tvshows['Title'][:15],
              x = netflix_genres_most_tvshows['Number of Genres'][:15],
              color = netflix_genres_most_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title = 'TV Shows with Highest Number of Genres : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:  
  
fig = px.bar(y = netflix_genres_least_tvshows['Title'][:15],  
             x = netflix_genres_least_tvshows['Number of Genres'][:15],  
             color = netflix_genres_least_tvshows['Number of Genres'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},  
             title  = 'TV Shows with Lowest Number of Genres : Netflix')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[34]:
```

```
hulu_genres_most_tvshows =  
df_genres_most_tvshows.loc[df_genres_most_tvshows['Hulu']==1].reset_index()  
hulu_genres_most_tvshows = hulu_genres_most_tvshows.drop(['index'], axis = 1)  
  
hulu_genres_least_tvshows =  
df_genres_least_tvshows.loc[df_genres_least_tvshows['Hulu']==1].reset_index()  
hulu_genres_least_tvshows = hulu_genres_least_tvshows.drop(['index'], axis = 1)  
  
hulu_genres_most_tvshows.head(5)
```

```
# In[35]:
```

```
fig = px.bar(y = hulu_genres_most_tvshows['Title'][:15],  
             x = hulu_genres_most_tvshows['Number of Genres'][:15],  
             color = hulu_genres_most_tvshows['Number of Genres'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},  
             title  = 'TV Shows with Highest Number of Genres : Hulu')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[36]:
```

```
fig = px.bar(y = hulu_genres_least_tvshows['Title'][:15],  
             x = hulu_genres_least_tvshows['Number of Genres'][:15],  
             color = hulu_genres_least_tvshows['Number of Genres'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},  
             title  = 'TV Shows with Lowest Number of Genres : Hulu')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[37]:
```

```

prime_video_genres_most_tvshows = df_genres_most_tvshows.loc[df_genres_most_tvshows['Prime Video']==1].reset_index()
prime_video_genres_most_tvshows = prime_video_genres_most_tvshows.drop(['index'], axis = 1)

prime_video_genres_least_tvshows =
df_genres_least_tvshows.loc[df_genres_least_tvshows['Prime Video']==1].reset_index()
prime_video_genres_least_tvshows = prime_video_genres_least_tvshows.drop(['index'], axis = 1)

prime_video_genres_most_tvshows.head(5)

# In[38]:


fig = px.bar(y = prime_video_genres_most_tvshows['Title'][:15],
              x = prime_video_genres_most_tvshows['Number of Genres'][:15],
              color = prime_video_genres_most_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title = 'TV Shows with Highest Number of Genres : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[39]:


fig = px.bar(y = prime_video_genres_least_tvshows['Title'][:15],
              x = prime_video_genres_least_tvshows['Number of Genres'][:15],
              color = prime_video_genres_least_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title = 'TV Shows with Lowest Number of Genres : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[40]:


disney_genres_most_tvshows =
df_genres_most_tvshows.loc[df_genres_most_tvshows['Disney+']==1].reset_index()
disney_genres_most_tvshows = disney_genres_most_tvshows.drop(['index'], axis = 1)

disney_genres_least_tvshows =
df_genres_least_tvshows.loc[df_genres_least_tvshows['Disney+']==1].reset_index()
disney_genres_least_tvshows = disney_genres_least_tvshows.drop(['index'], axis = 1)

disney_genres_most_tvshows.head(5)

# In[41]:


fig = px.bar(y = disney_genres_most_tvshows['Title'][:15],
              x = disney_genres_most_tvshows['Number of Genres'][:15],
              color = disney_genres_most_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',

```

```

        labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
        title  = 'TV Shows with Highest Number of Genres : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_genres_least_tvshows['Title'][:15],
              x = disney_genres_least_tvshows['Number of Genres'][:15],
              color = disney_genres_least_tvshows['Number of Genres'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Genres'},
              title  = 'TV Shows with Lowest Number of Genres : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''The TV Show with Highest Number of Genres Ever Got is
'{df_genres_most_tvshows['Title'][0]}' : '{df_genres_most_tvshows['Number of Genres']].max()}'\n
The TV Show with Lowest Number of Genres Ever Got is
'{df_genres_least_tvshows['Title'][0]}' : '{df_genres_least_tvshows['Number of Genres']].min()}'\n

The TV Show with Highest Number of Genres on 'Netflix' is
'{netflix_genres_most_tvshows['Title'][0]}' : '{netflix_genres_most_tvshows['Number of Genres']].max()}'\n
The TV Show with Lowest Number of Genres on 'Netflix' is
'{netflix_genres_least_tvshows['Title'][0]}' : '{netflix_genres_least_tvshows['Number of Genres']].min()}'\n

The TV Show with Highest Number of Genres on 'Hulu' is
'{hulu_genres_most_tvshows['Title'][0]}' : '{hulu_genres_most_tvshows['Number of Genres']].max()}'\n
The TV Show with Lowest Number of Genres on 'Hulu' is
'{hulu_genres_least_tvshows['Title'][0]}' : '{hulu_genres_least_tvshows['Number of Genres']].min()}'\n

The TV Show with Highest Number of Genres on 'Prime Video' is
'{prime_video_genres_most_tvshows['Title'][0]}' : '{prime_video_genres_most_tvshows['Number of Genres']].max()}'\n
The TV Show with Lowest Number of Genres on 'Prime Video' is
'{prime_video_genres_least_tvshows['Title'][0]}' :
'{prime_video_genres_least_tvshows['Number of Genres']].min()}'\n

The TV Show with Highest Number of Genres on 'Disney+' is
'{disney_genres_most_tvshows['Title'][0]}' : '{disney_genres_most_tvshows['Number of Genres']].max()}'\n
The TV Show with Lowest Number of Genres on 'Disney+' is
'{disney_genres_least_tvshows['Title'][0]}' : '{disney_genres_least_tvshows['Number of Genres']].min()}'\n
'''')

```

```

# In[44]:


print(f'''
    Accross All Platforms the Average Number of Genres is
'{round(df_tvshows_count_genres['Number of Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Netflix' is '{round(netflix_genres_tvshows['Number
of Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Hulu' is '{round(hulu_genres_tvshows['Number of
Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Prime Video' is
'{round(prime_video_genres_tvshows['Number of Genres'].mean(), ndigits = 2)}'\n
    The Average Number of Genres on 'Disney+' is '{round(disney_genres_tvshows['Number of
Genres'].mean(), ndigits = 2)}'\n
''')


# In[45]:


print(f'''
    Accross All Platforms Total Count of Genre is '{df_tvshows_count_genres['Number of
Genres'].max()}'\n
    Total Count of Genre on 'Netflix' is '{netflix_genres_tvshows['Number of
Genres'].max()}'\n
    Total Count of Genre on 'Hulu' is '{hulu_genres_tvshows['Number of Genres'].max()}'\n
    Total Count of Genre on 'Prime Video' is '{prime_video_genres_tvshows['Number of
Genres'].max()}'\n
    Total Count of Genre on 'Disney+' is '{disney_genres_tvshows['Number of
Genres'].max()}'\n
''')


# In[46]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_count_genres['Number of Genres'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_count_genres['Number of Genres'], ax = ax[1])
plt.show()


# In[47]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Number of Genres s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_genres_tvshows['Number of Genres'], color = 'lightblue', legend =
True, kde = True)
sns.histplot(netflix_genres_tvshows['Number of Genres'], color = 'red', legend = True, kde =
True)
sns.histplot(hulu_genres_tvshows['Number of Genres'], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_genres_tvshows['Number of Genres'], color = 'darkblue', legend = True,
kde = True)

# Setting the legend

```

```
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()
```

```
# In[48]:
```

```
df_lan = df_tvshows_genre['Genres'].str.split(',').apply(pd.Series).stack()
del df_tvshows_genre['Genres']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Genre'
df_tvshows_genre = df_tvshows_genre.join(df_lan)
df_tvshows_genre.drop_duplicates(inplace = True)
```

```
# In[49]:
```

```
df_tvshows_genre.head(5)
```

```
# In[50]:
```

```
genre_count = df_tvshows_genre.groupby('Genre')['Title'].count()
genre_tvshows = df_tvshows_genre.groupby('Genre')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
genre_data_tvshows = pd.concat([genre_count, genre_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
genre_data_tvshows = genre_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)
```

```
# In[51]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_genre_tvshows = netflix_genre_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

hulu_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_genre_tvshows = hulu_genre_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+',
'TV Shows Count'], axis = 1)

prime_video_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_genre_tvshows = prime_video_genre_tvshows.drop(['index', 'Netflix', 'Hulu',
'Disney+', 'TV Shows Count'], axis = 1)

disney_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_genre_tvshows = disney_genre_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime
Video', 'TV Shows Count'], axis = 1)
```

```
# In[52]:
```

```
# Genre with TV Shows Counts - All Platforms Combined  
genre_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]
```

```
# In[53]:
```

```
fig = px.bar(x = genre_data_tvshows['Genre'][:50],  
             y = genre_data_tvshows['TV Shows Count'][:50],  
             color = genre_data_tvshows['TV Shows Count'][:50],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'x' : 'Genre', 'y' : 'TV Shows Count'},  
             title = 'Major Genres : All Platforms')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[54]:
```

```
df_genre_high_tvshows = genre_data_tvshows.sort_values(by = 'TV Shows Count', ascending =  
False).reset_index()  
df_genre_high_tvshows = df_genre_high_tvshows.drop(['index'], axis = 1)  
# filter = (genre_data_tvshows['TV Shows Count'] == (genre_data_tvshows['TV Shows  
Count'].max()))  
# df_genre_high_tvshows = genre_data_tvshows[filter]  
  
# highestRated_tvshows = genre_data_tvshows.loc[genre_data_tvshows['TV Shows  
Count'].idxmax()]  
  
print('\nGenre with Highest Ever TV Shows Count are : All Platforms Combined\n')  
df_genre_high_tvshows.head(5)
```

```
# In[55]:
```

```
fig = px.bar(y = df_genre_high_tvshows['Genre'][:15],  
             x = df_genre_high_tvshows['TV Shows Count'][:15],  
             color = df_genre_high_tvshows['TV Shows Count'][:15],  
             color_continuous_scale = 'Teal_r',  
             labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},  
             title = 'Genre with Highest TV Shows : All Platforms')  
  
fig.update_layout(plot_bgcolor = 'white')  
fig.show()
```

```
# In[56]:
```

```
df_genre_low_tvshows = genre_data_tvshows.sort_values(by = 'TV Shows Count', ascending =  
True).reset_index()  
df_genre_low_tvshows = df_genre_low_tvshows.drop(['index'], axis = 1)  
# filter = (genre_data_tvshows['TV Shows Count'] == (genre_data_tvshows['TV Shows  
Count'].min()))  
# df_genre_low_tvshows = genre_data_tvshows[filter]  
  
print('\nGenre with Lowest Ever TV Shows Count are : All Platforms Combined\n')
```

```
df_genre_low_tvshows.head(5)
```

```
# In[57]:
```

```
fig = px.bar(y = df_genre_low_tvshows['Genre'][:15],
              x = df_genre_low_tvshows['TV Shows Count'][:15],
              color = df_genre_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},
              title = 'Genre with Lowest TV Shows Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[58]:
```

```
print(f'''
    Total '{genre_data_tvshows['Genre'].unique().shape[0]}' unique Genre Count s were
    Given, They were Like this,\n

    {genre_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Genre'].unique()[:5]}\n

    The Highest Ever TV Shows Count Ever Any TV Show Got is
'{df_genre_high_tvshows['Genre'][0]}' : '{df_genre_high_tvshows['TV Shows Count'].max()}'\n

    The Lowest Ever TV Shows Count Ever Any TV Show Got is
'{df_genre_low_tvshows['Genre'][0]}' : '{df_genre_low_tvshows['TV Shows Count'].min()}'\n
''' )
```

```
# In[59]:
```

```
fig = px.pie(genre_data_tvshows[:10], names = 'Genre', values = 'TV Shows Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Genre')
fig.show()
```

```
# In[60]:
```

```
# netflix_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Netflix'] !=
0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_genre_tvshows = netflix_genre_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

netflix_genre_high_tvshows = df_genre_high_tvshows.sort_values(by = 'Netflix', ascending =
False).reset_index()
netflix_genre_high_tvshows = netflix_genre_high_tvshows.drop(['index'], axis = 1)

netflix_genre_low_tvshows = df_genre_low_tvshows.sort_values(by = 'Netflix', ascending =
True).reset_index()
netflix_genre_low_tvshows = netflix_genre_low_tvshows.drop(['index'], axis = 1)
```

```
netflix_genre_high_tvshows.head(5)
```

```
# In[61]:
```

```
fig = px.bar(x = netflix_genre_high_tvshows['Genre'][:15],
              y = netflix_genre_high_tvshows['Netflix'][:15],
              color = netflix_genre_high_tvshows['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},
              title = 'Genre with Highest TV Shows : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[62]:
```

```
# hulu_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_genre_tvshows = hulu_genre_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_genre_high_tvshows = df_genre_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_genre_high_tvshows = hulu_genre_high_tvshows.drop(['index'], axis = 1)

hulu_genre_low_tvshows = df_genre_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_genre_low_tvshows = hulu_genre_low_tvshows.drop(['index'], axis = 1)

hulu_genre_high_tvshows.head(5)
```

```
# In[63]:
```

```
fig = px.bar(x = hulu_genre_high_tvshows['Genre'][:15],
              y = hulu_genre_high_tvshows['Hulu'][:15],
              color = hulu_genre_high_tvshows['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},
              title = 'Genre with Highest TV Shows : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[64]:
```

```
# prime_video_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_genre_tvshows = prime_video_genre_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_genre_high_tvshows = df_genre_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_genre_high_tvshows = prime_video_genre_high_tvshows.drop(['index'], axis = 1)
```

```

prime_video_genre_low_tvshows = df_genre_high_tvshows.sort_values(by = 'Prime Video',
ascending = True).reset_index()
prime_video_genre_low_tvshows = prime_video_genre_low_tvshows.drop(['index'], axis = 1)

prime_video_genre_high_tvshows.head(5)

# In[65]:


fig = px.bar(x = prime_video_genre_high_tvshows['Genre'][:15],
              y = prime_video_genre_high_tvshows['Prime Video'][:15],
              color = prime_video_genre_high_tvshows['Prime Video'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},
              title  = 'Genre with Highest TV Shows : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[66]:


# disney_genre_tvshows = genre_data_tvshows[genre_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_genre_tvshows = disney_genre_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_genre_high_tvshows = df_genre_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_genre_high_tvshows = disney_genre_high_tvshows.drop(['index'], axis = 1)

disney_genre_low_tvshows = df_genre_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_genre_low_tvshows = disney_genre_low_tvshows.drop(['index'], axis = 1)

disney_genre_high_tvshows.head(5)

# In[67]:


fig = px.bar(x = disney_genre_high_tvshows['Genre'][:15],
              y = disney_genre_high_tvshows['Disney+'][:15],
              color = disney_genre_high_tvshows['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Genre', 'x' : 'TV Shows Count'},
              title  = 'Genre with Highest TV Shows : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[68]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(genre_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(genre_data_tvshows['TV Shows Count'], ax = ax[1])

```

```
plt.show()
```

```
# In[69]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Genre TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_genre_tvshows['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_genre_tvshows['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_genre_tvshows['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_genre_tvshows['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()
```

```
# In[70]:
```

```
print(f'''
    The Genre with Highest TV Shows Count Ever Got is
'{df_genre_high_tvshows['Genre'][0]}' : '{df_genre_high_tvshows['TV Shows Count'].max()}'\n
    The Genre with Lowest TV Shows Count Ever Got is '{df_genre_low_tvshows['Genre'][0]}'
: '{df_genre_low_tvshows['TV Shows Count'].min()}'\n

    The Genre with Highest TV Shows Count on 'Netflix' is
'{netflix_genre_high_tvshows['Genre'][0]}' :
'{netflix_genre_high_tvshows['Netflix'].max()}'\n
    The Genre with Lowest TV Shows Count on 'Netflix' is
'{netflix_genre_low_tvshows['Genre'][0]}' :
'{netflix_genre_low_tvshows['Netflix'].min()}'\n

    The Genre with Highest TV Shows Count on 'Hulu' is
'{hulu_genre_high_tvshows['Genre'][0]}' : '{hulu_genre_high_tvshows['Hulu'].max()}'\n
    The Genre with Lowest TV Shows Count on 'Hulu' is
'{hulu_genre_low_tvshows['Genre'][0]}' : '{hulu_genre_low_tvshows['Hulu'].min()}'\n

    The Genre with Highest TV Shows Count on 'Prime Video' is
'{prime_video_genre_high_tvshows['Genre'][0]}' : '{prime_video_genre_high_tvshows['Prime
Video'].max()}'\n
    The Genre with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_genre_low_tvshows['Genre'][0]}' : '{prime_video_genre_low_tvshows['Prime
Video'].min()}'\n

    The Genre with Highest TV Shows Count on 'Disney+' is
'{disney_genre_high_tvshows['Genre'][0]}' :
'{disney_genre_high_tvshows['Disney+'].max()}'\n
    The Genre with Lowest TV Shows Count on 'Disney+' is
'{disney_genre_low_tvshows['Genre'][0]}' : '{disney_genre_low_tvshows['Disney+'].min()}'\n
'''')
```

```
# In[71]:
```

```
# Distribution of tvshows genre in each platform
plt.figure(figsize = (20, 5))
plt.title('Genre with TV Shows Count for All Platforms')
sns.violinplot(x = genre_data_tvshows['TV Shows Count'][:100], color = 'gold', legend =
True, kde = True, shade = False)
plt.show()
```

```
# In[72]:
```

```
# Distribution of Genre TV Shows Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_genre_tvshows['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_genre_tvshows['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_genre_tvshows['Prime Video'][:100], color = 'lightblue', ax
= ax2[0])
sns.violinplot(x = disney_genre_tvshows['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()
```

```
# In[73]:
```

```
print(f'''
    Accross All Platforms the Average TV Shows Count of Genre is
'{round(genre_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Genre on 'Netflix' is
'{round(netflix_genre_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Genre on 'Hulu' is
'{round(hulu_genre_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Genre on 'Prime Video' is
'{round(prime_video_genre_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Genre on 'Disney+' is
'{round(disney_genre_tvshows['Disney+'].mean(), ndigits = 2)}'
    ''')
```

```
# In[74]:
```

```
print(f'''
    Accross All Platforms Total Count of Genre is
'{genre_data_tvshows['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Netflix' is
'{netflix_genre_tvshows['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Hulu' is '{hulu_genre_tvshows['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Prime Video' is
'{prime_video_genre_tvshows['Genre'].unique().shape[0]}'\n
    Total Count of Genre on 'Disney+' is
'{disney_genre_tvshows['Genre'].unique().shape[0]}'
    ''')
```

```
# In[75]:
```

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = genre_data_tvshows['Genre'][:10], y = genre_data_tvshows['Netflix'][:10],
color = 'red')
sns.lineplot(x = genre_data_tvshows['Genre'][:10], y = genre_data_tvshows['Hulu'][:10],
color = 'lightgreen')
sns.lineplot(x = genre_data_tvshows['Genre'][:10], y = genre_data_tvshows['Prime
Video'][:10], color = 'lightblue')
sns.lineplot(x = genre_data_tvshows['Genre'][:10], y = genre_data_tvshows['Disney+'][:10],
color = 'darkblue')
plt.xlabel('Genre', fontsize = 20)
plt.ylabel('TV Shows Count', fontsize = 20)
plt.show()

```

In[76]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_g_ax1 = sns.lineplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_g_ax2 = sns.lineplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_g_ax3 = sns.lineplot(y = genre_data_tvshows['Genre'][:10], x = genre_data_tvshows['Prime
Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_g_ax4 = sns.lineplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()

```

In[77]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = netflix_genre_tvshows['Genre'][:10], x =
netflix_genre_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_g_ax2 = sns.barplot(y = hulu_genre_tvshows['Genre'][:10], x =
hulu_genre_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = prime_video_genre_tvshows['Genre'][:10], x =
prime_video_genre_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_g_ax4 = sns.barplot(y = disney_genre_tvshows['Genre'][:10], x =
disney_genre_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

```

```

plt.show()

# In[78]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Genre TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_genre_tvshows['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_genre_tvshows['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_genre_tvshows['Prime Video'][:10], color = 'lightblue', legend = True)
sns.kdeplot(disney_genre_tvshows['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()


# In[79]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_g_ax2 = sns.barplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = genre_data_tvshows['Genre'][:10], x = genre_data_tvshows['Prime
Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_g_ax4 = sns.barplot(y = genre_data_tvshows['Genre'][:10], x =
genre_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_g_ax1.title.set_text(labels[0])
h_g_ax2.title.set_text(labels[1])
p_g_ax3.title.set_text(labels[2])
d_g_ax4.title.set_text(labels[3])

plt.show()


# In[80]:


df_tvshows_mixed_genres.drop(df_tvshows_mixed_genres.loc[df_tvshows_mixed_genres['Genres']
== "NA"].index, inplace = True)
# df_tvshows_mixed_genres = df_tvshows_mixed_genres[df_tvshows_mixed_genres.Genre != "NA"]
df_tvshows_mixed_genres.drop(df_tvshows_mixed_genres.loc[df_tvshows_mixed_genres['Number of
Genres'] == 1].index, inplace = True)


# In[81]:


df_tvshows_mixed_genres.head(5)

```

```

# In[82]:


mixed_genres_count = df_tvshows_mixed_genres.groupby('Genres')['Title'].count()
mixed_genres_tvshows = df_tvshows_mixed_genres.groupby('Genres')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_genres_data_tvshows = pd.concat([mixed_genres_count, mixed_genres_tvshows], axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count', 'Genres' : 'Mixed Genre'})
mixed_genres_data_tvshows = mixed_genres_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[83]:


mixed_genres_data_tvshows.head(5)

# In[84]:


# Mixed Genre with TV Shows Counts - All Platforms Combined
mixed_genres_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[85]:


df_mixed_genres_high_tvshows = mixed_genres_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_mixed_genres_high_tvshows = df_mixed_genres_high_tvshows.drop(['index'], axis = 1)
# filter = (mixed_genres_data_tvshows['TV Shows Count'] == (mixed_genres_data_tvshows['TV Shows Count'].max()))
# df_mixed_genres_high_tvshows = mixed_genres_data_tvshows[filter]

# highestRated_tvshows = mixed_genres_data_tvshows.loc[mixed_genres_data_tvshows['TV Shows Count'].idxmax()]

print('\nMixed Genre with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_genres_high_tvshows.head(5)

# In[86]:


fig = px.bar(y = df_mixed_genres_high_tvshows['Mixed Genre'][:15],
              x = df_mixed_genres_high_tvshows['TV Shows Count'][:15],
              color = df_mixed_genres_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Genre'},
              title = 'TV Shows with Highest Number of Mixed Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[87]:

```

```

df_mixed_genres_low_tvshows = mixed_genres_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_mixed_genres_low_tvshows = df_mixed_genres_low_tvshows.drop(['index'], axis = 1)
# filter = (mixed_genres_data_tvshows['TV Shows Count'] == (mixed_genres_data_tvshows['TV Shows Count'].min()))
# df_mixed_genres_low_tvshows = mixed_genres_data_tvshows[filter]

print('\nMixed Genre with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_genres_low_tvshows.head(5)

```

In[88]:

```

fig = px.bar(y = df_mixed_genres_low_tvshows['Mixed Genre'][:15],
              x = df_mixed_genres_low_tvshows['TV Shows Count'][:15],
              color = df_mixed_genres_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Genre'},
              title = 'TV Shows with Lowest Number of Mixed Genres : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[89]:

```

print(f'''
    Total '{df_tvshows_genres['Genres'].count()}' Titles are available on All Platforms,
out of which\n
    You Can Choose to see TV Shows from Total '{mixed_genres_data_tvshows['Mixed
Genre'].unique().shape[0]}' Mixed Genre, They were Like this, \n

    {mixed_genres_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Mixed Genre'].head(5).unique()} etc. \n

    The Mixed Genre with Highest TV Shows Count have '{mixed_genres_data_tvshows['TV
Shows Count'].max()}' TV Shows Available is '{df_mixed_genres_high_tvshows['Mixed
Genre'][0]}', &\n
    The Mixed Genre with Lowest TV Shows Count have '{mixed_genres_data_tvshows['TV Shows
Count'].min()}' TV Shows Available is '{df_mixed_genres_low_tvshows['Mixed Genre'][0]}'
    ''')

```

In[90]:

```

fig = px.pie(mixed_genres_data_tvshows[:10], names = 'Mixed Genre', values = 'TV Shows
Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Mixed Genre')
fig.show()

```

In[91]:

```

# netflix_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()

```

```
# netflix_mixed_genres_tvshows = netflix_mixed_genres_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_mixed_genres_high_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_genres_high_tvshows = netflix_mixed_genres_high_tvshows.drop(['index'], axis = 1)

netflix_mixed_genres_low_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_mixed_genres_low_tvshows = netflix_mixed_genres_low_tvshows.drop(['index'], axis = 1)

netflix_mixed_genres_high_tvshows.head(5)
```

```
# In[92]:
```

```
# hulu_mixed_genres_tvshows = mixed_genres_data_tvshows[mixed_genres_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_mixed_genres_tvshows = hulu_mixed_genres_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_genres_high_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_genres_high_tvshows = hulu_mixed_genres_high_tvshows.drop(['index'], axis = 1)

hulu_mixed_genres_low_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_mixed_genres_low_tvshows = hulu_mixed_genres_low_tvshows.drop(['index'], axis = 1)

hulu_mixed_genres_high_tvshows.head(5)
```

```
# In[93]:
```

```
# prime_video_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_genres_tvshows = prime_video_mixed_genres_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_genres_high_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_genres_high_tvshows =
prime_video_mixed_genres_high_tvshows.drop(['index'], axis = 1)

prime_video_mixed_genres_low_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_mixed_genres_low_tvshows = prime_video_mixed_genres_low_tvshows.drop(['index'], axis = 1)

prime_video_mixed_genres_high_tvshows.head(5)
```

```
# In[94]:
```

```

# disney_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
# disney_mixed_genres_tvshows = disney_mixed_genres_tvshows.drop(['index', 'Netflix',
'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_mixed_genres_high_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_genres_high_tvshows = disney_mixed_genres_high_tvshows.drop(['index'], axis = 1)

disney_mixed_genres_low_tvshows = df_mixed_genres_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_genres_low_tvshows = disney_mixed_genres_low_tvshows.drop(['index'], axis = 1)

disney_mixed_genres_high_tvshows.head(5)

# In[95]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_genres_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_genres_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[96]:


# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_mixed_genres_tvshows = netflix_mixed_genres_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_genres_tvshows = mixed_genres_data_tvshows[mixed_genres_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_genres_tvshows = hulu_mixed_genres_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Prime Video'] != 0].sort_values(by =
'Prime Video', ascending = False).reset_index()
prime_video_mixed_genres_tvshows = prime_video_mixed_genres_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_mixed_genres_tvshows =
mixed_genres_data_tvshows[mixed_genres_data_tvshows['Disney+'] != 0].sort_values(by =
'Disney+', ascending = False).reset_index()
disney_mixed_genres_tvshows = disney_mixed_genres_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

# In[97]:


# Defining plot size and title

```

```

plt.figure(figsize = (20, 5))
plt.title('Mixed Genre TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_genres_tvshows['Prime Video'][:100], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_mixed_genres_tvshows['Netflix'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_mixed_genres_tvshows['Hulu'][:100], color = 'lightgreen', legend = True,
kde = True)
sns.histplot(disney_mixed_genres_tvshows['Disney+'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Genre with Highest TV Shows Count Ever Got is
'{df_mixed_genres_high_tvshows['Mixed Genre'][0]}' : '{df_mixed_genres_high_tvshows['TV
Shows Count'].max()}'\n
    The Mixed Genre with Lowest TV Shows Count Ever Got is
'{df_mixed_genres_low_tvshows['Mixed Genre'][0]}' : '{df_mixed_genres_low_tvshows['TV Shows
Count'].min()}'\n

    The Mixed Genre with Highest TV Shows Count on 'Netflix' is
'{netflix_mixed_genres_high_tvshows['Mixed Genre'][0]}' :
'{netflix_mixed_genres_high_tvshows['Netflix'].max()}'\n
    The Mixed Genre with Lowest TV Shows Count on 'Netflix' is
'{netflix_mixed_genres_low_tvshows['Mixed Genre'][0]}' :
'{netflix_mixed_genres_low_tvshows['Netflix'].min()}'\n

    The Mixed Genre with Highest TV Shows Count on 'Hulu' is
'{hulu_mixed_genres_high_tvshows['Mixed Genre'][0]}' :
'{hulu_mixed_genres_high_tvshows['Hulu'].max()}'\n
    The Mixed Genre with Lowest TV Shows Count on 'Hulu' is
'{hulu_mixed_genres_low_tvshows['Mixed Genre'][0]}' :
'{hulu_mixed_genres_low_tvshows['Hulu'].min()}'\n

    The Mixed Genre with Highest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_genres_high_tvshows['Mixed Genre'][0]}' :
'{prime_video_mixed_genres_high_tvshows['Prime Video'].max()}'\n
    The Mixed Genre with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_genres_low_tvshows['Mixed Genre'][0]}' :
'{prime_video_mixed_genres_low_tvshows['Prime Video'].min()}'\n

    The Mixed Genre with Highest TV Shows Count on 'Disney+' is
'{disney_mixed_genres_high_tvshows['Mixed Genre'][0]}' :
'{disney_mixed_genres_high_tvshows['Disney+'].max()}'\n
    The Mixed Genre with Lowest TV Shows Count on 'Disney+' is
'{disney_mixed_genres_low_tvshows['Mixed Genre'][0]}' :
'{disney_mixed_genres_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Accross All Platforms the Average TV Shows Count of Mixed Genre is
'{round(mixed_genres_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Genre on 'Netflix' is
'{round(netflix_mixed_genres_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Genre on 'Hulu' is
'{round(hulu_mixed_genres_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Genre on 'Prime Video' is
'{round(prime_video_mixed_genres_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Genre on 'Disney+' is
'{round(disney_mixed_genres_tvshows['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```

print(f'''
    Accross All Platforms Total Count of Mixed Genre is
'{mixed_genres_data_tvshows['Mixed Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Netflix' is '{netflix_mixed_genres_tvshows['Mixed
Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Hulu' is '{hulu_mixed_genres_tvshows['Mixed
Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Prime Video' is
'{prime_video_mixed_genres_tvshows['Mixed Genre'].unique().shape[0]}'\n
    Total Count of Mixed Genre on 'Disney+' is '{disney_mixed_genres_tvshows['Mixed
Genre'].unique().shape[0]}'\n
    ''')

```

In[101]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_genres_data_tvshows['Mixed Genre'][:5], y =
mixed_genres_data_tvshows['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_genres_data_tvshows['Mixed Genre'][:5], y =
mixed_genres_data_tvshows['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_genres_data_tvshows['Mixed Genre'][:5], y =
mixed_genres_data_tvshows['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_genres_data_tvshows['Mixed Genre'][:5], y =
mixed_genres_data_tvshows['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Genre', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

In[102]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_g_ax1 = sns.barplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =
mixed_genres_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_g_ax2 = sns.barplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =
mixed_genres_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_g_ax3 = sns.barplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =
mixed_genres_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])

```

```
d_g_ax4 = sns.barplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =  
mixed_genres_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])
```

```
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']
```

```
n_g_ax1.title.set_text(labels[0])  
h_g_ax2.title.set_text(labels[1])  
p_g_ax3.title.set_text(labels[2])  
d_g_ax4.title.set_text(labels[3])
```

```
plt.show()
```

```
# In[103]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))
```

```
n_mg_ax1 = sns.lineplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =  
mixed_genres_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])  
h_mg_ax2 = sns.lineplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =  
mixed_genres_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])  
p_mg_ax3 = sns.lineplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =  
mixed_genres_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])  
d_mg_ax4 = sns.lineplot(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =  
mixed_genres_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])
```

```
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']
```

```
n_mg_ax1.title.set_text(labels[0])  
h_mg_ax2.title.set_text(labels[1])  
p_mg_ax3.title.set_text(labels[2])  
d_mg_ax4.title.set_text(labels[3])
```

```
plt.show()
```

```
# In[104]:
```

```
# Defining plot size and title  
plt.figure(figsize = (20, 5))  
plt.title('Mixed Genre TV Shows Count Per Platform')  
  
# Plotting the information from each dataset into a histogram  
sns.kdeplot(netflix_mixed_genres_tvshows['Netflix'][:50], color = 'red', legend = True)  
sns.kdeplot(hulu_mixed_genres_tvshows['Hulu'][:50], color = 'green', legend = True)  
sns.kdeplot(prime_video_mixed_genres_tvshows['Prime Video'][:50], color = 'lightblue',  
legend = True)  
sns.kdeplot(disney_mixed_genres_tvshows['Disney+'][:50], color = 'darkblue', legend = True)
```

```
# Setting the legend
```

```
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])  
plt.show()
```

```
# In[105]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))
```

```

n_mg_ax1 = sns.barplot(y = netflix_mixed_genres_tvshows['Mixed Genre'][:10], x =
netflix_mixed_genres_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_mg_ax2 = sns.barplot(y = hulu_mixed_genres_tvshows['Mixed Genre'][:10], x =
hulu_mixed_genres_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_mg_ax3 = sns.barplot(y = prime_video_mixed_genres_tvshows['Mixed Genre'][:10], x =
prime_video_mixed_genres_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_mg_ax4 = sns.barplot(y = disney_mixed_genres_tvshows['Mixed Genre'][:10], x =
disney_mixed_genres_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_mg_ax1.title.set_text(labels[0])
h_mg_ax2.title.set_text(labels[1])
p_mg_ax3.title.set_text(labels[2])
d_mg_ax4.title.set_text(labels[3])

plt.show()

# In[106]:


fig = go.Figure(go.Funnel(y = mixed_genres_data_tvshows['Mixed Genre'][:10], x =
mixed_genres_data_tvshows['TV Shows Count'][:10]))
fig.show()

# In[126]:


genres = df_tvshows_genre_all['Genres'].str.get_dummies(',')
df_tvshows_genre_all = pd.concat([df_tvshows_genre_all, genres], axis = 1, sort = False)

# In[127]:


data_investigate(df_tvshows_genre_all)

# In[128]:


#Select the features on the basis of which you want to cluster
features = df_tvshows_genre_all[['Action', 'Adventure', 'Animation', 'Biography',
                                 'Comedy', 'Crime', 'Documentary', 'Drama', 'Family',
                                 'Fantasy',
                                 'Game-Show', 'History', 'Horror', 'Music', 'Musical',
                                 'Mystery', 'News',
                                 'Reality-TV', 'Romance', 'Sci-Fi', 'Short', 'Sport',
                                 'Talk-Show',
                                 'Thriller', 'War', 'Western']].astype(int)

#Scaling the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(features)

#Using TSNE
tsne = TSNE(n_components = 2)
transformed_genre = tsne.fit_transform(scaled_data)

```

```
# In[130]:  
  
#KMeans - Elbow Method  
distortions = []  
K = range(1, 100)  
for k in K:  
    kmean = KMeans(n_clusters = k)  
    kmean.fit(scaled_data)  
    distortions.append(kmean.inertia_)  
fig = px.line(x = K, y = distortions, title = 'The Elbow Method Showing The Optimal K',  
              labels = {'x':'No of Clusters', 'y':'Distortions'})  
fig.show()
```

```
# In[134]:
```

```
#Kmeans  
cluster = KMeans(n_clusters = 27)  
group_pred = cluster.fit_predict(scaled_data)  
  
tsne_df = pd.DataFrame(np.column_stack((transformed_genre, group_pred,  
df_tvshows_genre_all['Title'], df_tvshows_genre_all['Genres'],  
df_tvshows_genre_all['Service Provider'])), columns = ['X', 'Y', 'Group', 'Title',  
'Genres', 'Service Provider'])  
  
fig = px.scatter(tsne_df, x = 'X', y = 'Y', hover_data = ['Title', 'Genres', 'Service  
Provider'], color = 'Group',  
                 color_discrete_sequence = px.colors.cyclical.IceFire)  
fig.show()
```

otttvshows_imdb.ipynb

```
#!/usr/bin/env python  
# coding: utf-8  
  
# In[1]:  
  
# !pip install git+https://github.com/alberanid/imdbpy  
# !pip install pandas  
# !pip install numpy  
# !pip install matplotlib  
# !pip install seaborn  
# !pip install pandas_profiling --upgrade  
# !pip install plotly  
# !pip install wordcloud  
# !pip install Flask
```

```
# In[2]:
```

```
# Import Dataset  
# Import File from Loacal Drive  
# from google.colab import files
```

```

# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'
df_tvshows = pd.read_csv(path + 'otttvshows.csv')
df_tvshows.head()

# In[6]:


# profile = ProfileReport(df_tvshows)
# profile

# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])

```

```

print('***25)
print('Colums Names : \n', df.columns)
print('***25)
print('Datatype of Columns : \n', df.dtypes)
print('***25)
print('Missing Values : ')
c = df.isnull().sum()
c = c[c > 0]
print(c)
print('***25)
print('Missing vaules %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

```

In[8]:

```
data_investigate(df_tvshows)
```

In[9]:

```

# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)

```

```

df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:

```

```

data_investigate(df_tvshows)

```

```

# In[11]:

```

```

df_tvshows.head()

```

```

# In[12]:

```

```

df_tvshows.describe()

# In[13]:


df_tvshows.corr()

# In[14]:


# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]

# In[18]:


df_tvshows_imdb = df_tvshows.copy()

# In[19]:

```

```

df_tvshows_imdb.drop(df_tvshows_imdb.loc[df_tvshows_imdb['IMDb'] == "NA"].index, inplace = True)
# df_tvshows_imdb = df_tvshows_imdb[df_tvshows_imdb.IMDb != "NA"]
df_tvshows_imdb['IMDb'] = df_tvshows_imdb['IMDb'].astype(int)

# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_imdb_tvshows = df_tvshows_imdb.loc[df_tvshows_imdb['Netflix'] == 1]
hulu_imdb_tvshows = df_tvshows_imdb.loc[df_tvshows_imdb['Hulu'] == 1]
prime_video_imdb_tvshows = df_tvshows_imdb.loc[df_tvshows_imdb['Prime Video'] == 1]
disney_imdb_tvshows = df_tvshows_imdb.loc[df_tvshows_imdb['Disney+'] == 1]

# In[21]:


df_tvshows_imdb_group = df_tvshows_imdb.copy()

# In[22]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_imdb.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[23]:


df_imdb_high_tvshows = df_tvshows_imdb.sort_values(by = 'IMDb', ascending = False).reset_index()
df_imdb_high_tvshows = df_imdb_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_imdb['IMDb'] == (df_tvshows_imdb['IMDb'].max()))
# df_imdb_high_tvshows = df_tvshows_imdb[filter]

# highest_rated_tvshows = df_tvshows_imdb.loc[df_tvshows_imdb['IMDb'].idxmax()]

print('\nTV Shows with Highest Ever IMDb  are : \n')
df_imdb_high_tvshows.head(5)

# In[24]:


fig = px.bar(y = df_imdb_high_tvshows['Title'][:15],
              x = df_imdb_high_tvshows['IMDb'][:15],

```

```

color = df_imdb_high_tvshows['IMDb'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
title = 'TV Shows with Highest IMDb : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[25]:

```

df_imdb_low_tvshows = df_tvshows_imdb.sort_values(by = 'IMDb', ascending =
True).reset_index()
df_imdb_low_tvshows = df_imdb_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_imdb['IMDb'] == (df_tvshows_imdb['IMDb'].min()))
# df_imdb_low_tvshows = df_tvshows_imdb[filter]

print('\nTV Shows with Lowest Ever IMDb  are : \n')
df_imdb_low_tvshows.head(5)

```

In[26]:

```

fig = px.bar(y = df_imdb_low_tvshows['Title'][:15],
              x = df_imdb_low_tvshows['IMDb'][:15],
              color = df_imdb_low_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Lowest IMDb : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[27]:

```

print(f'''
    Total '{df_tvshows_imdb['IMDb'].unique().shape[0]}' unique IMDb s were Given, They
    were Like this,\n

{df_tvshows_imdb.sort_values(by = 'IMDb', ascending = False)['IMDb'].unique()}\n

    The Highest Ever IMDb Ever Any TV Show Got is '{df_imdb_high_tvshows['Title'][0]}' :
'{df_imdb_high_tvshows['IMDb'].max()}'\n

    The Lowest Ever IMDb Ever Any TV Show Got is '{df_imdb_low_tvshows['Title'][0]}' :
'{df_imdb_low_tvshows['IMDb'].min()}'\n
    ''')

```

In[28]:

```

netflix_imdb_high_tvshows =
df_imdb_high_tvshows.loc[df_imdb_high_tvshows['Netflix']==1].reset_index()
netflix_imdb_high_tvshows = netflix_imdb_high_tvshows.drop(['index'], axis = 1)

```

```
netflix_imdb_low_tvshows =
df_imdb_low_tvshows.loc[df_imdb_low_tvshows['Netflix']==1].reset_index()
netflix_imdb_low_tvshows = netflix_imdb_low_tvshows.drop(['index'], axis = 1)

netflix_imdb_high_tvshows.head(5)
```

In[29]:

```
fig = px.bar(y = netflix_imdb_high_tvshows['Title'][:15],
              x = netflix_imdb_high_tvshows['IMDb'][:15],
              color = netflix_imdb_high_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Highest IMDb : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[30]:

```
fig = px.bar(y = netflix_imdb_low_tvshows['Title'][:15],
              x = netflix_imdb_low_tvshows['IMDb'][:15],
              color = netflix_imdb_low_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Lowest IMDb : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

In[31]:

```
hulu_imdb_high_tvshows =
df_imdb_high_tvshows.loc[df_imdb_high_tvshows['Hulu']==1].reset_index()
hulu_imdb_high_tvshows = hulu_imdb_high_tvshows.drop(['index'], axis = 1)

hulu_imdb_low_tvshows =
df_imdb_low_tvshows.loc[df_imdb_low_tvshows['Hulu']==1].reset_index()
hulu_imdb_low_tvshows = hulu_imdb_low_tvshows.drop(['index'], axis = 1)

hulu_imdb_high_tvshows.head(5)
```

In[32]:

```
fig = px.bar(y = hulu_imdb_high_tvshows['Title'][:15],
              x = hulu_imdb_high_tvshows['IMDb'][:15],
              color = hulu_imdb_high_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Highest IMDb : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:
```

```
fig = px.bar(y = hulu_imdb_low_tvshows['Title'][:15],
              x = hulu_imdb_low_tvshows['IMDb'][:15],
              color = hulu_imdb_low_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating' },
              title = 'TV Shows with Lowest IMDb : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[34]:
```

```
prime_video_imdb_high_tvshows = df_imdb_high_tvshows.loc[df_imdb_high_tvshows['Prime Video']==1].reset_index()
prime_video_imdb_high_tvshows = prime_video_imdb_high_tvshows.drop(['index'], axis = 1)

prime_video_imdb_low_tvshows = df_imdb_low_tvshows.loc[df_imdb_low_tvshows['Prime Video']==1].reset_index()
prime_video_imdb_low_tvshows = prime_video_imdb_low_tvshows.drop(['index'], axis = 1)

prime_video_imdb_high_tvshows.head(5)
```



```
# In[35]:
```

```
fig = px.bar(y = prime_video_imdb_high_tvshows['Title'][:15],
              x = prime_video_imdb_high_tvshows['IMDb'][:15],
              color = prime_video_imdb_high_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating' },
              title = 'TV Shows with Highest IMDb : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[36]:
```

```
fig = px.bar(y = prime_video_imdb_low_tvshows['Title'][:15],
              x = prime_video_imdb_low_tvshows['IMDb'][:15],
              color = prime_video_imdb_low_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating' },
              title = 'TV Shows with Lowest IMDb : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[37]:
```

```
disney_imdb_high_tvshows =
df_imdb_high_tvshows.loc[df_imdb_high_tvshows['Disney+']==1].reset_index()
disney_imdb_high_tvshows = disney_imdb_high_tvshows.drop(['index'], axis = 1)
```

```
disney_imdb_low_tvshows =
df_imdb_low_tvshows.loc[df_imdb_low_tvshows['Disney+']==1].reset_index()
disney_imdb_low_tvshows = disney_imdb_low_tvshows.drop(['index'], axis = 1)
```

```
disney_imdb_high_tvshows.head(5)
```

```
# In[38]:
```

```
fig = px.bar(y = disney_imdb_high_tvshows['Title'][:15],
              x = disney_imdb_high_tvshows['IMDb'][:15],
              color = disney_imdb_high_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Highest IMDb : Disney+')
```

```
fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[39]:
```

```
fig = px.bar(y = disney_imdb_low_tvshows['Title'][:15],
              x = disney_imdb_low_tvshows['IMDb'][:15],
              color = disney_imdb_low_tvshows['IMDb'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Lowest IMDb : Disney+')
```

```
fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[40]:
```

```
print(f'''
    The TV Show with Highest IMDb Ever Got is '{df_imdb_high_tvshows['Title'][0]}':
'{df_imdb_high_tvshows['IMDb'].max()}'\n
    The TV Show with Lowest IMDb Ever Got is '{df_imdb_low_tvshows['Title'][0]}':
'{df_imdb_low_tvshows['IMDb'].min()}'\n

    The TV Show with Highest IMDb on 'Netflix' is
'{netflix_imdb_high_tvshows['Title'][0]}': '{netflix_imdb_high_tvshows['IMDb'].max()}'\n
    The TV Show with Lowest IMDb on 'Netflix' is
'{netflix_imdb_low_tvshows['Title'][0]}': '{netflix_imdb_low_tvshows['IMDb'].min()}'\n

    The TV Show with Highest IMDb on 'Hulu' is '{hulu_imdb_high_tvshows['Title'][0]}':
'{hulu_imdb_high_tvshows['IMDb'].max()}'\n
    The TV Show with Lowest IMDb on 'Hulu' is '{hulu_imdb_low_tvshows['Title'][0]}':
'{hulu_imdb_low_tvshows['IMDb'].min()}'\n

    The TV Show with Highest IMDb on 'Prime Video' is
'{prime_video_imdb_high_tvshows['Title'][0]}':
'{prime_video_imdb_high_tvshows['IMDb'].max()}'\n
```

```

    The TV Show with Lowest IMDb on 'Prime Video' is
'{prime_video_imdb_low_tvshows['Title'][0]} :
'{prime_video_imdb_low_tvshows['IMDb'].min()}'\n

    The TV Show with Highest IMDb on 'Disney+' is
'{disney_imdb_high_tvshows['Title'][0]} : '{disney_imdb_high_tvshows['IMDb'].max()}'\n
    The TV Show with Lowest IMDb on 'Disney+' is '{disney_imdb_low_tvshows['Title'][0]}'
: '{disney_imdb_low_tvshows['IMDb'].min()}'\n
'''')

```

In[41]:

```

print(f'''
    Accross All Platforms the Average IMDb is '{round(df_tvshows_imdb['IMDb'].mean(),
ndigits = 2)}'\n
    The Average IMDb on 'Netflix' is '{round(netflix_imdb_tvshows['IMDb'].mean(),
ndigits = 2)}'\n
    The Average IMDb on 'Hulu' is '{round(hulu_imdb_tvshows['IMDb'].mean(), ndigits =
2)}'\n
    The Average IMDb on 'Prime Video' is
'{round(prime_video_imdb_tvshows['IMDb'].mean(), ndigits = 2)}'\n
    The Average IMDb on 'Disney+' is '{round(disney_imdb_tvshows['IMDb'].mean(), ndigits
= 2)}'\n
''')

```

In[42]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_imdb['IMDb'],bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_imdb['IMDb'], ax = ax[1])
plt.show()

```

In[43]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('IMDb s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_imdb_tvshows['IMDb'][:100], color = 'lightblue', legend = True,
kde = True)
sns.histplot(netflix_imdb_tvshows['IMDb'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_imdb_tvshows['IMDb'][:100], color = 'lightgreen', legend = True, kde =
True)
sns.histplot(disney_imdb_tvshows['IMDb'][:100], color = 'darkblue', legend = True, kde =
True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[44]:

```

def round_val(data):
    if str(data) != 'nan':
        return round(data)

# In[45]:


df_tvshows_imdb_group['IMDb Group'] = df_tvshows_imdb['IMDb'].apply(round_val)

imdb_values = df_tvshows_imdb_group['IMDb Group'].value_counts().sort_index(ascending =
False).tolist()
imdb_index = df_tvshows_imdb_group['IMDb Group'].value_counts().sort_index(ascending =
False).index

# imdb_values, imdb_index

# In[46]:


imdb_group_count = df_tvshows_imdb_group.groupby('IMDb Group')['Title'].count()
imdb_group_tvshows = df_tvshows_imdb_group.groupby('IMDb Group')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
imdb_group_data_tvshows = pd.concat([imdb_group_count, imdb_group_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
imdb_group_data_tvshows = imdb_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)

# In[47]:


# IMDb Group with TV Shows Counts - All Platforms Combined
imdb_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[48]:


imdb_group_data_tvshows.sort_values(by = 'IMDb Group', ascending = False)

# In[49]:


fig = px.bar(y = imdb_group_data_tvshows['TV Shows Count'],
              x = imdb_group_data_tvshows['IMDb Group'],
              color = imdb_group_data_tvshows['IMDb Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'IMDb : Rating'},
              title = 'TV Shows with Group IMDb : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]:


fig = px.pie(imdb_group_data_tvshows[:10],

```

```

names = imdb_group_data_tvshows['IMDb Group'],
values = imdb_group_data_tvshows['TV Shows Count'],
color = imdb_group_data_tvshows['TV Shows Count'],
color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                   title = 'TV Shows Count based on IMDb Group')
fig.show()

# In[51]:


df_imdb_group_high_tvshows = imdb_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_imdb_group_high_tvshows = df_imdb_group_high_tvshows.drop(['index'], axis = 1)
# filter = (imdb_group_data_tvshows['TV Shows Count'] == (imdb_group_data_tvshows['TV Shows Count'].max()))
# df_imdb_group_high_tvshows = imdb_group_data_tvshows[filter]

# highestRated_tvshows = imdb_group_data_tvshows.loc[imdb_group_data_tvshows['TV Shows Count'].idxmax()]

# print('\nIMDb with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_imdb_group_high_tvshows.head(5)

# In[52]:


df_imdb_group_low_tvshows = imdb_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_imdb_group_low_tvshows = df_imdb_group_low_tvshows.drop(['index'], axis = 1)
# filter = (imdb_group_data_tvshows['TV Shows Count'] == (imdb_group_data_tvshows['TV Shows Count'].min()))
# df_imdb_group_low_tvshows = imdb_group_data_tvshows[filter]

# print('\nIMDb with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_imdb_group_low_tvshows.head(5)

# In[53]:


print(f'''
      Total '{df_tvshows_imdb['IMDb'].count()}' Titles are available on All Platforms, out
      of which\n
      You Can Choose to see TV Shows from Total '{imdb_group_data_tvshows['IMDb
      Group'].unique().shape[0]}' IMDb Group, They were Like this, \n

      {imdb_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[['IMDb
      Group']].unique()} etc. \n

      The IMDb Group with Highest TV Shows Count have '{imdb_group_data_tvshows['TV Shows
      Count'].max()}' TV Shows Available is '{df_imdb_group_high_tvshows['IMDb Group'][0]}', &\n
      The IMDb Group with Lowest TV Shows Count have '{imdb_group_data_tvshows['TV Shows
      Count'].min()}' TV Shows Available is '{df_imdb_group_low_tvshows['IMDb Group'][0]}'
      ''')

# In[54]:

```

```
netflix_imdb_group_tvshows = imdb_group_data_tvshows[imdb_group_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_imdb_group_tvshows = netflix_imdb_group_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_imdb_group_high_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_imdb_group_high_tvshows = netflix_imdb_group_high_tvshows.drop(['index'], axis = 1)

netflix_imdb_group_low_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_imdb_group_low_tvshows = netflix_imdb_group_low_tvshows.drop(['index'], axis = 1)

netflix_imdb_group_high_tvshows.head(5)
```

```
# In[55]:
```

```
hulu_imdb_group_tvshows = imdb_group_data_tvshows[imdb_group_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_imdb_group_tvshows = hulu_imdb_group_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_imdb_group_high_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_imdb_group_high_tvshows = hulu_imdb_group_high_tvshows.drop(['index'], axis = 1)

hulu_imdb_group_low_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_imdb_group_low_tvshows = hulu_imdb_group_low_tvshows.drop(['index'], axis = 1)

hulu_imdb_group_high_tvshows.head(5)
```

```
# In[56]:
```

```
prime_video_imdb_group_tvshows = imdb_group_data_tvshows[imdb_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_imdb_group_tvshows = prime_video_imdb_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_imdb_group_high_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_imdb_group_high_tvshows = prime_video_imdb_group_high_tvshows.drop(['index'], axis = 1)

prime_video_imdb_group_low_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_imdb_group_low_tvshows = prime_video_imdb_group_low_tvshows.drop(['index'], axis = 1)

prime_video_imdb_group_high_tvshows.head(5)
```

```
# In[57]:
```

```

disney_imdb_group_tvshows = imdb_group_data_tvshows[imdb_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_imdb_group_tvshows = disney_imdb_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_imdb_group_high_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_imdb_group_high_tvshows = disney_imdb_group_high_tvshows.drop(['index'], axis = 1)

disney_imdb_group_low_tvshows = df_imdb_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_imdb_group_low_tvshows = disney_imdb_group_low_tvshows.drop(['index'], axis = 1)

disney_imdb_group_high_tvshows.head(5)

# In[58]:


print(f'''
    The IMDb Group with Highest TV Shows Count Ever Got is
'{df_imdb_group_high_tvshows['IMDb Group'][0]}' : '{df_imdb_group_high_tvshows['TV Shows Count'].max()}'\n
    The IMDb Group with Lowest TV Shows Count Ever Got is
'{df_imdb_group_low_tvshows['IMDb Group'][0]}' : '{df_imdb_group_low_tvshows['TV Shows Count'].min()}'\n

    The IMDb Group with Highest TV Shows Count on 'Netflix' is
'{netflix_imdb_group_high_tvshows['IMDb Group'][0]}' :
'{netflix_imdb_group_high_tvshows['Netflix'].max()}'\n
    The IMDb Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_imdb_group_low_tvshows['IMDb Group'][0]}' :
'{netflix_imdb_group_low_tvshows['Netflix'].min()}'\n

    The IMDb Group with Highest TV Shows Count on 'Hulu' is
'{hulu_imdb_group_high_tvshows['IMDb Group'][0]}' :
'{hulu_imdb_group_high_tvshows['Hulu'].max()}'\n
    The IMDb Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_imdb_group_low_tvshows['IMDb Group'][0]}' :
'{hulu_imdb_group_low_tvshows['Hulu'].min()}'\n

    The IMDb Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_imdb_group_high_tvshows['IMDb Group'][0]}' :
'{prime_video_imdb_group_high_tvshows['Prime Video'].max()}'\n
    The IMDb Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_imdb_group_low_tvshows['IMDb Group'][0]}' :
'{prime_video_imdb_group_low_tvshows['Prime Video'].min()}'\n

    The IMDb Group with Highest TV Shows Count on 'Disney+' is
'{disney_imdb_group_high_tvshows['IMDb Group'][0]}' :
'{disney_imdb_group_high_tvshows['Disney+'].max()}'\n
    The IMDb Group with Lowest TV Shows Count on 'Disney+' is
'{disney_imdb_group_low_tvshows['IMDb Group'][0]}' :
'{disney_imdb_group_low_tvshows['Disney+'].min()}'\n
''')


# In[59]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

```

```

n_i_ax1 = sns.barplot(x = netflix_imdb_group_tvshows['IMDb Group'][:10], y =
netflix_imdb_group_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_i_ax2 = sns.barplot(x = hulu_imdb_group_tvshows['IMDb Group'][:10], y =
hulu_imdb_group_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_i_ax3 = sns.barplot(x = prime_video_imdb_group_tvshows['IMDb Group'][:10], y =
prime_video_imdb_group_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_i_ax4 = sns.barplot(x = disney_imdb_group_tvshows['IMDb Group'][:10], y =
disney_imdb_group_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()

```

In[60]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = imdb_group_data_tvshows['IMDb Group'], y =
imdb_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = imdb_group_data_tvshows['IMDb Group'], y =
imdb_group_data_tvshows['Hulu'], color = 'lightgreen')
sns.lineplot(x = imdb_group_data_tvshows['IMDb Group'], y = imdb_group_data_tvshows['Prime
Video'], color = 'lightblue')
sns.lineplot(x = imdb_group_data_tvshows['IMDb Group'], y =
imdb_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('IMDb Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

In[61]:

```

print(f'''
    Accross All Platforms Total Count of IMDb Group is '{imdb_group_data_tvshows['IMDb
Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Netflix' is '{netflix_imdb_group_tvshows['IMDb
Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Hulu' is '{hulu_imdb_group_tvshows['IMDb
Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Prime Video' is '{prime_video_imdb_group_tvshows['IMDb
Group'].unique().shape[0]}'\n
    Total Count of IMDb Group on 'Disney+' is '{disney_imdb_group_tvshows['IMDb
Group'].unique().shape[0]}'
    ''')

```

In[62]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_i_ax1 = sns.lineplot(y = imdb_group_data_tvshows['IMDb Group'], x =
imdb_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])

```

```

h_i_ax2 = sns.lineplot(y = imdb_group_data_tvshows['IMDb Group'], x =
imdb_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_i_ax3 = sns.lineplot(y = imdb_group_data_tvshows['IMDb Group'], x =
imdb_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_i_ax4 = sns.lineplot(y = imdb_group_data_tvshows['IMDb Group'], x =
imdb_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()

# In[63]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_i_ax1 = sns.barplot(x = imdb_group_data_tvshows['IMDb Group'][:10], y =
imdb_group_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_i_ax2 = sns.barplot(x = imdb_group_data_tvshows['IMDb Group'][:10], y =
imdb_group_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_i_ax3 = sns.barplot(x = imdb_group_data_tvshows['IMDb Group'][:10], y =
imdb_group_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_i_ax4 = sns.barplot(x = imdb_group_data_tvshows['IMDb Group'][:10], y =
imdb_group_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_i_ax1.title.set_text(labels[0])
h_i_ax2.title.set_text(labels[1])
p_i_ax3.title.set_text(labels[2])
d_i_ax4.title.set_text(labels[3])

plt.show()

```

otttvshows_language.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

```

```
# In[2]:  
  
# Import Dataset  
# Import File from Loacal Drive  
# from google.colab import files  
# data_to_load = files.upload()  
# from google.colab import drive  
# drive.mount('/content/drive')
```

```
# In[3]:
```

```
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import warnings  
import collections  
import plotly.express as px  
import plotly.graph_objects as go  
import nltk  
import re  
from nltk.corpus import stopwords  
from nltk.tokenize import word_tokenize  
from nltk.probability import FreqDist  
from nltk.util import ngrams  
from plotly.subplots import make_subplots  
from plotly.offline import iplot, init_notebook_mode  
from wordcloud import WordCloud, STOPWORDS  
from pandas_profiling import ProfileReport  
get_ipython().run_line_magic('matplotlib', 'inline')  
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'  
  
path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'  
  
df_tvshows = pd.read_csv(path + 'otttvshows.csv')  
  
df_tvshows.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_tvshows)  
# profile
```

```
# In[7]:
```

```
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()
```

```
# In[8]:
```

```
data_investigate(df_tvshows)
```

```
# In[9]:
```

```
# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
```

```

# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_tvshows)

# In[11]:

```

```

df_tvshows.head()

# In[12]:

df_tvshows.describe()

# In[13]:

df_tvshows.corr()

# In[14]:

# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:

# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:

# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]:

df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]


# In[18]:

```

```

df_tvshows_languages = df_tvshows.copy()

# In[19]:


df_tvshows_languages.drop(df_tvshows_languages.loc[df_tvshows_languages['Language'] == "NA"].index, inplace = True)
# df_tvshows_languages = df_tvshows_languages[df_tvshows_languages.Language != "NA"]
# df_tvshows_languages['Language'] = df_tvshows_languages['Language'].astype(str)

# In[20]:


df_tvshows_count_languages = df_tvshows_languages.copy()

# In[21]:


df_tvshows_language = df_tvshows_languages.copy()

# In[22]:


# Create languages dict where key=name and value = number of languages
languages = {}

for i in df_tvshows_count_languages['Language'].dropna():
    if i != "NA":
        #print(i,len(i.split(',')))
        languages[i] = len(i.split(','))
    else:
        languages[i] = 0

# Add this information to our dataframe as a new column

df_tvshows_count_languages['Number of Languages'] =
df_tvshows_count_languages['Language'].map(languages).astype(int)

# In[23]:


df_tvshows_mixed_languages = df_tvshows_count_languages.copy()

# In[24]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_languages_tvshows =
df_tvshows_count_languages.loc[df_tvshows_count_languages['Netflix'] == 1]
hulu_languages_tvshows = df_tvshows_count_languages.loc[df_tvshows_count_languages['Hulu'] == 1]

```

```
prime_video_languages_tvshows =
df_tvshows_count_languages.loc[df_tvshows_count_languages['Prime Video'] == 1]
disney_languages_tvshows =
df_tvshows_count_languages.loc[df_tvshows_count_languages['Disney+'] == 1]
```

```
# In[25]:
```

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_count_languages.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, atleast annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[26]:
```

```
df_languages_most_tvshows = df_tvshows_count_languages.sort_values(by = 'Number of Languages', ascending = False).reset_index()
df_languages_most_tvshows = df_languages_most_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_count_languages['Number of Languages'] ==
# df_tvshows_count_languages['Number of Languages'].max())
# df_languages_most_tvshows = df_tvshows_count_languages[filter]

# mostest_rated_tvshows = df_tvshows_count_languages.loc[df_tvshows_count_languages['Number of Languages'].idxmax()]

print('\nTV Shows with Highest Ever Number of Languages are : \n')
df_languages_most_tvshows.head(5)
```

```
# In[27]:
```

```
fig = px.bar(y = df_languages_most_tvshows['Title'][:15],
              x = df_languages_most_tvshows['Number of Languages'][:15],
              color = df_languages_most_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Highest Number of Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[28]:
```

```
df_languages_least_tvshows = df_tvshows_count_languages.sort_values(by = 'Number of Languages', ascending = True).reset_index()
df_languages_least_tvshows = df_languages_least_tvshows.drop(['index'], axis = 1)
```

```

# filter = (df_tvshows_count_languages['Number of Languages'] ==
(df_tvshows_count_languages['Number of Languages'].min()))
# df_languages_least_tvshows = df_tvshows_count_languages[filter]

print('\nTV Shows with Lowest Ever Number of Languages are : \n')
df_languages_least_tvshows.head(5)

# In[29]:


fig = px.bar(y = df_languages_least_tvshows['Title'][:15],
              x = df_languages_least_tvshows['Number of Languages'][:15],
              color = df_languages_least_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Lowest Number of Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:


print(f'''
    Total '{df_tvshows_count_languages['Number of Languages'].unique().shape[0]}' unique
Number of Languages s were Given, They were Like this,\n

    {df_tvshows_count_languages.sort_values(by = 'Number of Languages', ascending =
False)['Number of Languages'].unique()}\n

    The Highest Number of Languages Ever Any TV Show Got is
'{df_languages_most_tvshows['Title'][0]}' : '{df_languages_most_tvshows['Number of
Languages'].max()}'\n

    The Lowest Number of Languages Ever Any TV Show Got is
'{df_languages_least_tvshows['Title'][0]}' : '{df_languages_least_tvshows['Number of
Languages'].min()}'\n
''')


# In[31]:


netflix_languages_most_tvshows =
df_languages_most_tvshows.loc[df_languages_most_tvshows['Netflix']==1].reset_index()
netflix_languages_most_tvshows = netflix_languages_most_tvshows.drop(['index'], axis = 1)

netflix_languages_least_tvshows =
df_languages_least_tvshows.loc[df_languages_least_tvshows['Netflix']==1].reset_index()
netflix_languages_least_tvshows = netflix_languages_least_tvshows.drop(['index'], axis = 1)

netflix_languages_most_tvshows.head(5)

# In[32]:


fig = px.bar(y = netflix_languages_most_tvshows['Title'][:15],
              x = netflix_languages_most_tvshows['Number of Languages'][:15],

```

```

color = netflix_languages_most_tvshows['Number of Languages'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
title  = 'TV Shows with Highest Number of Languages : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[33]:

```

fig = px.bar(y = netflix_languages_least_tvshows['Title'][:15],
              x = netflix_languages_least_tvshows['Number of Languages'][:15],
              color = netflix_languages_least_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title  = 'TV Shows with Lowest Number of Languages : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[34]:

```

hulu_languages_most_tvshows =
df_languages_most_tvshows.loc[df_languages_most_tvshows['Hulu']==1].reset_index()
hulu_languages_most_tvshows = hulu_languages_most_tvshows.drop(['index'], axis = 1)

hulu_languages_least_tvshows =
df_languages_least_tvshows.loc[df_languages_least_tvshows['Hulu']==1].reset_index()
hulu_languages_least_tvshows = hulu_languages_least_tvshows.drop(['index'], axis = 1)

hulu_languages_most_tvshows.head(5)

```

In[35]:

```

fig = px.bar(y = hulu_languages_most_tvshows['Title'][:15],
              x = hulu_languages_most_tvshows['Number of Languages'][:15],
              color = hulu_languages_most_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title  = 'TV Shows with Highest Number of Languages : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[36]:

```

fig = px.bar(y = hulu_languages_least_tvshows['Title'][:15],
              x = hulu_languages_least_tvshows['Number of Languages'][:15],
              color = hulu_languages_least_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title  = 'TV Shows with Lowest Number of Languages : Hulu')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


prime_video_languages_most_tvshows =
df_languages_most_tvshows.loc[df_languages_most_tvshows['Prime Video']==1].reset_index()
prime_video_languages_most_tvshows = prime_video_languages_most_tvshows.drop(['index'],
axis = 1)

prime_video_languages_least_tvshows =
df_languages_least_tvshows.loc[df_languages_least_tvshows['Prime Video']==1].reset_index()
prime_video_languages_least_tvshows = prime_video_languages_least_tvshows.drop(['index'],
axis = 1)

prime_video_languages_most_tvshows.head(5)

```

```

# In[38]:


fig = px.bar(y = prime_video_languages_most_tvshows['Title'][:15],
              x = prime_video_languages_most_tvshows['Number of Languages'][:15],
              color = prime_video_languages_most_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Highest Number of Languages : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[39]:


fig = px.bar(y = prime_video_languages_least_tvshows['Title'][:15],
              x = prime_video_languages_least_tvshows['Number of Languages'][:15],
              color = prime_video_languages_least_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Lowest Number of Languages : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[40]:


disney_languages_most_tvshows =
df_languages_most_tvshows.loc[df_languages_most_tvshows['Disney+']==1].reset_index()
disney_languages_most_tvshows = disney_languages_most_tvshows.drop(['index'], axis = 1)

disney_languages_least_tvshows =
df_languages_least_tvshows.loc[df_languages_least_tvshows['Disney+']==1].reset_index()
disney_languages_least_tvshows = disney_languages_least_tvshows.drop(['index'], axis = 1)

disney_languages_most_tvshows.head(5)

```

```

# In[41]:


fig = px.bar(y = disney_languages_most_tvshows['Title'][:15],
              x = disney_languages_most_tvshows['Number of Languages'][:15],
              color = disney_languages_most_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Highest Number of Languages : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[42]:


fig = px.bar(y = disney_languages_least_tvshows['Title'][:15],
              x = disney_languages_least_tvshows['Number of Languages'][:15],
              color = disney_languages_least_tvshows['Number of Languages'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Languages'},
              title = 'TV Shows with Lowest Number of Languages : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[43]:


print(f'''The TV Show with Highest Number of Languages Ever Got is
'{df_languages_most_tvshows['Title'][0]}' : '{df_languages_most_tvshows['Number of Languages'].max()}'\n
The TV Show with Lowest Number of Languages Ever Got is
'{df_languages_least_tvshows['Title'][0]}' : '{df_languages_least_tvshows['Number of Languages'].min()}'\n

The TV Show with Highest Number of Languages on 'Netflix' is
'{netflix_languages_most_tvshows['Title'][0]}' : '{netflix_languages_most_tvshows['Number of Languages'].max()}'\n
The TV Show with Lowest Number of Languages on 'Netflix' is
'{netflix_languages_least_tvshows['Title'][0]}' : '{netflix_languages_least_tvshows['Number of Languages'].min()}'\n

The TV Show with Highest Number of Languages on 'Hulu' is
'{hulu_languages_most_tvshows['Title'][0]}' : '{hulu_languages_most_tvshows['Number of Languages'].max()}'\n
The TV Show with Lowest Number of Languages on 'Hulu' is
'{hulu_languages_least_tvshows['Title'][0]}' : '{hulu_languages_least_tvshows['Number of Languages'].min()}'\n

The TV Show with Highest Number of Languages on 'Prime Video' is
'{prime_video_languages_most_tvshows['Title'][0]}' :
'{prime_video_languages_most_tvshows['Number of Languages'].max()}'\n
The TV Show with Lowest Number of Languages on 'Prime Video' is
'{prime_video_languages_least_tvshows['Title'][0]}' :
'{prime_video_languages_least_tvshows['Number of Languages'].min()}'\n

```

```
The TV Show with Highest Number of Languages on 'Disney+' is
'{disney_languages_most_tvshows['Title'][0]}' : '{disney_languages_most_tvshows['Number of
Languages'].max()}'\n
The TV Show with Lowest Number of Languages on 'Disney+' is
'{disney_languages_least_tvshows['Title'][0]}' : '{disney_languages_least_tvshows['Number
of Languages'].min()}'\n
'''
```

```
# In[44]:
```

```
print(f'''
    Accross All Platforms the Average Number of Languages is
'{round(df_tvshows_count_languages['Number of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Netflix' is
'{round(netflix_languages_tvshows['Number of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Hulu' is '{round(hulu_languages_tvshows['Number
of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Prime Video' is
'{round(prime_video_languages_tvshows['Number of Languages'].mean(), ndigits = 2)}'\n
    The Average Number of Languages on 'Disney+' is
'{round(disney_languages_tvshows['Number of Languages'].mean(), ndigits = 2)}'
''')
```

```
# In[45]:
```

```
print(f'''
    Accross All Platforms Total Count of Language is '{df_tvshows_count_languages['Number
of Languages'].max()}'\n
    Total Count of Language on 'Netflix' is '{netflix_languages_tvshows['Number of
Languages'].max()}'\n
    Total Count of Language on 'Hulu' is '{hulu_languages_tvshows['Number of
Languages'].max()}'\n
    Total Count of Language on 'Prime Video' is '{prime_video_languages_tvshows['Number
of Languages'].max()}'\n
    Total Count of Language on 'Disney+' is '{disney_languages_tvshows['Number of
Languages'].max()}'\n
''')
```

```
# In[46]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_count_languages['Number of Languages'], bins = 20, kde = True, ax =
ax[0])
sns.boxplot(df_tvshows_count_languages['Number of Languages'], ax = ax[1])
plt.show()
```

```
# In[47]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Number of Languages s Per Platform')

# Plotting the information from each dataset into a histogram
```

```

sns.histplot(prime_video_languages_tvshows['Number of Languages'], color = 'lightblue',
legend = True, kde = True)
sns.histplot(netflix_languages_tvshows['Number of Languages'], color = 'red', legend =
True, kde = True)
sns.histplot(hulu_languages_tvshows['Number of Languages'], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_languages_tvshows['Number of Languages'], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

# In[48]:


df_lan = df_tvshows_language['Language'].str.split(',').apply(pd.Series).stack()
del df_tvshows_language['Language']
df_lan.index = df_lan.index.droplevel(-1)
df_lan.name = 'Language'
df_tvshows_language = df_tvshows_language.join(df_lan)
df_tvshows_language.drop_duplicates(inplace = True)

# In[49]:


df_tvshows_language.head(5)

# In[50]:


language_count = df_tvshows_language.groupby('Language')['Title'].count()
language_tvshows = df_tvshows_language.groupby('Language')[['Netflix', 'Hulu', 'Prime
Video', 'Disney+']].sum()
language_data_tvshows = pd.concat([language_count, language_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
language_data_tvshows = language_data_tvshows.sort_values(by = 'TV Shows Count', ascending
= False)

# In[51]:


# Language with TV Shows Counts - All Platforms Combined
language_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[52]:


fig = px.bar(x = language_data_tvshows['Language'][:50],
              y = language_data_tvshows['TV Shows Count'][:50],
              color = language_data_tvshows['TV Shows Count'][:50],
              color_continuous_scale = 'Teal_r',
              labels = { 'x' : 'Language', 'y' : 'TV Shows Count'},
              title = 'Major Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')

```

```

fig.show()

# In[53]:


df_language_high_tvshows = language_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_language_high_tvshows = df_language_high_tvshows.drop(['index'], axis = 1)
# filter = (language_data_tvshows['TV Shows Count'] == (language_data_tvshows['TV Shows
Count'].max()))
# df_language_high_tvshows = language_data_tvshows[filter]

# highestRated_tvshows = language_data_tvshows.loc[language_data_tvshows['TV Shows
Count'].idxmax()]

print('\nLanguage with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_language_high_tvshows.head(5)

# In[54]:


fig = px.bar(y = df_language_high_tvshows['Language'][:15],
              x = df_language_high_tvshows['TV Shows Count'][:15],
              color = df_language_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
              title = 'Language with Highest TV Shows : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[55]:


df_language_low_tvshows = language_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_language_low_tvshows = df_language_low_tvshows.drop(['index'], axis = 1)
# filter = (language_data_tvshows['TV Shows Count'] == (language_data_tvshows['TV Shows
Count'].min()))
# df_language_low_tvshows = language_data_tvshows[filter]

print('\nLanguage with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_language_low_tvshows.head(5)

# In[56]:


fig = px.bar(y = df_language_low_tvshows['Language'][:15],
              x = df_language_low_tvshows['TV Shows Count'][:15],
              color = df_language_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
              title = 'Language with Lowest TV Shows Count : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

```

# In[57]:


print(f'''
    Total '{language_data_tvshows['Language'].unique().shape[0]}' unique Language Count s
were Given, They were Like this,\n

    {language_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Language'].unique()[:5]}\n

    The Highest Ever TV Shows Count Ever Any TV Show Got is
'{df_language_high_tvshows['Language'][0]}' : '{df_language_high_tvshows['TV Shows
Count']].max()}'\n

    The Lowest Ever TV Shows Count Ever Any TV Show Got is
'{df_language_low_tvshows['Language'][0]}' : '{df_language_low_tvshows['TV Shows
Count']].min()}'\n
    ''')

```

```

# In[58]:


fig = px.pie(language_data_tvshows[:10], names = 'Language', values = 'TV Shows Count',
color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Language')
fig.show()

```

```

# In[59]:


# netflix_language_tvshows = language_data_tvshows[language_data_tvshows['Netflix'] !=
0].sort_values(by = 'Netflix', ascending = False).reset_index()
# netflix_language_tvshows = netflix_language_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

netflix_language_high_tvshows = df_language_high_tvshows.sort_values(by = 'Netflix',
ascending = False).reset_index()
netflix_language_high_tvshows = netflix_language_high_tvshows.drop(['index'], axis = 1)

netflix_language_low_tvshows = df_language_high_tvshows.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_language_low_tvshows = netflix_language_low_tvshows.drop(['index'], axis = 1)

netflix_language_high_tvshows.head(5)

```

```

# In[60]:


fig = px.bar(x = netflix_language_high_tvshows['Language'][:15],
              y = netflix_language_high_tvshows['Netflix'][:15],
              color = netflix_language_high_tvshows['Netflix'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
              title = 'Language with Highest TV Shows : Netflix')

fig.update_layout(plot_bgcolor = 'white')

```

```

fig.show()

# In[61]:


# hulu_language_tvshows = language_data_tvshows[language_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
# hulu_language_tvshows = hulu_language_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_language_high_tvshows = df_language_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_language_high_tvshows = hulu_language_high_tvshows.drop(['index'], axis = 1)

hulu_language_low_tvshows = df_language_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_language_low_tvshows = hulu_language_low_tvshows.drop(['index'], axis = 1)

hulu_language_high_tvshows.head(5)

# In[62]:


fig = px.bar(x = hulu_language_high_tvshows['Language'][:15],
              y = hulu_language_high_tvshows['Hulu'][:15],
              color = hulu_language_high_tvshows['Hulu'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
              title = 'Language with Highest TV Shows : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[63]:


# prime_video_language_tvshows = language_data_tvshows[language_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_language_tvshows = prime_video_language_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_language_high_tvshows = df_language_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_language_high_tvshows = prime_video_language_high_tvshows.drop(['index'], axis = 1)

prime_video_language_low_tvshows = df_language_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_language_low_tvshows = prime_video_language_low_tvshows.drop(['index'], axis = 1)

prime_video_language_high_tvshows.head(5)

# In[64]:


fig = px.bar(x = prime_video_language_high_tvshows['Language'][:15],

```

```

y = prime_video_language_high_tvshows['Prime Video'][:15],
color = prime_video_language_high_tvshows['Prime Video'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
title  = 'Language with Highest TV Shows : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[65]:


# disney_language_tvshows = language_data_tvshows[language_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_language_tvshows = disney_language_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_language_high_tvshows = df_language_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_language_high_tvshows = disney_language_high_tvshows.drop(['index'], axis = 1)

disney_language_low_tvshows = df_language_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_language_low_tvshows = disney_language_low_tvshows.drop(['index'], axis = 1)

disney_language_high_tvshows.head(5)

# In[66]:


fig = px.bar(x = disney_language_high_tvshows['Language'][:15],
              y = disney_language_high_tvshows['Disney+'][:15],
              color = disney_language_high_tvshows['Disney+'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'Language', 'x' : 'TV Shows Count'},
              title  = 'Language with Highest TV Shows : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[67]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(language_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(language_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()

# In[68]:


# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_language_tvshows = language_data_tvshows[language_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_language_tvshows = netflix_language_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

```

```

hulu_language_tvshows = language_data_tvshows[language_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_language_tvshows = hulu_language_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_language_tvshows = language_data_tvshows[language_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_language_tvshows = prime_video_language_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_language_tvshows = language_data_tvshows[language_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_language_tvshows = disney_language_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

# In[69]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Language TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(disney_language_tvshows['Disney+'][:50], color = 'darkblue', legend = True, kde = True)
sns.histplot(prime_video_language_tvshows['Prime Video'][:50], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_language_tvshows['Netflix'][:50], color = 'red', legend = True, kde = True)
sns.histplot(hulu_language_tvshows['Hulu'][:50], color = 'lightgreen', legend = True, kde = True)

# Setting the legend
plt.legend(['Disney+', 'Prime Video', 'Netflix', 'Hulu'])
plt.show()

# In[70]:


print(f'''
    The Language with Highest TV Shows Count Ever Got is
'{df_language_high_tvshows['Language'][0]}' : '{df_language_high_tvshows['TV Shows Count'].max()}'\n
    The Language with Lowest TV Shows Count Ever Got is
'{df_language_low_tvshows['Language'][0]}' : '{df_language_low_tvshows['TV Shows Count'].min()}'\n\n
    The Language with Highest TV Shows Count on 'Netflix' is
'{netflix_language_high_tvshows['Language'][0]}' :
'{netflix_language_high_tvshows['Netflix'].max()}'\n
    The Language with Lowest TV Shows Count on 'Netflix' is
'{netflix_language_low_tvshows['Language'][0]}' :
'{netflix_language_low_tvshows['Netflix'].min()}'\n\n
    The Language with Highest TV Shows Count on 'Hulu' is
'{hulu_language_high_tvshows['Language'][0]}' :
'{hulu_language_high_tvshows['Hulu'].max()}'\n

```

```

    The Language with Lowest TV Shows Count on 'Hulu' is
'{hulu_language_low_tvshows['Language'][0]} :
'{hulu_language_low_tvshows['Hulu'].min()}'\n

    The Language with Highest TV Shows Count on 'Prime Video' is
'{prime_video_language_high_tvshows['Language'][0]} :
'{prime_video_language_high_tvshows['Prime Video'].max()}'\n
    The Language with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_language_low_tvshows['Language'][0]} :
'{prime_video_language_low_tvshows['Prime Video'].min()}'\n

    The Language with Highest TV Shows Count on 'Disney+' is
'{disney_language_high_tvshows['Language'][0]} :
'{disney_language_high_tvshows['Disney+'].max()}'\n
    The Language with Lowest TV Shows Count on 'Disney+' is
'{disney_language_low_tvshows['Language'][0]} :
'{disney_language_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[71]:

```

# Distribution of tvshows language in each platform
plt.figure(figsize = (20, 5))
plt.title('Language with TV Shows Count for All Platforms')
sns.violinplot(x = language_data_tvshows['TV Shows Count'][:100], color = 'gold', legend = True, kde = True, shade = False)
plt.show()

```

In[72]:

```

# Distribution of Language TV Shows Count in each platform
f1, ax1 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = netflix_language_tvshows['Netflix'][:100], color = 'red', ax = ax1[0])
sns.violinplot(x = hulu_language_tvshows['Hulu'][:100], color = 'lightgreen', ax = ax1[1])

f2, ax2 = plt.subplots(1, 2 , figsize = (20, 5))
sns.violinplot(x = prime_video_language_tvshows['Prime Video'][:100], color = 'lightblue', ax = ax2[0])
sns.violinplot(x = disney_language_tvshows['Disney+'][:100], color = 'darkblue', ax = ax2[1])
plt.show()

```

In[73]:

```

print(f'''
    Accross All Platforms the Average TV Shows Count of Language is
'{round(language_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Language on 'Netflix' is
'{round(netflix_language_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Language on 'Hulu' is
'{round(hulu_language_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Language on 'Prime Video' is
'{round(prime_video_language_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Language on 'Disney+' is
'{round(disney_language_tvshows['Disney+'].mean(), ndigits = 2)}'\n

```

```
'''
```

```
# In[74]:
```

```
print(f'''  
    Across All Platforms Total Count of Language is  
'{language_data_tvshows['Language'].unique().shape[0]}'\n        Total Count of Language on 'Netflix' is  
'{netflix_language_tvshows['Language'].unique().shape[0]}'\n        Total Count of Language on 'Hulu' is  
'{hulu_language_tvshows['Language'].unique().shape[0]}'\n        Total Count of Language on 'Prime Video' is  
'{prime_video_language_tvshows['Language'].unique().shape[0]}'\n        Total Count of Language on 'Disney+' is  
'{disney_language_tvshows['Language'].unique().shape[0]}'\n'''')
```

```
# In[75]:
```

```
plt.figure(figsize = (20, 5))  
sns.lineplot(x = language_data_tvshows['Language'][:10], y =  
language_data_tvshows['Netflix'][:10], color = 'red')  
sns.lineplot(x = language_data_tvshows['Language'][:10], y =  
language_data_tvshows['Hulu'][:10], color = 'lightgreen')  
sns.lineplot(x = language_data_tvshows['Language'][:10], y =  
language_data_tvshows['Prime Video'][:10], color = 'lightblue')  
sns.lineplot(x = language_data_tvshows['Language'][:10], y =  
language_data_tvshows['Disney+'][:10], color = 'darkblue')  
plt.xlabel('Language', fontsize = 20)  
plt.ylabel('TV Shows Count', fontsize = 20)  
plt.show()
```

```
# In[76]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))  
  
n_l_ax1 = sns.lineplot(y = language_data_tvshows['Language'][:10], x =  
language_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])  
h_l_ax2 = sns.lineplot(y = language_data_tvshows['Language'][:10], x =  
language_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])  
p_l_ax3 = sns.lineplot(y = language_data_tvshows['Language'][:10], x =  
language_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])  
d_l_ax4 = sns.lineplot(y = language_data_tvshows['Language'][:10], x =  
language_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_l_ax1.title.set_text(labels[0])  
h_l_ax2.title.set_text(labels[1])  
p_l_ax3.title.set_text(labels[2])  
d_l_ax4.title.set_text(labels[3])  
  
plt.show()
```

```
# In[77]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = netflix_language_tvshows['Language'][:10], x =
netflix_language_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = hulu_language_tvshows['Language'][:10], x =
hulu_language_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = prime_video_language_tvshows['Language'][:10], x =
prime_video_language_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = disney_language_tvshows['Language'][:10], x =
disney_language_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[78]:
```

```
# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Language TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_language_tvshows['Netflix'][:10], color = 'red', legend = True)
sns.kdeplot(hulu_language_tvshows['Hulu'][:10], color = 'green', legend = True)
sns.kdeplot(prime_video_language_tvshows['Prime Video'][:10], color = 'lightblue', legend =
True)
sns.kdeplot(disney_language_tvshows['Disney+'][:10], color = 'darkblue', legend = True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()
```

```
# In[79]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = language_data_tvshows['Language'][:10], x =
language_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = language_data_tvshows['Language'][:10], x =
language_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = language_data_tvshows['Language'][:10], x =
language_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = language_data_tvshows['Language'][:10], x =
language_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
```

```

p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()

# In[80]:


df_tvshows_mixed_languages.drop(df_tvshows_mixed_languages.loc[df_tvshows_mixed_languages['Language'] == "NA"].index, inplace = True)
# df_tvshows_mixed_languages =
df_tvshows_mixed_languages[df_tvshows_mixed_languages.Language != "NA"]
df_tvshows_mixed_languages.drop(df_tvshows_mixed_languages.loc[df_tvshows_mixed_languages['Number of Languages'] == 1].index, inplace = True)

# In[81]:


df_tvshows_mixed_languages.head(5)

# In[82]:


mixed_languages_count = df_tvshows_mixed_languages.groupby('Language')['Title'].count()
mixed_languages_tvshows = df_tvshows_mixed_languages.groupby('Language')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
mixed_languages_data_tvshows = pd.concat([mixed_languages_count, mixed_languages_tvshows], axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count', 'Language' : 'Mixed Language'})
mixed_languages_data_tvshows = mixed_languages_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[83]:


mixed_languages_data_tvshows.head(5)

# In[84]:


# Mixed Language with TV Shows Counts - All Platforms Combined
mixed_languages_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)[:10]

# In[85]:


df_mixed_languages_high_tvshows = mixed_languages_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_mixed_languages_high_tvshows = df_mixed_languages_high_tvshows.drop(['index'], axis = 1)
# filter = (mixed_languages_data_tvshows['TV Shows Count'] == 
(mixed_languages_data_tvshows['TV Shows Count'].max()))
# df_mixed_languages_high_tvshows = mixed_languages_data_tvshows[filter]

# highestRated_tvshows = mixed_languages_data_tvshows.loc[mixed_languages_data_tvshows['TV Shows Count'].idxmax()]

```

```
print('\nMixed Language with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_languages_high_tvshows.head(5)
```

```
# In[86]:
```

```
fig = px.bar(y = df_mixed_languages_high_tvshows['Mixed Language'][:15],
              x = df_mixed_languages_high_tvshows['TV Shows Count'][:15],
              color = df_mixed_languages_high_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Language'},
              title = 'TV Shows with Highest Number of Mixed Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[87]:
```

```
df_mixed_languages_low_tvshows = mixed_languages_data_tvshows.sort_values(by = 'TV Shows Count', ascending = True).reset_index()
df_mixed_languages_low_tvshows = df_mixed_languages_low_tvshows.drop(['index'], axis = 1)
# filter = (mixed_languages_data_tvshows['TV Shows Count'] == 
(mixed_languages_data_tvshows['TV Shows Count'].min()))
# df_mixed_languages_low_tvshows = mixed_languages_data_tvshows[filter]

print('\nMixed Language with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_mixed_languages_low_tvshows.head(5)
```

```
# In[88]:
```

```
fig = px.bar(y = df_mixed_languages_low_tvshows['Mixed Language'][:15],
              x = df_mixed_languages_low_tvshows['TV Shows Count'][:15],
              color = df_mixed_languages_low_tvshows['TV Shows Count'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Number of Mixed Language'},
              title = 'TV Shows with Lowest Number of Mixed Languages : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[89]:
```

```
print(f'''
    Total '{df_tvshows_languages['Language'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see TV Shows from Total '{mixed_languages_data_tvshows['Mixed
Language'].unique().shape[0]}' Mixed Language, They were Like this, \n

    {mixed_languages_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Mixed Language'].head(5).unique()} etc. \n
```

```

    The Mixed Language with Highest TV Shows Count have
'{mixed_languages_data_tvshows['TV Shows Count'].max()}' TV Shows Available is
'{df_mixed_languages_high_tvshows['Mixed Language'][0]}', &\n
    The Mixed Language with Lowest TV Shows Count have '{mixed_languages_data_tvshows['TV
Shows Count'].min()}' TV Shows Available is '{df_mixed_languages_low_tvshows['Mixed
Language'][0]}'\n
    ''')

```

In[90]:

```

fig = px.pie(mixed_languages_data_tvshows[:10], names = 'Mixed Language', values = 'TV
Shows Count', color_discrete_sequence = px.colors.sequential.Teal)
fig.update_traces(textposition = 'inside', textinfo = 'percent+label', title = 'TV Shows
Count based on Mixed Language')
fig.show()

```

In[91]:

```

# netflix_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Netflix'] != 0].sort_values(by =
= 'Netflix', ascending = False).reset_index()
# netflix_mixed_languages_tvshows = netflix_mixed_languages_tvshows.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_mixed_languages_high_tvshows = df_mixed_languages_high_tvshows.sort_values(by =
= 'Netflix', ascending = False).reset_index()
netflix_mixed_languages_high_tvshows = netflix_mixed_languages_high_tvshows.drop(['index'],
axis = 1)

netflix_mixed_languages_low_tvshows = df_mixed_languages_high_tvshows.sort_values(by =
= 'Netflix', ascending = True).reset_index()
netflix_mixed_languages_low_tvshows = netflix_mixed_languages_low_tvshows.drop(['index'],
axis = 1)

netflix_mixed_languages_high_tvshows.head(5)

```

In[92]:

```

# hulu_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Hulu'] != 0].sort_values(by =
= 'Hulu', ascending = False).reset_index()
# hulu_mixed_languages_tvshows = hulu_mixed_languages_tvshows.drop(['index', 'Netflix',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_languages_high_tvshows = df_mixed_languages_high_tvshows.sort_values(by =
= 'Hulu', ascending = False).reset_index()
hulu_mixed_languages_high_tvshows = hulu_mixed_languages_high_tvshows.drop(['index'], axis
= 1)

hulu_mixed_languages_low_tvshows = df_mixed_languages_high_tvshows.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_mixed_languages_low_tvshows = hulu_mixed_languages_low_tvshows.drop(['index'], axis =
1)

hulu_mixed_languages_high_tvshows.head(5)

```

```
# In[93]:
```

```
# prime_video_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
# prime_video_mixed_languages_tvshows = prime_video_mixed_languages_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_languages_high_tvshows = df_mixed_languages_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_languages_high_tvshows =
prime_video_mixed_languages_high_tvshows.drop(['index'], axis = 1)

prime_video_mixed_languages_low_tvshows = df_mixed_languages_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_mixed_languages_low_tvshows =
prime_video_mixed_languages_low_tvshows.drop(['index'], axis = 1)

prime_video_mixed_languages_high_tvshows.head(5)
```

```
# In[94]:
```

```
# disney_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
# disney_mixed_languages_tvshows = disney_mixed_languages_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_mixed_languages_high_tvshows = df_mixed_languages_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_languages_high_tvshows = disney_mixed_languages_high_tvshows.drop(['index'], axis = 1)

disney_mixed_languages_low_tvshows = df_mixed_languages_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_mixed_languages_low_tvshows = disney_mixed_languages_low_tvshows.drop(['index'], axis = 1)

disney_mixed_languages_high_tvshows.head(5)
```

```
# In[95]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(mixed_languages_data_tvshows['TV Shows Count'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(mixed_languages_data_tvshows['TV Shows Count'], ax = ax[1])
plt.show()
```

```
# In[96]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming platforms
```

```

netflix_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_mixed_languages_tvshows = netflix_mixed_languages_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_mixed_languages_tvshows = hulu_mixed_languages_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_mixed_languages_tvshows = prime_video_mixed_languages_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

disney_mixed_languages_tvshows =
mixed_languages_data_tvshows[mixed_languages_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_mixed_languages_tvshows = disney_mixed_languages_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

```

In[97]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Language TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram

sns.histplot(prime_video_mixed_languages_tvshows['Prime Video'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_mixed_languages_tvshows['Netflix'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_mixed_languages_tvshows['Hulu'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_mixed_languages_tvshows['Disney+'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[98]:

```

print(f'''
    The Mixed Language with Highest TV Shows Count Ever Got is
'{df_mixed_languages_high_tvshows['Mixed Language'][0]}' :
'{df_mixed_languages_high_tvshows['TV Shows Count'].max()}'\n
    The Mixed Language with Lowest TV Shows Count Ever Got is
'{df_mixed_languages_low_tvshows['Mixed Language'][0]}' :
'{df_mixed_languages_low_tvshows['TV Shows Count'].min()}'\n

```

```

    The Mixed Language with Highest TV Shows Count on 'Netflix' is
'{netflix_mixed_languages_high_tvshows['Mixed Language'][0]}' :
'{netflix_mixed_languages_high_tvshows['Netflix'].max()}'\n
    The Mixed Language with Lowest TV Shows Count on 'Netflix' is
'{netflix_mixed_languages_low_tvshows['Mixed Language'][0]}' :
'{netflix_mixed_languages_low_tvshows['Netflix'].min()}'\n

    The Mixed Language with Highest TV Shows Count on 'Hulu' is
'{hulu_mixed_languages_high_tvshows['Mixed Language'][0]}' :
'{hulu_mixed_languages_high_tvshows['Hulu'].max()}'\n
    The Mixed Language with Lowest TV Shows Count on 'Hulu' is
'{hulu_mixed_languages_low_tvshows['Mixed Language'][0]}' :
'{hulu_mixed_languages_low_tvshows['Hulu'].min()}'\n

    The Mixed Language with Highest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_languages_high_tvshows['Mixed Language'][0]}' :
'{prime_video_mixed_languages_high_tvshows['Prime Video'].max()}'\n
    The Mixed Language with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_mixed_languages_low_tvshows['Mixed Language'][0]}' :
'{prime_video_mixed_languages_low_tvshows['Prime Video'].min()}'\n

    The Mixed Language with Highest TV Shows Count on 'Disney+' is
'{disney_mixed_languages_high_tvshows['Mixed Language'][0]}' :
'{disney_mixed_languages_high_tvshows['Disney+'].max()}'\n
    The Mixed Language with Lowest TV Shows Count on 'Disney+' is
'{disney_mixed_languages_low_tvshows['Mixed Language'][0]}' :
'{disney_mixed_languages_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[99]:

```

print(f'''
    Across All Platforms the Average TV Shows Count of Mixed Language is
'{round(mixed_languages_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Language on 'Netflix' is
'{round(netflix_mixed_languages_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Language on 'Hulu' is
'{round(hulu_mixed_languages_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Language on 'Prime Video' is
'{round(prime_video_mixed_languages_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Mixed Language on 'Disney+' is
'{round(disney_mixed_languages_tvshows['Disney+'].mean(), ndigits = 2)}'\n
    ''')

```

In[100]:

```

print(f'''
    Across All Platforms Total Count of Mixed Language is
'{mixed_languages_data_tvshows['Mixed Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Netflix' is
'{netflix_mixed_languages_tvshows['Mixed Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Hulu' is '{hulu_mixed_languages_tvshows['Mixed
Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Prime Video' is
'{prime_video_mixed_languages_tvshows['Mixed Language'].unique().shape[0]}'\n
    Total Count of Mixed Language on 'Disney+' is '{disney_mixed_languages_tvshows['Mixed
Language'].unique().shape[0]}'

```

```
'))
```

```
# In[101]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = mixed_languages_data_tvshows['Mixed Language'][:5], y =
mixed_languages_data_tvshows['Netflix'][:5], color = 'red')
sns.lineplot(x = mixed_languages_data_tvshows['Mixed Language'][:5], y =
mixed_languages_data_tvshows['Hulu'][:5], color = 'lightgreen')
sns.lineplot(x = mixed_languages_data_tvshows['Mixed Language'][:5], y =
mixed_languages_data_tvshows['Prime Video'][:5], color = 'lightblue')
sns.lineplot(x = mixed_languages_data_tvshows['Mixed Language'][:5], y =
mixed_languages_data_tvshows['Disney+'][:5], color = 'darkblue')
plt.xlabel('Mixed Language', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()
```

```
# In[102]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_l_ax1 = sns.barplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_l_ax2 = sns.barplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_l_ax3 = sns.barplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_l_ax4 = sns.barplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_l_ax1.title.set_text(labels[0])
h_l_ax2.title.set_text(labels[1])
p_l_ax3.title.set_text(labels[2])
d_l_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[103]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 10))

n_ml_ax1 = sns.lineplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Netflix'][:10], color = 'red', ax = axes[0, 0])
h_ml_ax2 = sns.lineplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Hulu'][:10], color = 'lightgreen', ax = axes[0, 1])
p_ml_ax3 = sns.lineplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Prime Video'][:10], color = 'lightblue', ax = axes[1, 0])
d_ml_ax4 = sns.lineplot(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['Disney+'][:10], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ml_ax1.title.set_text(labels[0])
```

```

h_ml_ax2.title.set_text(labels[1])
p_ml_ax3.title.set_text(labels[2])
d_ml_ax4.title.set_text(labels[3])

plt.show()

# In[104]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Mixed Language TV Shows Count Per Platform')

# Plotting the information from each dataset into a histogram
sns.kdeplot(netflix_mixed_languages_tvshows['Netflix'][:50], color = 'red', legend = True)
sns.kdeplot(hulu_mixed_languages_tvshows['Hulu'][:50], color = 'green', legend = True)
sns.kdeplot(prime_video_mixed_languages_tvshows['Prime Video'][:50], color = 'lightblue',
legend = True)
sns.kdeplot(disney_mixed_languages_tvshows['Disney+'][:50], color = 'darkblue', legend =
True)

# Setting the legend
plt.legend(['Netflix', 'Hulu', 'Prime Video', 'Disney+'])
plt.show()


# In[105]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ml_ax1 = sns.barplot(y = netflix_mixed_languages_tvshows['Mixed Language'][:10], x =
netflix_mixed_languages_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ml_ax2 = sns.barplot(y = hulu_mixed_languages_tvshows['Mixed Language'][:10], x =
hulu_mixed_languages_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ml_ax3 = sns.barplot(y = prime_video_mixed_languages_tvshows['Mixed Language'][:10], x =
prime_video_mixed_languages_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_ml_ax4 = sns.barplot(y = disney_mixed_languages_tvshows['Mixed Language'][:10], x =
disney_mixed_languages_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ml_ax1.title.set_text(labels[0])
h_ml_ax2.title.set_text(labels[1])
p_ml_ax3.title.set_text(labels[2])
d_ml_ax4.title.set_text(labels[3])

plt.show()

# In[106]:


fig = go.Funnel(y = mixed_languages_data_tvshows['Mixed Language'][:10], x =
mixed_languages_data_tvshows['TV Shows Count'][:10]))
fig.show()

```

otttvshows ott.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# ! pip install wordcloud
# ! pip install Flask


# In[2]:


# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")


# In[4]:


nltk.download('all')
```

```

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()


# In[6]:


# profile = ProfileReport(df_tvshows)
# profile


# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[8]:


data_investigate(df_tvshows)


# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)

```

```

df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)

```

```

# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]: 

data_investigate(df_tvshows)

# In[11]: 

df_tvshows.head()

# In[12]: 

df_tvshows.describe()

# In[13]: 

df_tvshows.corr()

# In[14]: 

# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]: 

# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]: 

# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]

```

```

# df_prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df_disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]:


df_netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df_hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df_prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df_disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_ott = df_tvshows.copy()


# In[19]:


df_tvshows_ott.drop(df_tvshows_ott.loc[df_tvshows_ott['Title'] == "NA"].index, inplace = True)
# df_tvshows_ott = df_tvshows_ott[df_tvshows_ott.Title != "NA"]


# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_ott_tvshows = df_tvshows_ott.loc[df_tvshows_ott['Netflix'] == 1]
hulu_ott_tvshows = df_tvshows_ott.loc[df_tvshows_ott['Hulu'] == 1]
prime_video_ott_tvshows = df_tvshows_ott.loc[df_tvshows_ott['Prime Video'] == 1]
disney_ott_tvshows = df_tvshows_ott.loc[df_tvshows_ott['Disney+'] == 1]


# In[21]:


df_tvshows_ott_group = df_tvshows_ott.copy()


# In[22]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_ott.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)

```

```

# show plot
plt.show()
fig.show()

# In[23]:


print('\nTV Shows Available on All Platforms Are : \n')
df_tvshows_ott.head(5)

# In[24]:


print('\nTV Shows Available on Netflix Are : \n')
netflix_ott_tvshows.head(5)

# In[25]:


print('\nTV Shows Available on Hulu Are : \n')
hulu_ott_tvshows.head(5)

# In[26]:


print('\nTV Shows Available on Prime Video Are : \n')
prime_video_ott_tvshows.head(5)

# In[27]:


print('\nTV Shows Available on Disney+ Are : \n')
disney_ott_tvshows.head(5)

# In[28]:


print(f'''
    Total '{df_tvshows_ott['Title'].count()}' Titles are available on All Platforms, out
    of Which,\n
        Total '{netflix_ott_tvshows['Title'].count()}' Titles are available on 'Netflix'
        Total '{hulu_ott_tvshows['Title'].count()}' Titles are available on 'Hulu'
        Total '{prime_video_ott_tvshows['Title'].count()}' Titles are available on 'Prime
        video'
        Total '{disney_ott_tvshows['Title'].count()}' Titles are available on 'Disney+'
    ''')

# In[29]:


Platform = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

Count = [netflix_ott_tvshows['Title'].count(),
         hulu_ott_tvshows['Title'].count(),

```

```

prime_video_ott_tvshows['Title'].count(),
disney_ott_tvshows['Title'].count()

fig = px.pie(names = Platform,
              values = Count,
              title = 'TV Shows Count Of Different Platforms',
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textposition = 'inside',
                   textinfo = 'percent + label')
fig.show()

# In[30]:


df_tvshows_ott['OTT Count'] = df_tvshows_ott['Netflix'] + df_tvshows_ott['Hulu'] +
df_tvshows_ott['Prime Video'] + df_tvshows_ott['Disney+']

# In[31]:


(df_tvshows_ott['OTT Count'].value_counts()/df_tvshows_ott.shape[0])*100

# In[32]:


print(f'''
    Total '{df_tvshows_ott[df_tvshows_ott['OTT Count'] == 4].shape[0]}' Titles are
available on All Platforms
    Total '{df_tvshows_ott[df_tvshows_ott['OTT Count'] == 3].shape[0]}' Titles are
available on at least 3 Platforms
    Total '{df_tvshows_ott[df_tvshows_ott['OTT Count'] == 2].shape[0]}' Titles are
available on at least 2 Platforms
    Total '{df_tvshows_ott[df_tvshows_ott['OTT Count'] == 1].shape[0]}' Titles are
available on at least 1 Platforms
    ''')

# In[33]:


# TV Shows Available on All Platforms

# df_tvshows_ott[(df_tvshows_ott['Netflix'] == 1) & (df_tvshows_ott['Hulu'] == 1) &
# (df_tvshows_ott['Prime Video'] == 1) & (df_tvshows_ott['Disney+'] == 1)]
print('\nTotal ', df_tvshows_ott[df_tvshows_ott['OTT Count'] == 4].shape[0], ' Titles are
available on All Platforms\n')
tvshows_on_4_platforms = df_tvshows_ott[df_tvshows_ott['OTT Count'] == 4]
tvshows_on_4_platforms

# In[34]:


# TV Shows Available on at least 3 Platforms

print('\nTotal ', df_tvshows_ott[df_tvshows_ott['OTT Count'] == 3].shape[0], ' Titles are
available on at least 3 Platforms\n')

```

```

tvshows_on_3_platforms = df_tvshows_ott[df_tvshows_ott['OTT Count'] == 3]
tvshows_on_3_platforms

# In[35]:


# TV Shows Available on at least 2 Platforms
print('\nTotal ', df_tvshows_ott[df_tvshows_ott['OTT Count'] == 2].shape[0], ' Titles are
available on at least 2 Platforms\n')
tvshows_on_2_platforms = df_tvshows_ott[df_tvshows_ott['OTT Count'] == 2]
tvshows_on_2_platforms

# In[36]:


# TV Shows Available on at least 1 Platform
print('\nTotal ', df_tvshows_ott[df_tvshows_ott['OTT Count'] == 1].shape[0], ' Titles are
available on at least 1 Platforms\n')
tvshows_on_1_platforms = df_tvshows_ott[df_tvshows_ott['OTT Count'] == 1]
tvshows_on_1_platforms

# In[37]:


df_netflix_only_tvshows_ott = df_tvshows_ott[(df_tvshows_ott['Netflix'] == 1) &
(df_tvshows_ott['Hulu'] == 0) & (df_tvshows_ott['Prime Video'] == 0 ) &
(df_tvshows_ott['Disney+'] == 0)]
df_hulu_only_tvshows_ott = df_tvshows_ott[(df_tvshows_ott['Netflix'] == 0) &
(df_tvshows_ott['Hulu'] == 1) & (df_tvshows_ott['Prime Video'] == 0 ) &
(df_tvshows_ott['Disney+'] == 0)]
df_prime_video_only_tvshows_ott = df_tvshows_ott[(df_tvshows_ott['Netflix'] == 0) &
(df_tvshows_ott['Hulu'] == 0) & (df_tvshows_ott['Prime Video'] == 1 ) &
(df_tvshows_ott['Disney+'] == 0)]
df_disney_only_tvshows_ott = df_tvshows_ott[(df_tvshows_ott['Netflix'] == 0) &
(df_tvshows_ott['Hulu'] == 0) & (df_tvshows_ott['Prime Video'] == 0 ) &
(df_tvshows_ott['Disney+'] == 1)]

# In[38]:


# TV Shows Available only on Netflix

print('\nTotal ', df_netflix_only_tvshows_ott['Title'].shape[0], ' Titles are available
only on Netflix\n')

df_netflix_only_tvshows_ott

# In[39]:


# TV Shows Available only on Hulu

print('\nTotal ', df_hulu_only_tvshows_ott['Title'].shape[0], ' Titles are available only
on Hulu\n')

df_hulu_only_tvshows_ott

```

```

# In[40]:


# TV Shows Available only on Prime Video

print('\nTotal ', df_prime_video_only_tvshows_ott['Title'].shape[0], ' Titles are available
only on Prime Video\n')

df_prime_video_only_tvshows_ott


# In[41]:


# TV Shows Available only on Disney+

print('\nTotal ', df_disney_only_tvshows_ott['Title'].shape[0], ' Titles are available only
on Disney+\n')

df_disney_only_tvshows_ott


# In[42]:


ott_group_count = df_tvshows_ott.groupby('OTT Count')['Title'].count()
ott_group_tvshows = df_tvshows_ott.groupby('OTT Count')[['Netflix', 'Hulu', 'Prime Video',
'Disney+']].sum()
ott_group_data_tvshows = pd.concat([ott_group_count, ott_group_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
ott_group_data_tvshows = ott_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)

# In[43]:


ott_group_data_tvshows


# In[44]:


# Title Group with TV Shows Counts - All Platforms Combined
ott_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)


# In[45]:


ott_group_data_tvshows.sort_values(by = 'OTT Count', ascending = False)

# In[46]:


fig = px.bar(y = ott_group_data_tvshows['TV Shows Count'],
              x = ott_group_data_tvshows['OTT Count'],
              color = ott_group_data_tvshows['OTT Count'],

```

```

color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows Count', 'x' : 'OTT Count'},
title = 'TV Shows with Group Title : All Platforms')
fig.update_layout(plot_bgcolor = "white")
fig.show()

```

In[47]:

```

fig = px.pie(ott_group_data_tvshows,
             names = ott_group_data_tvshows['OTT Count'],
             values = ott_group_data_tvshows['TV Shows Count'],
             color = ott_group_data_tvshows['TV Shows Count'],
             color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                   title = 'TV Shows Count based on OTT Count')
fig.show()

```

otttvshows_plotline.ipynb

```
#!/usr/bin/env python
# coding: utf-8
```

In[1]:

```

# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

```

In[2]:

```

# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

```

In[3]:

```

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

```

```
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_tvshows)
# profile
```

```
# In[7]:
```

```
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
```

```

print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

# In[8]:

data_investigate(df_tvshows)

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

```

```

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:

```

```

data_investigate(df_tvshows)

```

```

# In[11]:

```

```

df_tvshows.head()

```

```

# In[12]:

```

```

df_tvshows.describe()

```

```

# In[13]:

```

```

df_tvshows.corr()

```

```

# In[14]:

```

```

# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_plotline = df_tvshows.copy()

# In[19]:


df_tvshows_plotline.drop(df_tvshows_plotline.loc[df_tvshows_plotline['Plotline'] == "NA"].index, inplace = True)
# df_tvshows_plotline = df_tvshows_plotline[df_tvshows_plotline.Plotline != "NA"]


# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming
# platforms
netflix_plotline_tvshows = df_tvshows_plotline.loc[df_tvshows_plotline['Netflix'] == 1]
hulu_plotline_tvshows = df_tvshows_plotline.loc[df_tvshows_plotline['Hulu'] == 1]

```

```
prime_video_plotline_tvshows = df_tvshows_plotline.loc[df_tvshows_plotline['Prime Video']
== 1]
disney_plotline_tvshows = df_tvshows_plotline.loc[df_tvshows_plotline['Disney+'] == 1]
```

In[21]:

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_plotline.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

In[22]:

```
df_tvshows_plotline = df_tvshows_plotline['Plotline']
tvshows_plotline_w = ' '.join(df_tvshows_plotline)
```

In[23]:

```
stopwords = set(STOPWORDS)

wordcloud_all_plotline_tvshows = WordCloud(width = 1000, height = 500,
                                             background_color ='white',
                                             stopwords = stopwords,
                                             min_font_size = 10).generate(tvshows_plotline_w)

# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_all_plotline_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

In[24]:

```
tvshows_plotline_w = tvshows_plotline_w.lower()

stop_words_english_tvshows = set(STOPWORDS)

word_tokens_english_tvshows = word_tokenize(tvshows_plotline_w)

filtered_tvshow_plotline = [w for w in word_tokens_english_tvshows if not w in
stop_words_english_tvshows]
```

```

filtered_tvshow_plotline = " ".join(filtered_tvshow_plotline)

filtered_tvshow_plotline = re.sub("'", '', filtered_tvshow_plotline)

filtered_tvshow_plotline = re.sub(r'[0-9]+', '', filtered_tvshow_plotline)

final_tvshow_plotline = re.sub(r'[^w\s]', '', filtered_tvshow_plotline)

plotline_tvshows_corpus_len = len(filtered_tvshow_plotline.split())
plotline_tvshows_corpus_len

# In[25]:


def extract_ngrams(data, num):
    n_grams = ngrams(nltk.word_tokenize(data), num)
    return [ ' '.join(grams) for grams in n_grams]

# In[26]:


plotline_ngram1_tvshows = FreqDist()

plotline_ngram1 = extract_ngrams(final_tvshow_plotline[:plotline_tvshows_corpus_len], 1)

for word in plotline_ngram1:
    plotline_ngram1_tvshows[word.lower()] += 1

# In[27]:


plotline_ngram1_tvshows.most_common(10)

# In[28]:


plotline_ngram2_tvshows = FreqDist()

plotline_ngram2 = extract_ngrams(final_tvshow_plotline[:plotline_tvshows_corpus_len], 2)

for word in plotline_ngram2:
    plotline_ngram2_tvshows[word.lower()] += 1

# In[29]:


plotline_ngram2_tvshows.most_common(10)

# In[30]:


plotline_ngram3_tvshows = FreqDist()

plotline_ngram3 = extract_ngrams(final_tvshow_plotline[:plotline_tvshows_corpus_len], 3)

```

```

for word in plotline_ngram3:
    plotline_ngram3_tvshows[word.lower()] += 1

# In[31]:


plotline_ngram3_tvshows.most_common(10)

# In[32]:


plotline_ngram4_tvshows = FreqDist()

plotline_ngram4 = extract_ngrams(final_tvshow_plotline[:plotline_tvshows_corpus_len], 4)

for word in plotline_ngram4:
    plotline_ngram4_tvshows[word.lower()] += 1

# In[33]:


plotline_ngram4_tvshows.most_common(10)

# In[34]:


plotline_ngram5_tvshows = FreqDist()

plotline_ngram5 = extract_ngrams(final_tvshow_plotline[:plotline_tvshows_corpus_len], 5)

for word in plotline_ngram5:
    plotline_ngram5_tvshows[word.lower()] += 1

# In[35]:


plotline_ngram5_tvshows.most_common(10)

# In[36]:


# Netflix Wordcloud
netflix_plotline_tvshows_t = netflix_plotline_tvshows['Plotline']
netflix_tvshows_plotline_w = ' '.join(netflix_plotline_tvshows_t)

# In[37]:


stopwords = set(STOPWORDS)

wordcloud.netflix_plotline_tvshows = WordCloud(width = 1000, height = 500,
                                              background_color ='white',
                                              stopwords = stopwords,
                                              min_font_size = 10)

```

```
    ).generate(netflix_tvshows_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Netflix is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud.netflix_plotline_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

In[38]:

```
# Hulu Wordcloud
hulu_plotline_tvshows_t = hulu_plotline_tvshows['Plotline']
hulu_tvshows_plotline_w = ' '.join(hulu_plotline_tvshows_t)
```

In[39]:

```
stopwords = set(STOPWORDS)

wordcloud_hulu_plotline_tvshows = WordCloud(width = 1000, height = 500,
                                             background_color ='white',
                                             stopwords = stopwords,
                                             min_font_size = 10
                                             ).generate(hulu_tvshows_plotline_w)
```

```
print('\nThe Wordcloud Generated from Plotlines of Hulu is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_hulu_plotline_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)
```

plt.show()

In[40]:

```
# Prime Video Wordcloud
prime_video_plotline_tvshows_t = prime_video_plotline_tvshows['Plotline']
prime_video_tvshows_plotline_w = ' '.join(prime_video_plotline_tvshows_t)
```

In[41]:

```
stopwords = set(STOPWORDS)

wordcloud_prime_video_plotline_tvshows = WordCloud(width = 1000, height = 500,
                                                 background_color ='white',
                                                 stopwords = stopwords,
                                                 min_font_size = 10
                                                 ).generate(prime_video_tvshows_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Prime Video is : \n')
```

```

# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_prime_video_plotline_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

# In[42]:


# Disney+ Wordcloud
disney_plotline_tvshows_t = disney_plotline_tvshows['Plotline']
disney_tvshows_plotline_w = ' '.join(disney_plotline_tvshows_t)

# In[43]:


stopwords = set(STOPWORDS)

wordcloud_disney_plotline_tvshows = WordCloud(width = 1000, height = 500,
                                              background_color ='white',
                                              stopwords = stopwords,
                                              min_font_size = 10
                                              ).generate(disney_tvshows_plotline_w)

print('\nThe Wordcloud Generated from Plotlines of Disney+ is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_disney_plotline_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

```

otttvshows_roto.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/rotopy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

# In[2]:

```

```

# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')

# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()

# In[6]:


# profile = ProfileReport(df_tvshows)
# profile

# In[7]:

```

```

def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***25')
    print('Colums Names : \n', df.columns)
    print('***25')
    print('Datatype of Columns : \n', df.dtypes)
    print('***25')
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***25')
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***25')
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

```

In[8]:

```
data_investigate(df_tvshows)
```

In[9]:

```

# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)

```

```

# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix', 'Prime Video', 'Disney+', 'Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
```

data_investigate(df_tvshows)

```
# In[11]:
```

```

df_tvshows.head()

# In[12]:


df_tvshows.describe()

# In[13]:


df_tvshows.corr()

# In[14]:


# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('Rotten Tomatoes', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]

# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]

# In[18]:


df_tvshows_roto = df_tvshows.copy()

```

```
# In[19]:
```

```
df_tvshows_roto.drop(df_tvshows_roto.loc[df_tvshows_roto['Rotten Tomatoes'] == "NA"].index,
inplace = True)
# df_tvshows_roto = df_tvshows_roto[df_tvshows_roto.Rotten Tomatoes != "NA"]
df_tvshows_roto['Rotten Tomatoes'] = df_tvshows_roto['Rotten Tomatoes'].astype(int)
```



```
# In[20]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_roto_tvshows = df_tvshows_roto.loc[df_tvshows_roto['Netflix'] == 1]
hulu_roto_tvshows = df_tvshows_roto.loc[df_tvshows_roto['Hulu'] == 1]
prime_video_roto_tvshows = df_tvshows_roto.loc[df_tvshows_roto['Prime Video'] == 1]
disney_roto_tvshows = df_tvshows_roto.loc[df_tvshows_roto['Disney+'] == 1]
```



```
# In[21]:
```

```
df_tvshows_roto_group = df_tvshows_roto.copy()
```



```
# In[22]:
```

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_roto.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```



```
# In[23]:
```

```
df_roto_high_tvshows = df_tvshows_roto.sort_values(by = 'Rotten Tomatoes', ascending =
False).reset_index()
df_roto_high_tvshows = df_roto_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_roto['Rotten Tomatoes'] == (df_tvshows_roto['Rotten
Tomatoes'].max()))
# df_roto_high_tvshows = df_tvshows_roto[filter]

# highestRated_tvshows = df_tvshows_roto.loc[df_tvshows_roto['Rotten Tomatoes'].idxmax()]

print('\nTV Shows with Highest Ever Rotten Tomatoes are : \n')
df_roto_high_tvshows.head(5)
```

```
# In[24]:
```

```
fig = px.bar(y = df_roto_high_tvshows['Title'][:15],
              x = df_roto_high_tvshows['Rotten Tomatoes'][:15],
              color = df_roto_high_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title  = 'TV Shows with Highest Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[25]:
```

```
df_roto_low_tvshows = df_tvshows_roto.sort_values(by = 'Rotten Tomatoes', ascending =
True).reset_index()
df_roto_low_tvshows = df_roto_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_roto['Rotten Tomatoes'] == (df_tvshows_roto['Rotten
Tomatoes'].min()))
# df_roto_low_tvshows = df_tvshows_roto[filter]

print('\nTV Shows with Lowest Ever Rotten Tomatoes are : \n')
df_roto_low_tvshows.head(5)
```

```
# In[26]:
```

```
fig = px.bar(y = df_roto_low_tvshows['Title'][:15],
              x = df_roto_low_tvshows['Rotten Tomatoes'][:15],
              color = df_roto_low_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title  = 'TV Shows with Lowest Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[27]:
```

```
print(f'''
    Total '{df_tvshows_roto['Rotten Tomatoes'].unique().shape[0]}' unique Rotten Tomatoes
    were Given, They were Like this,\n

{df_tvshows_roto.sort_values(by = 'Rotten Tomatoes', ascending = False)['Rotten
Tomatoes'].unique()}\n

    The Highest Ever Rotten Tomatoes Ever Any TV Show Got is
'{df_roto_high_tvshows['Title'][0]}' : '{df_roto_high_tvshows['Rotten Tomatoes'].max()}'\n

    The Lowest Ever Rotten Tomatoes Ever Any TV Show Got is
'{df_roto_low_tvshows['Title'][0]}' : '{df_roto_low_tvshows['Rotten Tomatoes'].min()}'\n
'''')
```

```
# In[28]:
```

```
netflix_roto_high_tvshows =
df_roto_high_tvshows.loc[df_roto_high_tvshows['Netflix']==1].reset_index()
netflix_roto_high_tvshows = netflix_roto_high_tvshows.drop(['index'], axis = 1)

netflix_roto_low_tvshows =
df_roto_low_tvshows.loc[df_roto_low_tvshows['Netflix']==1].reset_index()
netflix_roto_low_tvshows = netflix_roto_low_tvshows.drop(['index'], axis = 1)

netflix_roto_high_tvshows.head(5)
```

```
# In[29]:
```

```
fig = px.bar(y = netflix_roto_high_tvshows['Title'][:15],
              x = netflix_roto_high_tvshows['Rotten Tomatoes'][:15],
              color = netflix_roto_high_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'TV Shows with Highest Rotten Tomatoes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[30]:
```

```
fig = px.bar(y = netflix_roto_low_tvshows['Title'][:15],
              x = netflix_roto_low_tvshows['Rotten Tomatoes'][:15],
              color = netflix_roto_low_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'TV Shows with Lowest Rotten Tomatoes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[31]:
```

```
hulu_roto_high_tvshows =
df_roto_high_tvshows.loc[df_roto_high_tvshows['Hulu']==1].reset_index()
hulu_roto_high_tvshows = hulu_roto_high_tvshows.drop(['index'], axis = 1)

hulu_roto_low_tvshows =
df_roto_low_tvshows.loc[df_roto_low_tvshows['Hulu']==1].reset_index()
hulu_roto_low_tvshows = hulu_roto_low_tvshows.drop(['index'], axis = 1)

hulu_roto_high_tvshows.head(5)
```

```
# In[32]:
```

```
fig = px.bar(y = hulu_roto_high_tvshows['Title'][:15],
              x = hulu_roto_high_tvshows['Rotten Tomatoes'][:15],
```

```

color = hulu_roto_high_tvshows['Rotten Tomatoes'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
title  = 'TV Shows with Highest Rotten Tomatoes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[33]:

```

fig = px.bar(y = hulu_roto_low_tvshows['Title'][:15],
              x = hulu_roto_low_tvshows['Rotten Tomatoes'][:15],
              color = hulu_roto_low_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title  = 'TV Shows with Lowest Rotten Tomatoes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[34]:

```

prime_video_roto_high_tvshows = df_roto_high_tvshows.loc[df_roto_high_tvshows['Prime
Video']==1].reset_index()
prime_video_roto_high_tvshows = prime_video_roto_high_tvshows.drop(['index'], axis = 1)

prime_video_roto_low_tvshows = df_roto_low_tvshows.loc[df_roto_low_tvshows['Prime
Video']==1].reset_index()
prime_video_roto_low_tvshows = prime_video_roto_low_tvshows.drop(['index'], axis = 1)

prime_video_roto_high_tvshows.head(5)

```

In[35]:

```

fig = px.bar(y = prime_video_roto_high_tvshows['Title'][:15],
              x = prime_video_roto_high_tvshows['Rotten Tomatoes'][:15],
              color = prime_video_roto_high_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title  = 'TV Shows with Highest Rotten Tomatoes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[36]:

```

fig = px.bar(y = prime_video_roto_low_tvshows['Title'][:15],
              x = prime_video_roto_low_tvshows['Rotten Tomatoes'][:15],
              color = prime_video_roto_low_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title  = 'TV Shows with Lowest Rotten Tomatoes : Prime Video')

```

```

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


disney_roto_high_tvshows =
df_roto_high_tvshows.loc[df_roto_high_tvshows['Disney+']==1].reset_index()
disney_roto_high_tvshows = disney_roto_high_tvshows.drop(['index'], axis = 1)

disney_roto_low_tvshows =
df_roto_low_tvshows.loc[df_roto_low_tvshows['Disney+']==1].reset_index()
disney_roto_low_tvshows = disney_roto_low_tvshows.drop(['index'], axis = 1)

disney_roto_high_tvshows.head(5)

```

In[38]:

```

fig = px.bar(y = disney_roto_high_tvshows['Title'][:15],
              x = disney_roto_high_tvshows['Rotten Tomatoes'][:15],
              color = disney_roto_high_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'TV Shows with Highest Rotten Tomatoes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[39]:

```

fig = px.bar(y = disney_roto_low_tvshows['Title'][:15],
              x = disney_roto_low_tvshows['Rotten Tomatoes'][:15],
              color = disney_roto_low_tvshows['Rotten Tomatoes'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'TV Shows with Lowest Rotten Tomatoes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[40]:

```

print(f'''
    The TV Show with Highest Rotten Tomatoes Ever Got is
'{df_roto_high_tvshows['Title'][0]}' : '{df_roto_high_tvshows['Rotten Tomatoes'].max()}'\n
    The TV Show with Lowest Rotten Tomatoes Ever Got is
'{df_roto_low_tvshows['Title'][0]}' : '{df_roto_low_tvshows['Rotten Tomatoes'].min()}'\n

    The TV Show with Highest Rotten Tomatoes on 'Netflix' is
'{netflix_roto_high_tvshows['Title'][0]}' : '{netflix_roto_high_tvshows['Rotten Tomatoes'].max()}'\n
    The TV Show with Lowest Rotten Tomatoes on 'Netflix' is
'{netflix_roto_low_tvshows['Title'][0]}' : '{netflix_roto_low_tvshows['Rotten Tomatoes'].min()}'\n

```

```

The TV Show with Highest Rotten Tomatoes on 'Hulu' is
'{hulu_roto_high_tvshows['Title'][0]} : '{hulu_roto_high_tvshows['Rotten
Tomatoes']].max()}'\n
The TV Show with Lowest Rotten Tomatoes on 'Hulu' is
'{hulu_roto_low_tvshows['Title'][0]} : '{hulu_roto_low_tvshows['Rotten
Tomatoes']].min()}'\n

The TV Show with Highest Rotten Tomatoes on 'Prime Video' is
'{prime_video_roto_high_tvshows['Title'][0]} : '{prime_video_roto_high_tvshows['Rotten
Tomatoes']].max()}'\n
The TV Show with Lowest Rotten Tomatoes on 'Prime Video' is
'{prime_video_roto_low_tvshows['Title'][0]} : '{prime_video_roto_low_tvshows['Rotten
Tomatoes']].min()}'\n

The TV Show with Highest Rotten Tomatoes on 'Disney+' is
'{disney_roto_high_tvshows['Title'][0]} : '{disney_roto_high_tvshows['Rotten
Tomatoes']].max()}'\n
The TV Show with Lowest Rotten Tomatoes on 'Disney+' is
'{disney_roto_low_tvshows['Title'][0]} : '{disney_roto_low_tvshows['Rotten
Tomatoes']].min()}'\n
'''')

```

```
# In[41]:
```

```

print(f'''
Accross All Platforms the Average Rotten Tomatoes is '{round(df_tvshows_roto['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
The Average Rotten Tomatoes on 'Netflix' is '{round(netflix_roto_tvshows['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
The Average Rotten Tomatoes on 'Hulu' is '{round(hulu_roto_tvshows['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
The Average Rotten Tomatoes on 'Prime Video' is
'{round(prime_video_roto_tvshows['Rotten Tomatoes'].mean(), ndigits = 2)}'\n
The Average Rotten Tomatoes on 'Disney+' is '{round(disney_roto_tvshows['Rotten
Tomatoes'].mean(), ndigits = 2)}'\n
'''')

```

```
# In[42]:
```

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_roto['Rotten Tomatoes'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_roto['Rotten Tomatoes'], ax = ax[1])
plt.show()

```

```
# In[43]:
```

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Rotten Tomatoes s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_roto_tvshows['Rotten Tomatoes'][:100], color = 'lightblue', legend
= True, kde = True)

```

```

sns.histplot(netflix_roto_tvshows['Rotten Tomatoes'][:100], color = 'red', legend = True,
kde = True)
sns.histplot(hulu_roto_tvshows['Rotten Tomatoes'][:100], color = 'lightgreen', legend =
True, kde = True)
sns.histplot(disney_roto_tvshows['Rotten Tomatoes'][:100], color = 'darkblue', legend =
True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[44]:

```

def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(0,11):
        # print(data)
        return 10
    if data in range(11,21):
        return 20
    if data in range(21,31):
        return 30
    if data in range(31,41):
        return 40
    if data in range(41,51):
        return 50
    if data in range(51,61):
        return 60
    if data in range(61,71):
        return 70
    if data in range(71,81):
        return 80
    if data in range(81,91):
        return 90
    if data in range(91,101):
        return 100

```

In[45]:

```

df_tvshows_roto_group['Rotten Tomatoes Group'] = df_tvshows_roto['Rotten
Tomatoes'].apply(round_fix)

roto_values = df_tvshows_roto_group['Rotten Tomatoes
Group'].value_counts().sort_index(ascending = False).tolist()
roto_index = df_tvshows_roto_group['Rotten Tomatoes
Group'].value_counts().sort_index(ascending = False).index

# roto_values, roto_index

```

In[46]:

```
roto_group_count = df_tvshows_roto_group.groupby('Rotten Tomatoes Group')['Title'].count()
```

```

roto_group_tvshows = df_tvshows_roto_group.groupby('Rotten Tomatoes Group')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
roto_group_data_tvshows = pd.concat([roto_group_count, roto_group_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
roto_group_data_tvshows = roto_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)

# In[47]:


# Rotten Tomatoes Group with TV Shows Counts - All Platforms Combined
roto_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[48]:


roto_group_data_tvshows.sort_values(by = 'Rotten Tomatoes Group', ascending = False)

# In[49]:


fig = px.bar(y = roto_group_data_tvshows['TV Shows Count'],
              x = roto_group_data_tvshows['Rotten Tomatoes Group'],
              color = roto_group_data_tvshows['Rotten Tomatoes Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Rotten Tomatoes : Rating'},
              title = 'TV Shows with Group Rotten Tomatoes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]:


fig = px.pie(roto_group_data_tvshows[:10],
              names = roto_group_data_tvshows['Rotten Tomatoes Group'],
              values = roto_group_data_tvshows['TV Shows Count'],
              color = roto_group_data_tvshows['TV Shows Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'TV Shows Count based on Rotten Tomatoes Group')
fig.show()

# In[51]:


df_roto_group_high_tvshows = roto_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_roto_group_high_tvshows = df_roto_group_high_tvshows.drop(['index'], axis = 1)
# filter = (roto_group_data_tvshows['TV Shows Count'] == (roto_group_data_tvshows['TV
Shows Count'].max()))
# df_roto_group_high_tvshows = roto_group_data_tvshows[filter]

# highestRated_tvshows = roto_group_data_tvshows.loc[roto_group_data_tvshows['TV Shows
Count'].idxmax()]

```

```

# print('\nRotten Tomatoes with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_roto_group_high_tvshows.head(5)

# In[52]:


df_roto_group_low_tvshows = roto_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_roto_group_low_tvshows = df_roto_group_low_tvshows.drop(['index'], axis = 1)
# filter = (roto_group_data_tvshows['TV Shows Count'] == (roto_group_data_tvshows['TV Shows Count'].min()))
# df_roto_group_low_tvshows = roto_group_data_tvshows[filter]

# print('\nRotten Tomatoes with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_roto_group_low_tvshows.head(5)

```

```
# In[53]:
```

```

print(f'''
    Total '{df_tvshows_roto['Rotten Tomatoes'].count()}' Titles are available on All Platforms, out of which\n
    You Can Choose to see TV Shows from Total '{roto_group_data_tvshows['Rotten Tomatoes Group'].unique().shape[0]}' Rotten Tomatoes Group, They were Like this, \n

    {roto_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)['Rotten Tomatoes Group'].unique()} etc. \n

    The Rotten Tomatoes Group with Highest TV Shows Count have
'{roto_group_data_tvshows['TV Shows Count'].max()}' TV Shows Available is
'{df_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}', &\n
    The Rotten Tomatoes Group with Lowest TV Shows Count have
'{roto_group_data_tvshows['TV Shows Count'].min()}' TV Shows Available is
'{df_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}'
    ''')

```

```
# In[54]:
```

```

netflix_roto_group_tvshows = roto_group_data_tvshows[roto_group_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_roto_group_tvshows = netflix_roto_group_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_roto_group_high_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_roto_group_high_tvshows = netflix_roto_group_high_tvshows.drop(['index'], axis = 1)

netflix_roto_group_low_tvshows = df_roto_group_low_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_roto_group_low_tvshows = netflix_roto_group_low_tvshows.drop(['index'], axis = 1)

netflix_roto_group_high_tvshows.head(5)

```

```
# In[55]:
```

```
hulu_roto_group_tvshows = roto_group_data_tvshows[roto_group_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_roto_group_tvshows = hulu_roto_group_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_roto_group_high_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_roto_group_high_tvshows = hulu_roto_group_high_tvshows.drop(['index'], axis = 1)

hulu_roto_group_low_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_roto_group_low_tvshows = hulu_roto_group_low_tvshows.drop(['index'], axis = 1)

hulu_roto_group_high_tvshows.head(5)
```

In[56]:

```
prime_video_roto_group_tvshows = roto_group_data_tvshows[roto_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_roto_group_tvshows = prime_video_roto_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_roto_group_high_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_roto_group_high_tvshows = prime_video_roto_group_high_tvshows.drop(['index'], axis = 1)

prime_video_roto_group_low_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_roto_group_low_tvshows = prime_video_roto_group_low_tvshows.drop(['index'], axis = 1)

prime_video_roto_group_high_tvshows.head(5)
```

In[57]:

```
disney_roto_group_tvshows = roto_group_data_tvshows[roto_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_roto_group_tvshows = disney_roto_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_roto_group_high_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_roto_group_high_tvshows = disney_roto_group_high_tvshows.drop(['index'], axis = 1)

disney_roto_group_low_tvshows = df_roto_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_roto_group_low_tvshows = disney_roto_group_low_tvshows.drop(['index'], axis = 1)

disney_roto_group_high_tvshows.head(5)
```

In[58]:

```

print(f'''
    The Rotten Tomatoes Group with Highest TV Shows Count Ever Got is
'{df_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}':
'{df_roto_group_high_tvshows['TV Shows Count'].max()}'\n
    The Rotten Tomatoes Group with Lowest TV Shows Count Ever Got is
'{df_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}': '{df_roto_group_low_tvshows['TV
Shows Count'].min()}'\n

    The Rotten Tomatoes Group with Highest TV Shows Count on 'Netflix' is
'{netflix_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}':
'{netflix_roto_group_high_tvshows['Netflix'].max()}'\n
    The Rotten Tomatoes Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}':
'{netflix_roto_group_low_tvshows['Netflix'].min()}'\n

    The Rotten Tomatoes Group with Highest TV Shows Count on 'Hulu' is
'{hulu_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}':
'{hulu_roto_group_high_tvshows['Hulu'].max()}'\n
    The Rotten Tomatoes Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}':
'{hulu_roto_group_low_tvshows['Hulu'].min()}'\n

    The Rotten Tomatoes Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}':
'{prime_video_roto_group_high_tvshows['Prime Video'].max()}'\n
    The Rotten Tomatoes Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}':
'{prime_video_roto_group_low_tvshows['Prime Video'].min()}'\n

    The Rotten Tomatoes Group with Highest TV Shows Count on 'Disney+' is
'{disney_roto_group_high_tvshows['Rotten Tomatoes Group'][0]}':
'{disney_roto_group_high_tvshows['Disney+'].max()}'\n
    The Rotten Tomatoes Group with Lowest TV Shows Count on 'Disney+' is
'{disney_roto_group_low_tvshows['Rotten Tomatoes Group'][0]}':
'{disney_roto_group_low_tvshows['Disney+'].min()}'\n
    ''')

# In[59]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.barplot(x = netflix_roto_group_tvshows['Rotten Tomatoes Group'][:10], y =
netflix_roto_group_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ro_ax2 = sns.barplot(x = hulu_roto_group_tvshows['Rotten Tomatoes Group'][:10], y =
hulu_roto_group_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ro_ax3 = sns.barplot(x = prime_video_roto_group_tvshows['Rotten Tomatoes Group'][:10], y =
prime_video_roto_group_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ro_ax4 = sns.barplot(x = disney_roto_group_tvshows['Rotten Tomatoes Group'][:10], y =
disney_roto_group_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[60]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'], y =
roto_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'], y =
roto_group_data_tvshows['Hulu'], color = 'lightgreen')
sns.lineplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'], y =
roto_group_data_tvshows['Prime Video'], color = 'lightblue')
sns.lineplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'], y =
roto_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Rotten Tomatoes Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()
```

```
# In[61]:
```

```
print(f'''
    Across All Platforms Total Count of Rotten Tomatoes Group is
'{roto_group_data_tvshows['Rotten Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Netflix' is
'{netflix_roto_group_tvshows['Rotten Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Hulu' is '{hulu_roto_group_tvshows['Rotten
Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Prime Video' is
'{prime_video_roto_group_tvshows['Rotten Tomatoes Group'].unique().shape[0]}'\n
    Total Count of Rotten Tomatoes Group on 'Disney+' is
'{disney_roto_group_tvshows['Rotten Tomatoes Group'].unique().shape[0]}'\n
    ''')
```

```
# In[62]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.lineplot(y = roto_group_data_tvshows['Rotten Tomatoes Group'], x =
roto_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_ro_ax2 = sns.lineplot(y = roto_group_data_tvshows['Rotten Tomatoes Group'], x =
roto_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ro_ax3 = sns.lineplot(y = roto_group_data_tvshows['Rotten Tomatoes Group'], x =
roto_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ro_ax4 = sns.lineplot(y = roto_group_data_tvshows['Rotten Tomatoes Group'], x =
roto_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[63]:
```

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ro_ax1 = sns.barplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'][:10], y =
roto_group_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ro_ax2 = sns.barplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'][:10], y =
roto_group_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ro_ax3 = sns.barplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'][:10], y =
roto_group_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ro_ax4 = sns.barplot(x = roto_group_data_tvshows['Rotten Tomatoes Group'][:10], y =
roto_group_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ro_ax1.title.set_text(labels[0])
h_ro_ax2.title.set_text(labels[1])
p_ro_ax3.title.set_text(labels[2])
d_ro_ax4.title.set_text(labels[3])

plt.show()

```

otttvshows_runtime.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


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

```

```

import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')


# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()


# In[6]:


# profile = ProfileReport(df_tvshows)
# profile


# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index)))))


```

```

print('*****')
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

# In[8]:

data_investigate(df_tvshows)

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language

```

```

df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
```

data_investigate(df_tvshows)

```
# In[11]:
```

df_tvshows.head()

```
# In[12]:
```

df_tvshows.describe()

```
# In[13]:
```

df_tvshows.corr()

```
# In[14]:
```

```

# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_runtimes = df_tvshows.copy()

# In[19]:


df_tvshows_runtimes.drop(df_tvshows_runtimes.loc[df_tvshows_runtimes['Runtime'] == "NA"].index, inplace = True)
# df_tvshows_runtimes = df_tvshows_runtimes[df_tvshows_runtimes.Runtime != "NA"]
df_tvshows_runtimes['Runtime'] = df_tvshows_runtimes['Runtime'].astype(int)

# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_runtimes_tvshows = df_tvshows_runtimes.loc[df_tvshows_runtimes['Netflix'] == 1]

```

```

hulu_runtimes_tvshows = df_tvshows_runtimes.loc[df_tvshows_runtimes['Hulu'] == 1]
prime_video_runtimes_tvshows = df_tvshows_runtimes.loc[df_tvshows_runtimes['Prime Video'] == 1]
disney_runtimes_tvshows = df_tvshows_runtimes.loc[df_tvshows_runtimes['Disney+'] == 1]

# In[21]:


df_tvshows_runtimes_group = df_tvshows_runtimes.copy()

# In[22]:


df_tvshows_screentimes = df_tvshows_runtimes.copy()
df_tvshows_screentimes['Screentime'] = round(df_tvshows_runtimes['Runtime']/60, ndigits = 2)

# In[23]:


# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_screentimes_tvshows = df_tvshows_screentimes.loc[df_tvshows_screentimes['Netflix'] == 1]
hulu_screentimes_tvshows = df_tvshows_screentimes.loc[df_tvshows_screentimes['Hulu'] == 1]
prime_video_screentimes_tvshows = df_tvshows_screentimes.loc[df_tvshows_screentimes['Prime Video'] == 1]
disney_screentimes_tvshows = df_tvshows_screentimes.loc[df_tvshows_screentimes['Disney+'] == 1]

# In[24]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_runtimes.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()

# In[25]:


df_runtimes_high_tvshows = df_tvshows_runtimes.sort_values(by = 'Runtime', ascending = False).reset_index()
df_runtimes_high_tvshows = df_runtimes_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_runtimes['Runtime'] == (df_tvshows_runtimes['Runtime'].max()))
# df_runtimes_high_tvshows = df_tvshows_runtimes[filter]

```

```

# highestRated_tvshows = df_tvshows_runtimes.loc[df_tvshows_runtimes['Runtime'].idxmax()]

print('\nTV Shows with Highest Ever Runtime are : \n')
df_runtimes_high_tvshows.head(5)

# In[26]:


fig = px.bar(y = df_runtimes_high_tvshows['Title'][:15],
              x = df_runtimes_high_tvshows['Runtime'][:15],
              color = df_runtimes_high_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title = 'TV Shows with Highest Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[27]:


df_runtimes_low_tvshows = df_tvshows_runtimes.sort_values(by = 'Runtime', ascending = True).reset_index()
df_runtimes_low_tvshows = df_runtimes_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_runtimes['Runtime'] == (df_tvshows_runtimes['Runtime'].min()))
# df_runtimes_low_tvshows = df_tvshows_runtimes[filter]

print('\nTV Shows with Lowest Ever Runtime are : \n')
df_runtimes_low_tvshows.head(5)

# In[28]:


fig = px.bar(y = df_runtimes_low_tvshows['Title'][:15],
              x = df_runtimes_low_tvshows['Runtime'][:15],
              color = df_runtimes_low_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title = 'TV Shows with Lowest Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[29]:


print(f'''
    Total '{df_tvshows_runtimes['Runtime'].unique().shape[0]}' unique Runtime s were
Given, They were Like this,\n

{df_tvshows_runtimes.sort_values(by = 'Runtime', ascending = False)['Runtime'].unique()}\n

    The Highest Ever Runtime Ever Any TV Show Got is
'{df_runtimes_high_tvshows['Title'][0]}' : '{df_runtimes_high_tvshows['Runtime'].max()}'\n

    The Lowest Ever Runtime Ever Any TV Show Got is
'{df_runtimes_low_tvshows['Title'][0]}' : '{df_runtimes_low_tvshows['Runtime'].min()}'\n

```

```
'''
```

```
# In[30]:
```

```
netflix_runtimes_high_tvshows =
df_runtimes_high_tvshows.loc[df_runtimes_high_tvshows['Netflix']==1].reset_index()
netflix_runtimes_high_tvshows = netflix_runtimes_high_tvshows.drop(['index'], axis = 1)

netflix_runtimes_low_tvshows =
df_runtimes_low_tvshows.loc[df_runtimes_low_tvshows['Netflix']==1].reset_index()
netflix_runtimes_low_tvshows = netflix_runtimes_low_tvshows.drop(['index'], axis = 1)

netflix_runtimes_high_tvshows.head(5)
```

```
# In[31]:
```

```
fig = px.bar(y = netflix_runtimes_high_tvshows['Title'][:15],
              x = netflix_runtimes_high_tvshows['Runtime'][:15],
              color = netflix_runtimes_high_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title = 'TV Shows with Highest Runtime in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[32]:
```

```
fig = px.bar(y = netflix_runtimes_low_tvshows['Title'][:15],
              x = netflix_runtimes_low_tvshows['Runtime'][:15],
              color = netflix_runtimes_low_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title = 'TV Shows with Lowest Runtime in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:
```

```
hulu_runtimes_high_tvshows =
df_runtimes_high_tvshows.loc[df_runtimes_high_tvshows['Hulu']==1].reset_index()
hulu_runtimes_high_tvshows = hulu_runtimes_high_tvshows.drop(['index'], axis = 1)

hulu_runtimes_low_tvshows =
df_runtimes_low_tvshows.loc[df_runtimes_low_tvshows['Hulu']==1].reset_index()
hulu_runtimes_low_tvshows = hulu_runtimes_low_tvshows.drop(['index'], axis = 1)

hulu_runtimes_high_tvshows.head(5)
```

```
# In[34]:
```

```

fig = px.bar(y = hulu_runtimes_high_tvshows['Title'][:15],
              x = hulu_runtimes_high_tvshows['Runtime'][:15],
              color = hulu_runtimes_high_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title  = 'TV Shows with Highest Runtime in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[35]:

```

fig = px.bar(y = hulu_runtimes_low_tvshows['Title'][:15],
              x = hulu_runtimes_low_tvshows['Runtime'][:15],
              color = hulu_runtimes_low_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title  = 'TV Shows with Lowest Runtime in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[36]:

```

prime_video_runtimes_high_tvshows =
df_runtimes_high_tvshows.loc[df_runtimes_high_tvshows['Prime Video']==1].reset_index()
prime_video_runtimes_high_tvshows = prime_video_runtimes_high_tvshows.drop(['index'], axis = 1)

prime_video_runtimes_low_tvshows =
df_runtimes_low_tvshows.loc[df_runtimes_low_tvshows['Prime Video']==1].reset_index()
prime_video_runtimes_low_tvshows = prime_video_runtimes_low_tvshows.drop(['index'], axis = 1)

prime_video_runtimes_high_tvshows.head(5)

```

In[37]:

```

fig = px.bar(y = prime_video_runtimes_high_tvshows['Title'][:15],
              x = prime_video_runtimes_high_tvshows['Runtime'][:15],
              color = prime_video_runtimes_high_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title  = 'TV Shows with Highest Runtime in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[38]:

```

fig = px.bar(y = prime_video_runtimes_low_tvshows['Title'][:15],
              x = prime_video_runtimes_low_tvshows['Runtime'][:15],

```

```

color = prime_video_runtimes_low_tvshows['Runtime'][:15],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
title  = 'TV Shows with Lowest Runtime in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[39]:

```

disney_runtimes_high_tvshows =
df_runtimes_high_tvshows.loc[df_runtimes_high_tvshows['Disney+']==1].reset_index()
disney_runtimes_high_tvshows = disney_runtimes_high_tvshows.drop(['index'], axis = 1)

disney_runtimes_low_tvshows =
df_runtimes_low_tvshows.loc[df_runtimes_low_tvshows['Disney+']==1].reset_index()
disney_runtimes_low_tvshows = disney_runtimes_low_tvshows.drop(['index'], axis = 1)

disney_runtimes_high_tvshows.head(5)

```

In[40]:

```

fig = px.bar(y = disney_runtimes_high_tvshows['Title'][:15],
              x = disney_runtimes_high_tvshows['Runtime'][:15],
              color = disney_runtimes_high_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title  = 'TV Shows with Highest Runtime in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[41]:

```

fig = px.bar(y = disney_runtimes_low_tvshows['Title'][:15],
              x = disney_runtimes_low_tvshows['Runtime'][:15],
              color = disney_runtimes_low_tvshows['Runtime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Runtime : In Minutes'},
              title  = 'TV Shows with Lowest Runtime in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[42]:

```

print(f'''
    The TV Show with Highest Runtime Ever Got is
'{df_runtimes_high_tvshows['Title'][0]}' : '{df_runtimes_high_tvshows['Runtime'].max()}'\n
    The TV Show with Lowest Runtime Ever Got is '{df_runtimes_low_tvshows['Title'][0]}'
: '{df_runtimes_low_tvshows['Runtime'].min()}'\n

```

```

    The TV Show with Highest Runtime on 'Netflix' is
'{netflix_runtimes_high_tvshows['Title'][0]}' :
'{netflix_runtimes_high_tvshows['Runtime'].max()}'\n
    The TV Show with Lowest Runtime on 'Netflix' is
'{netflix_runtimes_low_tvshows['Title'][0]}' :
'{netflix_runtimes_low_tvshows['Runtime'].min()}'\n

    The TV Show with Highest Runtime on 'Hulu' is
'{hulu_runtimes_high_tvshows['Title'][0]}' :
'{hulu_runtimes_high_tvshows['Runtime'].max()}'\n
    The TV Show with Lowest Runtime on 'Hulu' is
'{hulu_runtimes_low_tvshows['Title'][0]}' :
'{hulu_runtimes_low_tvshows['Runtime'].min()}'\n

    The TV Show with Highest Runtime on 'Prime Video' is
'{prime_video_runtimes_high_tvshows['Title'][0]}' :
'{prime_video_runtimes_high_tvshows['Runtime'].max()}'\n
    The TV Show with Lowest Runtime on 'Prime Video' is
'{prime_video_runtimes_low_tvshows['Title'][0]}' :
'{prime_video_runtimes_low_tvshows['Runtime'].min()}'\n

    The TV Show with Highest Runtime on 'Disney+' is
'{disney_runtimes_high_tvshows['Title'][0]}' :
'{disney_runtimes_high_tvshows['Runtime'].max()}'\n
    The TV Show with Lowest Runtime on 'Disney+' is
'{disney_runtimes_low_tvshows['Title'][0]}' :
'{disney_runtimes_low_tvshows['Runtime'].min()}'\n
    ''')

```

In[43]:

```

print(f'''
    Across All Platforms the Average Runtime is
'{round(df_tvshows_runtimes['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Netflix' is
'{round(netflix_runtimes_tvshows['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Hulu' is '{round(hulu_runtimes_tvshows['Runtime'].mean(),
ndigits = 2)}'\n
    The Average Runtime on 'Prime Video' is
'{round(prime_video_runtimes_tvshows['Runtime'].mean(), ndigits = 2)}'\n
    The Average Runtime on 'Disney+' is
'{round(disney_runtimes_tvshows['Runtime'].mean(), ndigits = 2)}'\n
    ''')

```

In[44]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_runtimes['Runtime'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_runtimes['Runtime'], ax = ax[1])
plt.show()

```

In[45]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))

```

```

plt.title('Runtime s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_runtimes_tvshows['Runtime'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_runtimes_tvshows['Runtime'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_runtimes_tvshows['Runtime'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_runtimes_tvshows['Runtime'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[46]:

```

def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(0,51):
        # print(data)
        return 50
    if data in range(51,101):
        return 100
    if data in range(101,151):
        return 150
    if data in range(151,201):
        return 200
    if data in range(201,251):
        return 250
    if data in range(251,301):
        return 300
    if data in range(301,351):
        return 350
    if data in range(351,401):
        return 400
    if data in range(401,451):
        return 450
    if data in range(451,501):
        return 500
    if data in range(501,551):
        return 550
    if data in range(551,601):
        return 600
    if data in range(601,651):
        return 650
    if data in range(651,701):
        return 700
    if data in range(701,751):
        return 750
    if data in range(751,801):
        return 800
    if data in range(801,851):
        return 850
    if data in range(851,901):

```

```
        return 900
    if data in range(901,951):
        return 950
    if data in range(951,1001):
        return 1000
    if data in range(1001,1051):
        return 1050
    if data in range(1051,1101):
        return 1100
    if data in range(1101,1151):
        return 1150
    if data in range(1151,1201):
        return 1200
    if data in range(1201,1251):
        return 1250
    if data in range(1251,1301):
        return 1300
    if data in range(1301,1351):
        return 1350
    if data in range(1351,2001):
        return 2000
```

```
# In[47]:
```

```
df_tvshows_runtimes_group['Runtime Group'] =
df_tvshows_runtimes['Runtime'].apply(round_fix)

runtimes_values = df_tvshows_runtimes_group['Runtime
Group'].value_counts().sort_index(ascending = False).tolist()
runtimes_index = df_tvshows_runtimes_group['Runtime
Group'].value_counts().sort_index(ascending = False).index

# runtimes_values, runtimes_index
```

```
# In[48]:
```

```
runtimes_group_count = df_tvshows_runtimes_group.groupby('Runtime Group')['Title'].count()
runtimes_group_tvshows = df_tvshows_runtimes_group.groupby('Runtime Group')[['Netflix',
'Hulu', 'Prime Video', 'Disney+']].sum()
runtimes_group_data_tvshows = pd.concat([runtimes_group_count, runtimes_group_tvshows],
axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
runtimes_group_data_tvshows = runtimes_group_data_tvshows.sort_values(by = 'TV Shows
Count', ascending = False)
```

```
# In[49]:
```

```
# Runtime Group with TV Shows Counts - All Platforms Combined
runtimes_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)
```

```
# In[50]:
```

```
runtimes_group_data_tvshows.sort_values(by = 'Runtime Group', ascending = False)
```

```
# In[51]:
```

```
fig = px.bar(y = runtimes_group_data_tvshows['TV Shows Count'],
              x = runtimes_group_data_tvshows['Runtime Group'],
              color = runtimes_group_data_tvshows['Runtime Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Runtime : In Minutes'},
              title = 'TV Shows with Group Runtime in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()
```

```
# In[52]:
```

```
fig = px.pie(runtimes_group_data_tvshows[:10],
              names = runtimes_group_data_tvshows['Runtime Group'],
              values = runtimes_group_data_tvshows['TV Shows Count'],
              color = runtimes_group_data_tvshows['TV Shows Count'],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'TV Shows Count based on Runtime Group')
fig.show()
```

```
# In[53]:
```

```
df_runtimes_group_high_tvshows = runtimes_group_data_tvshows.sort_values(by = 'TV Shows
Count', ascending = False).reset_index()
df_runtimes_group_high_tvshows = df_runtimes_group_high_tvshows.drop(['index'], axis = 1)
# filter = (runtimes_group_data_tvshows['TV Shows Count'] ==
(runtimes_group_data_tvshows['TV Shows Count'].max()))
# df_runtimes_group_high_tvshows = runtimes_group_data_tvshows[filter]

# highestRated_tvshows = runtimes_group_data_tvshows.loc[runtimes_group_data_tvshows['TV
Shows Count'].idxmax()]

# print('\nRuntime with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_runtimes_group_high_tvshows.head(5)
```

```
# In[54]:
```

```
df_runtimes_group_low_tvshows = runtimes_group_data_tvshows.sort_values(by = 'TV Shows
Count', ascending = True).reset_index()
df_runtimes_group_low_tvshows = df_runtimes_group_low_tvshows.drop(['index'], axis = 1)
# filter = (runtimes_group_data_tvshows['TV Shows Count'] ==
(runtimes_group_data_tvshows['TV Shows Count'].min()))
# df_runtimes_group_low_tvshows = runtimes_group_data_tvshows[filter]

# print('\nRuntime with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_runtimes_group_low_tvshows.head(5)
```

```
# In[55]:
```

```

print(f'''
    Total '{df_tvshows_runtimes['Runtime'].count()}' Titles are available on All
Platforms, out of which\n
    You Can Choose to see TV Shows from Total '{runtimes_group_data_tvshows['Runtime
Group'].unique().shape[0]}' Runtime Group, They were Like this, \n

    {runtimes_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Runtime Group'].unique()} etc. \n

    The Runtime Group with Highest TV Shows Count have '{runtimes_group_data_tvshows['TV
Shows Count'].max()}' TV Shows Available is '{df_runtimes_group_high_tvshows['Runtime
Group'][0]}', &\n
    The Runtime Group with Lowest TV Shows Count have '{runtimes_group_data_tvshows['TV
Shows Count'].min()}' TV Shows Available is '{df_runtimes_group_low_tvshows['Runtime
Group'][0]}'
    ''')

```

In[56]:

```

netflix_runtimes_group_tvshows =
runtimes_group_data_tvshows[runtimes_group_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_runtimes_group_tvshows = netflix_runtimes_group_tvshows.drop(['index', 'Hulu',
'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_runtimes_group_high_tvshows = df_runtimes_group_high_tvshows.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_runtimes_group_high_tvshows = netflix_runtimes_group_high_tvshows.drop(['index'],
axis = 1)

netflix_runtimes_group_low_tvshows = df_runtimes_group_high_tvshows.sort_values(by =
'Netflix', ascending = True).reset_index()
netflix_runtimes_group_low_tvshows = netflix_runtimes_group_low_tvshows.drop(['index'],
axis = 1)

netflix_runtimes_group_high_tvshows.head(5)

```

In[57]:

```

hulu_runtimes_group_tvshows =
runtimes_group_data_tvshows[runtimes_group_data_tvshows['Hulu'] != 0].sort_values(by =
'Hulu', ascending = False).reset_index()
hulu_runtimes_group_tvshows = hulu_runtimes_group_tvshows.drop(['index', 'Netflix', 'Prime
Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_runtimes_group_high_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_runtimes_group_high_tvshows = hulu_runtimes_group_high_tvshows.drop(['index'], axis =
1)

hulu_runtimes_group_low_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_runtimes_group_low_tvshows = hulu_runtimes_group_low_tvshows.drop(['index'], axis =
1)

hulu_runtimes_group_high_tvshows.head(5)

```

```
# In[58]:
```

```
prime_video_runtimes_group_tvshows =
runtimes_group_data_tvshows[runtimes_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_runtimes_group_tvshows = prime_video_runtimes_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_runtimes_group_high_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_runtimes_group_high_tvshows =
prime_video_runtimes_group_high_tvshows.drop(['index'], axis = 1)

prime_video_runtimes_group_low_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_runtimes_group_low_tvshows =
prime_video_runtimes_group_low_tvshows.drop(['index'], axis = 1)

prime_video_runtimes_group_high_tvshows.head(5)
```



```
# In[59]:
```

```
disney_runtimes_group_tvshows =
runtimes_group_data_tvshows[runtimes_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_runtimes_group_tvshows = disney_runtimes_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_runtimes_group_high_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_runtimes_group_high_tvshows = disney_runtimes_group_high_tvshows.drop(['index'], axis = 1)

disney_runtimes_group_low_tvshows = df_runtimes_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_runtimes_group_low_tvshows = disney_runtimes_group_low_tvshows.drop(['index'], axis = 1)

disney_runtimes_group_high_tvshows.head(5)
```



```
# In[60]:
```

```
print(f'''  
    The Runtime Group with Highest TV Shows Count Ever Got is  
'{df_runtimes_group_high_tvshows['Runtime Group'][0]}' :  
'{df_runtimes_group_high_tvshows['TV Shows Count'].max()}'\n  
    The Runtime Group with Lowest TV Shows Count Ever Got is  
'{df_runtimes_group_low_tvshows['Runtime Group'][0]}' : '{df_runtimes_group_low_tvshows['TV Shows Count'].min()}'\n  
  
    The Runtime Group with Highest TV Shows Count on 'Netflix' is  
'{netflix_runtimes_group_high_tvshows['Runtime Group'][0]}' :  
'{netflix_runtimes_group_high_tvshows['Netflix'].max()}'\n
```

```

    The Runtime Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_runtimes_group_low_tvshows['Runtime Group'][0]} :
'{netflix_runtimes_group_low_tvshows['Netflix'].min()}'\n

    The Runtime Group with Highest TV Shows Count on 'Hulu' is
'{hulu_runtimes_group_high_tvshows['Runtime Group'][0]} :
'{hulu_runtimes_group_high_tvshows['Hulu'].max()}'\n
    The Runtime Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_runtimes_group_low_tvshows['Runtime Group'][0]} :
'{hulu_runtimes_group_low_tvshows['Hulu'].min()}'\n

    The Runtime Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_runtimes_group_high_tvshows['Runtime Group'][0]} :
'{prime_video_runtimes_group_high_tvshows['Prime Video'].max()}'\n
    The Runtime Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_runtimes_group_low_tvshows['Runtime Group'][0]} :
'{prime_video_runtimes_group_low_tvshows['Prime Video'].min()}'\n

    The Runtime Group with Highest TV Shows Count on 'Disney+' is
'{disney_runtimes_group_high_tvshows['Runtime Group'][0]} :
'{disney_runtimes_group_high_tvshows['Disney+'].max()}'\n
    The Runtime Group with Lowest TV Shows Count on 'Disney+' is
'{disney_runtimes_group_low_tvshows['Runtime Group'][0]} :
'{disney_runtimes_group_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[61]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = netflix_runtimes_group_tvshows['Runtime Group'][:10], y =
netflix_runtimes_group_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = hulu_runtimes_group_tvshows['Runtime Group'][:10], y =
hulu_runtimes_group_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = prime_video_runtimes_group_tvshows['Runtime Group'][:10], y =
prime_video_runtimes_group_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1,
0])
d_ru_ax4 = sns.barplot(x = disney_runtimes_group_tvshows['Runtime Group'][:10], y =
disney_runtimes_group_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[62]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = runtimes_group_data_tvshows['Runtime Group'], y =
runtimes_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = runtimes_group_data_tvshows['Runtime Group'], y =
runtimes_group_data_tvshows['Hulu'], color = 'lightgreen')

```

```

sns.lineplot(x = runtimes_group_data_tvshows['Runtime Group'], y =
runtimes_group_data_tvshows['Prime Video'], color = 'lightblue')
sns.lineplot(x = runtimes_group_data_tvshows['Runtime Group'], y =
runtimes_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Runtime Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

```

In[63]:

```

print(f'''
    Across All Platforms Total Count of Runtime Group is
'{runtimes_group_data_tvshows['Runtime Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Netflix' is
'{netflix_runtimes_group_tvshows['Runtime Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Hulu' is '{hulu_runtimes_group_tvshows['Runtime
Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Prime Video' is
'{prime_video_runtimes_group_tvshows['Runtime Group'].unique().shape[0]}'\n
    Total Count of Runtime Group on 'Disney+' is '{disney_runtimes_group_tvshows['Runtime
Group'].unique().shape[0]}'\n
    ''')

```

In[64]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.lineplot(y = runtimes_group_data_tvshows['Runtime Group'], x =
runtimes_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_ru_ax2 = sns.lineplot(y = runtimes_group_data_tvshows['Runtime Group'], x =
runtimes_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ru_ax3 = sns.lineplot(y = runtimes_group_data_tvshows['Runtime Group'], x =
runtimes_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ru_ax4 = sns.lineplot(y = runtimes_group_data_tvshows['Runtime Group'], x =
runtimes_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[65]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = runtimes_group_data_tvshows['Runtime Group'][:10], y =
runtimes_group_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = runtimes_group_data_tvshows['Runtime Group'][:10], y =
runtimes_group_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = runtimes_group_data_tvshows['Runtime Group'][:10], y =
runtimes_group_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])

```

```

d_ru_ax4 = sns.barplot(x = runtimes_group_data_tvshows['Runtime Group'][:10], y = runtimes_group_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[66]:

```

df_screentimes_high_tvshows = df_tvshows_screentimes.sort_values(by = 'Screentime', ascending = False).reset_index()
df_screentimes_high_tvshows = df_screentimes_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_screentimes['Screentime'] == (df_tvshows_screentimes['Screentime'].max()))
# df_screentimes_high_tvshows = df_tvshows_screentimes[filter]

# highestRated_tvshows =
df_tvshows_screentimes.loc[df_tvshows_screentimes['Screentime'].idxmax()]

print('\nTV Shows with Highest Ever Screentime are : \n')
df_screentimes_high_tvshows.head(5)

```

In[67]:

```

fig = px.bar(y = df_screentimes_high_tvshows['Title'][:15],
              x = df_screentimes_high_tvshows['Screentime'][:15],
              color = df_screentimes_high_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title = 'TV Shows with Highest Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[68]:

```

df_screentimes_low_tvshows = df_tvshows_screentimes.sort_values(by = 'Screentime', ascending = True).reset_index()
df_screentimes_low_tvshows = df_screentimes_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_screentimes['Screentime'] == (df_tvshows_screentimes['Screentime'].min()))
# df_screentimes_low_tvshows = df_tvshows_screentimes[filter]

print('\nTV Shows with Lowest Ever Screentime are : \n')
df_screentimes_low_tvshows.head(5)

```

In[69]:

```

fig = px.bar(y = df_screentimes_low_tvshows['Title'][:15],
              x = df_screentimes_low_tvshows['Screentime'][:15],
              color = df_screentimes_low_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title  = 'TV Shows with Lowest Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[70]:


print(f'''
    Total '{df_tvshows_screentimes['Screentime'].unique().shape[0]}' unique Screentime s
    were Given, They were Like this,\n

{df_tvshows_screentimes.sort_values(by = 'Screentime', ascending =
False)['Screentime'].unique()}\n

    The Highest Ever Screentime Ever Any TV Show Got is
'{df_screentimes_high_tvshows['Title'][0]}':
'{df_screentimes_high_tvshows['Screentime'].max()}'\n

    The Lowest Ever Screentime Ever Any TV Show Got is
'{df_screentimes_low_tvshows['Title'][0]}':
'{df_screentimes_low_tvshows['Screentime'].min()}'\n
''')


# In[71]:


netflix_screentimes_high_tvshows =
df_screentimes_high_tvshows.loc[df_screentimes_high_tvshows['Netflix']==1].reset_index()
netflix_screentimes_high_tvshows = netflix_screentimes_high_tvshows.drop(['index'], axis = 1)

netflix_screentimes_low_tvshows =
df_screentimes_low_tvshows.loc[df_screentimes_low_tvshows['Netflix']==1].reset_index()
netflix_screentimes_low_tvshows = netflix_screentimes_low_tvshows.drop(['index'], axis = 1)

netflix_screentimes_high_tvshows.head(5)

# In[72]:


fig = px.bar(y = netflix_screentimes_high_tvshows['Title'][:15],
              x = netflix_screentimes_high_tvshows['Screentime'][:15],
              color = netflix_screentimes_high_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title  = 'TV Shows with Highest Screentime in Hours : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[73]:

```

```

fig = px.bar(y = netflix_screentimes_low_tvshows['Title'][:15],
              x = netflix_screentimes_low_tvshows['Screentime'][:15],
              color = netflix_screentimes_low_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title  = 'TV Shows with Lowest Screentime in Hours : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[74]:


hulu_screentimes_high_tvshows =
df_screentimes_high_tvshows.loc[df_screentimes_high_tvshows['Hulu']==1].reset_index()
hulu_screentimes_high_tvshows = hulu_screentimes_high_tvshows.drop(['index'], axis = 1)

hulu_screentimes_low_tvshows =
df_screentimes_low_tvshows.loc[df_screentimes_low_tvshows['Hulu']==1].reset_index()
hulu_screentimes_low_tvshows = hulu_screentimes_low_tvshows.drop(['index'], axis = 1)

hulu_screentimes_high_tvshows.head(5)

# In[75]:


fig = px.bar(y = hulu_screentimes_high_tvshows['Title'][:15],
              x = hulu_screentimes_high_tvshows['Screentime'][:15],
              color = hulu_screentimes_high_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title  = 'TV Shows with Highest Screentime in Hours : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[76]:


fig = px.bar(y = hulu_screentimes_low_tvshows['Title'][:15],
              x = hulu_screentimes_low_tvshows['Screentime'][:15],
              color = hulu_screentimes_low_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title  = 'TV Shows with Lowest Screentime in Hours : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[77]:


prime_video_screentimes_high_tvshows =
df_screentimes_high_tvshows.loc[df_screentimes_high_tvshows['Prime Video']==1].reset_index()

```

```

prime_video_screentimes_high_tvshows = prime_video_screentimes_high_tvshows.drop(['index'],
axis = 1)

prime_video_screentimes_low_tvshows =
df_screentimes_low_tvshows.loc[df_screentimes_low_tvshows['Prime Video']==1].reset_index()
prime_video_screentimes_low_tvshows = prime_video_screentimes_low_tvshows.drop(['index'],
axis = 1)

prime_video_screentimes_high_tvshows.head(5)

# In[78]:


fig = px.bar(y = prime_video_screentimes_high_tvshows['Title'][:15],
              x = prime_video_screentimes_high_tvshows['Screentime'][:15],
              color = prime_video_screentimes_high_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title = 'TV Shows with Highest Screentime in Hours : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[79]:


fig = px.bar(y = prime_video_screentimes_low_tvshows['Title'][:15],
              x = prime_video_screentimes_low_tvshows['Screentime'][:15],
              color = prime_video_screentimes_low_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
              title = 'TV Shows with Lowest Screentime in Hours : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[80]:


disney_screentimes_high_tvshows =
df_screentimes_high_tvshows.loc[df_screentimes_high_tvshows['Disney+']==1].reset_index()
disney_screentimes_high_tvshows = disney_screentimes_high_tvshows.drop(['index'], axis = 1)

disney_screentimes_low_tvshows =
df_screentimes_low_tvshows.loc[df_screentimes_low_tvshows['Disney+']==1].reset_index()
disney_screentimes_low_tvshows = disney_screentimes_low_tvshows.drop(['index'], axis = 1)

disney_screentimes_high_tvshows.head(5)

# In[81]:


fig = px.bar(y = disney_screentimes_high_tvshows['Title'][:15],
              x = disney_screentimes_high_tvshows['Screentime'][:15],
              color = disney_screentimes_high_tvshows['Screentime'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screentime : In Hours'},
```

```

        title = 'TV Shows with Highest Screenshot in Hours : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[82]:

```

fig = px.bar(y = disney_screentimes_low_tvshows['Title'][:15],
              x = disney_screentimes_low_tvshows['Screenshot'][:15],
              color = disney_screentimes_low_tvshows['Screenshot'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Screenshot : In Hours'},
              title = 'TV Shows with Lowest Screenshot in Hours : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[83]:

```

print(f'''

    The TV Show with Highest Screenshot Ever Got is
'{df_screentimes_high_tvshows['Title'][0]}':
'{df_screentimes_high_tvshows['Screenshot'].max()}'\n
    The TV Show with Lowest Screenshot Ever Got is
'{df_screentimes_low_tvshows['Title'][0]}':
'{df_screentimes_low_tvshows['Screenshot'].min()}'\n

    The TV Show with Highest Screenshot on 'Netflix' is
'{netflix_screentimes_high_tvshows['Title'][0]}':
'{netflix_screentimes_high_tvshows['Screenshot'].max()}'\n
    The TV Show with Lowest Screenshot on 'Netflix' is
'{netflix_screentimes_low_tvshows['Title'][0]}':
'{netflix_screentimes_low_tvshows['Screenshot'].min()}'\n

    The TV Show with Highest Screenshot on 'Hulu' is
'{hulu_screentimes_high_tvshows['Title'][0]}':
'{hulu_screentimes_high_tvshows['Screenshot'].max()}'\n
    The TV Show with Lowest Screenshot on 'Hulu' is
'{hulu_screentimes_low_tvshows['Title'][0]}':
'{hulu_screentimes_low_tvshows['Screenshot'].min()}'\n

    The TV Show with Highest Screenshot on 'Prime Video' is
'{prime_video_screentimes_high_tvshows['Title'][0]}':
'{prime_video_screentimes_high_tvshows['Screenshot'].max()}'\n
    The TV Show with Lowest Screenshot on 'Prime Video' is
'{prime_video_screentimes_low_tvshows['Title'][0]}':
'{prime_video_screentimes_low_tvshows['Screenshot'].min()}'\n

    The TV Show with Highest Screenshot on 'Disney+' is
'{disney_screentimes_high_tvshows['Title'][0]}':
'{disney_screentimes_high_tvshows['Screenshot'].max()}'\n
    The TV Show with Lowest Screenshot on 'Disney+' is
'{disney_screentimes_low_tvshows['Title'][0]}':
'{disney_screentimes_low_tvshows['Screenshot'].min()}'\n
''')

```

```
# In[84]:
```

```
print(f'''  
    Accross All Platforms the Average Screentime is  
'{round(df_tvshows_screentimes['Screentime'].mean(), ndigits = 2)}'\n        The Average Screentime on 'Netflix' is  
'{round(netflix_screentimes_tvshows['Screentime'].mean(), ndigits = 2)}'\n        The Average Screentime on 'Hulu' is  
'{round(hulu_screentimes_tvshows['Screentime'].mean(), ndigits = 2)}'\n        The Average Screentime on 'Prime Video' is  
'{round(prime_video_screentimes_tvshows['Screentime'].mean(), ndigits = 2)}'\n        The Average Screentime on 'Disney+' is  
'{round(disney_screentimes_tvshows['Screentime'].mean(), ndigits = 2)}'\n''')
```

```
# In[85]:
```

```
f, ax = plt.subplots(1, 2 , figsize = (20, 5))  
sns.distplot(df_tvshows_screentimes['Screentime'], bins = 20, kde = True, ax = ax[0])  
sns.boxplot(df_tvshows_screentimes['Screentime'], ax = ax[1])  
plt.show()
```

```
# In[86]:
```

```
# Defining plot size and title  
plt.figure(figsize = (20, 10))  
plt.title('Screentime s Per Platform')  
  
# Plotting the information from each dataset into a histogram  
sns.histplot(prime_video_screentimes_tvshows['Screentime'][:100], color = 'lightblue',  
legend = True, kde = True)  
sns.histplot(netflix_screentimes_tvshows['Screentime'][:100], color = 'red', legend = True,  
kde = True)  
sns.histplot(hulu_screentimes_tvshows['Screentime'][:100], color = 'lightgreen', legend =  
True, kde = True)  
sns.histplot(disney_screentimes_tvshows['Screentime'][:100], color = 'darkblue', legend =  
True, kde = True)  
  
# Setting the legend  
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])  
plt.show()
```

```
# In[87]:
```

```
def round_val(data):  
    if str(data) != 'nan':  
        return round(data)
```

```
# In[88]:
```

```
df_tvshows_screentimes_group = df_tvshows_screentimes.copy()
```

```

# In[89]:


df_tvshows_screentimes_group['Screentime Group'] =
df_tvshows_screentimes['Screentime'].apply(round_val)

screentimes_values = df_tvshows_screentimes_group['Screentime
Group'].value_counts().sort_index(ascending = False).tolist()
screentimes_index = df_tvshows_screentimes_group['Screentime
Group'].value_counts().sort_index(ascending = False).index

# screentimes_values, screentimes_index

# In[90]:


screentimes_group_count = df_tvshows_screentimes_group.groupby('Screentime
Group')['Title'].count()
screentimes_group_tvshows = df_tvshows_screentimes_group.groupby('Screentime
Group')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
screentimes_group_data_tvshows = pd.concat([screentimes_group_count,
screentimes_group_tvshows], axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows
Count'})
screentimes_group_data_tvshows = screentimes_group_data_tvshows.sort_values(by = 'TV Shows
Count', ascending = False)

# In[91]:


# Screentime Group with TV Shows Counts - All Platforms Combined
screentimes_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[92]:


screentimes_group_data_tvshows.sort_values(by = 'Screentime Group', ascending = False)

# In[93]:


fig = px.bar(y = screentimes_group_data_tvshows['TV Shows Count'],
              x = screentimes_group_data_tvshows['Screentime Group'],
              color = screentimes_group_data_tvshows['Screentime Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Screentime : In Hours'},
              title = 'TV Shows with Group Screentime in Hours : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[94]:


fig = px.pie(screentimes_group_data_tvshows[:10],
             names = screentimes_group_data_tvshows['Screentime Group'],

```

```

        values = screentimes_group_data_tvshows['TV Shows Count'],
        color = screentimes_group_data_tvshows['TV Shows Count'],
        color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                   title = 'TV Shows Count based on Screentime Group')
fig.show()

# In[95]:


df_screentimes_group_high_tvshows = screentimes_group_data_tvshows.sort_values(by = 'TV
Shows Count', ascending = False).reset_index()
df_screentimes_group_high_tvshows = df_screentimes_group_high_tvshows.drop(['index'], axis
= 1)
# filter = (screentimes_group_data_tvshows['TV Shows Count'] ==
(screentimes_group_data_tvshows['TV Shows Count'].max()))
# df_screentimes_group_high_tvshows = screentimes_group_data_tvshows[filter]

# highestRated_tvshows =
screentimes_group_data_tvshows.loc[screentimes_group_data_tvshows['TV Shows
Count'].idxmax()]

# print('\nScreentime with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_screentimes_group_high_tvshows.head(5)

# In[96]:


df_screentimes_group_low_tvshows = screentimes_group_data_tvshows.sort_values(by = 'TV
Shows Count', ascending = True).reset_index()
df_screentimes_group_low_tvshows = df_screentimes_group_low_tvshows.drop(['index'], axis
= 1)
# filter = (screentimes_group_data_tvshows['TV Shows Count'] ==
(screentimes_group_data_tvshows['TV Shows Count'].min()))
# df_screentimes_group_low_tvshows = screentimes_group_data_tvshows[filter]

# print('\nScreentime with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_screentimes_group_low_tvshows.head(5)

# In[97]:


print(f'''
    Total '{df_tvshows_screentimes['Screentime'].count()}' Titles are available on All
Platforms, out of which
        You Can Choose to see TV Shows from Total
'{screentimes_group_data_tvshows['Screentime Group'].unique().shape[0]}' Screentime Group,
They were Like this, \n

    {screentimes_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Screentime Group'].unique()} etc. \n

    The Screentime Group with Highest TV Shows Count have
'{screentimes_group_data_tvshows['TV Shows Count'].max()}' TV Shows Available is
'{df_screentimes_group_high_tvshows['Screentime Group'][0]}', &\n

```

```
The Screentime Group with Lowest TV Shows Count have
'{screentimes_group_data_tvshows['TV Shows Count'].min()}' TV Shows Available is
'{df_screentimes_group_low_tvshows['Screentime Group'][0]}'
'''')
```

```
# In[98]:
```

```
netflix_screentimes_group_tvshows =
screentimes_group_data_tvshows[screentimes_group_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_screentimes_group_tvshows = netflix_screentimes_group_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_screentimes_group_high_tvshows = df_screentimes_group_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_screentimes_group_high_tvshows =
netflix_screentimes_group_high_tvshows.drop(['index'], axis = 1)

netflix_screentimes_group_low_tvshows = df_screentimes_group_high_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_screentimes_group_low_tvshows =
netflix_screentimes_group_low_tvshows.drop(['index'], axis = 1)

netflix_screentimes_group_high_tvshows.head(5)
```

```
# In[99]:
```

```
hulu_screentimes_group_tvshows =
screentimes_group_data_tvshows[screentimes_group_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_screentimes_group_tvshows = hulu_screentimes_group_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_screentimes_group_high_tvshows = df_screentimes_group_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_screentimes_group_high_tvshows = hulu_screentimes_group_high_tvshows.drop(['index'], axis = 1)

hulu_screentimes_group_low_tvshows = df_screentimes_group_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_screentimes_group_low_tvshows = hulu_screentimes_group_low_tvshows.drop(['index'], axis = 1)

hulu_screentimes_group_high_tvshows.head(5)
```

```
# In[100]:
```

```
prime_video_screentimes_group_tvshows =
screentimes_group_data_tvshows[screentimes_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_screentimes_group_tvshows =
prime_video_screentimes_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)
```

```

prime_video_screentimes_group_high_tvshows =
df_screentimes_group_high_tvshows.sort_values(by = 'Prime Video', ascending =
False).reset_index()
prime_video_screentimes_group_high_tvshows =
prime_video_screentimes_group_high_tvshows.drop(['index'], axis = 1)

prime_video_screentimes_group_low_tvshows =
df_screentimes_group_high_tvshows.sort_values(by = 'Prime Video', ascending =
True).reset_index()
prime_video_screentimes_group_low_tvshows =
prime_video_screentimes_group_low_tvshows.drop(['index'], axis = 1)

prime_video_screentimes_group_high_tvshows.head(5)

```

In[101]:

```

disney_screentimes_group_tvshows =
screentimes_group_data_tvshows[screentimes_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_screentimes_group_tvshows = disney_screentimes_group_tvshows.drop(['index',
'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_screentimes_group_high_tvshows = df_screentimes_group_high_tvshows.sort_values(by =
'Disney+', ascending = False).reset_index()
disney_screentimes_group_high_tvshows =
disney_screentimes_group_high_tvshows.drop(['index'], axis = 1)

disney_screentimes_group_low_tvshows = df_screentimes_group_high_tvshows.sort_values(by =
'Disney+', ascending = True).reset_index()
disney_screentimes_group_low_tvshows = disney_screentimes_group_low_tvshows.drop(['index'],
axis = 1)

disney_screentimes_group_high_tvshows.head(5)

```

In[102]:

```

print(f'''
    The Screentime Group with Highest TV Shows Count Ever Got is
'{df_screentimes_group_high_tvshows['Screentime Group'][0]}' :
'{df_screentimes_group_high_tvshows['TV Shows Count'].max()}'\n
    The Screentime Group with Lowest TV Shows Count Ever Got is
'{df_screentimes_group_low_tvshows['Screentime Group'][0]}' :
'{df_screentimes_group_low_tvshows['TV Shows Count'].min()}'\n

    The Screentime Group with Highest TV Shows Count on 'Netflix' is
'{netflix_screentimes_group_high_tvshows['Screentime Group'][0]}' :
'{netflix_screentimes_group_high_tvshows['Netflix'].max()}'\n
    The Screentime Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_screentimes_group_low_tvshows['Screentime Group'][0]}' :
'{netflix_screentimes_group_low_tvshows['Netflix'].min()}'\n

    The Screentime Group with Highest TV Shows Count on 'Hulu' is
'{hulu_screentimes_group_high_tvshows['Screentime Group'][0]}' :
'{hulu_screentimes_group_high_tvshows['Hulu'].max()}'\n
    The Screentime Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_screentimes_group_low_tvshows['Screentime Group'][0]}' :
'{hulu_screentimes_group_low_tvshows['Hulu'].min()}'\n

```

```
The Screentime Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_screentimes_group_high_tvshows['Screentime Group'][0]}':
'{prime_video_screentimes_group_high_tvshows['Prime Video'].max()}\n
The Screentime Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_screentimes_group_low_tvshows['Screentime Group'][0]}':
'{prime_video_screentimes_group_low_tvshows['Prime Video'].min()}\n
```

```
The Screentime Group with Highest TV Shows Count on 'Disney+' is
'{disney_screentimes_group_high_tvshows['Screentime Group'][0]}':
'{disney_screentimes_group_high_tvshows['Disney+'].max()}\n
The Screentime Group with Lowest TV Shows Count on 'Disney+' is
'{disney_screentimes_group_low_tvshows['Screentime Group'][0]}':
'{disney_screentimes_group_low_tvshows['Disney+'].min()}\n
''')
```

```
# In[103]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_scr_ax1 = sns.barplot(x = netflix_screentimes_group_tvshows['Screentime Group'][:10], y =
netflix_screentimes_group_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_scr_ax2 = sns.barplot(x = hulu_screentimes_group_tvshows['Screentime Group'][:10], y =
hulu_screentimes_group_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_scr_ax3 = sns.barplot(x = prime_video_screentimes_group_tvshows['Screentime Group'][:10],
y = prime_video_screentimes_group_tvshows['Prime Video'][:10], palette = 'Blues_r', ax =
axes[1, 0])
d_scr_ax4 = sns.barplot(x = disney_screentimes_group_tvshows['Screentime Group'][:10], y =
disney_screentimes_group_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_scr_ax1.title.set_text(labels[0])
h_scr_ax2.title.set_text(labels[1])
p_scr_ax3.title.set_text(labels[2])
d_scr_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[104]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = screentimes_group_data_tvshows['Screentime Group'], y =
screentimes_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = screentimes_group_data_tvshows['Screentime Group'], y =
screentimes_group_data_tvshows['Hulu'], color = 'lightgreen')
sns.lineplot(x = screentimes_group_data_tvshows['Screentime Group'], y =
screentimes_group_data_tvshows['Prime Video'], color = 'lightblue')
sns.lineplot(x = screentimes_group_data_tvshows['Screentime Group'], y =
screentimes_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Screentime Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()
```

```
# In[105]:
```

```
print(f'''  
    Accross All Platforms Total Count of Screentime Group is  
'{screentimes_group_data_tvshows['Screentime Group'].unique().shape[0]}'\n  
    Total Count of Screentime Group on 'Netflix' is  
'{netflix_screentimes_group_tvshows['Screentime Group'].unique().shape[0]}'\n  
    Total Count of Screentime Group on 'Hulu' is  
'{hulu_screentimes_group_tvshows['Screentime Group'].unique().shape[0]}'\n  
    Total Count of Screentime Group on 'Prime Video' is  
'{prime_video_screentimes_group_tvshows['Screentime Group'].unique().shape[0]}'\n  
    Total Count of Screentime Group on 'Disney+' is  
'{disney_screentimes_group_tvshows['Screentime Group'].unique().shape[0]}'\n''')
```

```
# In[106]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))  
  
n_scr_ax1 = sns.lineplot(y = screentimes_group_data_tvshows['Screentime Group'], x =  
screentimes_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])  
h_scr_ax2 = sns.lineplot(y = screentimes_group_data_tvshows['Screentime Group'], x =  
screentimes_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])  
p_scr_ax3 = sns.lineplot(y = screentimes_group_data_tvshows['Screentime Group'], x =  
screentimes_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])  
d_scr_ax4 = sns.lineplot(y = screentimes_group_data_tvshows['Screentime Group'], x =  
screentimes_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_scr_ax1.title.set_text(labels[0])  
h_scr_ax2.title.set_text(labels[1])  
p_scr_ax3.title.set_text(labels[2])  
d_scr_ax4.title.set_text(labels[3])  
  
plt.show()
```

```
# In[107]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))  
  
n_ru_ax1 = sns.barplot(x = screentimes_group_data_tvshows['Screentime Group'][:10], y =  
screentimes_group_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])  
h_ru_ax2 = sns.barplot(x = screentimes_group_data_tvshows['Screentime Group'][:10], y =  
screentimes_group_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])  
p_ru_ax3 = sns.barplot(x = screentimes_group_data_tvshows['Screentime Group'][:10], y =  
screentimes_group_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])  
d_ru_ax4 = sns.barplot(x = screentimes_group_data_tvshows['Screentime Group'][:10], y =  
screentimes_group_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])  
  
labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']  
  
n_ru_ax1.title.set_text(labels[0])  
h_ru_ax2.title.set_text(labels[1])  
p_ru_ax3.title.set_text(labels[2])  
d_ru_ax4.title.set_text(labels[3])  
  
plt.show()
```

otttvshows_season.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Loacal Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")


# In[4]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')
```

```

df_tvshows.head()

# In[5]:


# profile = ProfileReport(df_tvshows)
# profile

# In[6]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('*'*25)
    print('Colums Names : \n', df.columns)
    print('*'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('*'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('*'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('*'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()

# In[7]:


data_investigate(df_tvshows)

# In[8]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)

```

```

# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)

```

```
df_tvshows.drop_duplicates(inplace = True)

# In[9]:  
  
data_investigate(df_tvshows)  
  
# In[10]:  
  
df_tvshows.head()  
  
# In[11]:  
  
df_tvshows.describe()  
  
# In[12]:  
  
df_tvshows.corr()  
  
# In[13]:  
  
# df_tvshows.sort_values('Year', ascending = True)  
# df_tvshows.sort_values('Seasons', ascending = False)  
  
# In[14]:  
  
# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',  
index = False)  
  
# path = '/content/drive/MyDrive/Files/'  
  
# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')  
  
# udf_tvshows  
  
# In[15]:  
  
# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]  
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]  
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]  
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]  
  
# In[16]:  
  
df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
```

```
df_hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1)
& (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df_prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu']
== 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df_disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] ==
0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]
```

```
# In[17]:
```

```
df_tvshows_season = df_tvshows.copy()
```

```
# In[18]:
```

```
df_tvshows_season.loc[df_tvshows_season['Seasons'] == "NA"] = 1
df_tvshows_season.drop(df_tvshows_season.loc[df_tvshows_season['Seasons'] == "NA"].index,
inplace = True)
# df_tvshows_season = df_tvshows_season[df_tvshows_season.Seasons != "NA"]
df_tvshows_season['Seasons'] = df_tvshows_season['Seasons'].astype(int)
```

```
# In[19]:
```

```
# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_season_tvshows = df_tvshows_season.loc[df_tvshows_season['Netflix'] == 1]
hulu_season_tvshows = df_tvshows_season.loc[df_tvshows_season['Hulu'] == 1]
prime_video_season_tvshows = df_tvshows_season.loc[df_tvshows_season['Prime Video'] == 1]
disney_season_tvshows = df_tvshows_season.loc[df_tvshows_season['Disney+'] == 1]
```

```
# In[20]:
```

```
df_tvshows_season_group = df_tvshows_season.copy()
```

```
# In[21]:
```

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_season.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[22]:
```

```

df_season_high_tvshows = df_tvshows_season.sort_values(by = 'Seasons', ascending = False).reset_index()
df_season_high_tvshows = df_season_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_season['Seasons'] == (df_tvshows_season['Seasons'].max()))
# df_season_high_tvshows = df_tvshows_season[filter]

# highestRated_tvshows = df_tvshows_season.loc[df_tvshows_season['Seasons'].idxmax()]

print('\nTV Shows with Highest Ever Seasons are : \n')
df_season_high_tvshows.head(5)

```

In[23]:

```

fig = px.bar(y = df_season_high_tvshows['Title'][:15],
              x = df_season_high_tvshows['Seasons'][:15],
              color = df_season_high_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Highest Seasons in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[24]:

```

df_season_low_tvshows = df_tvshows_season.sort_values(by = 'Seasons', ascending = True).reset_index()
df_season_low_tvshows = df_season_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_season['Seasons'] == (df_tvshows_season['Seasons'].min()))
# df_season_low_tvshows = df_tvshows_season[filter]

print('\nTV Shows with Lowest Ever Seasons are : \n')
df_season_low_tvshows.head(5)

```

In[25]:

```

fig = px.bar(y = df_season_low_tvshows['Title'][:15],
              x = df_season_low_tvshows['Seasons'][:15],
              color = df_season_low_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Lowest Seasons in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[26]:

```

print(f'''
    Total '{df_tvshows_season['Seasons'].unique().shape[0]}' unique Seasons s were Given,
They were Like this,\n

```

```

{df_tvshows_season.sort_values(by = 'Seasons', ascending = False)['Seasons'].unique()}\n\n
The Highest Ever Seasons Ever Any TV Show Got is
'{df_season_high_tvshows['Title'][0]}' : '{df_season_high_tvshows['Seasons'].max()}'\n\n
The Lowest Ever Seasons Ever Any TV Show Got is '{df_season_low_tvshows['Title'][0]}'
: '{df_season_low_tvshows['Seasons'].min()}'\n
''')

# In[27]:\n\n

netflix_season_high_tvshows =
df_season_high_tvshows.loc[df_season_high_tvshows['Netflix']==1].reset_index()
netflix_season_high_tvshows = netflix_season_high_tvshows.drop(['index'], axis = 1)

netflix_season_low_tvshows =
df_season_low_tvshows.loc[df_season_low_tvshows['Netflix']==1].reset_index()
netflix_season_low_tvshows = netflix_season_low_tvshows.drop(['index'], axis = 1)

netflix_season_high_tvshows.head(5)

# In[28]:\n\n

fig = px.bar(y = netflix_season_high_tvshows['Title'][:15],
              x = netflix_season_high_tvshows['Seasons'][:15],
              color = netflix_season_high_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Highest Seasons in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[29]:\n\n

fig = px.bar(y = netflix_season_low_tvshows['Title'][:15],
              x = netflix_season_low_tvshows['Seasons'][:15],
              color = netflix_season_low_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Lowest Seasons in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[30]:\n\n

hulu_season_high_tvshows =
df_season_high_tvshows.loc[df_season_high_tvshows['Hulu']==1].reset_index()
hulu_season_high_tvshows = hulu_season_high_tvshows.drop(['index'], axis = 1)

hulu_season_low_tvshows =
df_season_low_tvshows.loc[df_season_low_tvshows['Hulu']==1].reset_index()

```

```
hulu_season_low_tvshows = hulu_season_low_tvshows.drop(['index'], axis = 1)

hulu_season_high_tvshows.head(5)
```

```
# In[31]:
```

```
fig = px.bar(y = hulu_season_high_tvshows['Title'][:15],
              x = hulu_season_high_tvshows['Seasons'][:15],
              color = hulu_season_high_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Highest Seasons in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[32]:
```

```
fig = px.bar(y = hulu_season_low_tvshows['Title'][:15],
              x = hulu_season_low_tvshows['Seasons'][:15],
              color = hulu_season_low_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Lowest Seasons in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[33]:
```

```
prime_video_season_high_tvshows = df_season_high_tvshows.loc[df_season_high_tvshows['Prime
Video']==1].reset_index()
prime_video_season_high_tvshows = prime_video_season_high_tvshows.drop(['index'], axis = 1)

prime_video_season_low_tvshows = df_season_low_tvshows.loc[df_season_low_tvshows['Prime
Video']==1].reset_index()
prime_video_season_low_tvshows = prime_video_season_low_tvshows.drop(['index'], axis = 1)

prime_video_season_high_tvshows.head(5)
```

```
# In[34]:
```

```
fig = px.bar(y = prime_video_season_high_tvshows['Title'][:15],
              x = prime_video_season_high_tvshows['Seasons'][:15],
              color = prime_video_season_high_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Highest Seasons in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[35]:
```

```
fig = px.bar(y = prime_video_season_low_tvshows['Title'][:15],
              x = prime_video_season_low_tvshows['Seasons'][:15],
              color = prime_video_season_low_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Lowest Seasons in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[36]:
```

```
disney_season_high_tvshows =
df_season_high_tvshows.loc[df_season_high_tvshows['Disney+']==1].reset_index()
disney_season_high_tvshows = disney_season_high_tvshows.drop(['index'], axis = 1)

disney_season_low_tvshows =
df_season_low_tvshows.loc[df_season_low_tvshows['Disney+']==1].reset_index()
disney_season_low_tvshows = disney_season_low_tvshows.drop(['index'], axis = 1)

disney_season_high_tvshows.head(5)
```

```
# In[37]:
```

```
fig = px.bar(y = disney_season_high_tvshows['Title'][:15],
              x = disney_season_high_tvshows['Seasons'][:15],
              color = disney_season_high_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Highest Seasons in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[38]:
```

```
fig = px.bar(y = disney_season_low_tvshows['Title'][:15],
              x = disney_season_low_tvshows['Seasons'][:15],
              color = disney_season_low_tvshows['Seasons'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Lowest Seasons in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[39]:
```

```
print(f'''
```

```

    The TV Show with Highest Seasons Ever Got is '{df_season_high_tvshows['Title'][0]}'
: '{df_season_high_tvshows['Seasons'].max()}'\n
    The TV Show with Lowest Seasons Ever Got is '{df_season_low_tvshows['Title'][0]} :
'{df_season_low_tvshows['Seasons'].min()}'\n

    The TV Show with Highest Seasons on 'Netflix' is
'{netflix_season_high_tvshows['Title'][0]}' :
'{netflix_season_high_tvshows['Seasons'].max()}'\n
    The TV Show with Lowest Seasons on 'Netflix' is
'{netflix_season_low_tvshows['Title'][0]}' :
'{netflix_season_low_tvshows['Seasons'].min()}'\n

    The TV Show with Highest Seasons on 'Hulu' is
'{hulu_season_high_tvshows['Title'][0]}' : '{hulu_season_high_tvshows['Seasons'].max()}'\n
    The TV Show with Lowest Seasons on 'Hulu' is '{hulu_season_low_tvshows['Title'][0]}'
: '{hulu_season_low_tvshows['Seasons'].min()}'\n

    The TV Show with Highest Seasons on 'Prime Video' is
'{prime_video_season_high_tvshows['Title'][0]}' :
'{prime_video_season_high_tvshows['Seasons'].max()}'\n
    The TV Show with Lowest Seasons on 'Prime Video' is
'{prime_video_season_low_tvshows['Title'][0]}' :
'{prime_video_season_low_tvshows['Seasons'].min()}'\n

    The TV Show with Highest Seasons on 'Disney+' is
'{disney_season_high_tvshows['Title'][0]}' :
'{disney_season_high_tvshows['Seasons'].max()}'\n
    The TV Show with Lowest Seasons on 'Disney+' is
'{disney_season_low_tvshows['Title'][0]}' :
'{disney_season_low_tvshows['Seasons'].min()}'\n
    ''')

```

In[40]:

```

print(f'''
    Accross All Platforms the Average Seasons is
'{round(df_tvshows_season['Seasons'].mean(), ndigits = 2)}'\n
    The Average Seasons on 'Netflix' is
'{round(netflix_season_tvshows['Seasons'].mean(), ndigits = 2)}'\n
    The Average Seasons on 'Hulu' is '{round(hulu_season_tvshows['Seasons'].mean(),
ndigits = 2)}'\n
    The Average Seasons on 'Prime Video' is
'{round(prime_video_season_tvshows['Seasons'].mean(), ndigits = 2)}'\n
    The Average Seasons on 'Disney+' is '{round(disney_season_tvshows['Seasons'].mean(),
ndigits = 2)}'\n
    ''')

```

In[41]:

```

f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_season['Seasons'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_season['Seasons'], ax = ax[1])
plt.show()

```

In[42]:

```

# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Seasons s Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_season_tvshows['Seasons'][:100], color = 'lightblue', legend =
True, kde = True)
sns.histplot(netflix_season_tvshows['Seasons'][:100], color = 'red', legend = True, kde =
True)
sns.histplot(hulu_season_tvshows['Seasons'][:100], color = 'lightgreen', legend = True, kde =
True)
sns.histplot(disney_season_tvshows['Seasons'][:100], color = 'darkblue', legend = True, kde =
True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()

```

In[43]:

```

def round_val(data):
    if str(data) != 'nan':
        return round(data)

```

In[44]:

```

df_tvshows_season_group['Seasons Group'] = df_tvshows_season['Seasons'].apply(round_val)

season_values = df_tvshows_season_group['Seasons
Group'].value_counts().sort_index(ascending = False).tolist()
season_index = df_tvshows_season_group['Seasons Group'].value_counts().sort_index(ascending =
False).index

# season_values, season_index

```

In[45]:

```

season_group_count = df_tvshows_season_group.groupby('Seasons Group')['Title'].count()
season_group_tvshows = df_tvshows_season_group.groupby('Seasons Group')[['Netflix', 'Hulu',
'Prime Video', 'Disney+']].sum()
season_group_data_tvshows = pd.concat([season_group_count, season_group_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
season_group_data_tvshows = season_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)

```

In[46]:

```

# Seasons Group with TV Shows Counts - All Platforms Combined
season_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

```

In[47]:

```
season_group_data_tvshows.sort_values(by = 'Seasons Group', ascending = False)
```

```
# In[48]:
```

```
fig = px.bar(y = season_group_data_tvshows['TV Shows Count'],
              x = season_group_data_tvshows['Seasons Group'],
              color = season_group_data_tvshows['Seasons Group'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Seasons : In Minutes'},
              title = 'TV Shows with Group Seasons in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()
```

```
# In[49]:
```

```
fig = px.pie(season_group_data_tvshows[:10],
              names = season_group_data_tvshows['Seasons Group'][:10],
              values = season_group_data_tvshows['TV Shows Count'][:10],
              color = season_group_data_tvshows['TV Shows Count'][:10],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'TV Shows Count based on Seasons Group')
fig.show()
```

```
# In[50]:
```

```
df_season_group_high_tvshows = season_group_data_tvshows.sort_values(by = 'TV Shows Count',
                                                                    ascending = False).reset_index()
df_season_group_high_tvshows = df_season_group_high_tvshows.drop(['index'], axis = 1)
# filter = (season_group_data_tvshows['TV Shows Count'] == (season_group_data_tvshows['TV Shows Count'].max()))
# df_season_group_high_tvshows = season_group_data_tvshows[filter]

# highestRated_tvshows = season_group_data_tvshows.loc[season_group_data_tvshows['TV Shows Count'].idxmax()]

# print('\nSeasons with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_season_group_high_tvshows.head(5)
```

```
# In[51]:
```

```
df_season_group_low_tvshows = season_group_data_tvshows.sort_values(by = 'TV Shows Count',
                                                                    ascending = True).reset_index()
df_season_group_low_tvshows = df_season_group_low_tvshows.drop(['index'], axis = 1)
# filter = (season_group_data_tvshows['TV Shows Count'] == (season_group_data_tvshows['TV Shows Count'].min()))
# df_season_group_low_tvshows = season_group_data_tvshows[filter]

# print('\nSeasons with Lowest Ever TV Shows Count are : All Platforms Combined\n')
```

```

df_season_group_low_tvshows.head(5)

# In[52]:


print(f'''
    Total '{df_tvshows_season['Seasons'].count()}' Titles are available on All Platforms,
out of which\n
    You Can Choose to see TV Shows from Total '{season_group_data_tvshows['Seasons
Group'].unique().shape[0]}' Seasons Group, They were Like this, \n
    {season_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
False)['Seasons Group'].unique()} etc. \n
    The Seasons Group with Highest TV Shows Count have '{season_group_data_tvshows['TV
Shows Count'].max()}' TV Shows Available is '{df_season_group_high_tvshows['Seasons
Group'][0]}', &\n
    The Seasons Group with Lowest TV Shows Count have '{season_group_data_tvshows['TV
Shows Count'].min()}' TV Shows Available is '{df_season_group_low_tvshows['Seasons
Group'][0]}'
    ''')

```

```

# In[53]:


netflix_season_group_tvshows =
season_group_data_tvshows[season_group_data_tvshows['Netflix'] != 0].sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_season_group_tvshows = netflix_season_group_tvshows.drop(['index', 'Hulu', 'Prime
Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_season_group_high_tvshows = df_season_group_high_tvshows.sort_values(by =
'Netflix', ascending = False).reset_index()
netflix_season_group_high_tvshows = netflix_season_group_high_tvshows.drop(['index'], axis =
1)

netflix_season_group_low_tvshows = df_season_group_low_tvshows.sort_values(by = 'Netflix',
ascending = True).reset_index()
netflix_season_group_low_tvshows = netflix_season_group_low_tvshows.drop(['index'], axis =
1)

netflix_season_group_high_tvshows.head(5)

# In[54]:


hulu_season_group_tvshows = season_group_data_tvshows[season_group_data_tvshows['Hulu'] !=
0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_season_group_tvshows = hulu_season_group_tvshows.drop(['index', 'Netflix', 'Prime
Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_season_group_high_tvshows = df_season_group_high_tvshows.sort_values(by = 'Hulu',
ascending = False).reset_index()
hulu_season_group_high_tvshows = hulu_season_group_high_tvshows.drop(['index'], axis = 1)

hulu_season_group_low_tvshows = df_season_group_low_tvshows.sort_values(by = 'Hulu',
ascending = True).reset_index()
hulu_season_group_low_tvshows = hulu_season_group_low_tvshows.drop(['index'], axis = 1)

```

```
hulu_season_group_high_tvshows.head(5)
```

```
# In[55]:
```

```
prime_video_season_group_tvshows =
season_group_data_tvshows[season_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_season_group_tvshows = prime_video_season_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_season_group_high_tvshows = df_season_group_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_season_group_high_tvshows =
prime_video_season_group_high_tvshows.drop(['index'], axis = 1)

prime_video_season_group_low_tvshows = df_season_group_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_season_group_low_tvshows = prime_video_season_group_low_tvshows.drop(['index'], axis = 1)

prime_video_season_group_high_tvshows.head(5)
```

```
# In[56]:
```

```
disney_season_group_tvshows =
season_group_data_tvshows[season_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_season_group_tvshows = disney_season_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_season_group_high_tvshows = df_season_group_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_season_group_high_tvshows = disney_season_group_high_tvshows.drop(['index'], axis = 1)

disney_season_group_low_tvshows = df_season_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_season_group_low_tvshows = disney_season_group_low_tvshows.drop(['index'], axis = 1)

disney_season_group_high_tvshows.head(5)
```

```
# In[57]:
```

```
print(f'''
    The Seasons Group with Highest TV Shows Count Ever Got is
'{df_season_group_high_tvshows['Seasons Group'][0]}' : '{df_season_group_high_tvshows['TV Shows Count'].max()}'\n
    The Seasons Group with Lowest TV Shows Count Ever Got is
'{df_season_group_low_tvshows['Seasons Group'][0]}' : '{df_season_group_low_tvshows['TV Shows Count'].min()}'\n

    The Seasons Group with Highest TV Shows Count on 'Netflix' is
'{netflix_season_group_high_tvshows['Seasons Group'][0]}' :
'{netflix_season_group_high_tvshows['Netflix'].max()}'\n
```

```

The Seasons Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_season_group_low_tvshows['Seasons Group'][0]} :
'{netflix_season_group_low_tvshows['Netflix'].min()}'\n

The Seasons Group with Highest TV Shows Count on 'Hulu' is
'{hulu_season_group_high_tvshows['Seasons Group'][0]} :
'{hulu_season_group_high_tvshows['Hulu'].max()}'\n
The Seasons Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_season_group_low_tvshows['Seasons Group'][0]} :
'{hulu_season_group_low_tvshows['Hulu'].min()}'\n

The Seasons Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_season_group_high_tvshows['Seasons Group'][0]} :
'{prime_video_season_group_high_tvshows['Prime Video'].max()}'\n
The Seasons Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_season_group_low_tvshows['Seasons Group'][0]} :
'{prime_video_season_group_low_tvshows['Prime Video'].min()}'\n

The Seasons Group with Highest TV Shows Count on 'Disney+' is
'{disney_season_group_high_tvshows['Seasons Group'][0]} :
'{disney_season_group_high_tvshows['Disney+'].max()}'\n
The Seasons Group with Lowest TV Shows Count on 'Disney+' is
'{disney_season_group_low_tvshows['Seasons Group'][0]} :
'{disney_season_group_low_tvshows['Disney+'].min()}'\n
''')

```

In[58]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = netflix_season_group_tvshows['Seasons Group'][:10], y =
netflix_season_group_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = hulu_season_group_tvshows['Seasons Group'][:10], y =
hulu_season_group_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = prime_video_season_group_tvshows['Seasons Group'][:10], y =
prime_video_season_group_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = disney_season_group_tvshows['Seasons Group'][:10], y =
disney_season_group_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

In[59]:

```

plt.figure(figsize = (20, 5))
sns.lineplot(x = season_group_data_tvshows['Seasons Group'], y =
season_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = season_group_data_tvshows['Seasons Group'], y =
season_group_data_tvshows['Hulu'], color = 'lightgreen')
sns.lineplot(x = season_group_data_tvshows['Seasons Group'], y =
season_group_data_tvshows['Prime Video'], color = 'lightblue')

```

```

sns.lineplot(x = season_group_data_tvshows['Seasons Group'], y =
season_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Seasons Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

# In[60]:


print(f'''
    Across All Platforms Total Count of Seasons Group is
'{season_group_data_tvshows['Seasons Group'].unique().shape[0]}'\n
    Total Count of Seasons Group on 'Netflix' is '{netflix_season_group_tvshows['Seasons
Group'].unique().shape[0]}'\n
    Total Count of Seasons Group on 'Hulu' is '{hulu_season_group_tvshows['Seasons
Group'].unique().shape[0]}'\n
    Total Count of Seasons Group on 'Prime Video' is
'{prime_video_season_group_tvshows['Seasons Group'].unique().shape[0]}'\n
    Total Count of Seasons Group on 'Disney+' is '{disney_season_group_tvshows['Seasons
Group'].unique().shape[0]}'
''')


# In[61]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.lineplot(y = season_group_data_tvshows['Seasons Group'], x =
season_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_ru_ax2 = sns.lineplot(y = season_group_data_tvshows['Seasons Group'], x =
season_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ru_ax3 = sns.lineplot(y = season_group_data_tvshows['Seasons Group'], x =
season_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ru_ax4 = sns.lineplot(y = season_group_data_tvshows['Seasons Group'], x =
season_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

# In[62]:


fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = season_group_data_tvshows['Seasons Group'][:10], y =
season_group_data_tvshows['Netflix'][:10], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = season_group_data_tvshows['Seasons Group'][:10], y =
season_group_data_tvshows['Hulu'][:10], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = season_group_data_tvshows['Seasons Group'][:10], y =
season_group_data_tvshows['Prime Video'][:10], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = season_group_data_tvshows['Seasons Group'][:10], y =
season_group_data_tvshows['Disney+'][:10], palette = 'BuPu_r', ax = axes[1, 1])

```

```

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

otttvshows_title.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask


# In[2]:


# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')


# In[3]:


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots

```

```

from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")

# In[4]:


nltk.download('all')


# In[5]:


# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()


# In[6]:


# profile = ProfileReport(df_tvshows)
# profile


# In[7]:


def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Columns Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
    print(c)
    print('***'*25)
    print('Missing vaules %age wise :\n')
    print((100*(df.isnull().sum()/len(df.index))))
    print('***'*25)
    print('Pictorial Representation : ')
    plt.figure(figsize = (10, 10))
    sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
    plt.show()


# In[8]:


data_investigate(df_tvshows)

```

```

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind

```

```

# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_tvshows)

# In[11]:
df_tvshows.head()

# In[12]:
df_tvshows.describe()

# In[13]:
df_tvshows.corr()

# In[14]:
# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:
# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

```

```

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_title = df_tvshows.copy()


# In[19]:


df_tvshows_title.drop(df_tvshows_title.loc[df_tvshows_title['Title'] == "NA"].index,
inplace = True)
# df_tvshows_title = df_tvshows_title[df_tvshows_title.Title != "NA"]


# In[20]:


# Creating distinct dataframes only with the tvshows present on individual streaming
platforms
netflix_title_tvshows = df_tvshows_title.loc[df_tvshows_title['Netflix'] == 1]
hulu_title_tvshows = df_tvshows_title.loc[df_tvshows_title['Hulu'] == 1]
prime_video_title_tvshows = df_tvshows_title.loc[df_tvshows_title['Prime Video'] == 1]
disney_title_tvshows = df_tvshows_title.loc[df_tvshows_title['Disney+'] == 1]


# In[21]:


plt.figure(figsize = (10, 10))
corr = df_tvshows_title.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))

```

```
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[22]:
```

```
df_tvshows_title = df_tvshows_title['Title']
tvshows_title_w = ' '.join(df_tvshows_title)
```

```
# In[23]:
```

```
stopwords = set(STOPWORDS)

wordcloud_all_title_tvshows = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10).generate(tvshows_title_w)

# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_all_title_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

```
# In[24]:
```

```
tvshows_title_w = tvshows_title_w.lower()

stop_words_english_tvshows = set(STOPWORDS)

word_tokens_english_tvshows = word_tokenize(tvshows_title_w)

filtered_tvshow_title = [w for w in word_tokens_english_tvshows if not w in
stop_words_english_tvshows]

filtered_tvshow_title = " ".join(filtered_tvshow_title)

filtered_tvshow_title = re.sub("\s", ' ', filtered_tvshow_title)

filtered_tvshow_title = re.sub(r'[0-9]+', ' ', filtered_tvshow_title)

final_tvshow_title = re.sub(r'[^w\s]', ' ', filtered_tvshow_title)

title_tvshows_corpus_len = len(filtered_tvshow_title.split())
title_tvshows_corpus_len
```

```

# In[25]:


def extract_ngrams(data, num):
    n_grams = ngrams(nltk.word_tokenize(data), num)
    return [ ' '.join(grams) for grams in n_grams]

# In[26]:


title_ngram1_tvshows = FreqDist()

title_ngram1 = extract_ngrams(final_tvshow_title[:title_tvshows_corpus_len], 1)

for word in title_ngram1:
    title_ngram1_tvshows[word.lower()] += 1

# In[27]:


title_ngram1_tvshows.most_common(10)

# In[28]:


title_ngram2_tvshows = FreqDist()

title_ngram2 = extract_ngrams(final_tvshow_title[:title_tvshows_corpus_len], 2)

for word in title_ngram2:
    title_ngram2_tvshows[word.lower()] += 1

# In[29]:


title_ngram2_tvshows.most_common(10)

# In[30]:


title_ngram3_tvshows = FreqDist()

title_ngram3 = extract_ngrams(final_tvshow_title[:title_tvshows_corpus_len], 3)

for word in title_ngram3:
    title_ngram3_tvshows[word.lower()] += 1

# In[31]:


title_ngram3_tvshows.most_common(10)

# In[32]:

```

```

title_ngram4_tvshows = FreqDist()

title_ngram4 = extract_ngrams(final_tvshow_title[:title_tvshows_corpus_len], 4)

for word in title_ngram4:
    title_ngram4_tvshows[word.lower()] += 1

# In[33]: 

title_ngram4_tvshows.most_common(10)

# In[34]: 

title_ngram5_tvshows = FreqDist()

title_ngram5 = extract_ngrams(final_tvshow_title[:title_tvshows_corpus_len], 5)

for word in title_ngram5:
    title_ngram5_tvshows[word.lower()] += 1

# In[35]: 

title_ngram5_tvshows.most_common(10)

# In[36]: 

# Netflix Wordcloud
netflix_title_tvshows_t = netflix_title_tvshows['Title']
netflix_tvshows_title_w = ' '.join(netflix_title_tvshows_t)

# In[37]: 

stopwords = set(STOPWORDS)

wordcloud.netflix_title_tvshows = WordCloud(width = 1000, height = 500,
                                             background_color ='white',
                                             stopwords = stopwords,
                                             min_font_size = 10
                                             ).generate(netflix_tvshows_title_w)

print('\nThe Wordcloud Generated from Titles of Netflix is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud.netflix_title_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

```

```
# In[38]:
```

```
# Hulu Wordcloud
hulu_title_tvshows_t = hulu_title_tvshows['Title']
hulu_tvshows_title_w = ' '.join(hulu_title_tvshows_t)
```

```
# In[39]:
```

```
stopwords = set(STOPWORDS)

wordcloud_hulu_title_tvshows = WordCloud(width = 1000, height = 500,
                                         background_color ='white',
                                         stopwords = stopwords,
                                         min_font_size = 10
                                         ).generate(hulu_tvshows_title_w)

print('\nThe Wordcloud Generated from Titles of Hulu is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_hulu_title_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

```
# In[40]:
```

```
# Prime Video Wordcloud
prime_video_title_tvshows_t = prime_video_title_tvshows['Title']
prime_video_tvshows_title_w = ' '.join(prime_video_title_tvshows_t)
```

```
# In[41]:
```

```
stopwords = set(STOPWORDS)

wordcloud_prime_video_title_tvshows = WordCloud(width = 1000, height = 500,
                                                 background_color ='white',
                                                 stopwords = stopwords,
                                                 min_font_size = 10
                                                 ).generate(prime_video_tvshows_title_w)

print('\nThe Wordcloud Generated from Titles of Prime Video is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_prime_video_title_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

```
# In[42]:
```

```

# Disney+ Wordcloud
disney_title_tvshows_t = disney_title_tvshows['Title']
disney_tvshows_title_w = ' '.join(disney_title_tvshows_t)

# In[43]:


stopwords = set(STOPWORDS)

wordcloud_disney_title_tvshows = WordCloud(width = 1000, height = 500,
                                             background_color ='white',
                                             stopwords = stopwords,
                                             min_font_size = 10
                                           ).generate(disney_tvshows_title_w)

print('\nThe Wordcloud Generated from Titles of Disney+ is : \n')
# plot the WordCloud image
plt.figure(figsize = (20, 10), facecolor = None)
plt.imshow(wordcloud_disney_title_tvshows)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()

```

otttvshows_year.ipynb

```

#!/usr/bin/env python
# coding: utf-8

# In[1]:


# !pip install git+https://github.com/alberanid/imdbpy
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install seaborn
# !pip install pandas_profiling --upgrade
# !pip install plotly
# !pip install wordcloud
# !pip install Flask

```

```
# In[2]:
```

```

# Import Dataset
# Import File from Local Drive
# from google.colab import files
# data_to_load = files.upload()
# from google.colab import drive
# drive.mount('/content/drive')

```

```
# In[3]:
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import collections
import plotly.express as px
import plotly.graph_objects as go
import nltk
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.util import ngrams
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
from wordcloud import WordCloud, STOPWORDS
from pandas_profiling import ProfileReport
get_ipython().run_line_magic('matplotlib', 'inline')
warnings.filterwarnings("ignore")
```

```
# In[4]:
```

```
nltk.download('all')
```

```
# In[5]:
```

```
# path = '/content/drive/MyDrive/Files/'

path = 'C:\\\\Users\\\\pawan\\\\OneDrive\\\\Desktop\\\\ott\\\\Data\\\\'

df_tvshows = pd.read_csv(path + 'otttvshows.csv')

df_tvshows.head()
```

```
# In[6]:
```

```
# profile = ProfileReport(df_tvshows)
# profile
```

```
# In[7]:
```

```
def data_investigate(df):
    print('No of Rows : ', df.shape[0])
    print('No of Coloums : ', df.shape[1])
    print('***'*25)
    print('Colums Names : \n', df.columns)
    print('***'*25)
    print('Datatype of Columns : \n', df.dtypes)
    print('***'*25)
    print('Missing Values : ')
    c = df.isnull().sum()
    c = c[c > 0]
```

```

print(c)
print('***25)
print('Missing values %age wise :\n')
print((100*(df.isnull().sum()/len(df.index))))
print('***25)
print('Pictorial Representation : ')
plt.figure(figsize = (10, 10))
sns.heatmap(df.isnull(), yticklabels = False, cbar = False)
plt.show()

# In[8]:


data_investigate(df_tvshows)

# In[9]:


# ID
# df_tvshows = df_tvshows.drop(['ID'], axis = 1)

# Age
df_tvshows.loc[df_tvshows['Age'].isnull() & df_tvshows['Disney+'] == 1, "Age"] = '13'
# df_tvshows.fillna({'Age' : 18}, inplace = True)
df_tvshows.fillna({'Age' : 'NR'}, inplace = True)
df_tvshows['Age'].replace({'all': '0'}, inplace = True)
df_tvshows['Age'].replace({'7+': '7'}, inplace = True)
df_tvshows['Age'].replace({'13+': '13'}, inplace = True)
df_tvshows['Age'].replace({'16+': '16'}, inplace = True)
df_tvshows['Age'].replace({'18+': '18'}, inplace = True)
# df_tvshows['Age'] = df_tvshows['Age'].astype(int)

# IMDb
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].mean()}, inplace = True)
# df_tvshows.fillna({'IMDb' : df_tvshows['IMDb'].median()}, inplace = True)
df_tvshows.fillna({'IMDb' : "NA"}, inplace = True)

# Rotten Tomatoes
df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].str.replace('%', '').astype(int)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'][df_tvshows['Rotten Tomatoes'].notnull()].astype(int)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].mean()}, inplace = True)
# df_tvshows.fillna({'Rotten Tomatoes' : df_tvshows['Rotten Tomatoes'].median()}, inplace = True)
# df_tvshows['Rotten Tomatoes'] = df_tvshows['Rotten Tomatoes'].astype(int)
df_tvshows.fillna({'Rotten Tomatoes' : "NA"}, inplace = True)

# Directors
# df_tvshows = df_tvshows.drop(['Directors'], axis = 1)
df_tvshows.fillna({'Directors' : "NA"}, inplace = True)

# Cast
df_tvshows.fillna({'Cast' : "NA"}, inplace = True)

# Genres
df_tvshows.fillna({'Genres': "NA"}, inplace = True)

```

```

# Country
df_tvshows.fillna({'Country': "NA"}, inplace = True)

# Language
df_tvshows.fillna({'Language': "NA"}, inplace = True)

# Plotline
df_tvshows.fillna({'Plotline': "NA"}, inplace = True)

# Runtime
# df_tvshows.fillna({'Runtime' : df_tvshows['Runtime'].mean()}, inplace = True)
# df_tvshows['Runtime'] = df_tvshows['Runtime'].astype(int)
df_tvshows.fillna({'Runtime' : "NA"}, inplace = True)

# Kind
# df_tvshows.fillna({'Kind': "NA"}, inplace = True)

# Type
# df_tvshows.fillna({'Type': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Type'], axis = 1)

# Seasons
# df_tvshows.fillna({'Seasons': 1}, inplace = True)
df_tvshows.fillna({'Seasons': "NA"}, inplace = True)
# df_tvshows = df_tvshows.drop(['Seasons'], axis = 1)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)
# df_tvshows.fillna({'Seasons' : df_tvshows['Seasons'].mean()}, inplace = True)
# df_tvshows['Seasons'] = df_tvshows['Seasons'].astype(int)

# Service Provider
df_tvshows['Service Provider'] = df_tvshows.loc[:, ['Netflix', 'Prime Video', 'Disney+', 'Hulu']].idxmax(axis = 1)
# df_tvshows.drop(['Netflix','Prime Video','Disney+','Hulu'], axis = 1)

# Removing Duplicate and Missing Entries
df_tvshows.dropna(how = 'any', inplace = True)
df_tvshows.drop_duplicates(inplace = True)

# In[10]:
data_investigate(df_tvshows)

# In[11]:
df_tvshows.head()

# In[12]:
df_tvshows.describe()

# In[13]:
df_tvshows.corr()

```

```

# In[14]:


# df_tvshows.sort_values('Year', ascending = True)
# df_tvshows.sort_values('IMDb', ascending = False)

# In[15]:


# df_tvshows.to_csv(path_or_buf= '/content/drive/MyDrive/Files/updated_otttvshows.csv',
index = False)

# path = '/content/drive/MyDrive/Files/'

# udf_tvshows = pd.read_csv(path + 'updated_otttvshows.csv')

# udf_tvshows

# In[16]:


# df.netflix_tvshows = df_tvshows.loc[(df_tvshows['Netflix'] > 0)]
# df.hulu_tvshows = df_tvshows.loc[(df_tvshows['Hulu'] > 0)]
# df.prime_video_tvshows = df_tvshows.loc[(df_tvshows['Prime Video'] > 0)]
# df.disney_tvshows = df_tvshows.loc[(df_tvshows['Disney+'] > 0)]


# In[17]:


df.netflix_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 1) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.hulu_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 1) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 0)]
df.prime_video_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 1 ) & (df_tvshows['Disney+'] == 0)]
df.disney_only_tvshows = df_tvshows[(df_tvshows['Netflix'] == 0) & (df_tvshows['Hulu'] == 0) & (df_tvshows['Prime Video'] == 0 ) & (df_tvshows['Disney+'] == 1)]


# In[18]:


df_tvshows_years = df_tvshows.copy()

# In[19]:


df_tvshows_years.drop(df_tvshows_years.loc[df_tvshows_years['Year'] == "NA"].index, inplace = True)
# df_tvshows_years = df_tvshows_years[df_tvshows_years.Year != "NA"]
df_tvshows_years['Year'] = df_tvshows_years['Year'].astype(int)

# In[20]:

```

```
# Creating distinct dataframes only with the tvshows present on individual streaming platforms
netflix_years_tvshows = df_tvshows_years.loc[df_tvshows_years['Netflix'] == 1]
hulu_years_tvshows = df_tvshows_years.loc[df_tvshows_years['Hulu'] == 1]
prime_video_years_tvshows = df_tvshows_years.loc[df_tvshows_years['Prime Video'] == 1]
disney_years_tvshows = df_tvshows_years.loc[df_tvshows_years['Disney+'] == 1]
```

```
# In[21]:
```

```
df_tvshows_years_group = df_tvshows_years.copy()
```

```
# In[22]:
```

```
plt.figure(figsize = (10, 10))
corr = df_tvshows_years.corr()
# Plot figsize
fig, ax = plt.subplots(figsize=(10, 8))
# Generate Heat Map, allow annotations and place floats in map
sns.heatmap(corr, cmap = 'magma', annot = True, fmt = ".2f")
# Apply xticks
plt.xticks(range(len(corr.columns)), corr.columns);
# Apply yticks
plt.yticks(range(len(corr.columns)), corr.columns)
# show plot
plt.show()
fig.show()
```

```
# In[23]:
```

```
df_years_high_tvshows = df_tvshows_years.sort_values(by = 'Year', ascending = False).reset_index()
df_years_high_tvshows = df_years_high_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_years['Year'] == (df_tvshows_years['Year'].max()))
# df_years_high_tvshows = df_tvshows_years[filter]

# highestRated_tvshows = df_tvshows_years.loc[df_tvshows_years['Year'].idxmax()]

print('\nTV Shows with Highest Ever Year are : \n')
df_years_high_tvshows.head(5)
```

```
# In[24]:
```

```
fig = px.bar(y = df_years_high_tvshows['Title'][:15],
              x = df_years_high_tvshows['Year'][:15],
              color = df_years_high_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Highest Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[25]:
```

```
df_years_low_tvshows = df_tvshows_years.sort_values(by = 'Year', ascending = True).reset_index()
df_years_low_tvshows = df_years_low_tvshows.drop(['index'], axis = 1)
# filter = (df_tvshows_years['Year'] == (df_tvshows_years['Year'].min()))
# df_years_low_tvshows = df_tvshows_years[filter]

print('\nTV Shows with Lowest Ever Year are : \n')
df_years_low_tvshows.head(5)
```

```
# In[26]:
```

```
fig = px.bar(y = df_years_low_tvshows['Title'][:15],
              x = df_years_low_tvshows['Year'][:15],
              color = df_years_low_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Lowest Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```

```
# In[27]:
```

```
print(f'''
    Total '{df_tvshows_years['Year'].unique().shape[0]}' unique Year s were Given, They
    were Like this,\n

{df_tvshows_years.sort_values(by = 'Year', ascending = False)['Year'].unique()}\n

    The Highest Ever Year Ever Any TV Show Got is '{df_years_high_tvshows['Title'][0]}' :
    '{df_years_high_tvshows['Year'].max()}'\n

    The Lowest Ever Year Ever Any TV Show Got is '{df_years_low_tvshows['Title'][0]}' :
    '{df_years_low_tvshows['Year'].min()}'\n
    ''')
```

```
# In[28]:
```

```
netflix_years_high_tvshows =
df_years_high_tvshows.loc[df_years_high_tvshows['Netflix']==1].reset_index()
netflix_years_high_tvshows = netflix_years_high_tvshows.drop(['index'], axis = 1)

netflix_years_low_tvshows =
df_years_low_tvshows.loc[df_years_low_tvshows['Netflix']==1].reset_index()
netflix_years_low_tvshows = netflix_years_low_tvshows.drop(['index'], axis = 1)

netflix_years_high_tvshows.head(5)
```

```
# In[29]:
```

```

fig = px.bar(y = netflix_years_high_tvshows['Title'][:15],
              x = netflix_years_high_tvshows['Year'][:15],
              color = netflix_years_high_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title  = 'TV Shows with Highest Year in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[30]:

```

fig = px.bar(y = netflix_years_low_tvshows['Title'][:15],
              x = netflix_years_low_tvshows['Year'][:15],
              color = netflix_years_low_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title  = 'TV Shows with Lowest Year in Minutes : Netflix')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[31]:

```

hulu_years_high_tvshows =
df_years_high_tvshows.loc[df_years_high_tvshows['Hulu']==1].reset_index()
hulu_years_high_tvshows = hulu_years_high_tvshows.drop(['index'], axis = 1)

hulu_years_low_tvshows =
df_years_low_tvshows.loc[df_years_low_tvshows['Hulu']==1].reset_index()
hulu_years_low_tvshows = hulu_years_low_tvshows.drop(['index'], axis = 1)

hulu_years_high_tvshows.head(5)

```

In[32]:

```

fig = px.bar(y = hulu_years_high_tvshows['Title'][:15],
              x = hulu_years_high_tvshows['Year'][:15],
              color = hulu_years_high_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title  = 'TV Shows with Highest Year in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

```

In[33]:

```

fig = px.bar(y = hulu_years_low_tvshows['Title'][:15],
              x = hulu_years_low_tvshows['Year'][:15],
              color = hulu_years_low_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',

```

```

labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
title  = 'TV Shows with Lowest Year in Minutes : Hulu')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[34]:


prime_video_years_high_tvshows = df_years_high_tvshows.loc[df_years_high_tvshows['Prime Video']==1].reset_index()
prime_video_years_high_tvshows = prime_video_years_high_tvshows.drop(['index'], axis = 1)

prime_video_years_low_tvshows = df_years_low_tvshows.loc[df_years_low_tvshows['Prime Video']==1].reset_index()
prime_video_years_low_tvshows = prime_video_years_low_tvshows.drop(['index'], axis = 1)

prime_video_years_high_tvshows.head(5)

# In[35]:


fig = px.bar(y = prime_video_years_high_tvshows['Title'][:15],
              x = prime_video_years_high_tvshows['Year'][:15],
              color = prime_video_years_high_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Highest Year in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[36]:


fig = px.bar(y = prime_video_years_low_tvshows['Title'][:15],
              x = prime_video_years_low_tvshows['Year'][:15],
              color = prime_video_years_low_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Lowest Year in Minutes : Prime Video')

fig.update_layout(plot_bgcolor = 'white')
fig.show()

# In[37]:


disney_years_high_tvshows =
df_years_high_tvshows.loc[df_years_high_tvshows['Disney+']==1].reset_index()
disney_years_high_tvshows = disney_years_high_tvshows.drop(['index'], axis = 1)

disney_years_low_tvshows =
df_years_low_tvshows.loc[df_years_low_tvshows['Disney+']==1].reset_index()
disney_years_low_tvshows = disney_years_low_tvshows.drop(['index'], axis = 1)

disney_years_high_tvshows.head(5)

```

```
# In[38]:
```

```
fig = px.bar(y = disney_years_high_tvshows['Title'][:15],
              x = disney_years_high_tvshows['Year'][:15],
              color = disney_years_high_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Highest Year in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[39]:
```

```
fig = px.bar(y = disney_years_low_tvshows['Title'][:15],
              x = disney_years_low_tvshows['Year'][:15],
              color = disney_years_low_tvshows['Year'][:15],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Lowest Year in Minutes : Disney+')

fig.update_layout(plot_bgcolor = 'white')
fig.show()
```



```
# In[40]:
```

```
print(f'''  

    The TV Show with Highest Year Ever Got is '{df_years_high_tvshows['Title'][0]} :  

'{df_years_high_tvshows['Year'].max()}'\n
    The TV Show with Lowest Year Ever Got is '{df_years_low_tvshows['Title'][0]} :  

'{df_years_low_tvshows['Year'].min()}'\n
  

    The TV Show with Highest Year on 'Netflix' is  

'{netflix_years_high_tvshows['Title'][0]} : '{netflix_years_high_tvshows['Year'].max()}'\n
    The TV Show with Lowest Year on 'Netflix' is  

'{netflix_years_low_tvshows['Title'][0]} : '{netflix_years_low_tvshows['Year'].min()}'\n
  

    The TV Show with Highest Year on 'Hulu' is '{hulu_years_high_tvshows['Title'][0]} :  

'{hulu_years_high_tvshows['Year'].max()}'\n
    The TV Show with Lowest Year on 'Hulu' is '{hulu_years_low_tvshows['Title'][0]} :  

'{hulu_years_low_tvshows['Year'].min()}'\n
  

    The TV Show with Highest Year on 'Prime Video' is  

'{prime_video_years_high_tvshows['Title'][0]} :  

'{prime_video_years_high_tvshows['Year'].max()}'\n
    The TV Show with Lowest Year on 'Prime Video' is  

'{prime_video_years_low_tvshows['Title'][0]} :  

'{prime_video_years_low_tvshows['Year'].min()}'\n
  

    The TV Show with Highest Year on 'Disney+' is  

'{disney_years_high_tvshows['Title'][0]} : '{disney_years_high_tvshows['Year'].max()}'\n
    The TV Show with Lowest Year on 'Disney+' is  

'{disney_years_low_tvshows['Title'][0]} : '{disney_years_low_tvshows['Year'].min()}'\n
''')
```

```

# In[41]:


print(f'''
    Accross All Platforms the Average Year  is '{round(df_tvshows_years['Year'].mean(), ndigits = 2)}'\n
    The Average Year  on 'Netflix' is '{round(netflix_years_tvshows['Year'].mean(), ndigits = 2)}'\n
    The Average Year  on 'Hulu' is '{round(hulu_years_tvshows['Year'].mean(), ndigits = 2)}'\n
    The Average Year  on 'Prime Video' is
'{round(prime_video_years_tvshows['Year'].mean(), ndigits = 2)}'\n
    The Average Year  on 'Disney+' is '{round(disney_years_tvshows['Year'].mean(), ndigits = 2)}'
''')


# In[42]:


f, ax = plt.subplots(1, 2 , figsize = (20, 5))
sns.distplot(df_tvshows_years['Year'], bins = 20, kde = True, ax = ax[0])
sns.boxplot(df_tvshows_years['Year'], ax = ax[1])
plt.show()


# In[43]:


# Defining plot size and title
plt.figure(figsize = (20, 5))
plt.title('Years Per Platform')

# Plotting the information from each dataset into a histogram
sns.histplot(prime_video_years_tvshows['Year'][:100], color = 'lightblue', legend = True, kde = True)
sns.histplot(netflix_years_tvshows['Year'][:100], color = 'red', legend = True, kde = True)
sns.histplot(hulu_years_tvshows['Year'][:100], color = 'lightgreen', legend = True, kde = True)
sns.histplot(disney_years_tvshows['Year'][:100], color = 'darkblue', legend = True, kde = True)

# Setting the legend
plt.legend(['Prime Video', 'Netflix', 'Hulu', 'Disney+'])
plt.show()


# In[44]:


year_count = df_tvshows_years.groupby('Year')['Title'].count()
year_tvshows = df_tvshows_years.groupby('Year')[['Netflix', 'Hulu', 'Prime Video', 'Disney+']].sum()
year_data_tvshows = pd.concat([year_count, year_tvshows], axis = 1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
year_data_tvshows = year_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)


# In[45]:

```

```

# TV Shows Count per Year - All Platforms Combined
year_data_tvshows.head()

# In[46]:


fig = px.bar(y = year_data_tvshows['TV Shows Count'],
              x = year_data_tvshows['Year'],
              color = year_data_tvshows['Year'],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Year : In Minutes'},
              title = 'TV Shows with Year : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[47]:


fig = px.pie(year_data_tvshows[:10],
              names = year_data_tvshows['Year'][:10],
              values = year_data_tvshows['TV Shows Count'][:10],
              color = year_data_tvshows['TV Shows Count'][:10],
              color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
                  title = 'TV Shows Count based on Year Group')
fig.show()

# In[48]:


# Highest TV Shows Count per Year - All Platforms Combined
df_year_high_tvshows = year_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False).reset_index()
df_year_high_tvshows = df_year_high_tvshows.drop(['index'], axis = 1)
# filter = (year_data_tvshows['TV Shows Count'] == (year_data_tvshows['TV Shows Count'].max()))
# df_year_high_tvshows = year_data_tvshows[filter]

# highestRated_tvshows = year_data_tvshows.loc[year_data_tvshows['TV Shows Count'].idxmax()]

print('\nYear with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_year_high_tvshows.head(5)

# In[49]:


fig = px.bar(y = df_year_high_tvshows['TV Shows Count'][:10],
              x = df_year_high_tvshows['Year'][:10],
              color = df_year_high_tvshows['TV Shows Count'][:10],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Year : In Minutes'},
              title = 'Year with Highest TV Shows Count : All Platforms')

```

```

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[50]:


# Lowest TV Shows Count per Year - All Platforms Combined
df_year_low_tvshows = year_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
True).reset_index()
df_year_low_tvshows = df_year_low_tvshows.drop(['index'], axis = 1)
# filter = (year_data_tvshows['TV Shows Count'] == (year_data_tvshows['TV Shows
Count'].min()))
# df_year_low_tvshows = year_data_tvshows[filter]

print('\nYear with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_year_low_tvshows.head(5)

# In[51]:


fig = px.bar(y = df_year_low_tvshows['TV Shows Count'][:10],
              x = df_year_low_tvshows['Year'][:10],
              color = df_year_low_tvshows['TV Shows Count'][:10],
              color_continuous_scale = 'Teal_r',
              labels = { 'y' : 'TV Shows Count', 'x' : 'Year : In Minutes'},
              title = 'Year with Lowest TV Shows Count : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[52]:


print(f'''
      Total '{df_tvshows_years['Year'].count()}' Titles are available on All Platforms, out
      of which\n
      You Can Choose to see TV Shows from Total
      '{year_data_tvshows['Year'].unique().shape[0]}' Year, They were Like this, \n

      {year_data_tvshows.sort_values(by = 'TV Shows Count', ascending =
      False)['Year'].head(5).unique()} etc. \n

      The Year with Highest TV Shows Count have '{year_data_tvshows['TV Shows
      Count'].max()}' TV Shows Available is '{df_year_high_tvshows['Year'][0]}', &\n
      The Year with Lowest TV Shows Count have '{year_data_tvshows['TV Shows
      Count'].min()}' TV Shows Available is '{df_year_low_tvshows['Year'][0]}'
      ''')

# In[53]:


# Highest TV Shows Count per Year - Netflix
netflix_year_tvshows = year_data_tvshows[year_data_tvshows['Netflix'] != 0].sort_values(by
= 'Netflix', ascending = False).reset_index()
netflix_year_tvshows = netflix_year_tvshows.drop(['index', 'Hulu', 'Prime Video',
'Disney+', 'TV Shows Count'], axis = 1)

```

```
netflix_year_high_tvshows = df_year_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_year_high_tvshows = netflix_year_high_tvshows.drop(['index'], axis = 1)

netflix_year_low_tvshows = df_year_high_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_year_low_tvshows = netflix_year_low_tvshows.drop(['index'], axis = 1)

netflix_year_high_tvshows.head(5)
```

In[54]:

```
# Highest TV Shows Count per Year - Hulu
hulu_year_tvshows = year_data_tvshows[year_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_year_tvshows = hulu_year_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_year_high_tvshows = df_year_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_year_high_tvshows = hulu_year_high_tvshows.drop(['index'], axis = 1)

hulu_year_low_tvshows = df_year_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_year_low_tvshows = hulu_year_low_tvshows.drop(['index'], axis = 1)

hulu_year_high_tvshows.head(5)
```

In[55]:

```
# Highest TV Shows Count per Year - Prime Video
prime_video_year_tvshows = year_data_tvshows[year_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_year_tvshows = prime_video_year_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_year_high_tvshows = df_year_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_year_high_tvshows = prime_video_year_high_tvshows.drop(['index'], axis = 1)

prime_video_year_low_tvshows = df_year_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_year_low_tvshows = prime_video_year_low_tvshows.drop(['index'], axis = 1)

prime_video_year_high_tvshows.head(5)
```

In[56]:

```
# Highest TV Shows Count per Year - Disney+
disney_year_tvshows = year_data_tvshows[year_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_year_tvshows = disney_year_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)
```

```

disney_year_high_tvshows = df_year_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_year_high_tvshows = disney_year_high_tvshows.drop(['index'], axis = 1)

disney_year_low_tvshows = df_year_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_year_low_tvshows = disney_year_low_tvshows.drop(['index'], axis = 1)

disney_year_high_tvshows.head(5)

# In[57]:


print(f'''
    The Year with Highest TV Shows Count Ever Got is '{df_year_high_tvshows['Year'][0]}'
: '{df_year_high_tvshows['TV Shows Count'].max()}'\n
    The Year with Lowest TV Shows Count Ever Got is '{df_year_low_tvshows['Year'][0]}'
: '{df_year_low_tvshows['TV Shows Count'].min()}'\n

    The Year with Highest TV Shows Count on 'Netflix' is
'{netflix_year_high_tvshows['Year'][0]} : '{netflix_year_high_tvshows['Netflix'].max()}'\n
    The Year with Lowest TV Shows Count on 'Netflix' is
'{netflix_year_low_tvshows['Year'][0]} : '{netflix_year_low_tvshows['Netflix'].min()}'\n

    The Year with Highest TV Shows Count on 'Hulu' is
'{hulu_year_high_tvshows['Year'][0]} : '{hulu_year_high_tvshows['Hulu'].max()}'\n
    The Year with Lowest TV Shows Count on 'Hulu' is '{hulu_year_low_tvshows['Year'][0]}'
: '{hulu_year_low_tvshows['Hulu'].min()}'\n

    The Year with Highest TV Shows Count on 'Prime Video' is
'{prime_video_year_high_tvshows['Year'][0]} : '{prime_video_year_high_tvshows['Prime
Video'].max()}'\n
    The Year with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_year_low_tvshows['Year'][0]} : '{prime_video_year_low_tvshows['Prime
Video'].min()}'\n

    The Year with Highest TV Shows Count on 'Disney+' is
'{disney_year_high_tvshows['Year'][0]} : '{disney_year_high_tvshows['Disney+'].max()}'\n
    The Year with Lowest TV Shows Count on 'Disney+' is
'{disney_year_low_tvshows['Year'][0]} : '{disney_year_low_tvshows['Disney+'].min()}'\n
''')


# In[58]:


print(f'''

    Accross All Platforms the Average TV Shows Count of Year is
'{round(year_data_tvshows['TV Shows Count'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Year on 'Netflix' is
'{round(netflix_year_tvshows['Netflix'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Year on 'Hulu' is
'{round(hulu_year_tvshows['Hulu'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Year on 'Prime Video' is
'{round(prime_video_year_tvshows['Prime Video'].mean(), ndigits = 2)}'\n
    The Average TV Shows Count of Year on 'Disney+' is
'{round(disney_year_tvshows['Disney+'].mean(), ndigits = 2)}'\n
'''')

```

```
# In[59]:
print(f'''
    Accross All Platforms Total Count of Year is
'{year_data_tvshows['Year'].unique().shape[0]}'\n
    Total Count of Year on 'Netflix' is
'{netflix_year_tvshows['Year'].unique().shape[0]}'\n
    Total Count of Year on 'Hulu' is '{hulu_year_tvshows['Year'].unique().shape[0]}'\n
    Total Count of Year on 'Prime Video' is
'{prime_video_year_tvshows['Year'].unique().shape[0]}'\n
    Total Count of Year on 'Disney+' is
'{disney_year_tvshows['Year'].unique().shape[0]}'
''')

# In[60]:
fig = plt.figure(figsize = (20, 10))
sns.lineplot(data = year_data_tvshows, x = 'Year', y = 'TV Shows Count')
plt.show()

# In[61]:
plt.figure(figsize = (20, 10))
sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Netflix'], color =
'red')
sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Hulu'], color =
'lightgreen')
sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Prime Video'], color =
'lightblue')
sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Disney+'], color =
'darkblue')
plt.xlabel('Release Year', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()

# In[62]:
fig, axes = plt.subplots(2, 2, figsize=(20 ,20))

n_y_ax1 = sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Netflix'],
color = 'red', ax = axes[0, 0])
h_y_ax2 = sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Hulu'], color =
'lightgreen', ax = axes[0, 1])
p_y_ax3 = sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Prime Video'],
color = 'lightblue', ax = axes[1, 0])
d_y_ax4 = sns.lineplot(x = year_data_tvshows['Year'], y = year_data_tvshows['Disney+'],
color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_y_ax1.title.set_text(labels[0])
h_y_ax2.title.set_text(labels[1])
p_y_ax3.title.set_text(labels[2])
d_y_ax4.title.set_text(labels[3])
```

```
plt.show()
```

```
# In[63]:
```

```
def round_val(data):
    if str(data) != 'nan':
        return round(data)

def round_fix(data):
    if data in range(1801,1901):
        # print(data)
        return 1900
    if data in range(1901,1911):
        return 1910
    if data in range(1911,1921):
        return 1920
    if data in range(1921,1931):
        return 1930
    if data in range(1931,1941):
        return 1940
    if data in range(1941,1951):
        return 1950
    if data in range(1951,1961):
        return 1960
    if data in range(1961,1971):
        return 1970
    if data in range(1971,1981):
        return 1980
    if data in range(1981,1991):
        return 1990
    if data in range(1991,2001):
        return 2000
    if data in range(2000,2011):
        return 2010
    if data in range(2010,2021):
        return 2020
    if data in range(2020,2031):
        return 2030
    else:
        return 2100
```

```
# In[64]:
```

```
df_tvshows_years_group['Year Group'] =
df_tvshows_years_group['Year'].apply(round_fix).astype(int)

years_values = df_tvshows_years_group['Year Group'].value_counts().sort_index(ascending =
False).tolist()
years_index = df_tvshows_years_group['Year Group'].value_counts().sort_index(ascending =
False).index

# years_values, years_index
```

```
# In[65]:
```

```

years_group_count = df_tvshows_years_group.groupby('Year Group')['Title'].count()
years_group_tvshows = df_tvshows_years_group.groupby('Year Group')[['Netflix', 'Hulu',
'Prime Video', 'Disney+']].sum()
years_group_data_tvshows = pd.concat([years_group_count, years_group_tvshows], axis =
1).reset_index().rename(columns = {'Title' : 'TV Shows Count'})
years_group_data_tvshows = years_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False)

# In[66]:


# Year Group with TV Shows Counts - All Platforms Combined
years_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)

# In[67]:


years_group_data_tvshows.sort_values(by = 'Year Group', ascending = False)

# In[68]:


fig = px.bar(y = years_group_data_tvshows['TV Shows Count'],
x = years_group_data_tvshows['Year Group'],
color = years_group_data_tvshows['Year Group'],
color_continuous_scale = 'Teal_r',
labels = { 'y' : 'TV Shows Count', 'x' : 'Year : In Minutes'},
title = 'TV Shows with Group Year in Minutes : All Platforms')

fig.update_layout(plot_bgcolor = "white")
fig.show()

# In[69]:


fig = px.pie(years_group_data_tvshows[:10],
names = years_group_data_tvshows['Year Group'],
values = years_group_data_tvshows['TV Shows Count'],
color = years_group_data_tvshows['TV Shows Count'],
color_discrete_sequence = px.colors.sequential.Teal)

fig.update_traces(textinfo = 'percent+label',
title = 'TV Shows Count based on Year Group')
fig.show()

# In[70]:


df_years_group_high_tvshows = years_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = False).reset_index()
df_years_group_high_tvshows = df_years_group_high_tvshows.drop(['index'], axis = 1)
# filter = (years_group_data_tvshows['TV Shows Count'] == (years_group_data_tvshows['TV
Shows Count'].max()))
# df_years_group_high_tvshows = years_group_data_tvshows[filter]

```

```

# highest_rated_tvshows = years_group_data_tvshows.loc[years_group_data_tvshows['TV Shows Count'].idxmax()]

# print('\nYear with Highest Ever TV Shows Count are : All Platforms Combined\n')
df_years_group_high_tvshows.head(5)

# In[71]:


df_years_group_low_tvshows = years_group_data_tvshows.sort_values(by = 'TV Shows Count',
ascending = True).reset_index()
df_years_group_low_tvshows = df_years_group_low_tvshows.drop(['index'], axis = 1)
# filter = (years_group_data_tvshows['TV Shows Count'] == (years_group_data_tvshows['TV Shows Count'].min()))
# df_years_group_low_tvshows = years_group_data_tvshows[filter]

# print('\nYear with Lowest Ever TV Shows Count are : All Platforms Combined\n')
df_years_group_low_tvshows.head(5)

# In[72]:


print(f'''
    Total '{df_tvshows_years['Year'].count()}' Titles are available on All Platforms, out
    of which\n
        You Can Choose to see TV Shows from Total '{years_group_data_tvshows['Year Group'].unique().shape[0]}' Year Group, They were Like this, \n
            {years_group_data_tvshows.sort_values(by = 'TV Shows Count', ascending = False)['Year Group'].unique()} etc. \n
    The Year Group with Highest TV Shows Count have '{years_group_data_tvshows['TV Shows Count'].max()}' TV Shows Available is '{df_years_group_high_tvshows['Year Group'][0]}', &\n
    The Year Group with Lowest TV Shows Count have '{years_group_data_tvshows['TV Shows Count'].min()}' TV Shows Available is '{df_years_group_low_tvshows['Year Group'][0]}'
''')


# In[73]:


netflix_years_group_tvshows = years_group_data_tvshows[years_group_data_tvshows['Netflix'] != 0].sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_years_group_tvshows = netflix_years_group_tvshows.drop(['index', 'Hulu', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

netflix_years_group_high_tvshows = df_years_group_high_tvshows.sort_values(by = 'Netflix', ascending = False).reset_index()
netflix_years_group_high_tvshows = netflix_years_group_high_tvshows.drop(['index'], axis = 1)

netflix_years_group_low_tvshows = df_years_group_low_tvshows.sort_values(by = 'Netflix', ascending = True).reset_index()
netflix_years_group_low_tvshows = netflix_years_group_low_tvshows.drop(['index'], axis = 1)

netflix_years_group_high_tvshows.head(5)

# In[74]:

```

```

hulu_years_group_tvshows = years_group_data_tvshows[years_group_data_tvshows['Hulu'] != 0].sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_years_group_tvshows = hulu_years_group_tvshows.drop(['index', 'Netflix', 'Prime Video', 'Disney+', 'TV Shows Count'], axis = 1)

hulu_years_group_high_tvshows = df_years_group_high_tvshows.sort_values(by = 'Hulu', ascending = False).reset_index()
hulu_years_group_high_tvshows = hulu_years_group_high_tvshows.drop(['index'], axis = 1)

hulu_years_group_low_tvshows = df_years_group_high_tvshows.sort_values(by = 'Hulu', ascending = True).reset_index()
hulu_years_group_low_tvshows = hulu_years_group_low_tvshows.drop(['index'], axis = 1)

hulu_years_group_high_tvshows.head(5)

# In[75]:


prime_video_years_group_tvshows = years_group_data_tvshows[years_group_data_tvshows['Prime Video'] != 0].sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_years_group_tvshows = prime_video_years_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Disney+', 'TV Shows Count'], axis = 1)

prime_video_years_group_high_tvshows = df_years_group_high_tvshows.sort_values(by = 'Prime Video', ascending = False).reset_index()
prime_video_years_group_high_tvshows = prime_video_years_group_high_tvshows.drop(['index'], axis = 1)

prime_video_years_group_low_tvshows = df_years_group_high_tvshows.sort_values(by = 'Prime Video', ascending = True).reset_index()
prime_video_years_group_low_tvshows = prime_video_years_group_low_tvshows.drop(['index'], axis = 1)

prime_video_years_group_high_tvshows.head(5)

# In[76]:


disney_years_group_tvshows = years_group_data_tvshows[years_group_data_tvshows['Disney+'] != 0].sort_values(by = 'Disney+', ascending = False).reset_index()
disney_years_group_tvshows = disney_years_group_tvshows.drop(['index', 'Netflix', 'Hulu', 'Prime Video', 'TV Shows Count'], axis = 1)

disney_years_group_high_tvshows = df_years_group_high_tvshows.sort_values(by = 'Disney+', ascending = False).reset_index()
disney_years_group_high_tvshows = disney_years_group_high_tvshows.drop(['index'], axis = 1)

disney_years_group_low_tvshows = df_years_group_high_tvshows.sort_values(by = 'Disney+', ascending = True).reset_index()
disney_years_group_low_tvshows = disney_years_group_low_tvshows.drop(['index'], axis = 1)

disney_years_group_high_tvshows.head(5)

# In[77]:

```

```

print(f'''
    The Year Group with Highest TV Shows Count Ever Got is
'{df_years_group_high_tvshows['Year Group'][0]} : '{df_years_group_high_tvshows['TV Shows
Count'].max()}'\n
    The Year Group with Lowest TV Shows Count Ever Got is
'{df_years_group_low_tvshows['Year Group'][0]} : '{df_years_group_low_tvshows['TV Shows
Count'].min()}'\n

    The Year Group with Highest TV Shows Count on 'Netflix' is
'{netflix_years_group_high_tvshows['Year Group'][0]} :
'{netflix_years_group_high_tvshows['Netflix'].max()}'\n
    The Year Group with Lowest TV Shows Count on 'Netflix' is
'{netflix_years_group_low_tvshows['Year Group'][0]} :
'{netflix_years_group_low_tvshows['Netflix'].min()}'\n

    The Year Group with Highest TV Shows Count on 'Hulu' is
'{hulu_years_group_high_tvshows['Year Group'][0]} :
'{hulu_years_group_high_tvshows['Hulu'].max()}'\n
    The Year Group with Lowest TV Shows Count on 'Hulu' is
'{hulu_years_group_low_tvshows['Year Group'][0]} :
'{hulu_years_group_low_tvshows['Hulu'].min()}'\n

    The Year Group with Highest TV Shows Count on 'Prime Video' is
'{prime_video_years_group_high_tvshows['Year Group'][0]} :
'{prime_video_years_group_high_tvshows['Prime Video'].max()}'\n
    The Year Group with Lowest TV Shows Count on 'Prime Video' is
'{prime_video_years_group_low_tvshows['Year Group'][0]} :
'{prime_video_years_group_low_tvshows['Prime Video'].min()}'\n

    The Year Group with Highest TV Shows Count on 'Disney+' is
'{disney_years_group_high_tvshows['Year Group'][0]} :
'{disney_years_group_high_tvshows['Disney+'].max()}'\n
    The Year Group with Lowest TV Shows Count on 'Disney+' is
'{disney_years_group_low_tvshows['Year Group'][0]} :
'{disney_years_group_low_tvshows['Disney+'].min()}'\n
    ''')

```

In[78]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = netflix_years_group_tvshows['Year Group'], y =
netflix_years_group_tvshows['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = hulu_years_group_tvshows['Year Group'], y =
hulu_years_group_tvshows['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = prime_video_years_group_tvshows['Year Group'], y =
prime_video_years_group_tvshows['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = disney_years_group_tvshows['Year Group'], y =
disney_years_group_tvshows['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()

```

```
# In[79]:
```

```
plt.figure(figsize = (20, 5))
sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Netflix'], color = 'red')
sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Hulu'], color = 'lightgreen')
sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Prime Video'], color = 'lightblue')
sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Disney+'], color = 'darkblue')
plt.xlabel('Year Group', fontsize = 15)
plt.ylabel('TV Shows Count', fontsize = 15)
plt.show()
```

```
# In[80]:
```

```
print(f'''
    Across All Platforms Total Count of Year Group is '{years_group_data_tvshows['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Netflix' is '{netflix_years_group_tvshows['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Hulu' is '{hulu_years_group_tvshows['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Prime Video' is '{prime_video_years_group_tvshows['Year
Group'].unique().shape[0]}'\n
    Total Count of Year Group on 'Disney+' is '{disney_years_group_tvshows['Year
Group'].unique().shape[0]}'\n
''')
```

```
# In[81]:
```

```
fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.lineplot(y = years_group_data_tvshows['Year Group'], x =
years_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_ru_ax2 = sns.lineplot(y = years_group_data_tvshows['Year Group'], x =
years_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_ru_ax3 = sns.lineplot(y = years_group_data_tvshows['Year Group'], x =
years_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_ru_ax4 = sns.lineplot(y = years_group_data_tvshows['Year Group'], x =
years_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

plt.show()
```

```
# In[82]:
```

```

fig, axes = plt.subplots(2, 2, figsize=(20 ,20))

n_yg_ax1 = sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Netflix'], color = 'red', ax = axes[0, 0])
h_yg_ax2 = sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Hulu'], color = 'lightgreen', ax = axes[0, 1])
p_yg_ax3 = sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Prime Video'], color = 'lightblue', ax = axes[1, 0])
d_yg_ax4 = sns.lineplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Disney+'], color = 'darkblue', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_yg_ax1.title.set_text(labels[0])
h_yg_ax2.title.set_text(labels[1])
p_yg_ax3.title.set_text(labels[2])
d_yg_ax4.title.set_text(labels[3])

plt.show()

```

In[83]:

```

fig, axes = plt.subplots(2, 2, figsize = (20 , 20))

n_ru_ax1 = sns.barplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Netflix'], palette = 'Reds_r', ax = axes[0, 0])
h_ru_ax2 = sns.barplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Hulu'], palette = 'Greens_r', ax = axes[0, 1])
p_ru_ax3 = sns.barplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Prime Video'], palette = 'Blues_r', ax = axes[1, 0])
d_ru_ax4 = sns.barplot(x = years_group_data_tvshows['Year Group'], y =
years_group_data_tvshows['Disney+'], palette = 'BuPu_r', ax = axes[1, 1])

labels = ['Netflix', 'Hulu', 'Prime Video', 'Disney+']

n_ru_ax1.title.set_text(labels[0])
h_ru_ax2.title.set_text(labels[1])
p_ru_ax3.title.set_text(labels[2])
d_ru_ax4.title.set_text(labels[3])

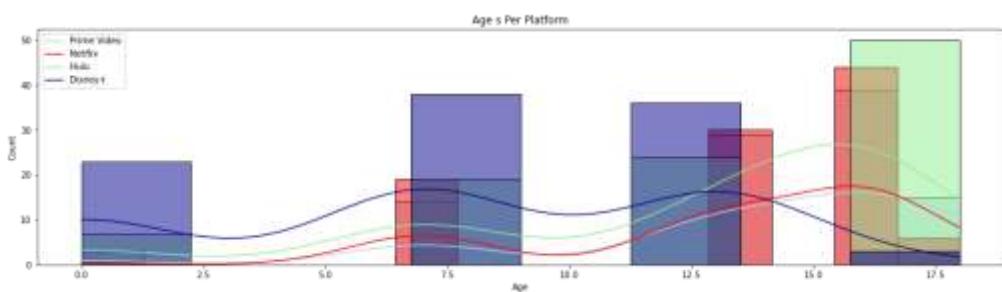
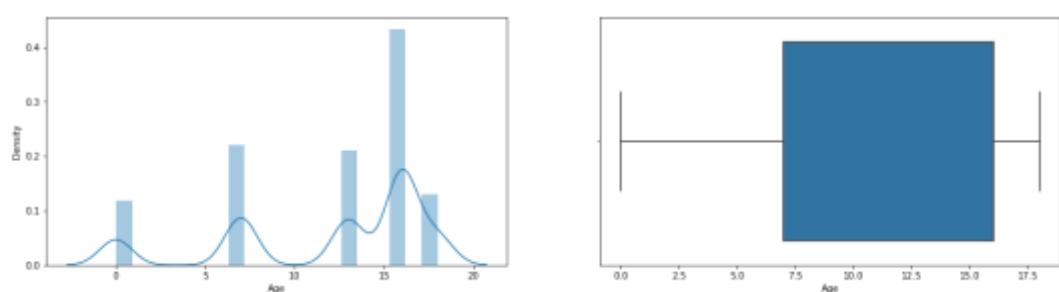
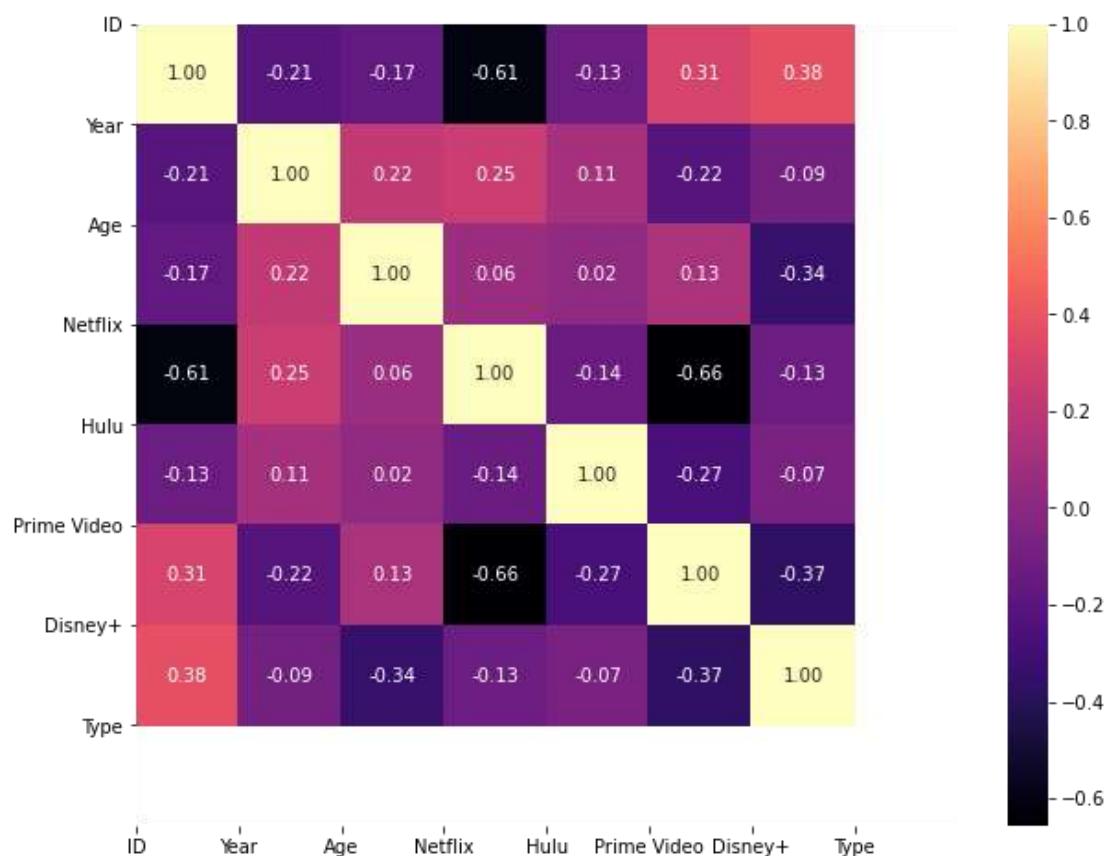
plt.show()

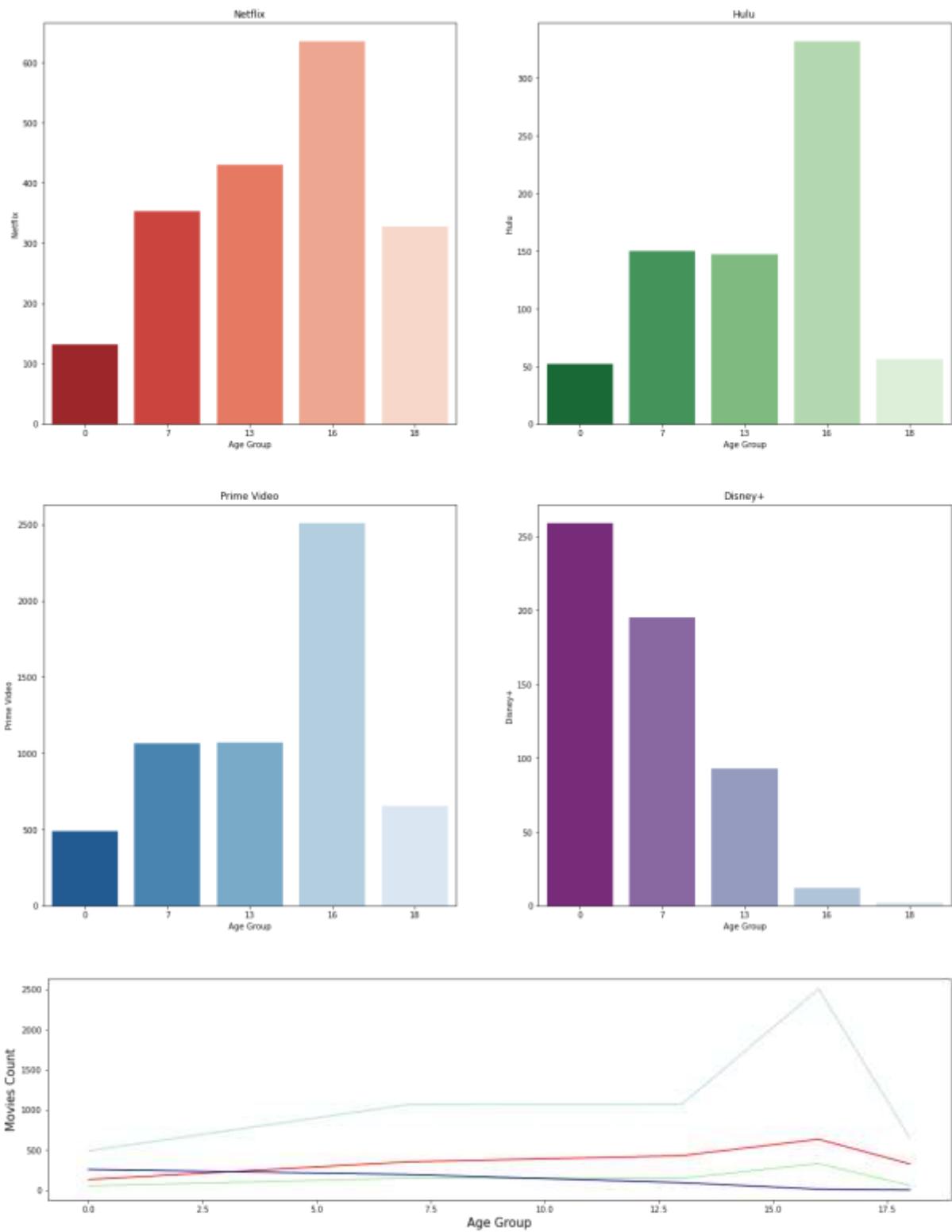
```

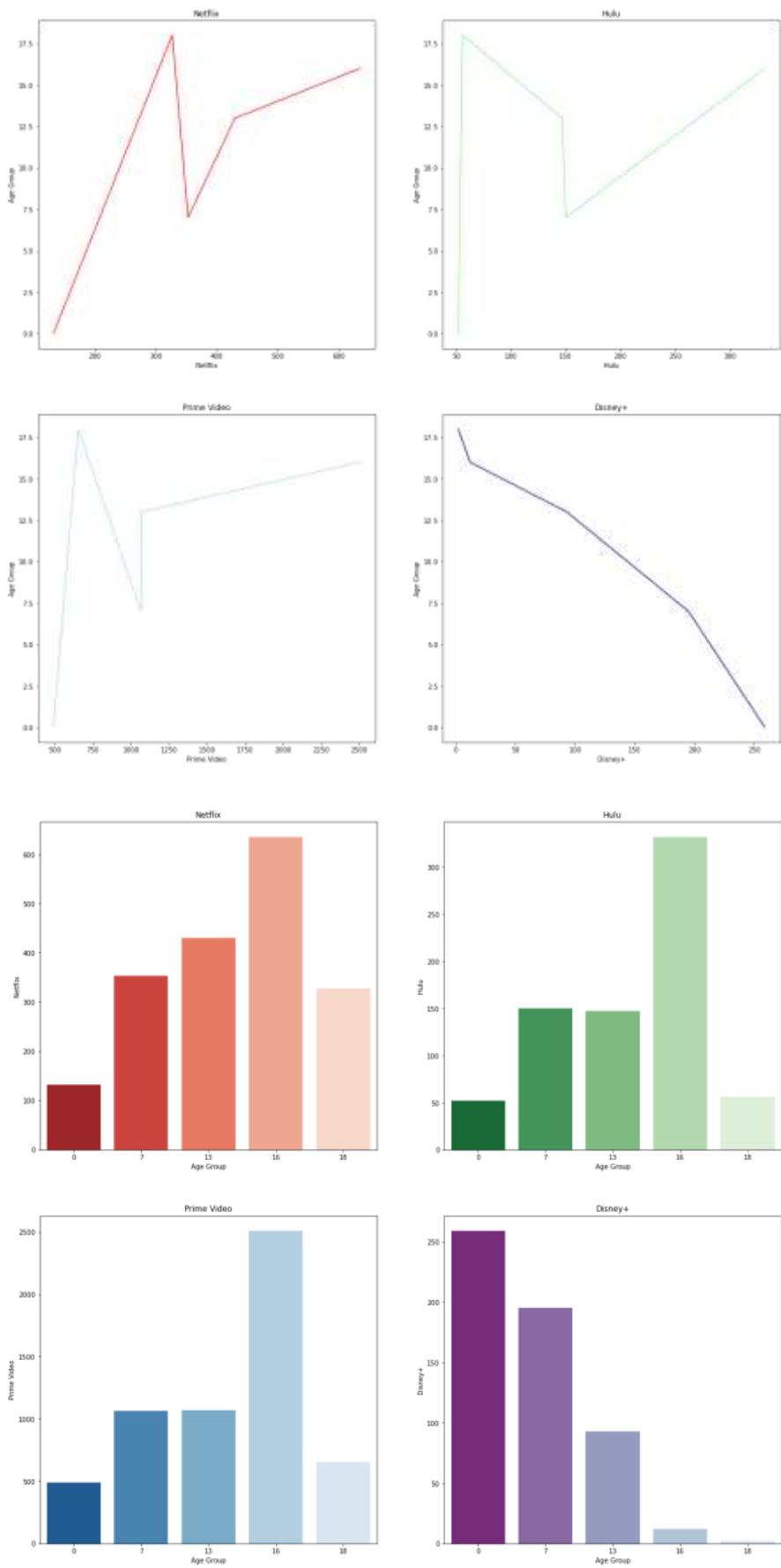
CHAPTER: SEVEN

Output: Movies

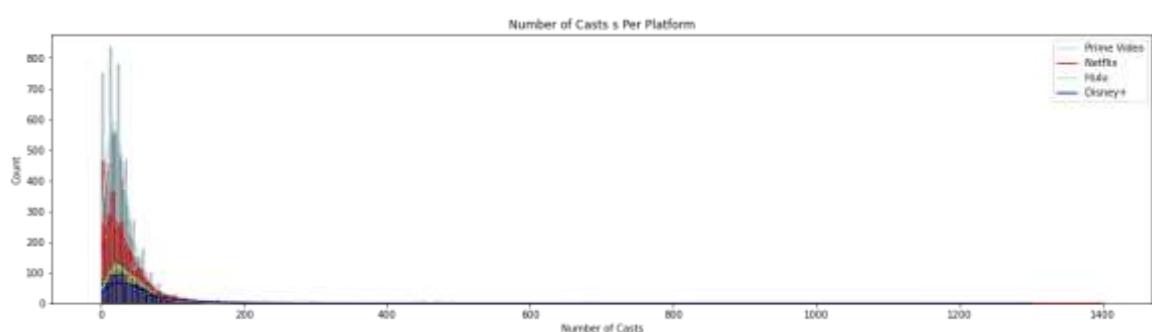
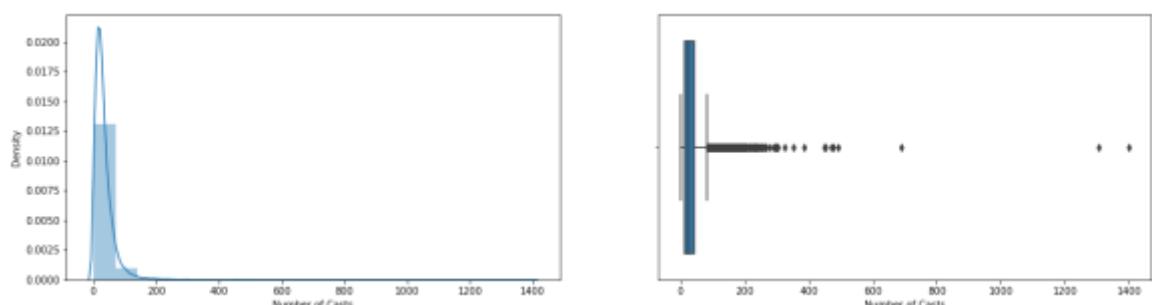
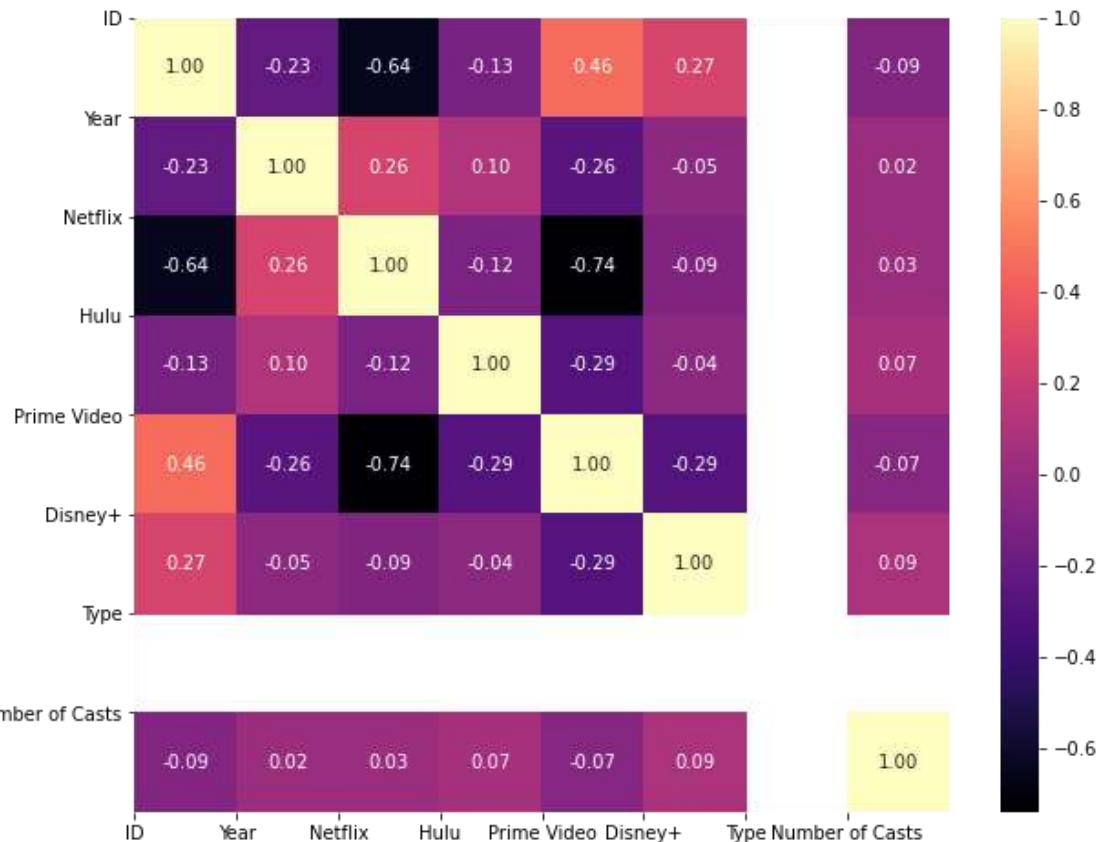
AGE

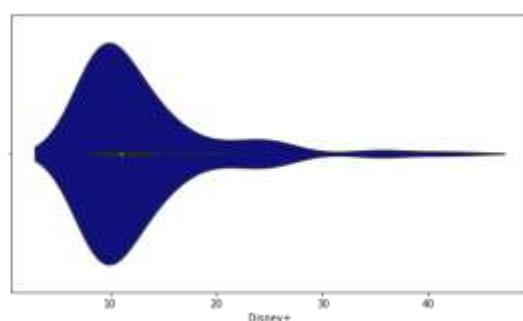
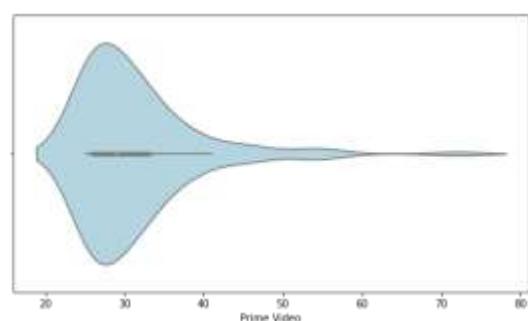
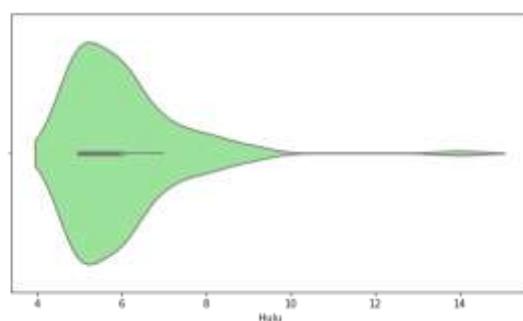
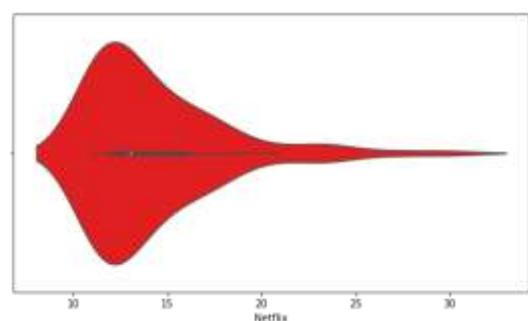
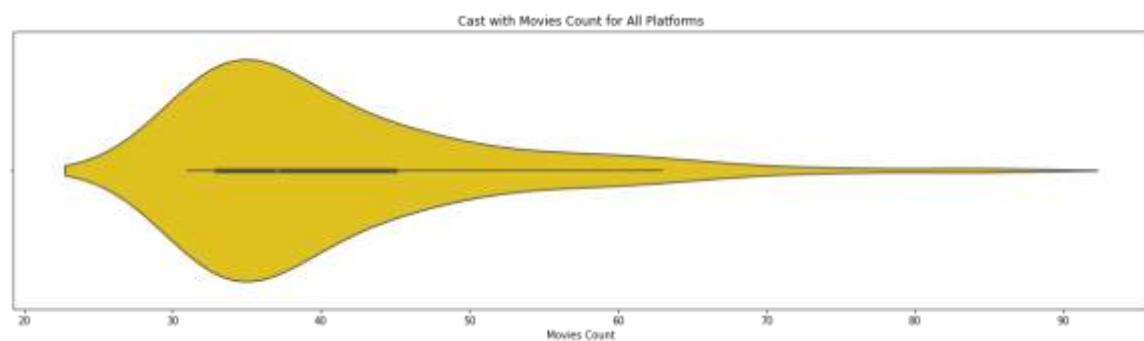
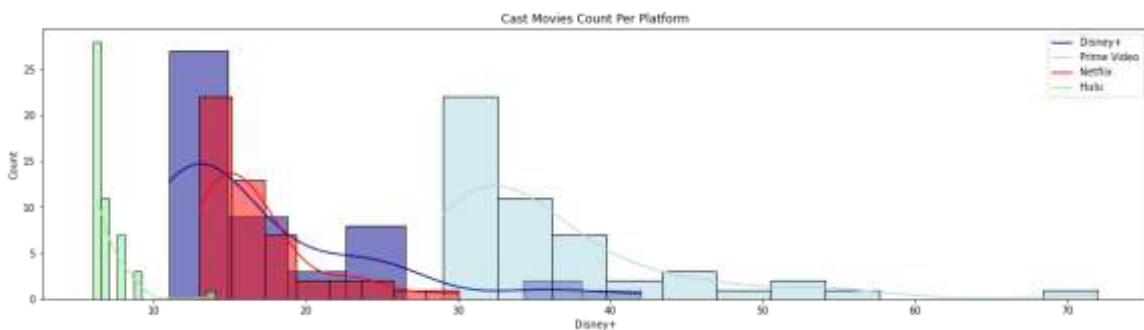
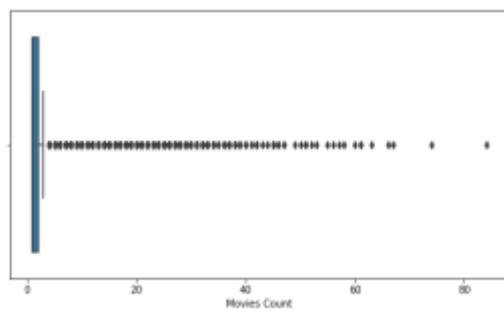
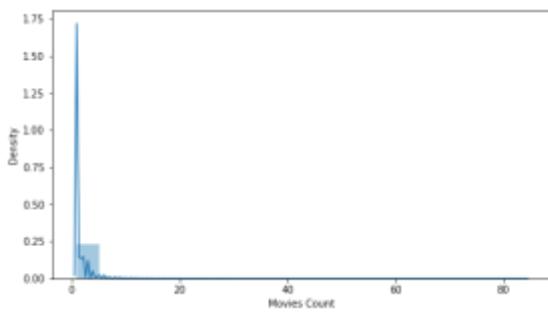


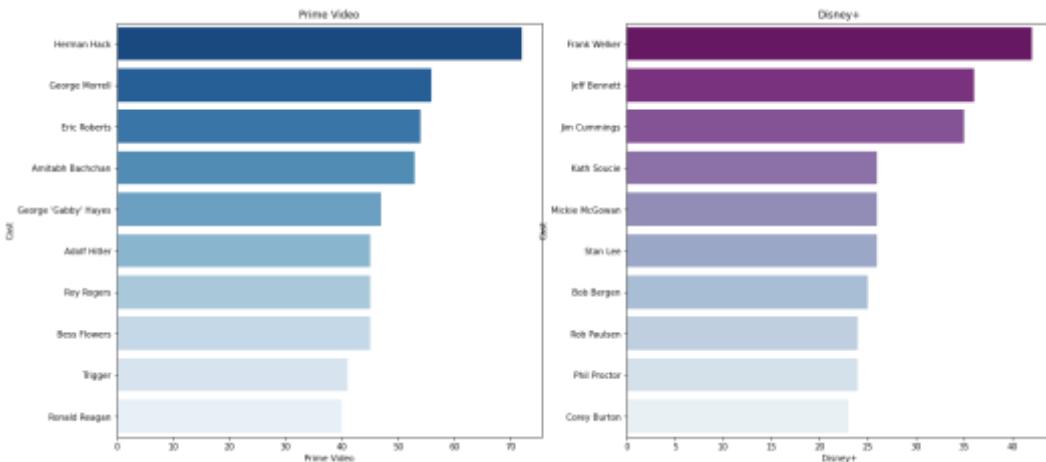
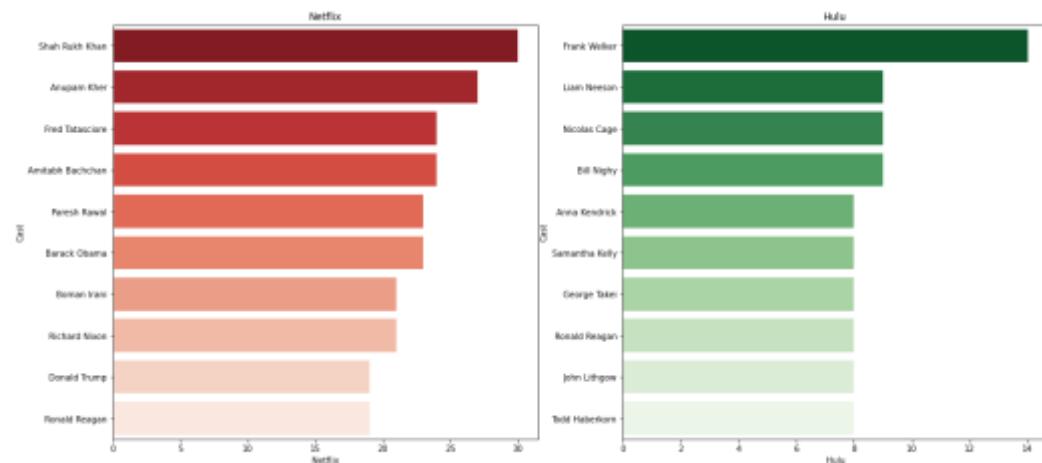
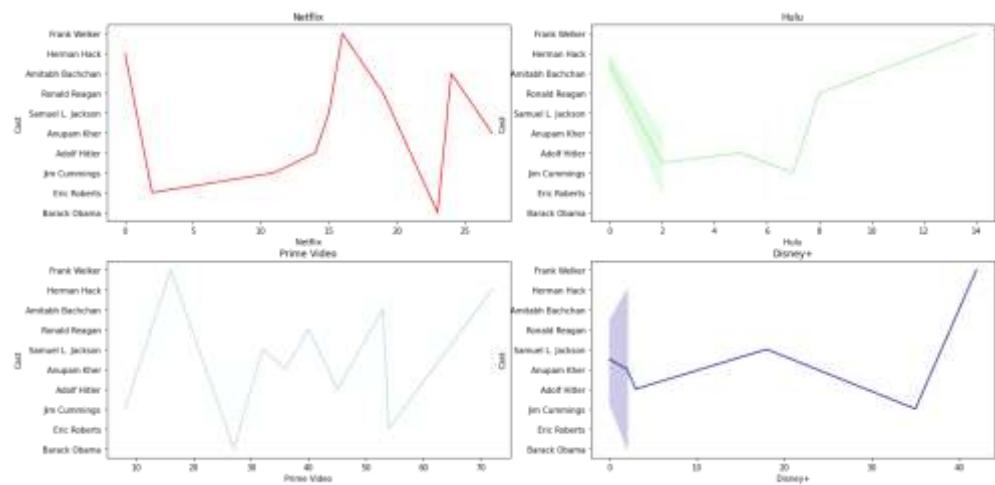
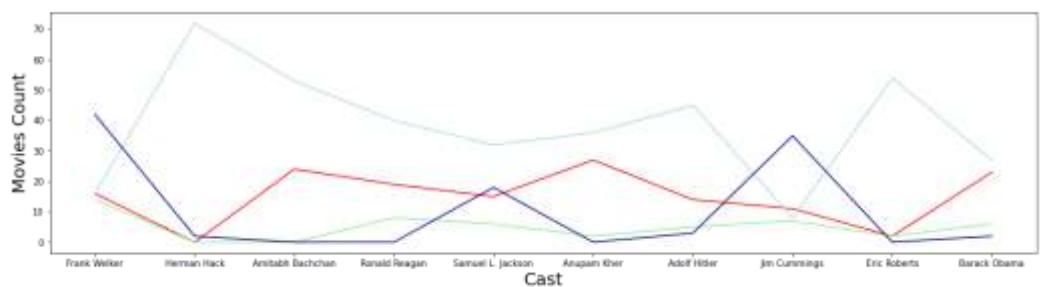


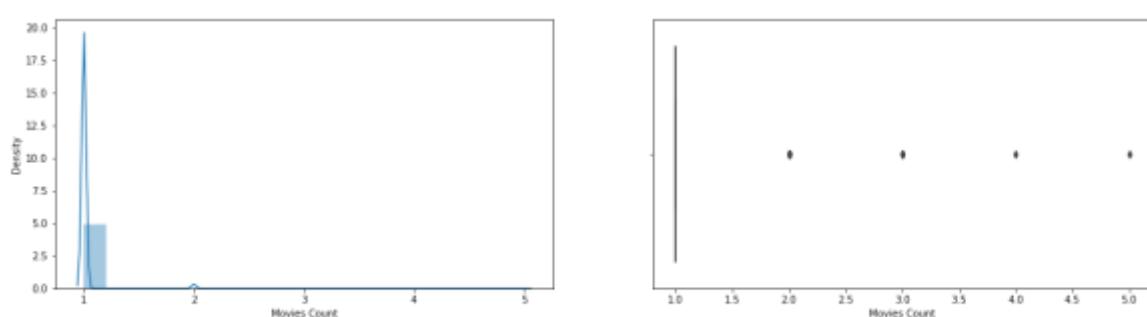
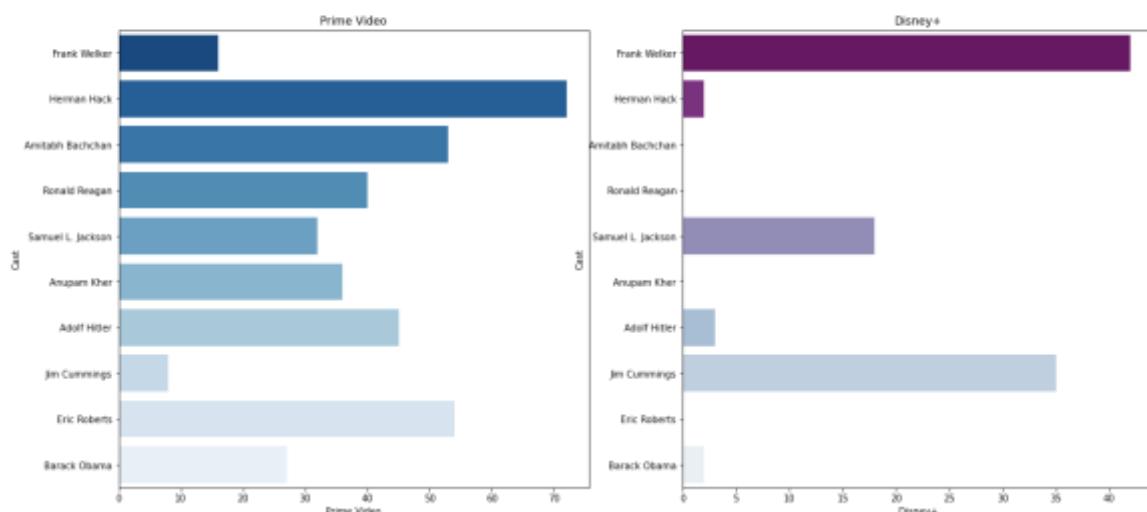
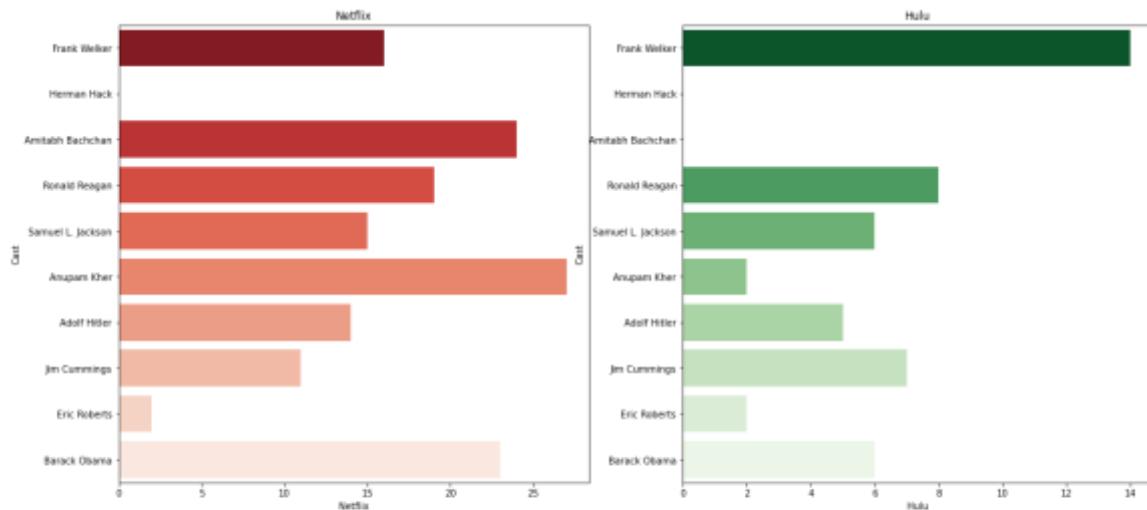
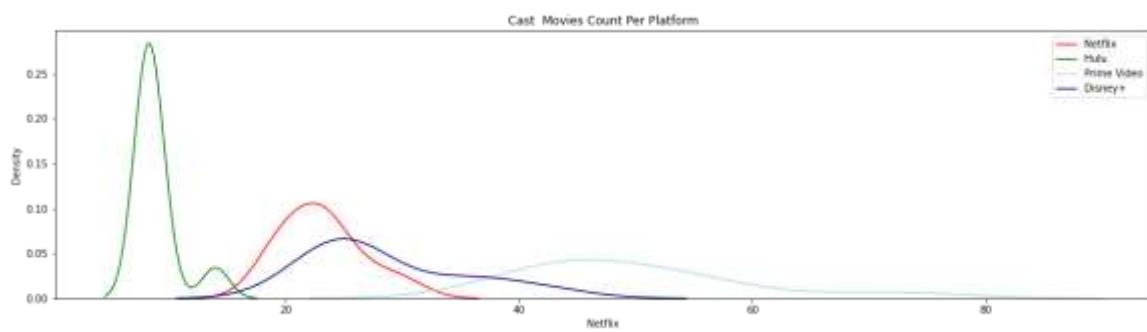


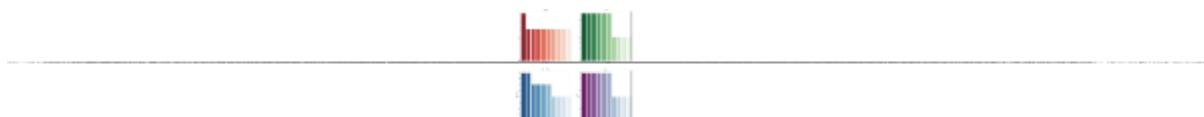
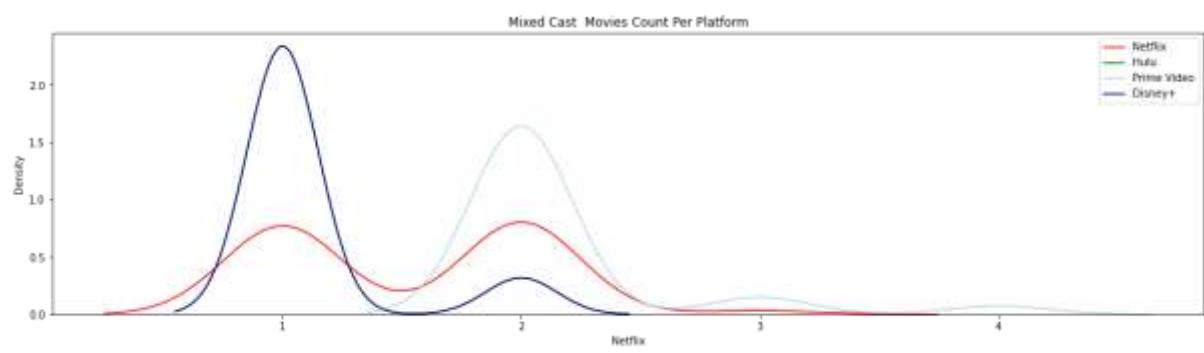
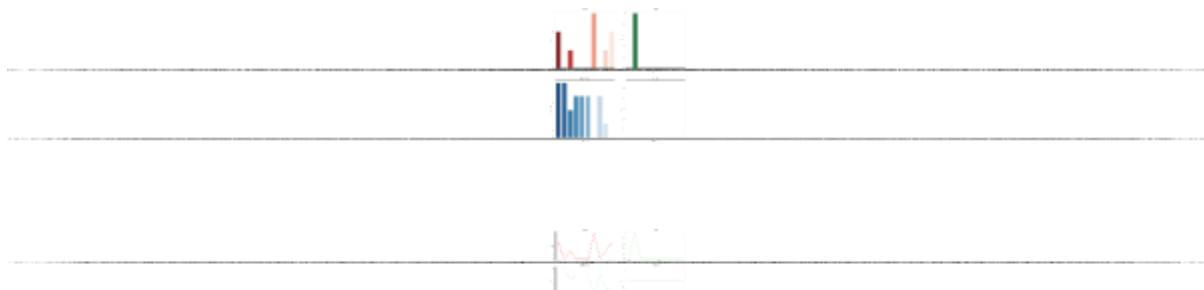
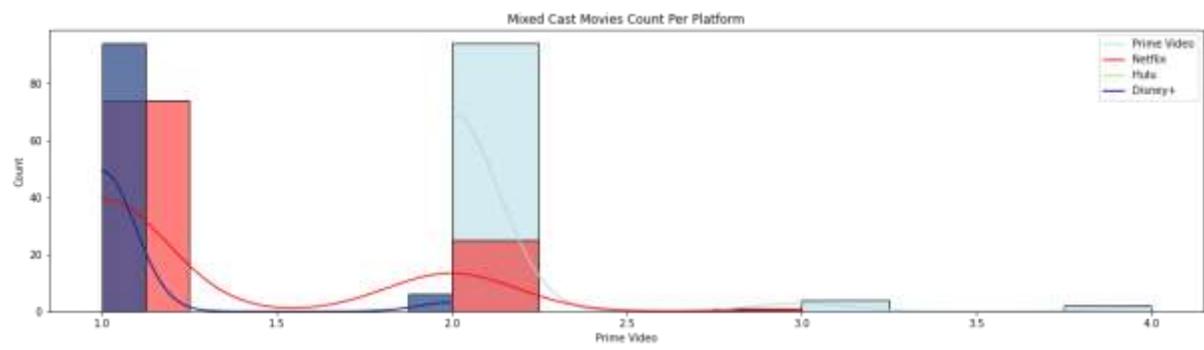
CAST



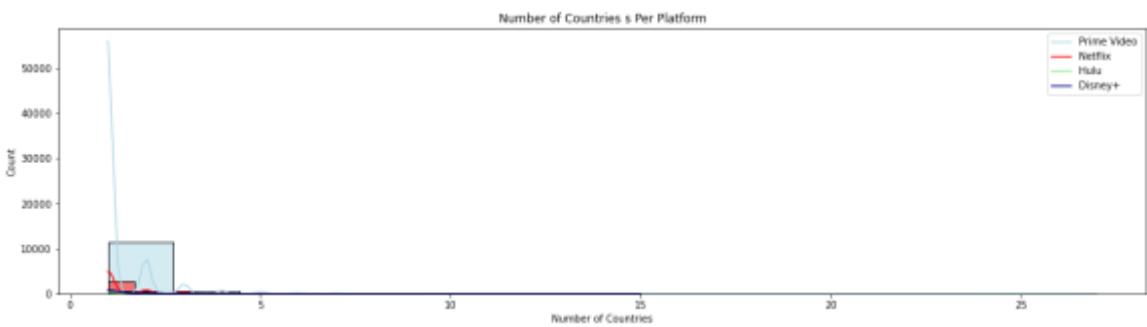
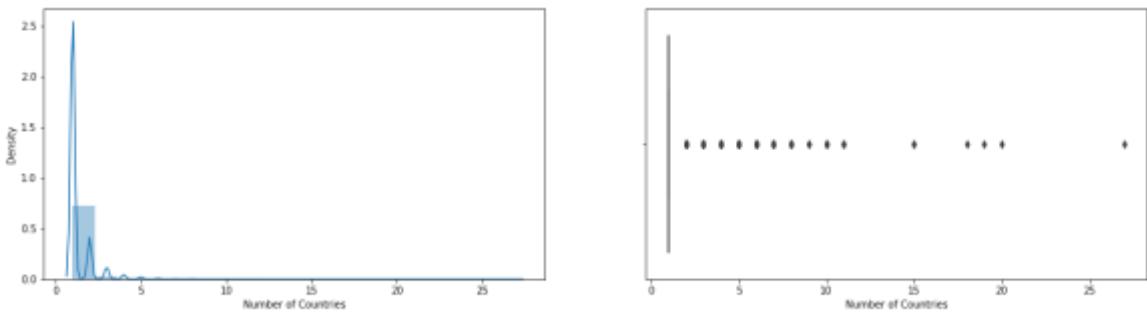
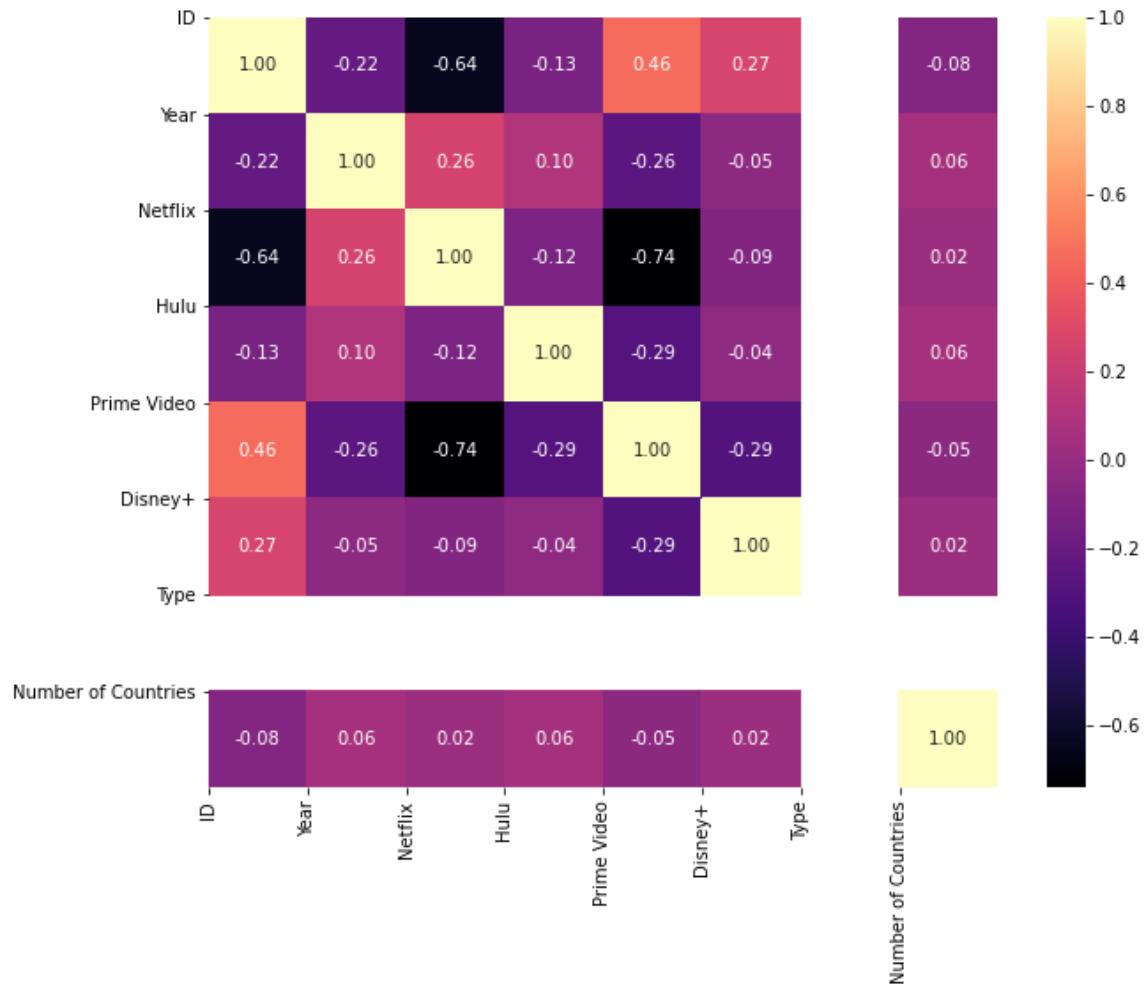


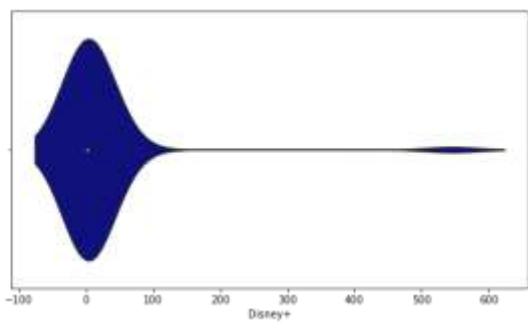
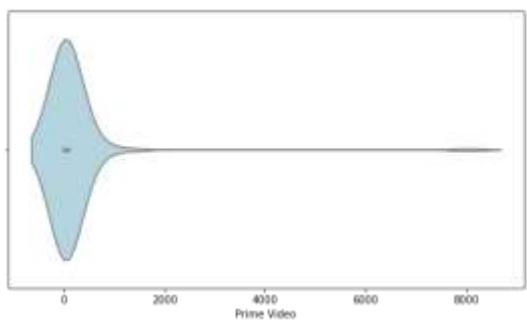
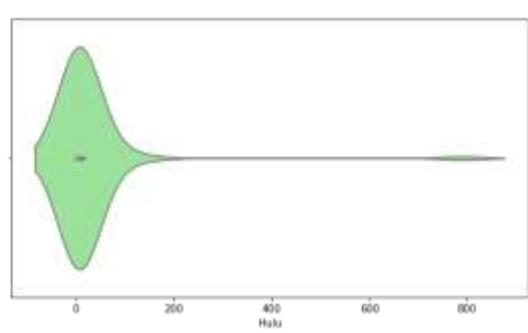
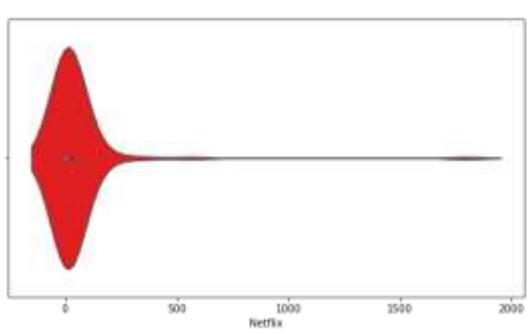
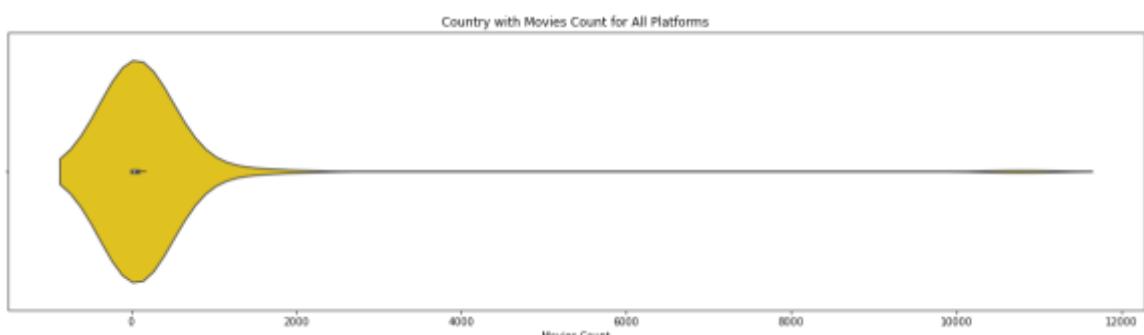
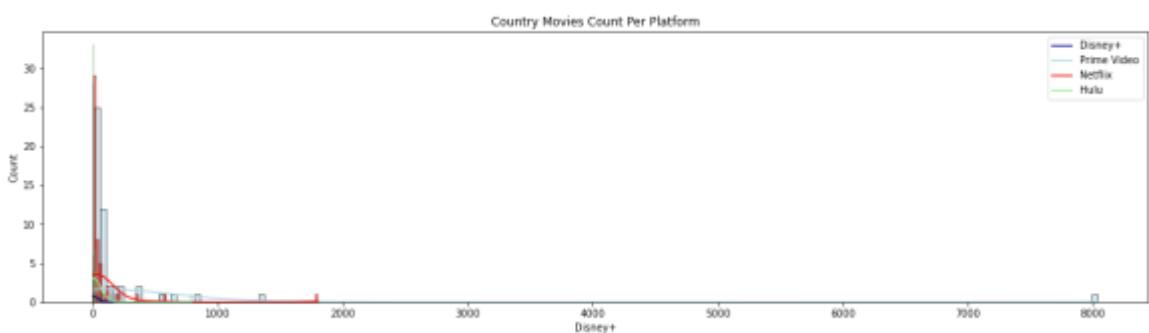
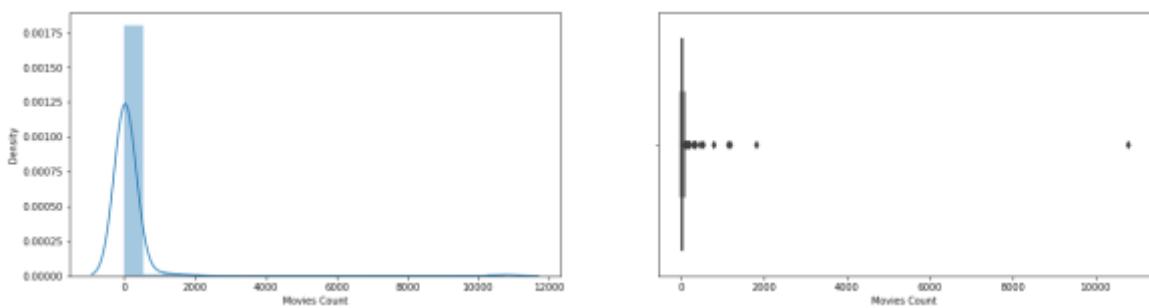


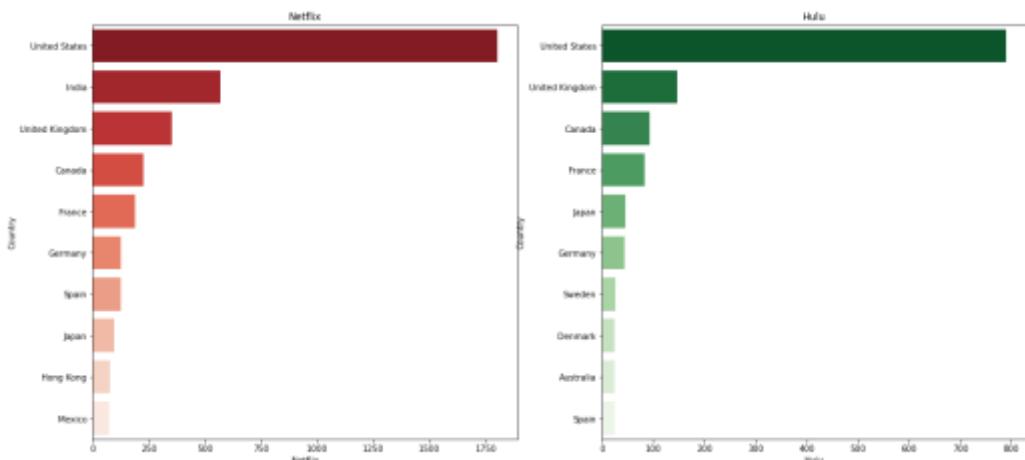
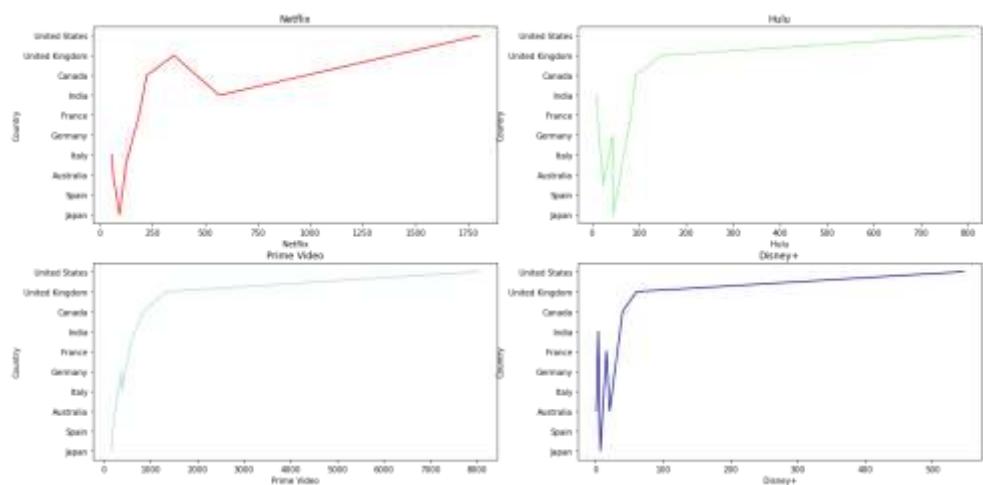
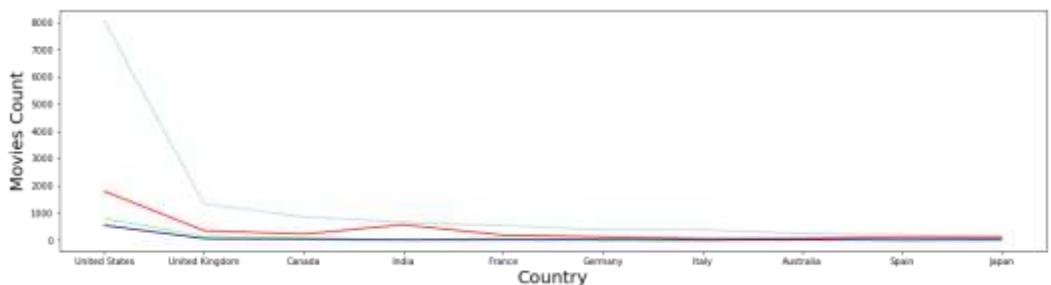


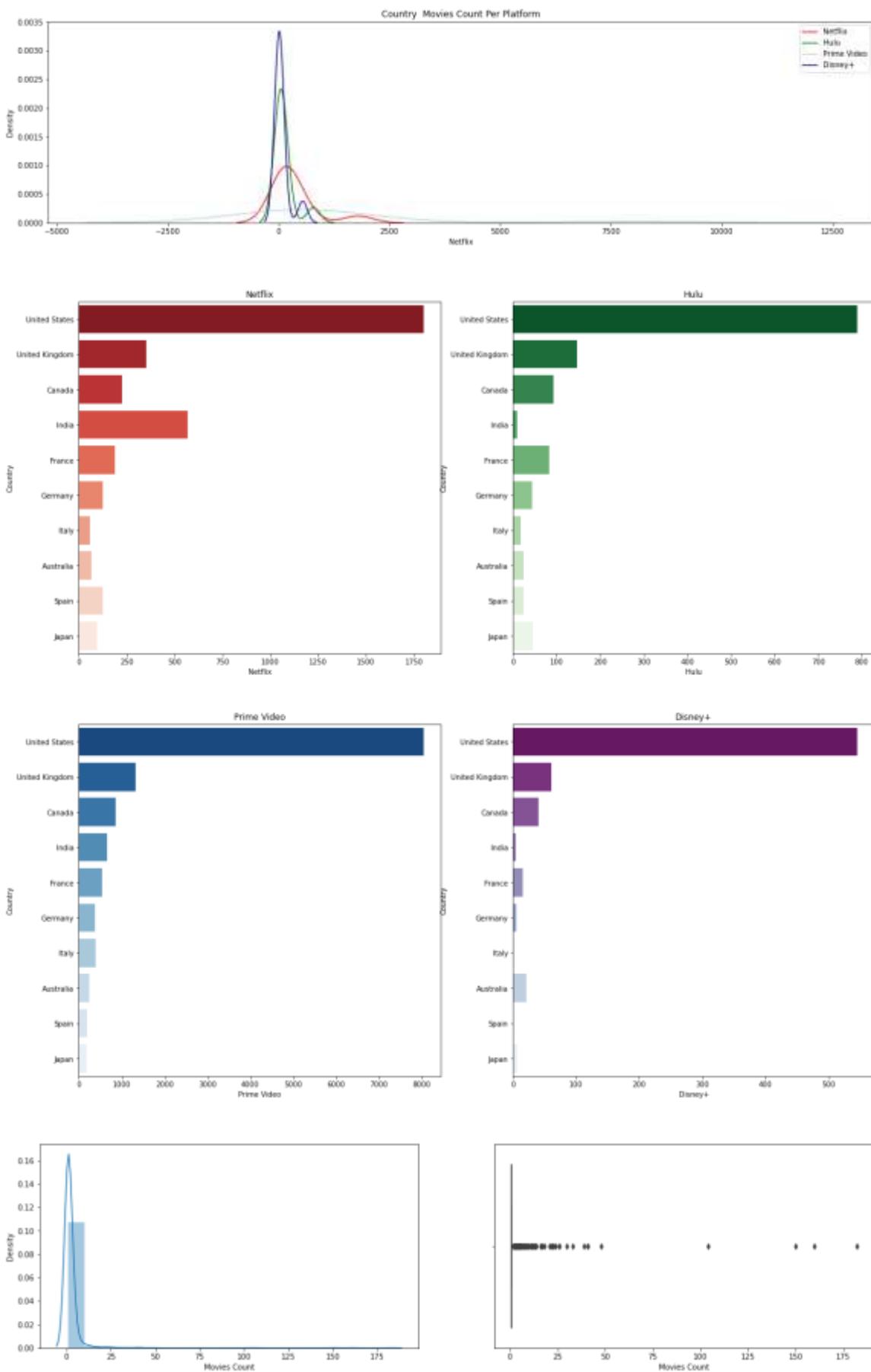


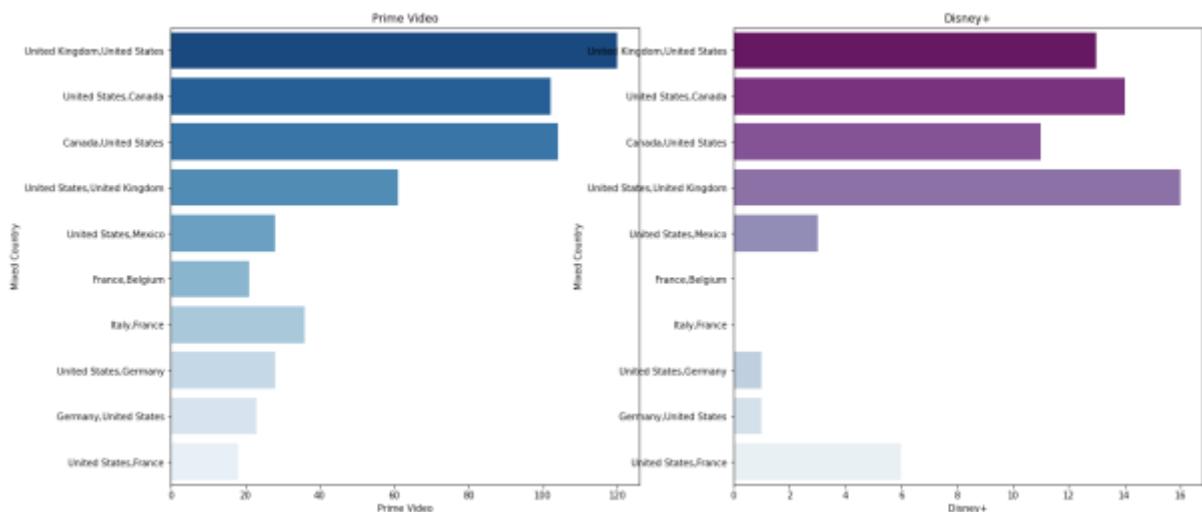
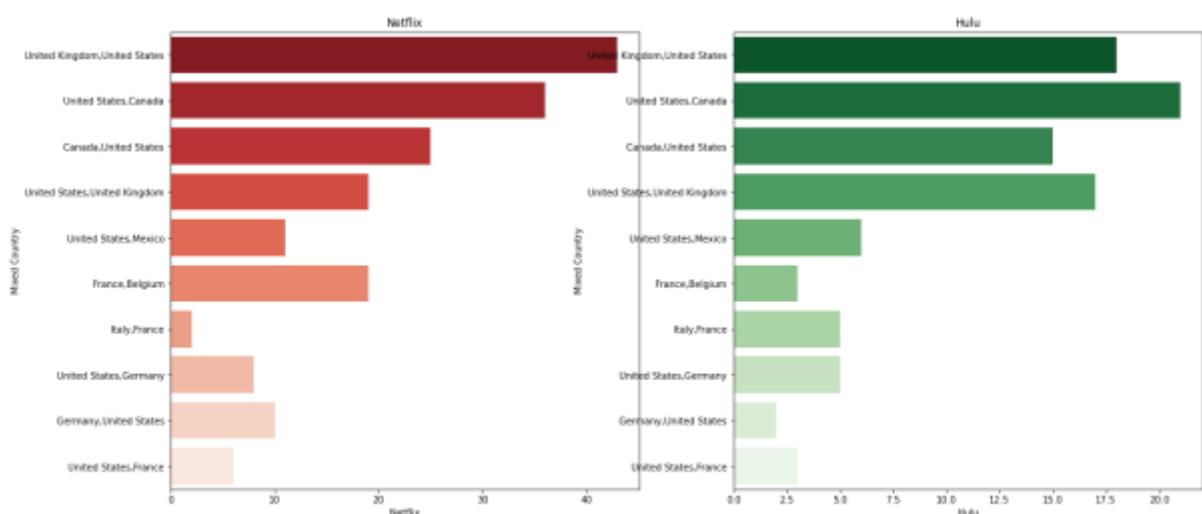
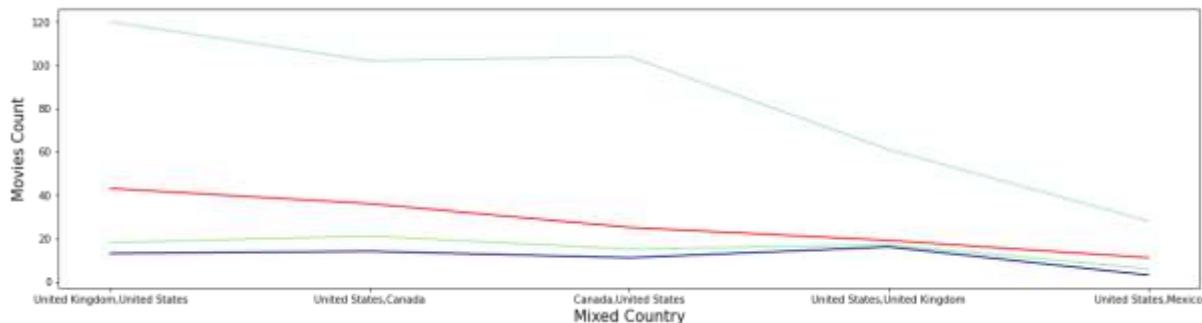
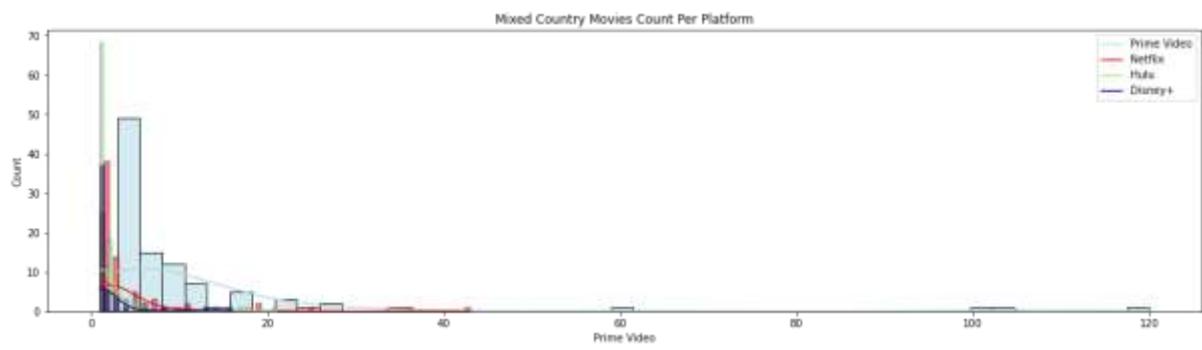
COUNTRY





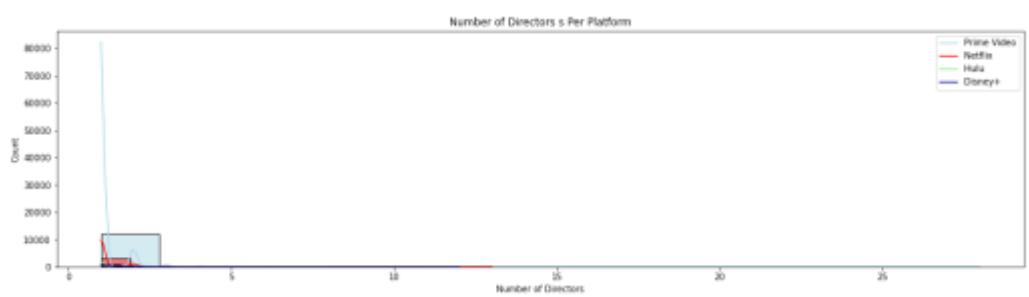
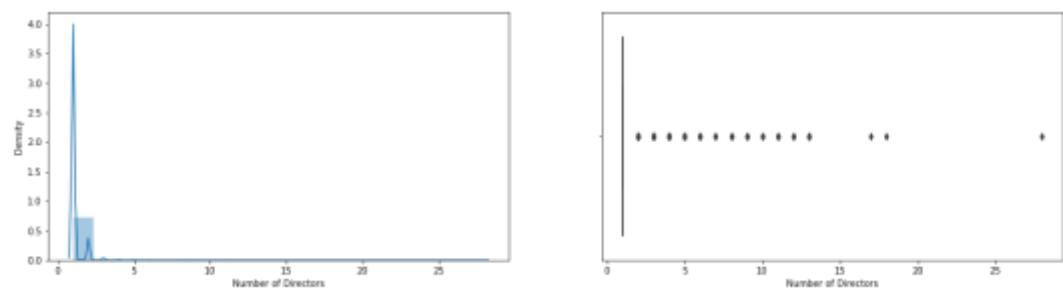
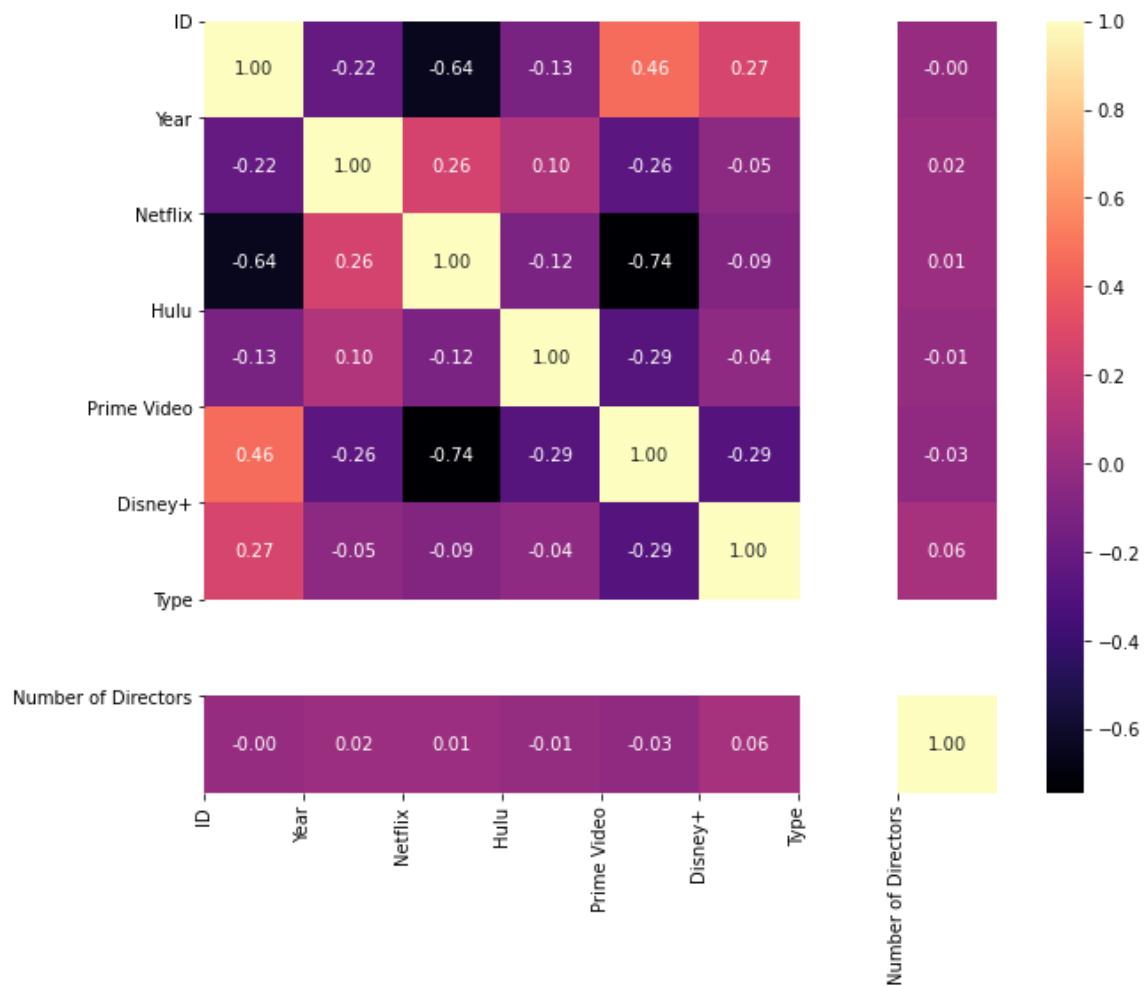


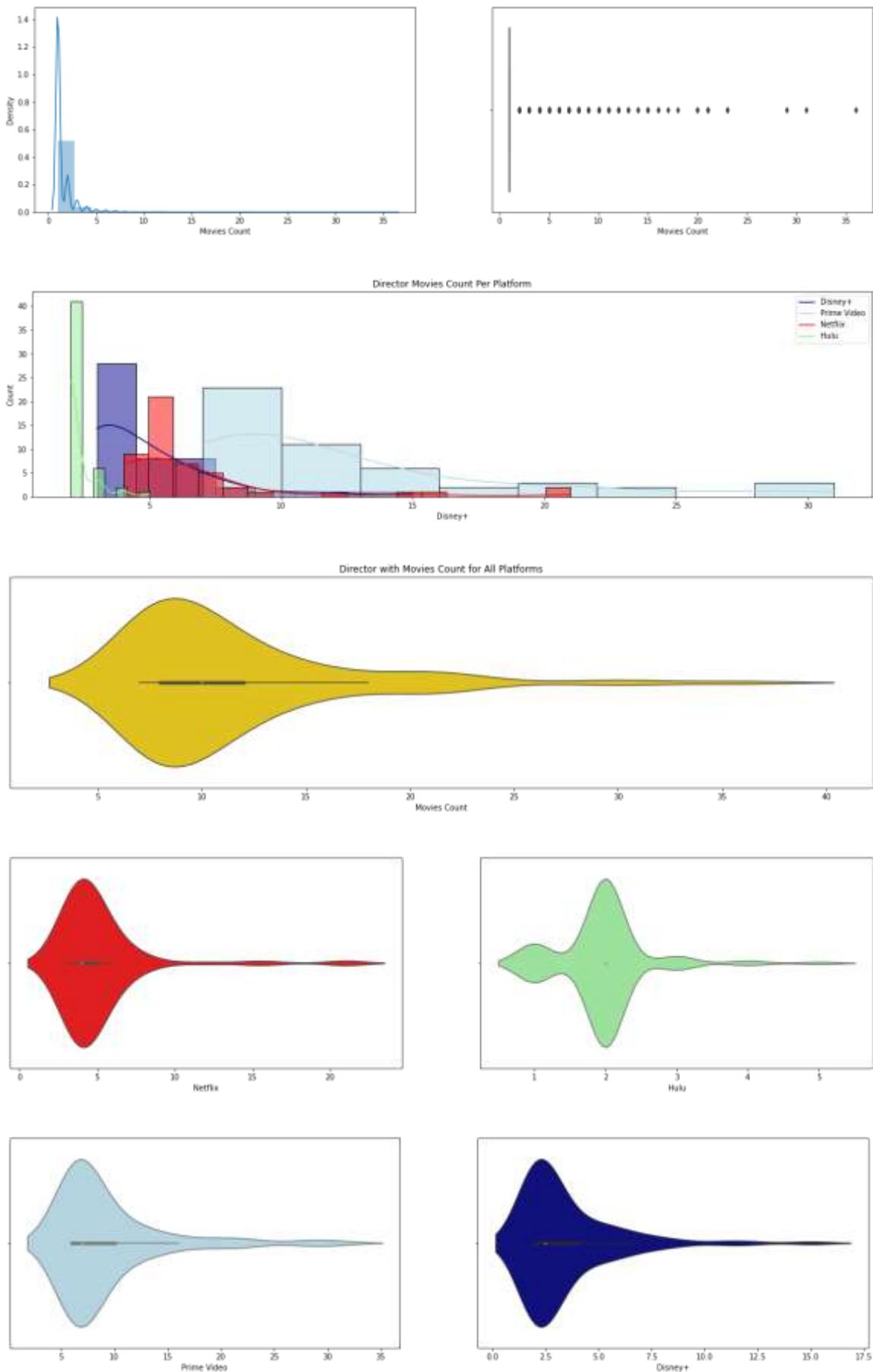


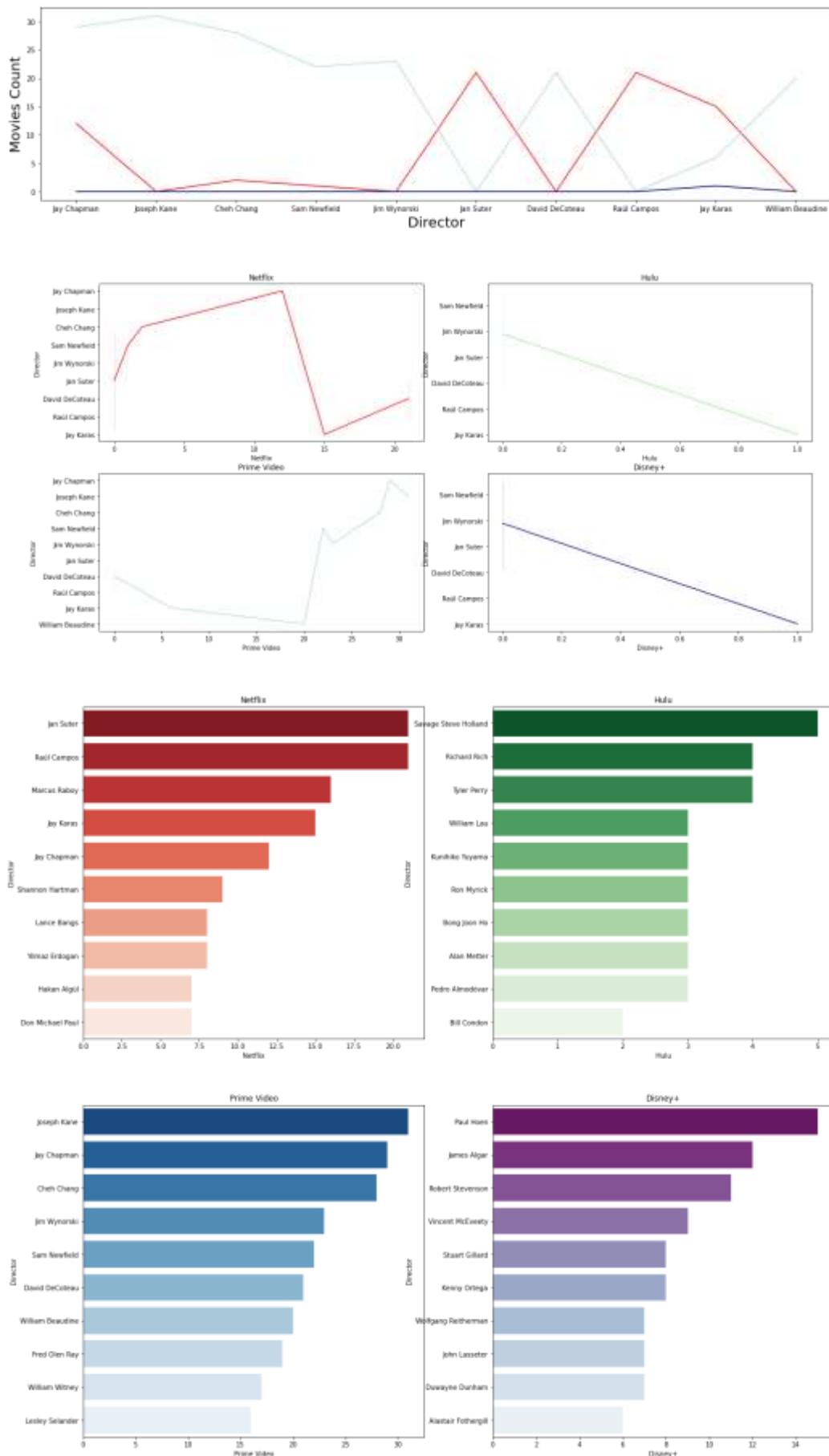


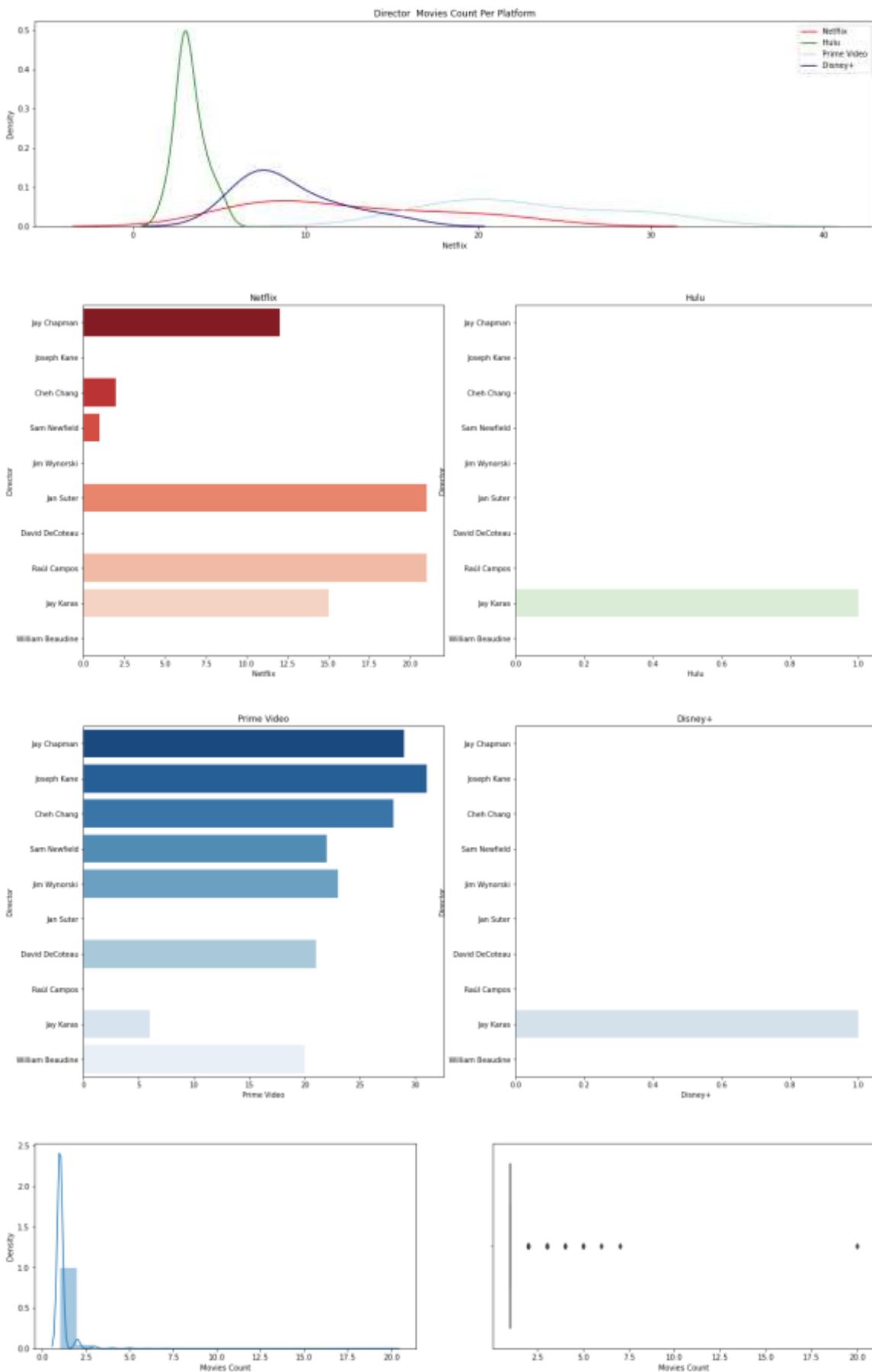


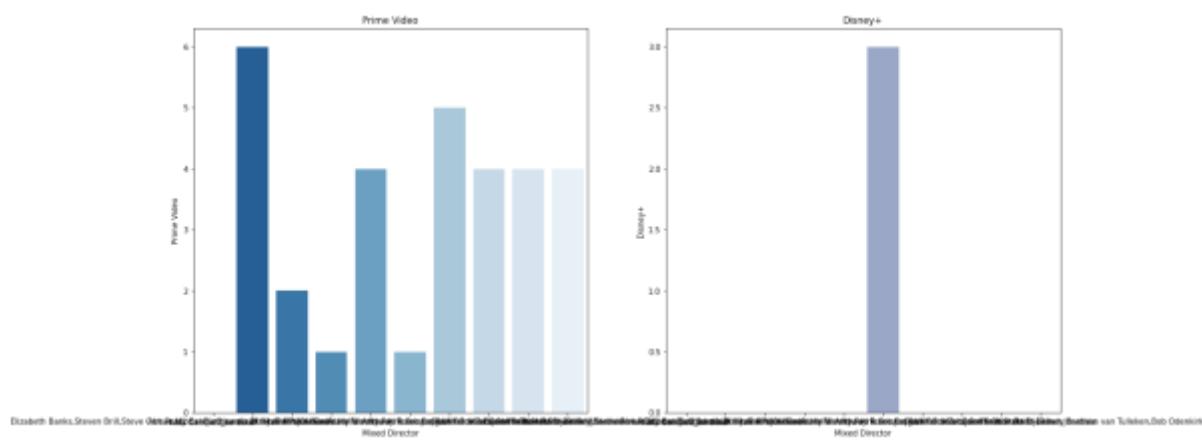
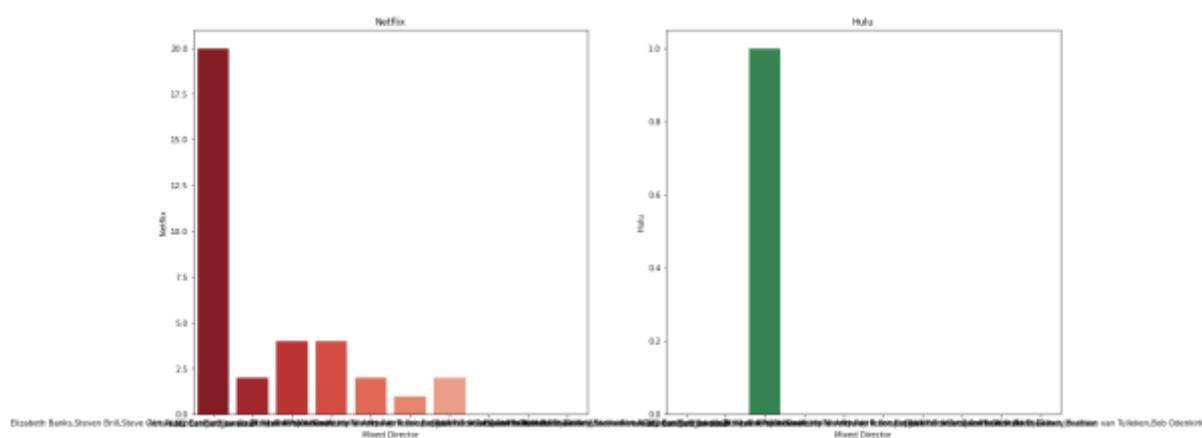
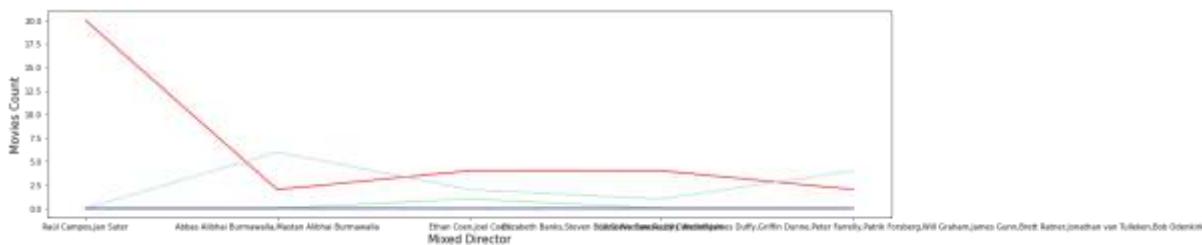
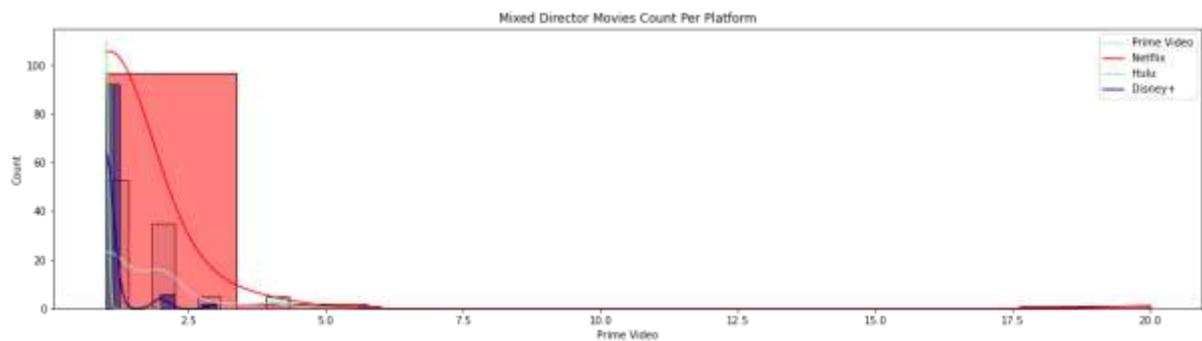
DIRECTOR

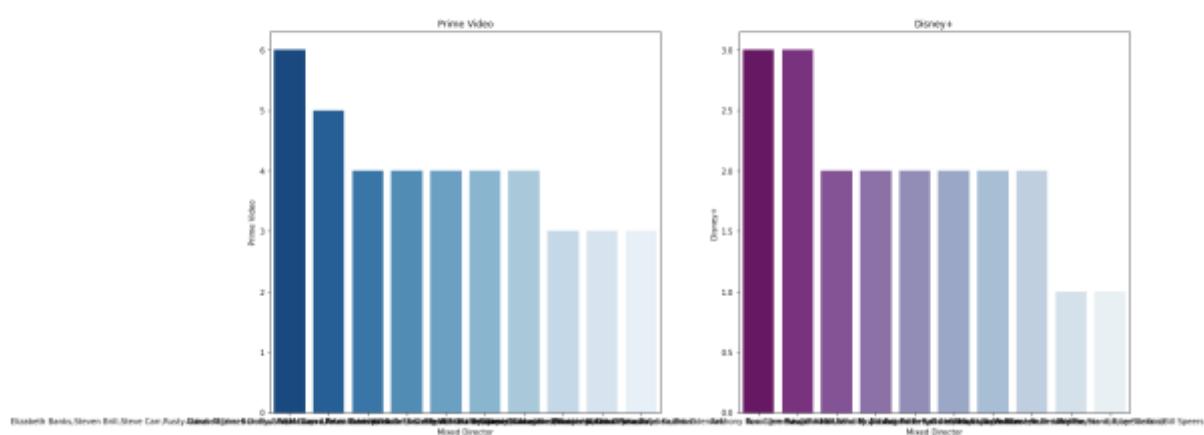
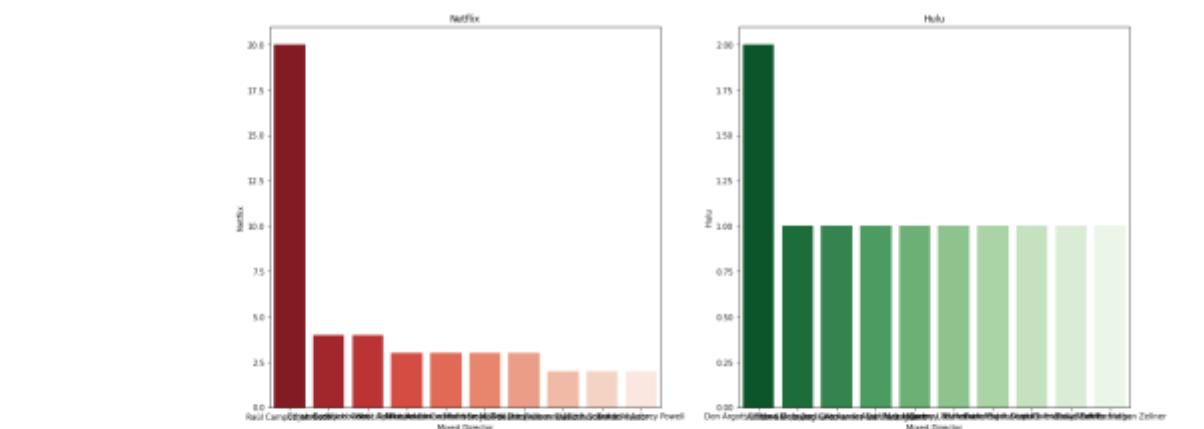
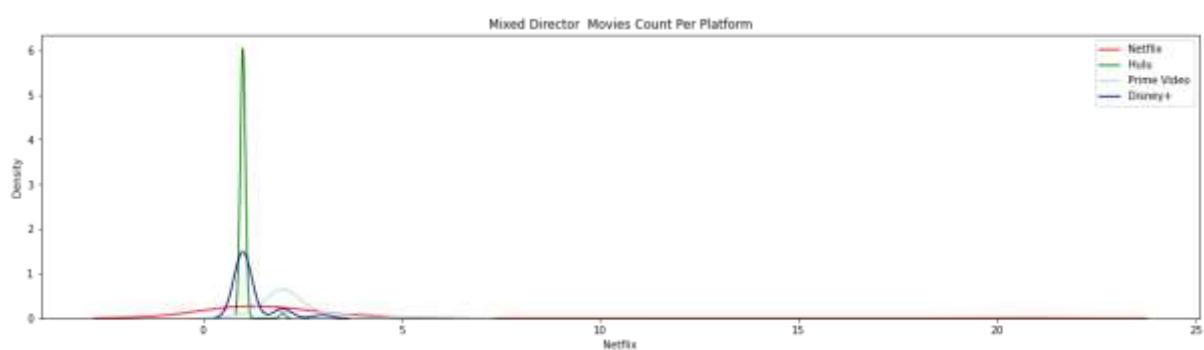
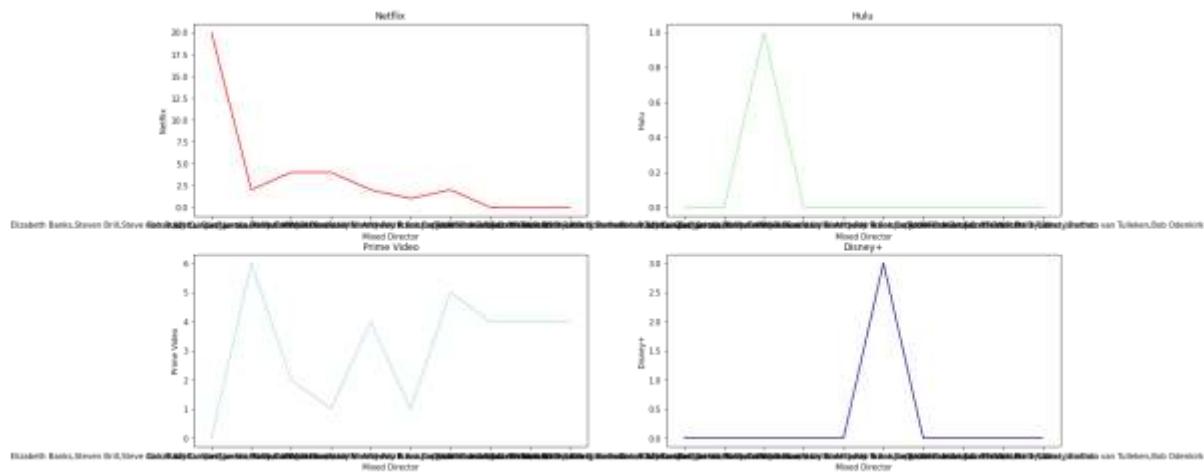


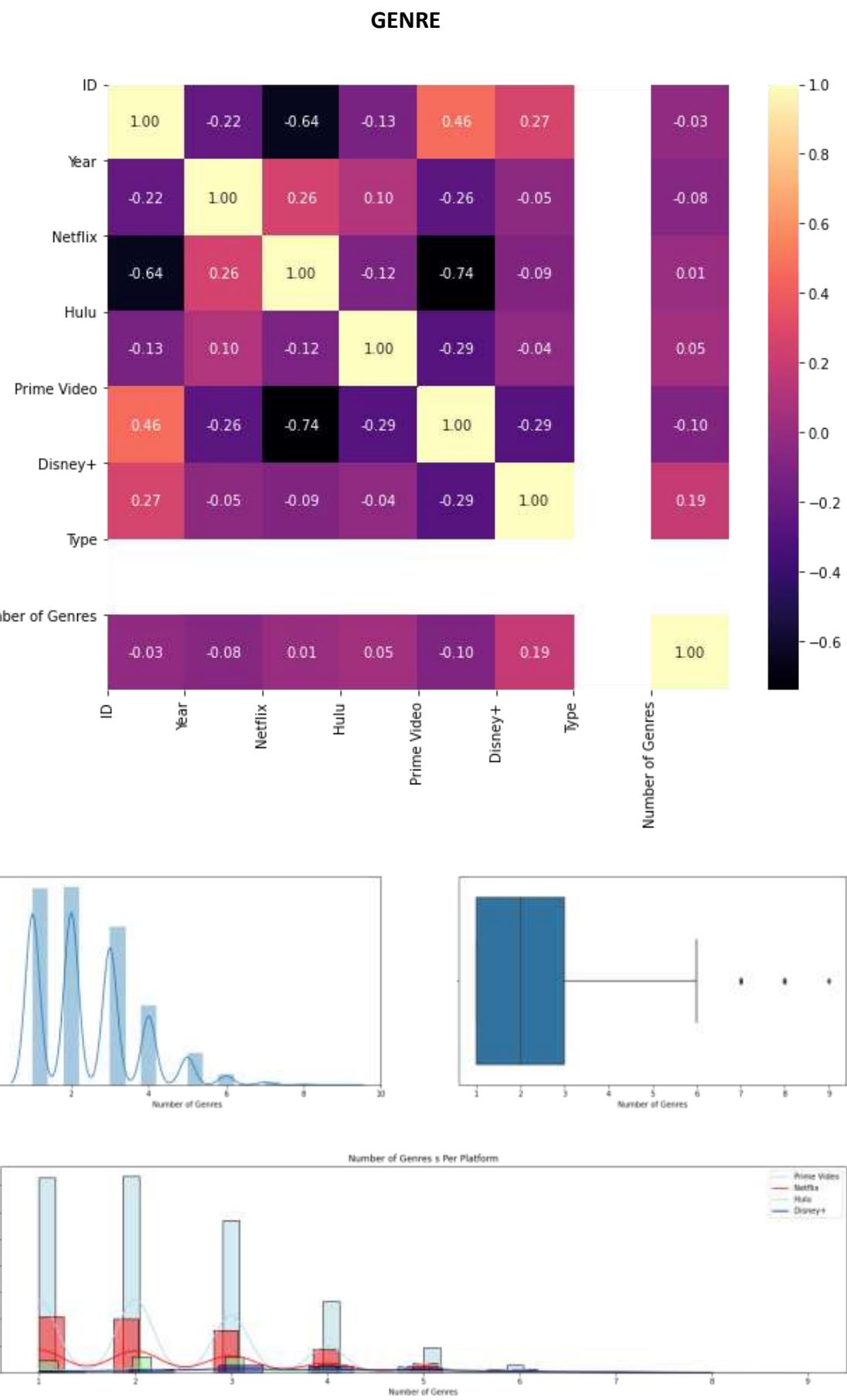


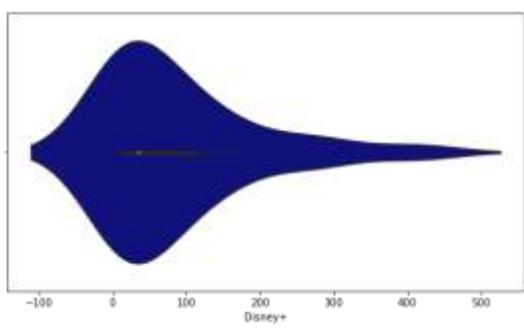
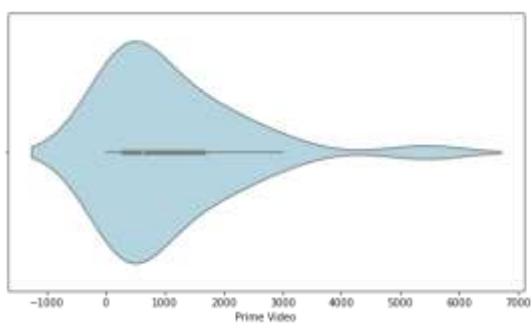
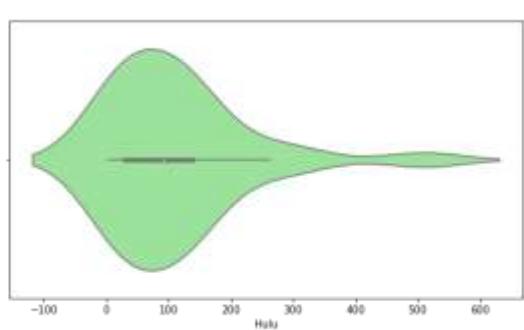
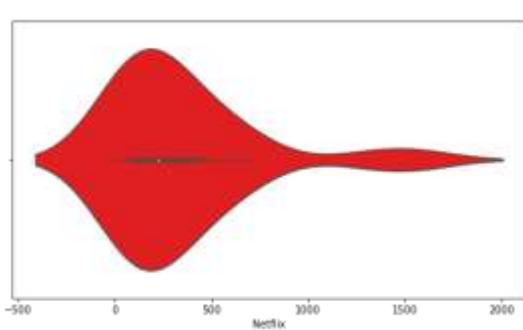
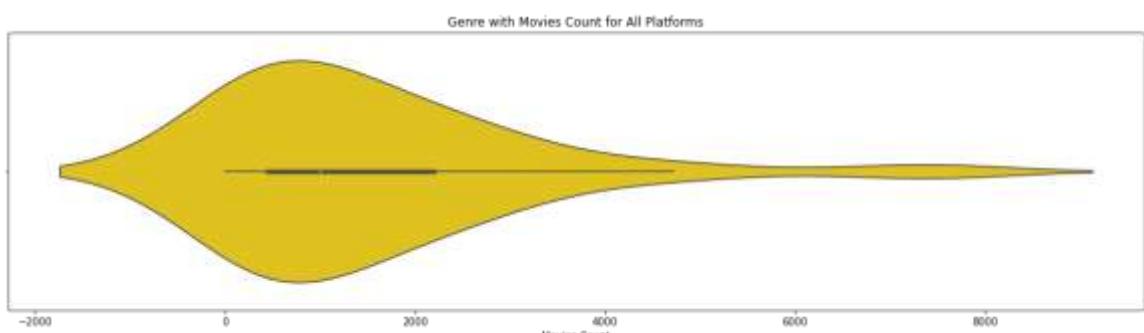
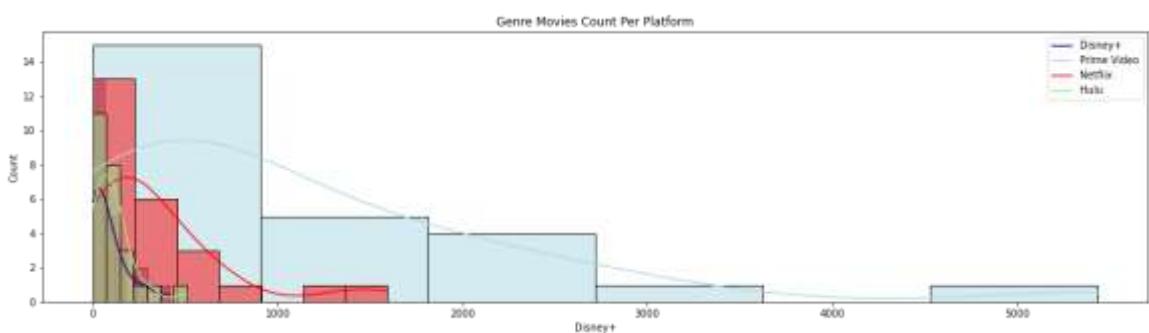
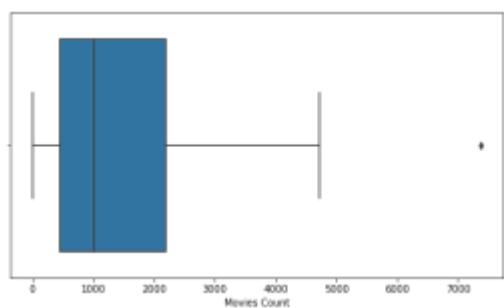
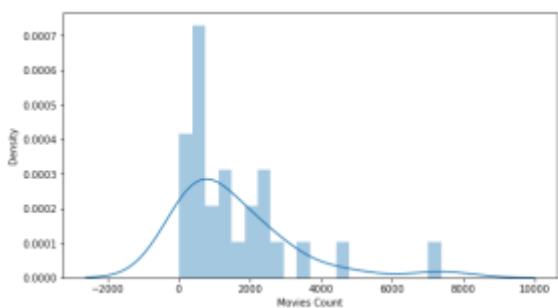


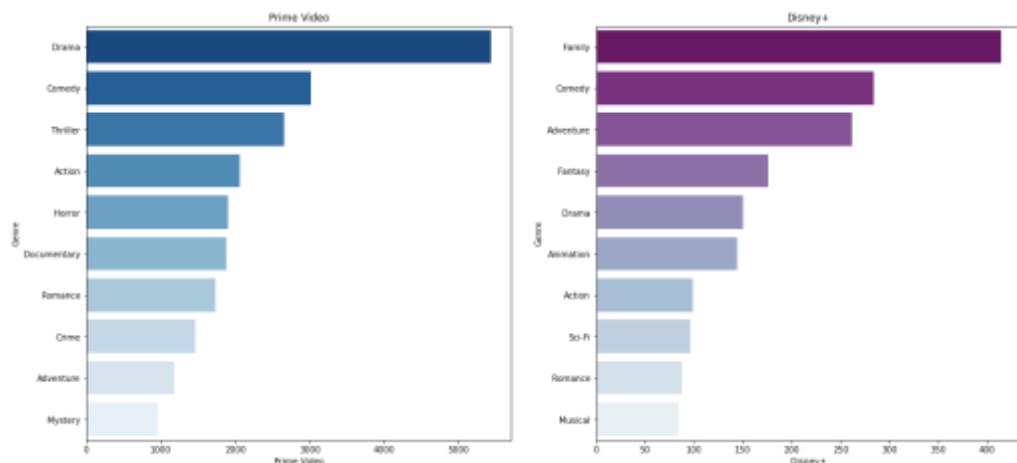
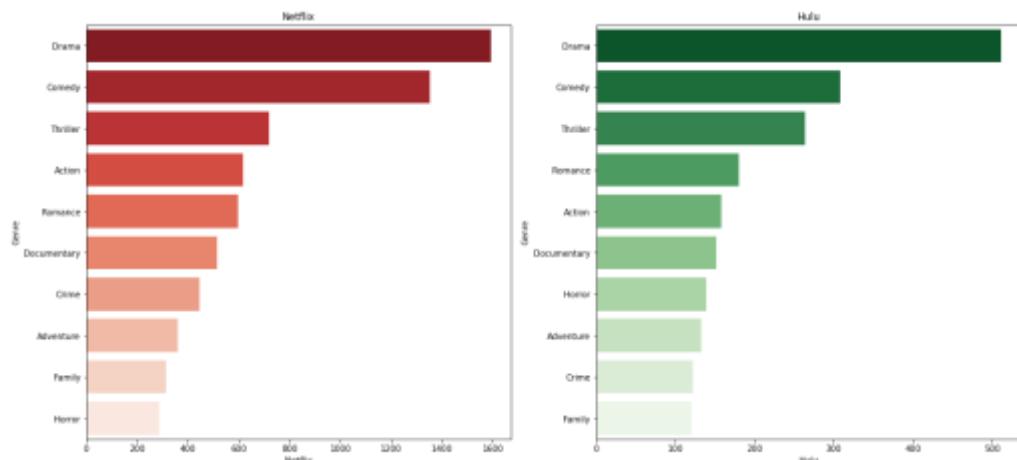
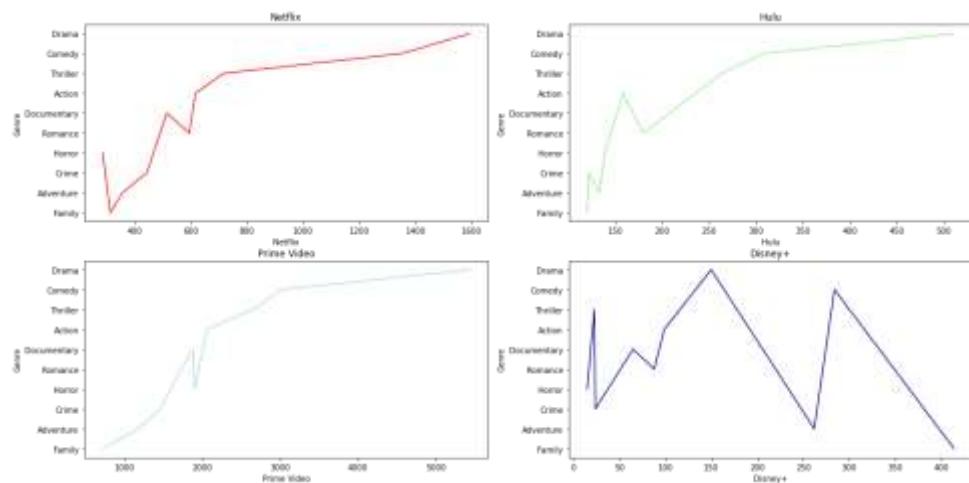
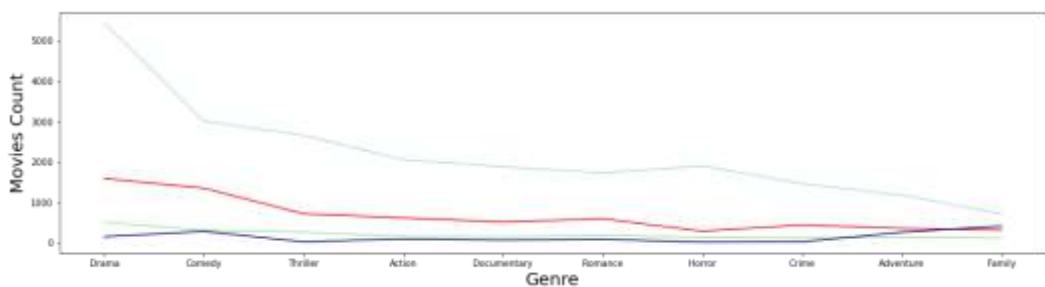


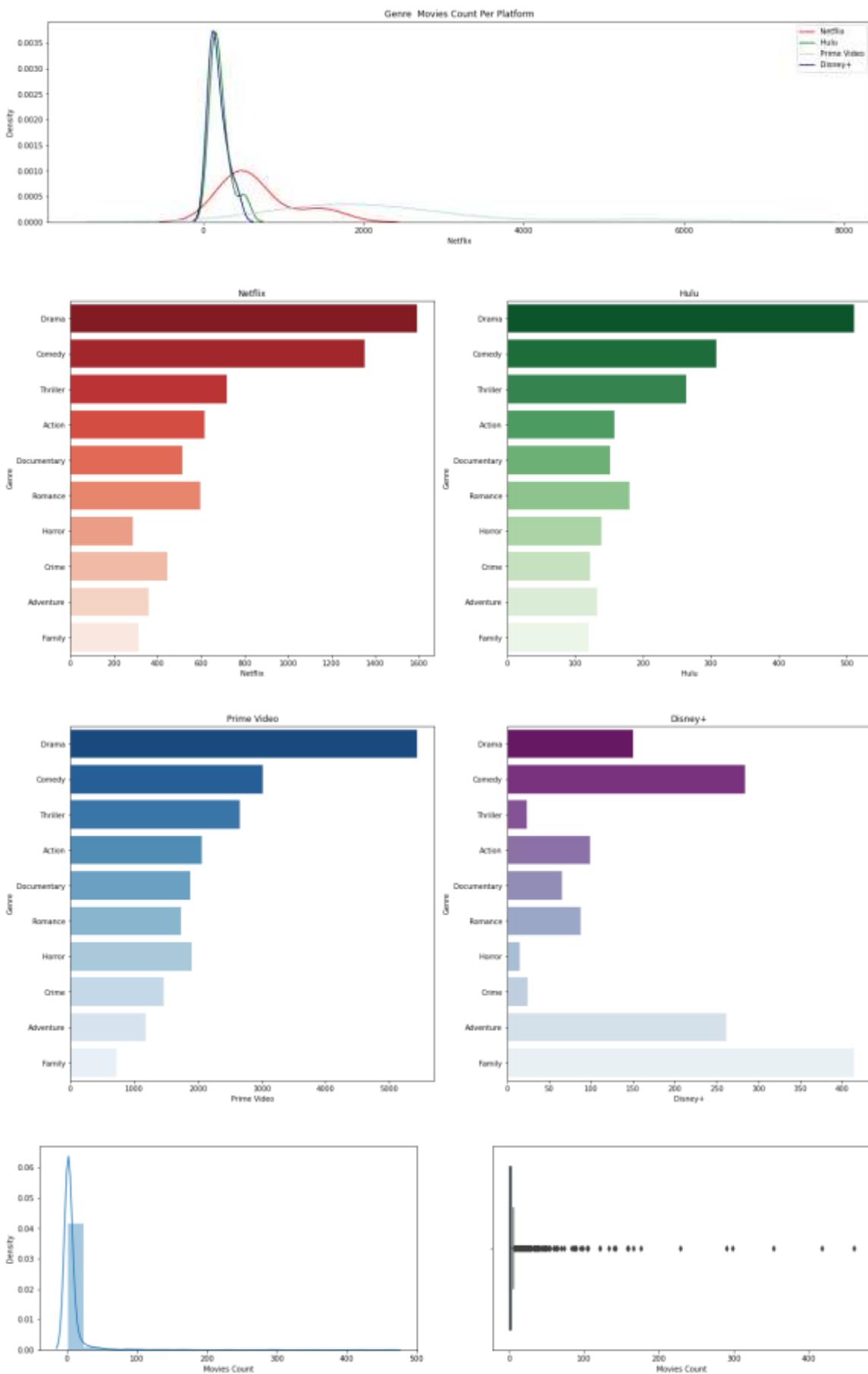


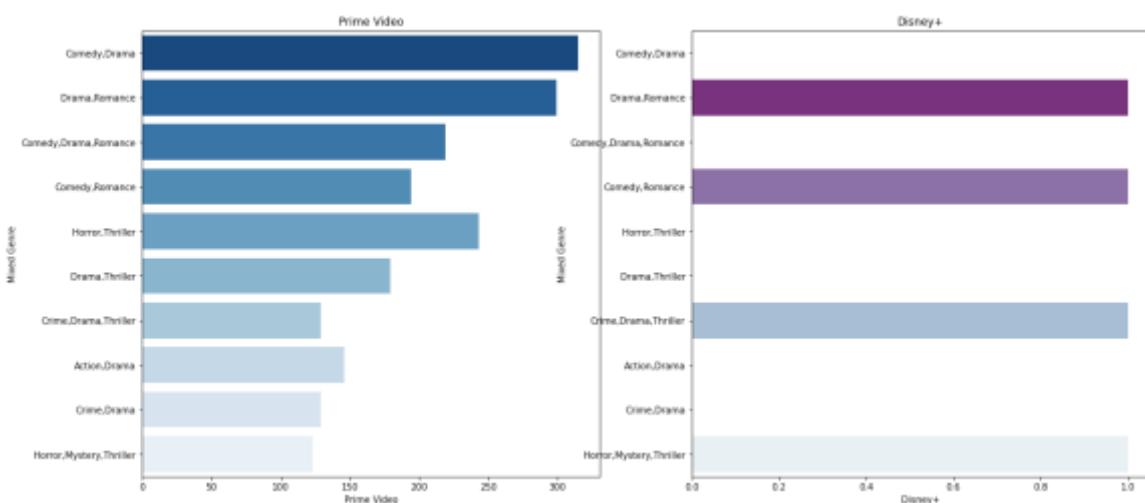
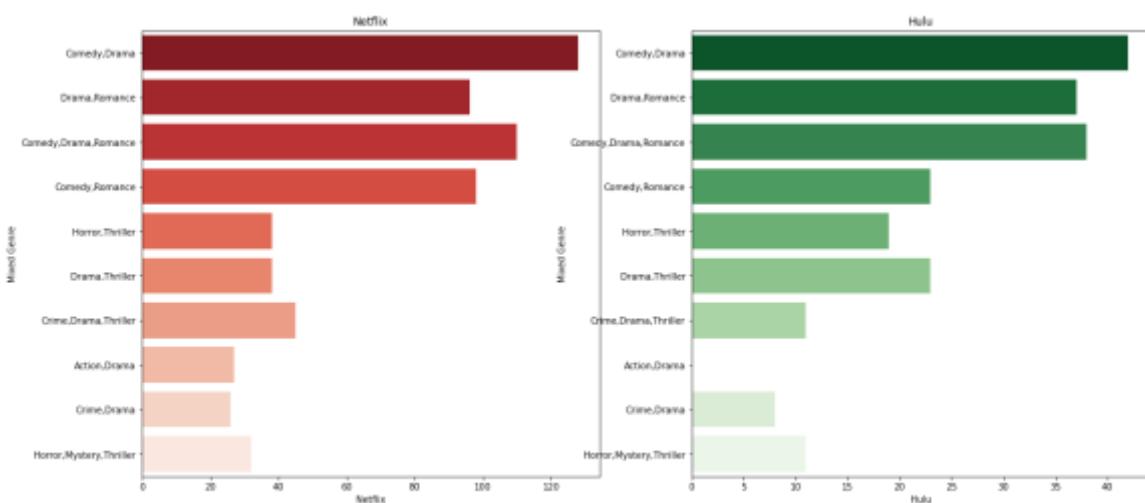
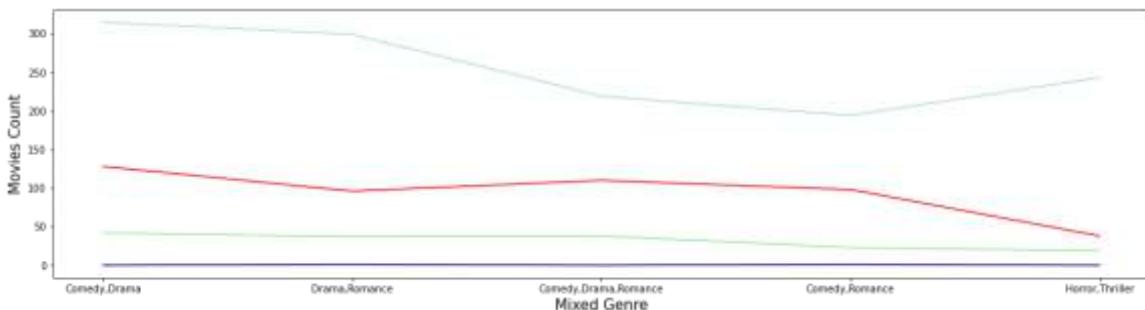
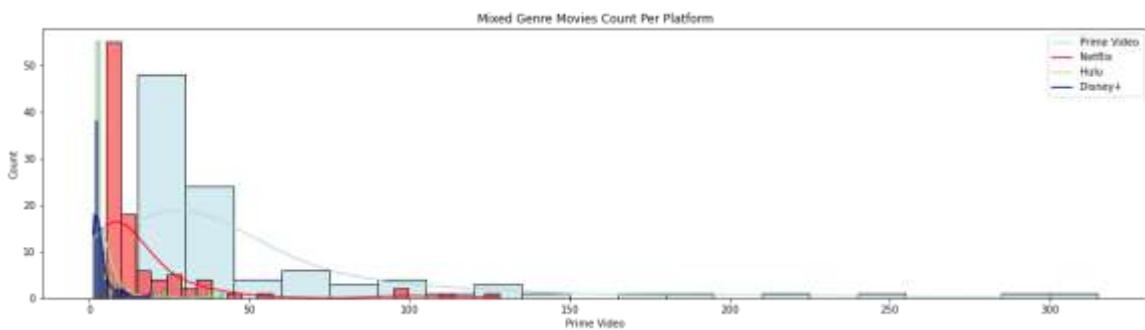


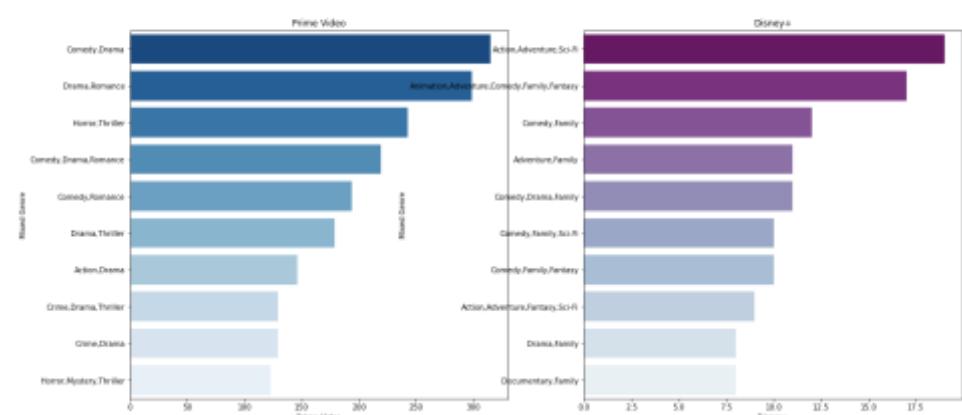
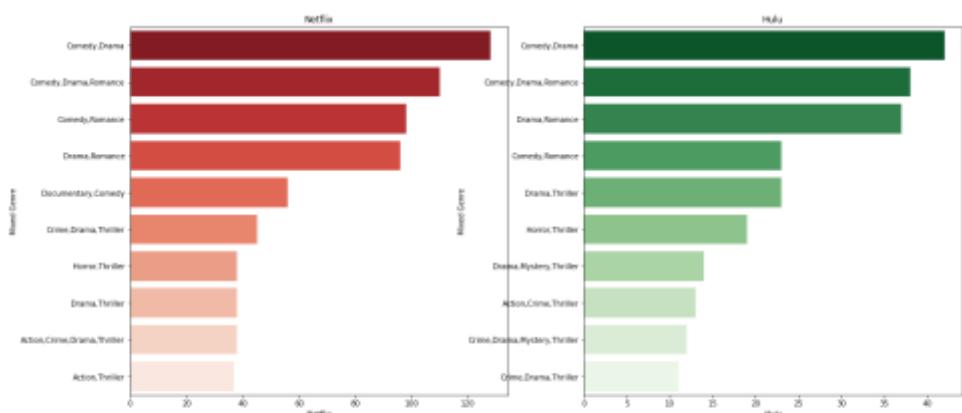
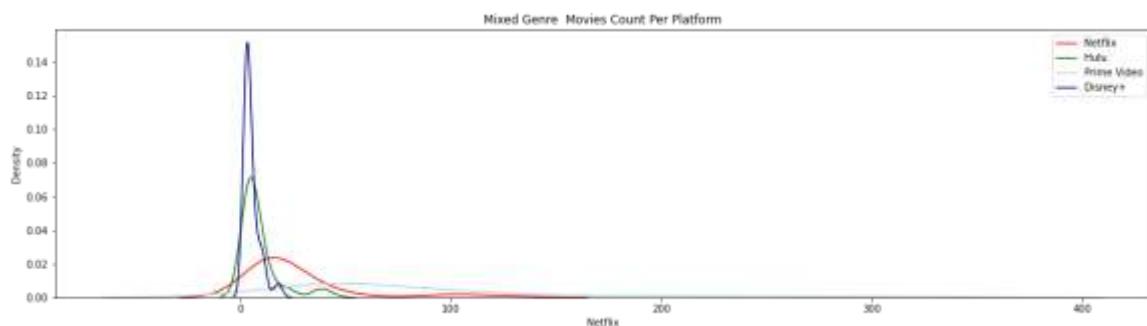
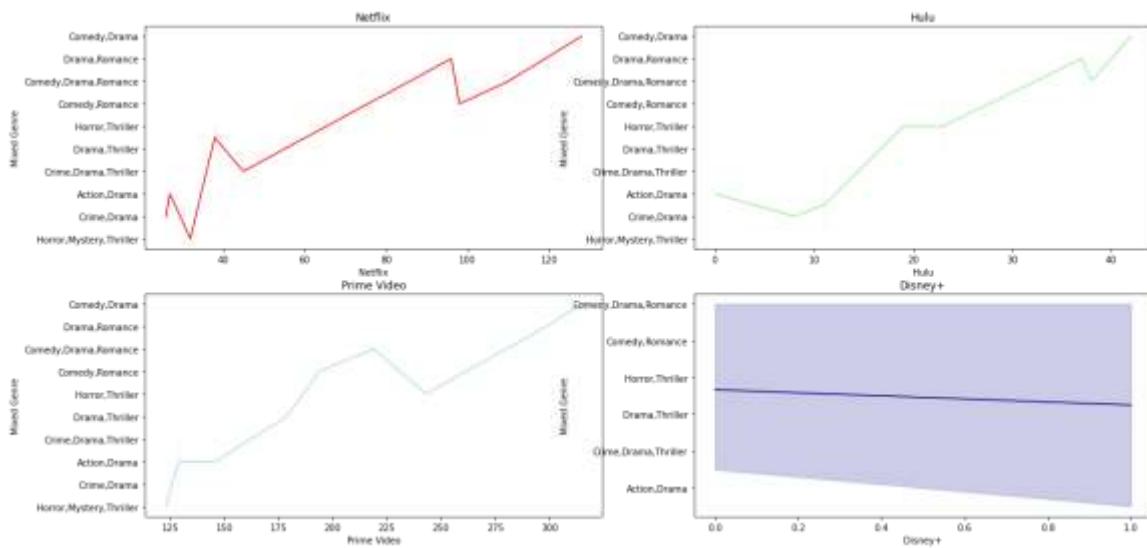




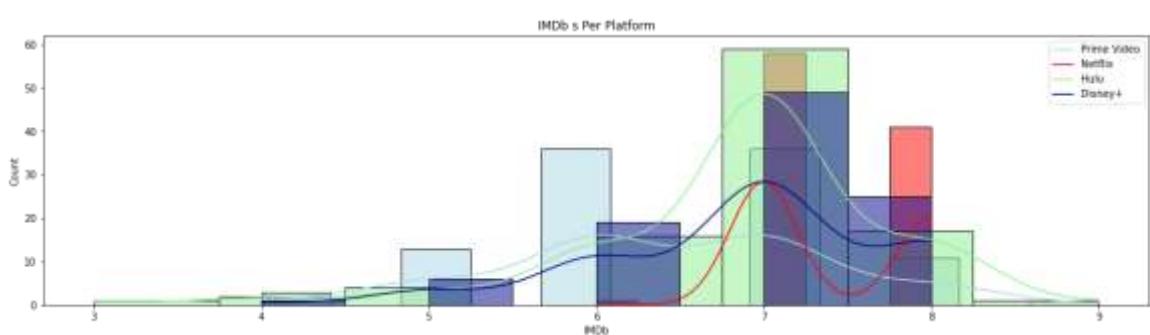
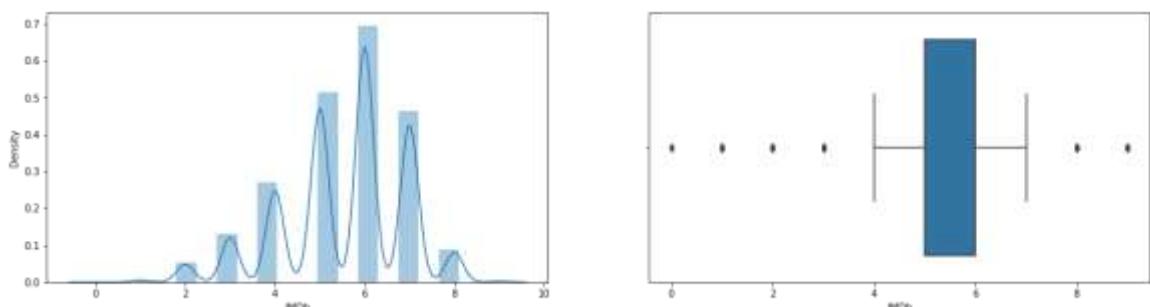


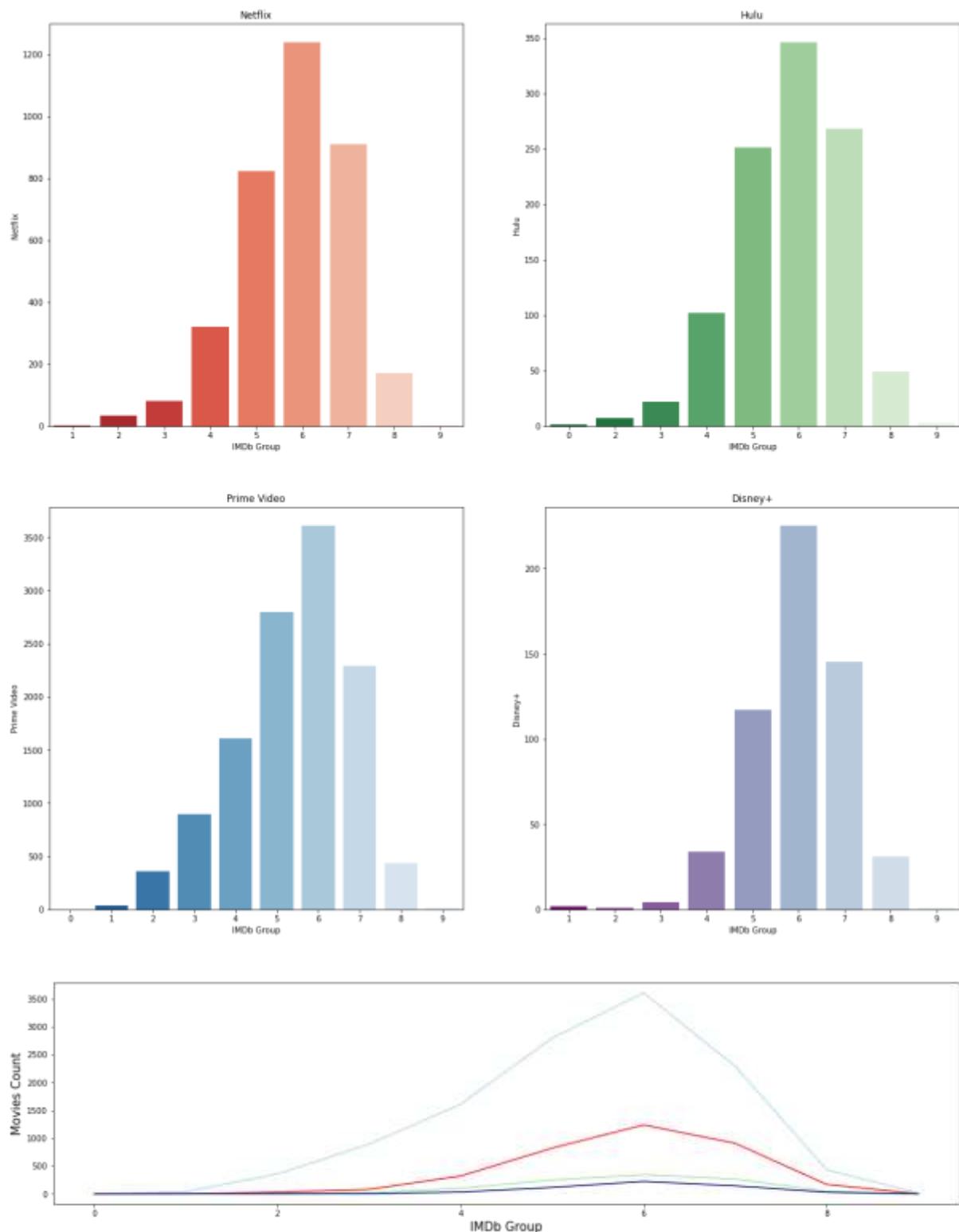


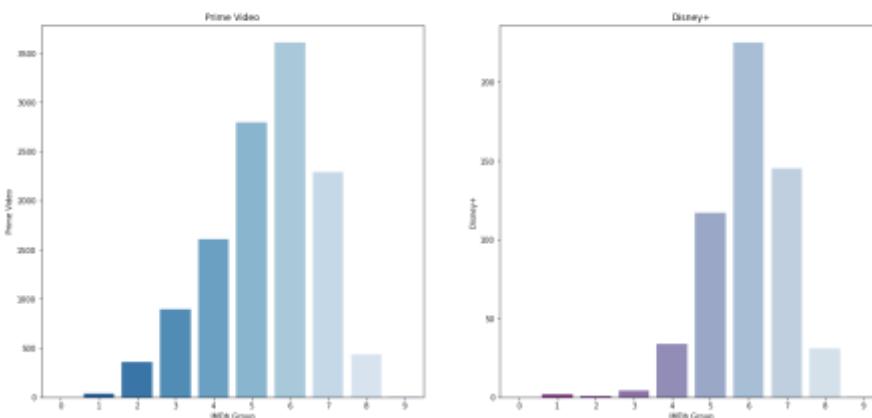
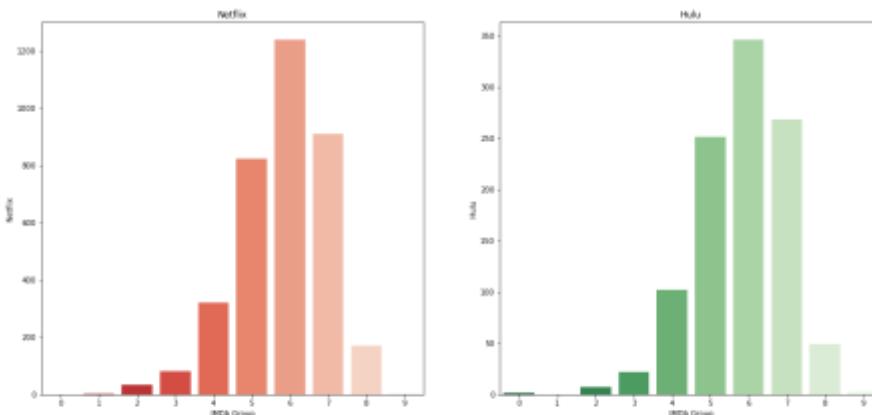
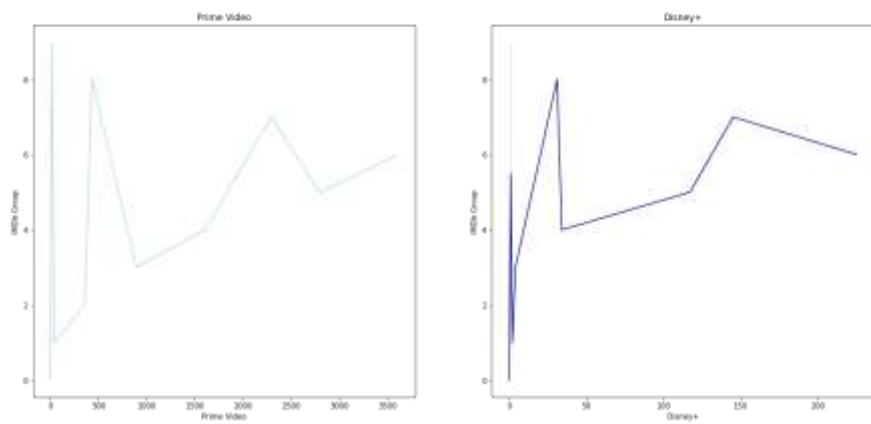
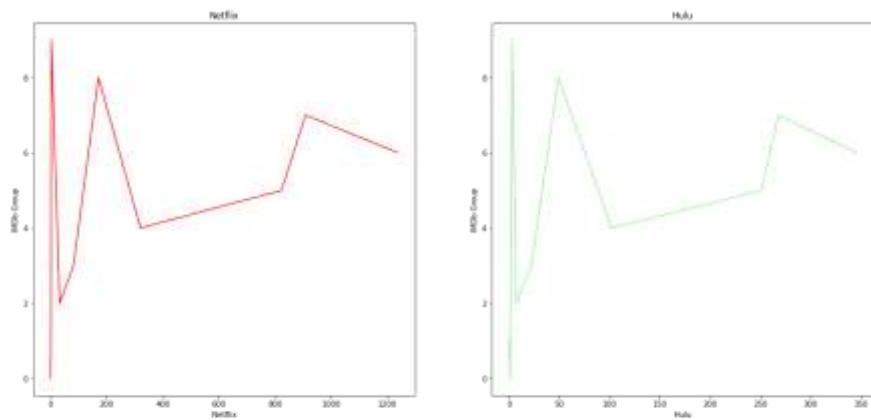




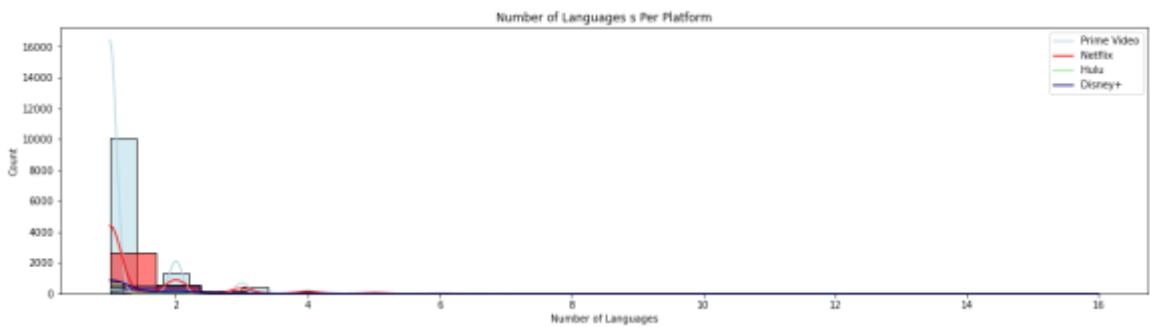
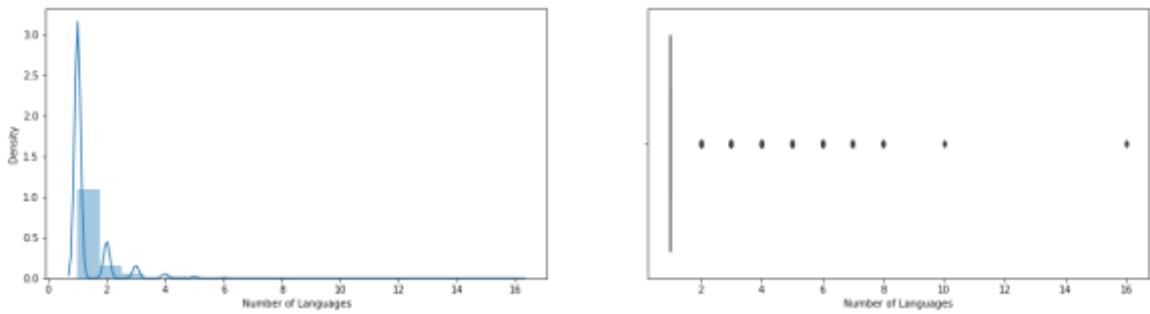
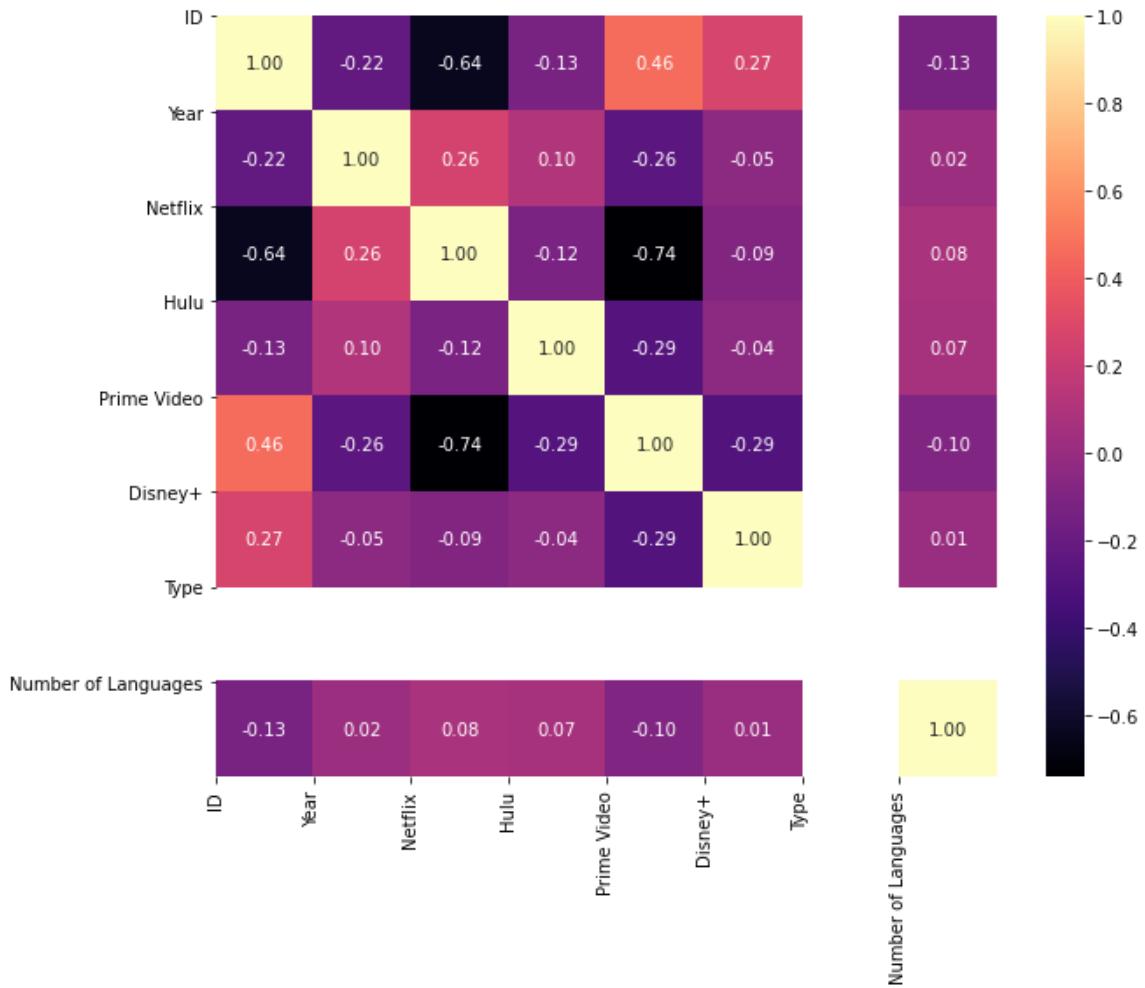
IMDB

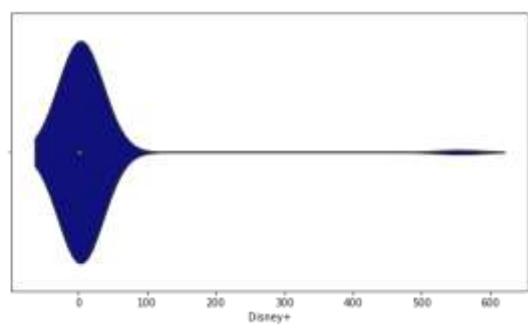
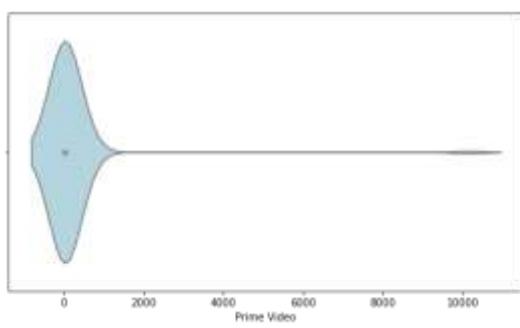
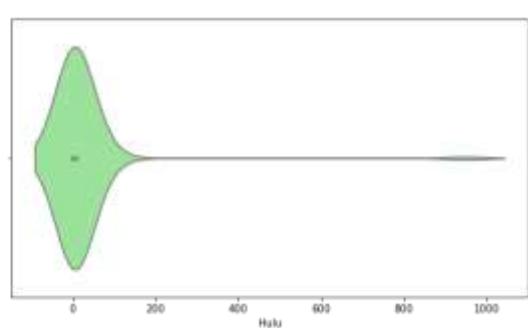
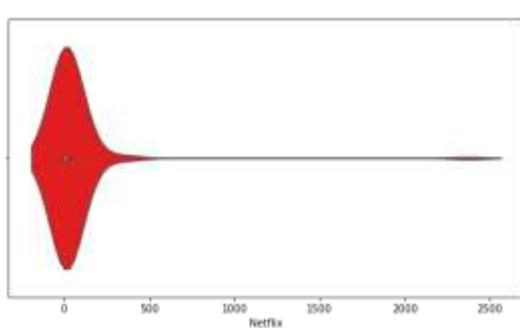
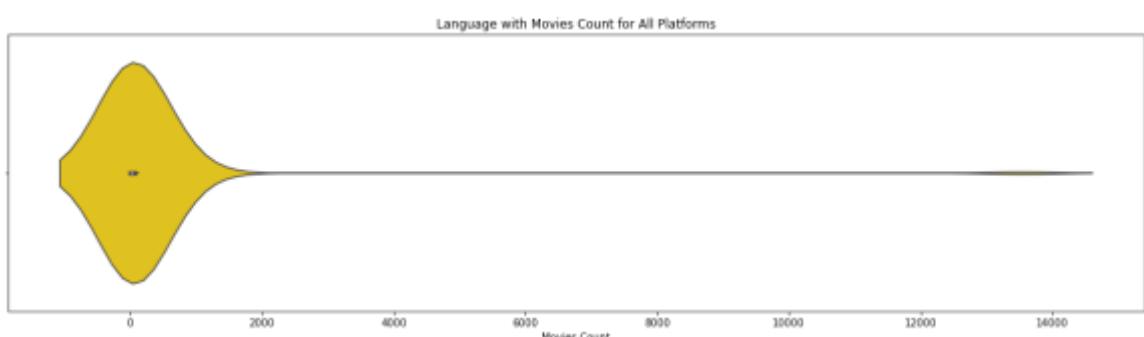
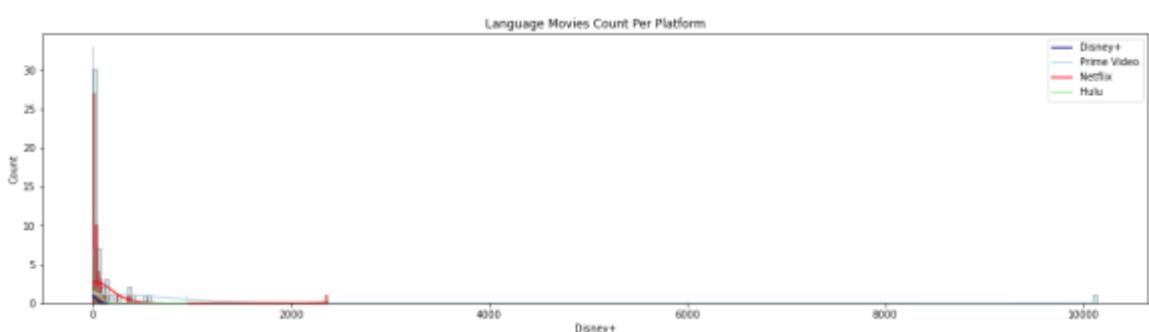
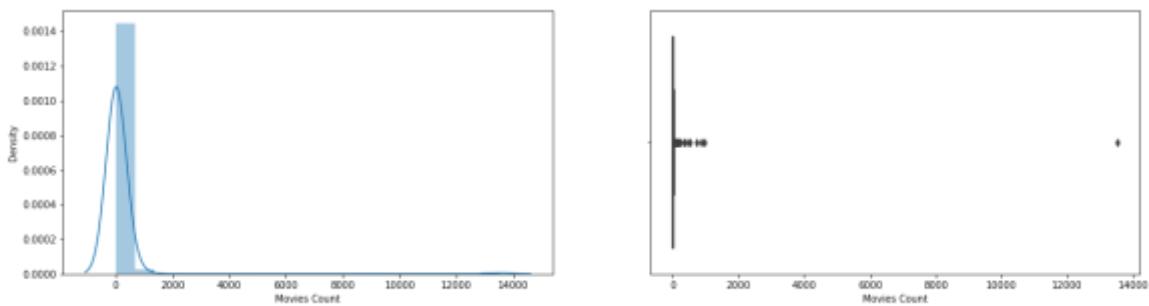


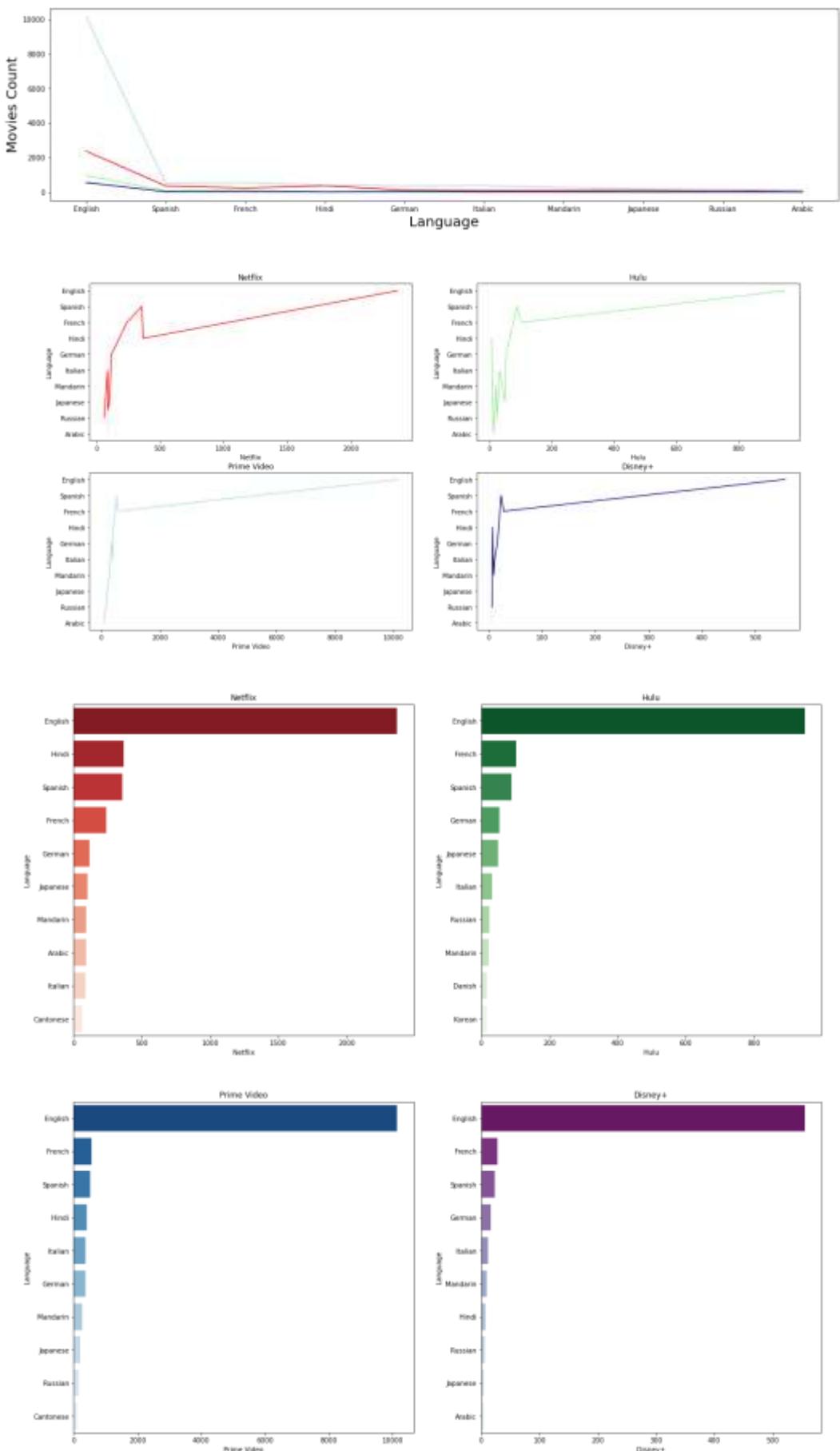


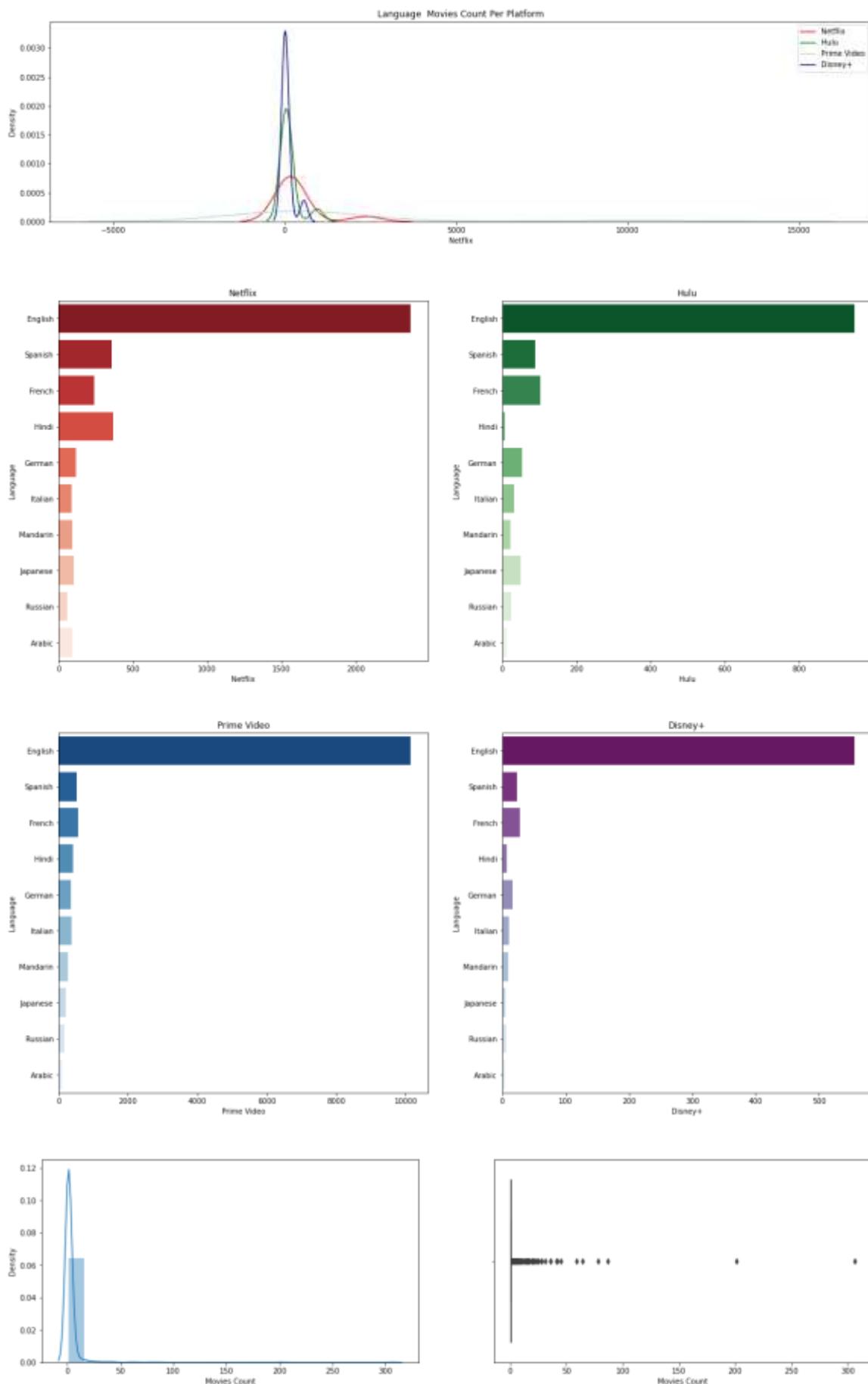


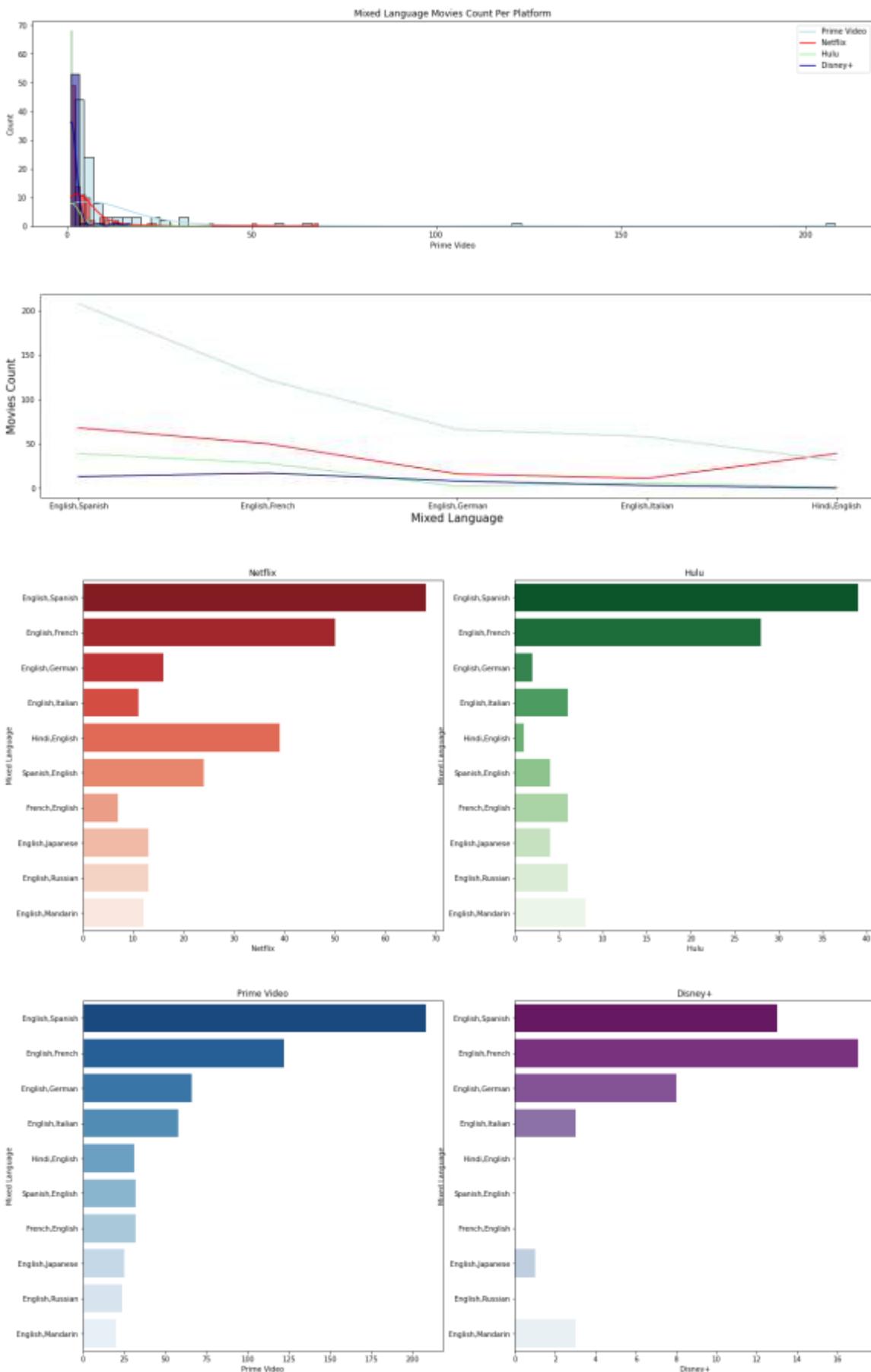
LANGUAGE

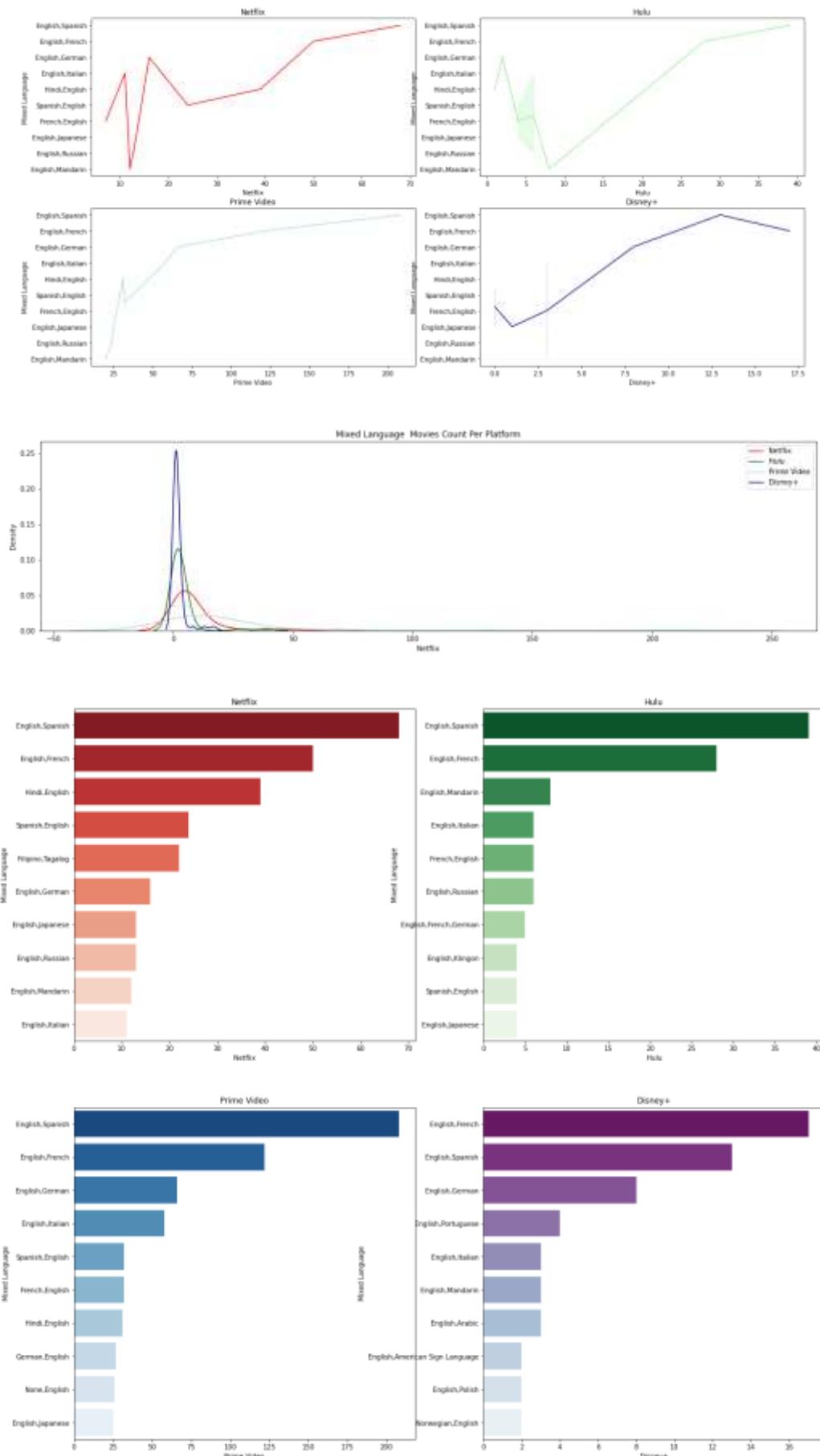






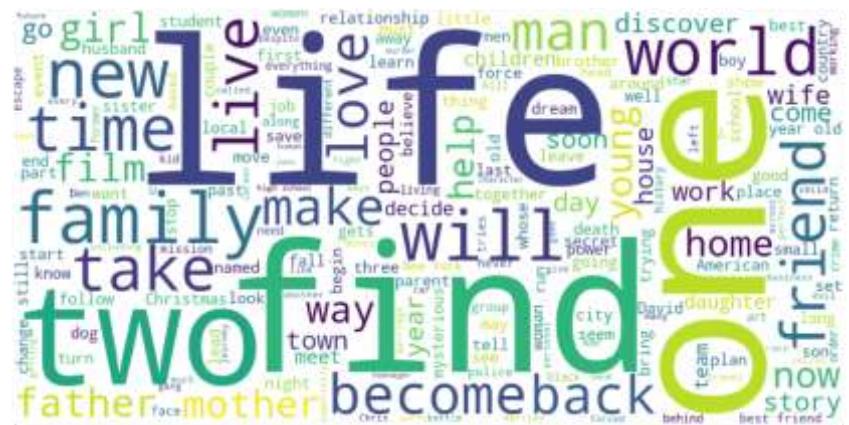




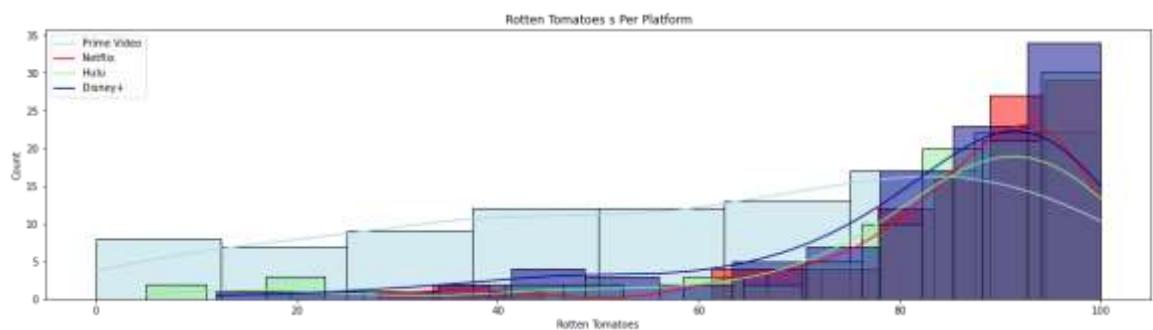
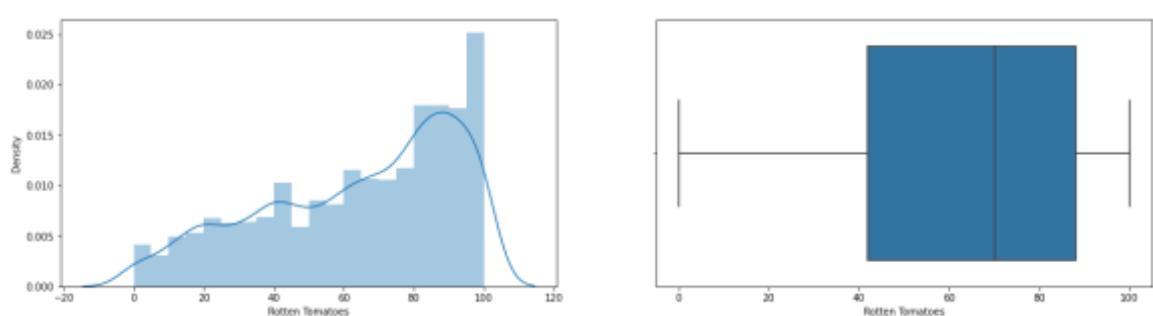
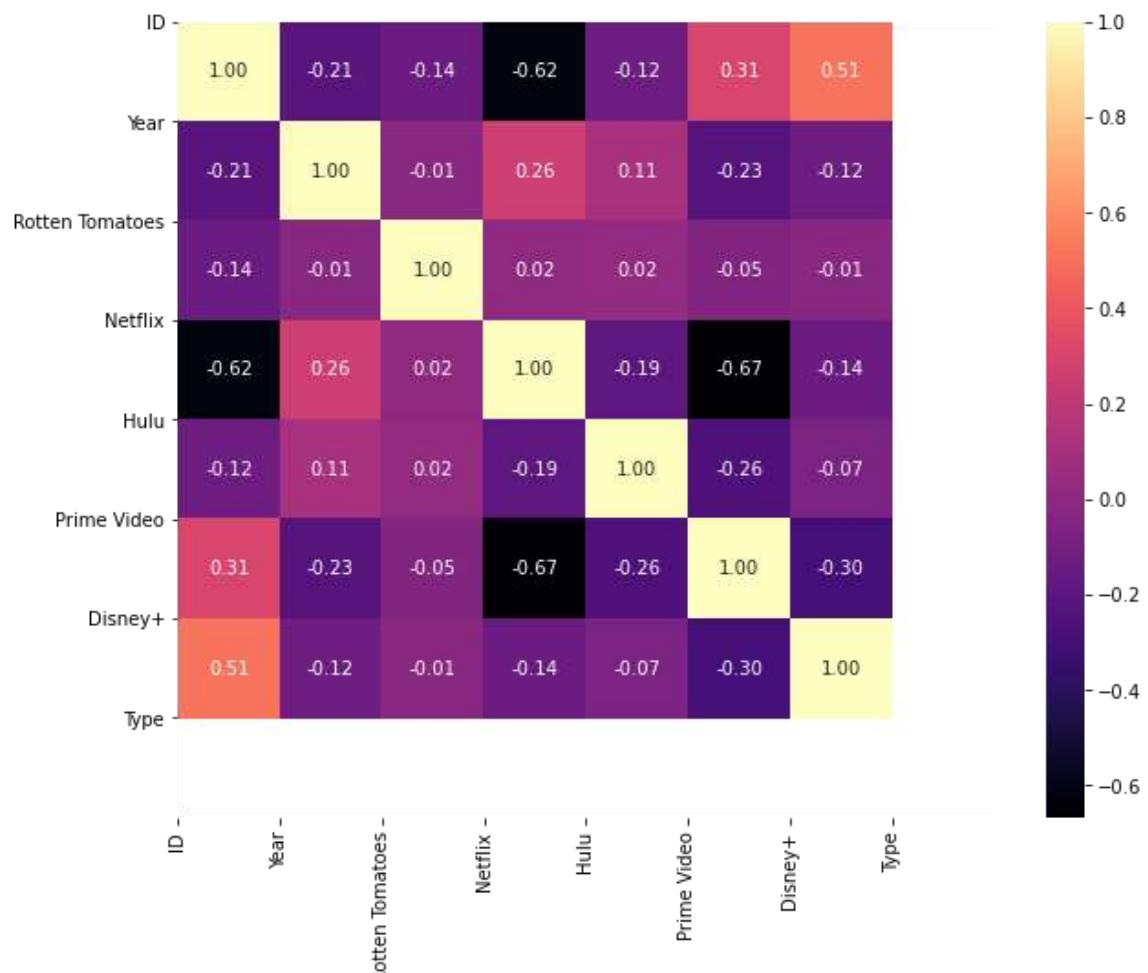


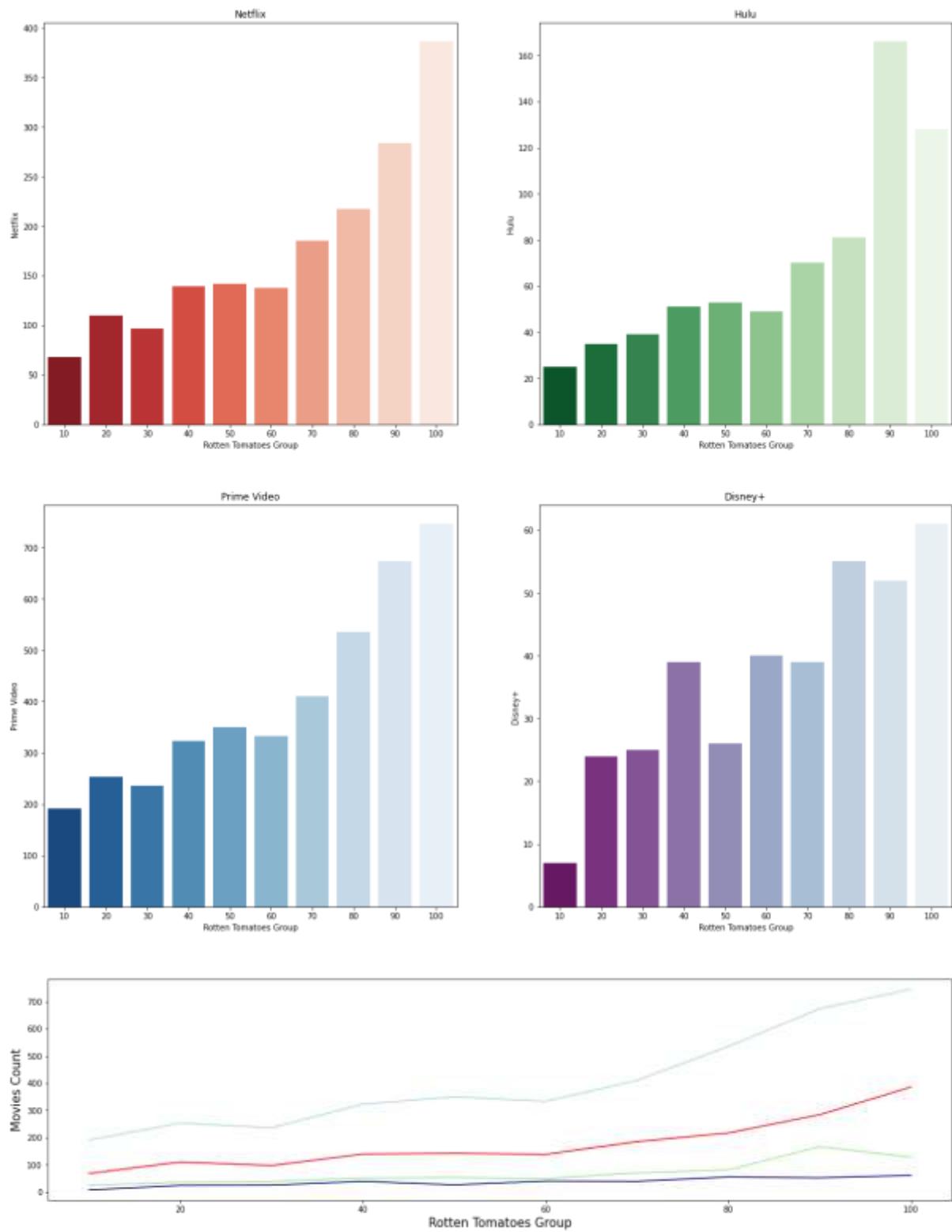
PLOTLINE

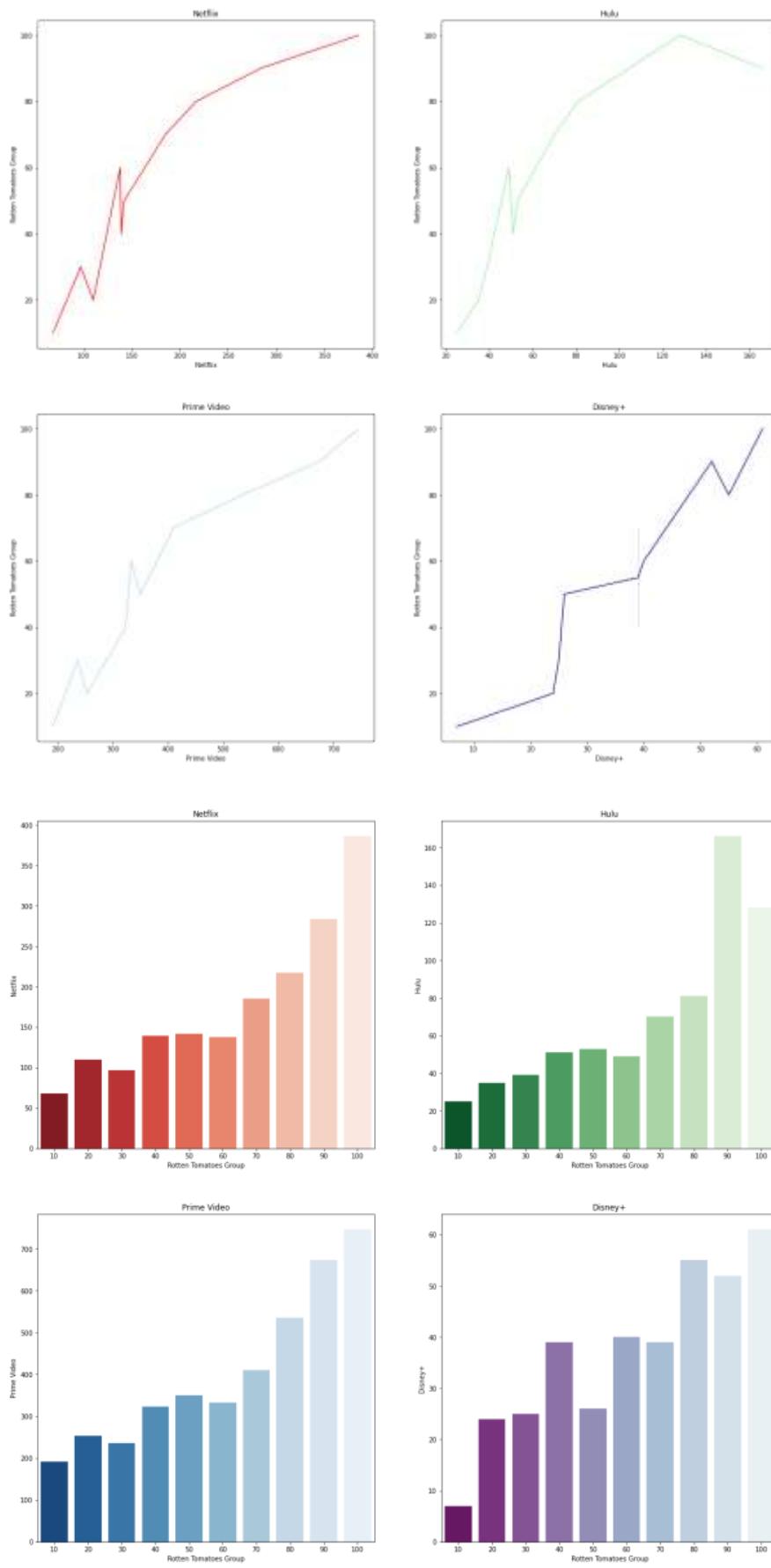




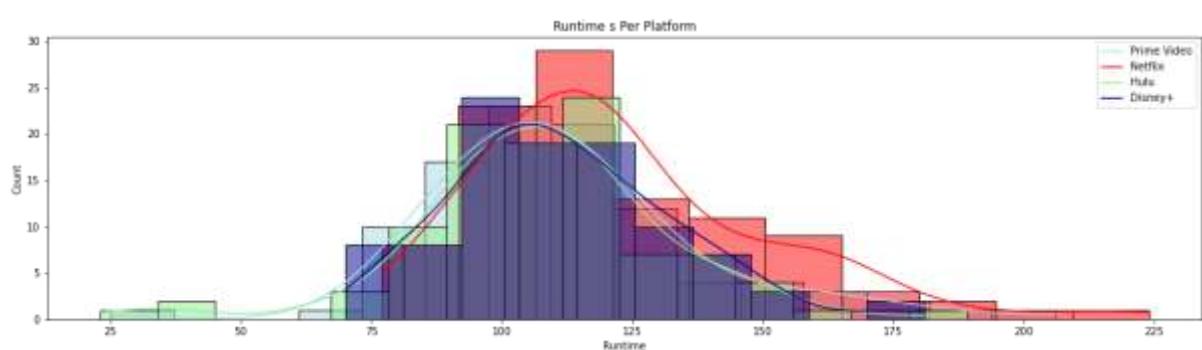
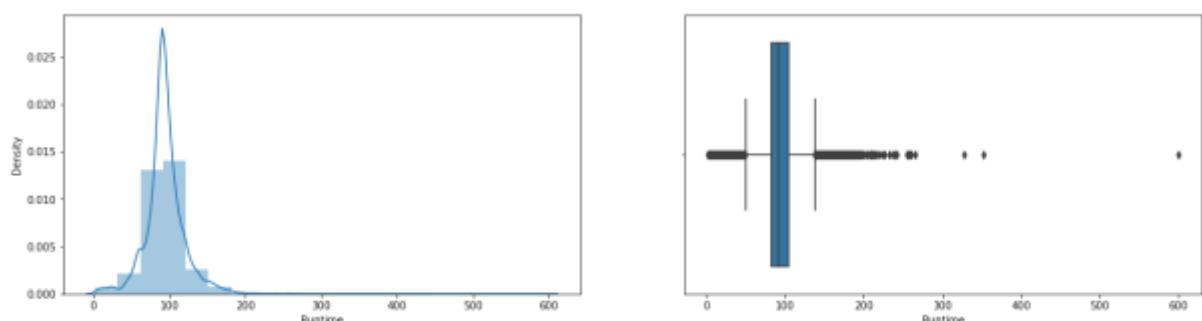
ROTTEN TOMATOES

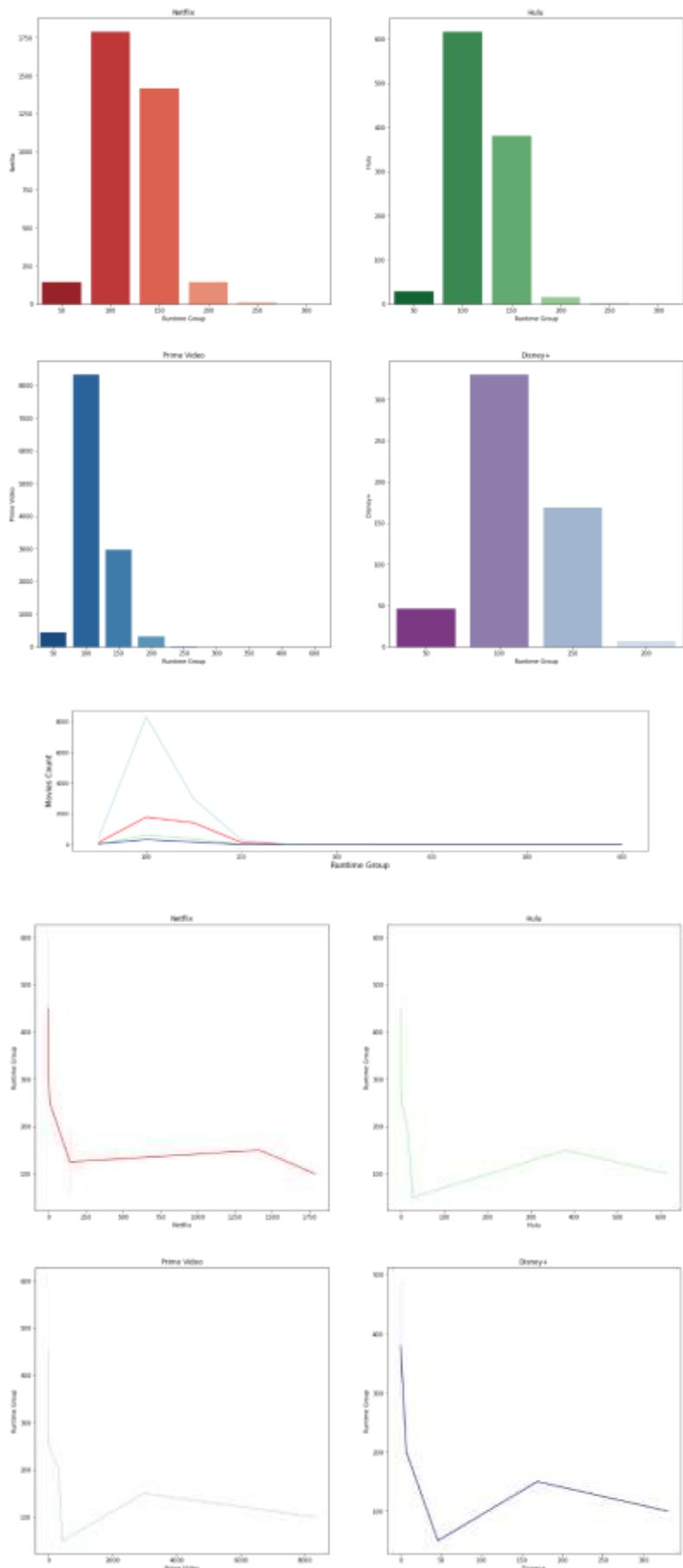


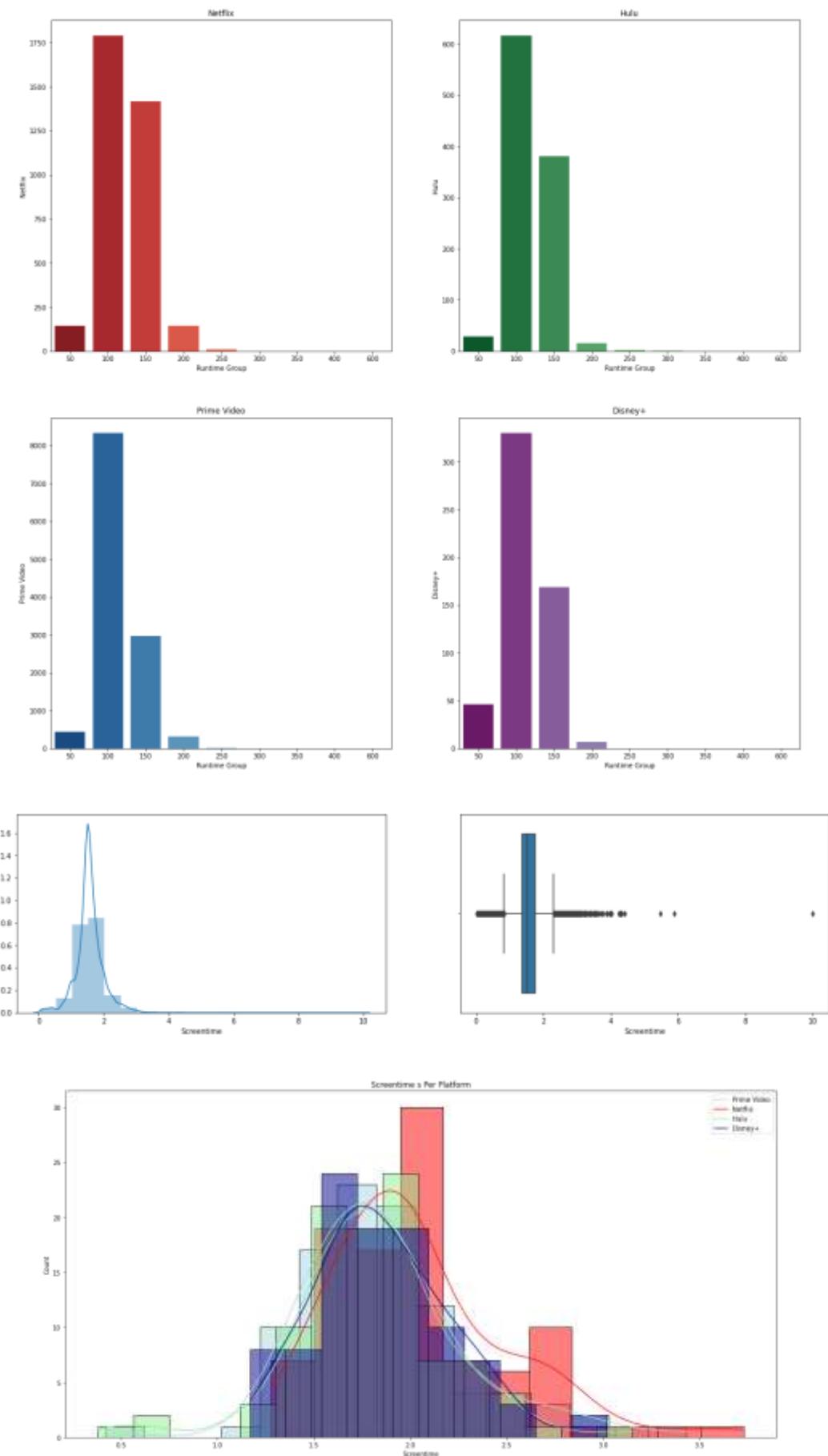


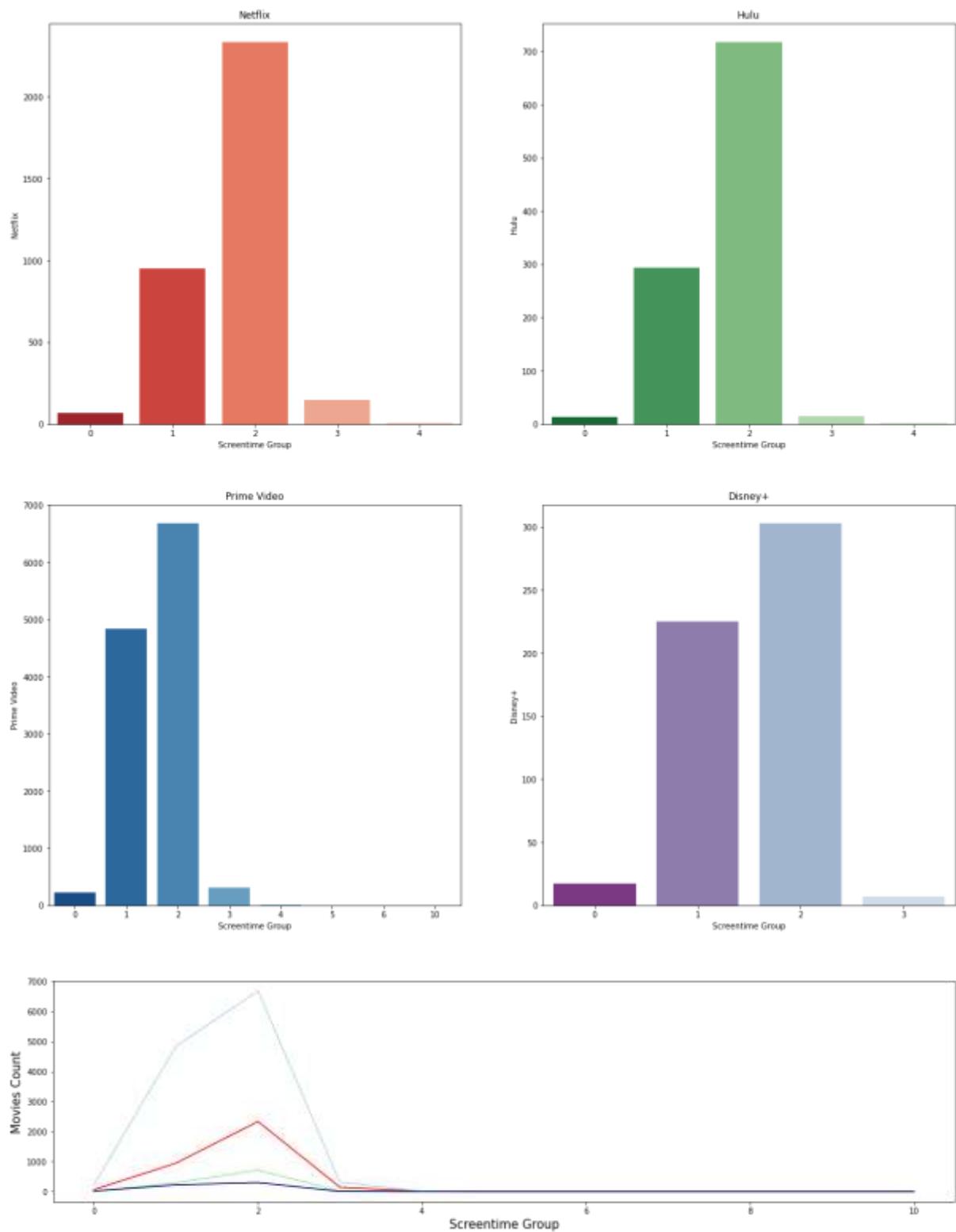


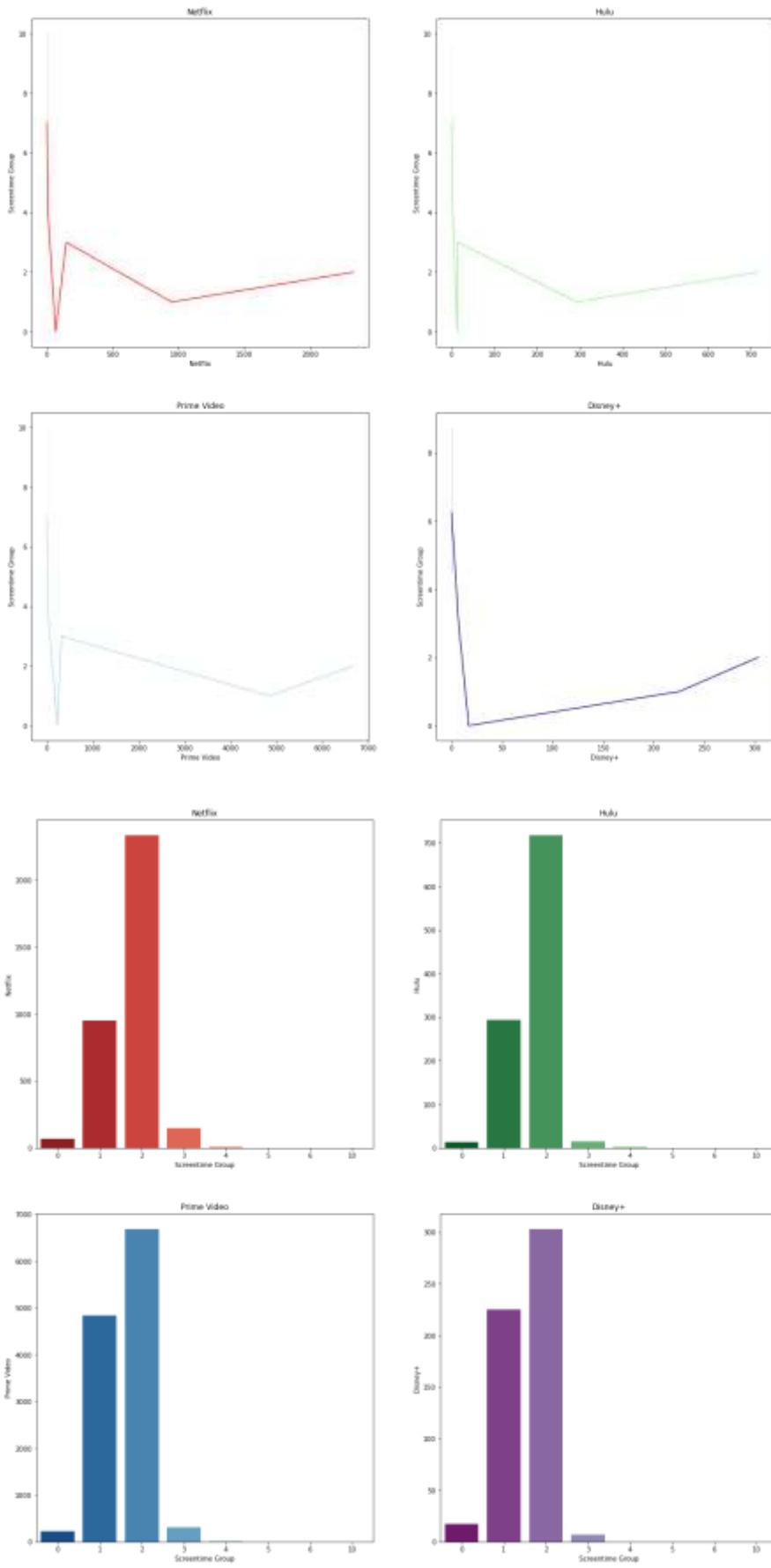
RUNTIME

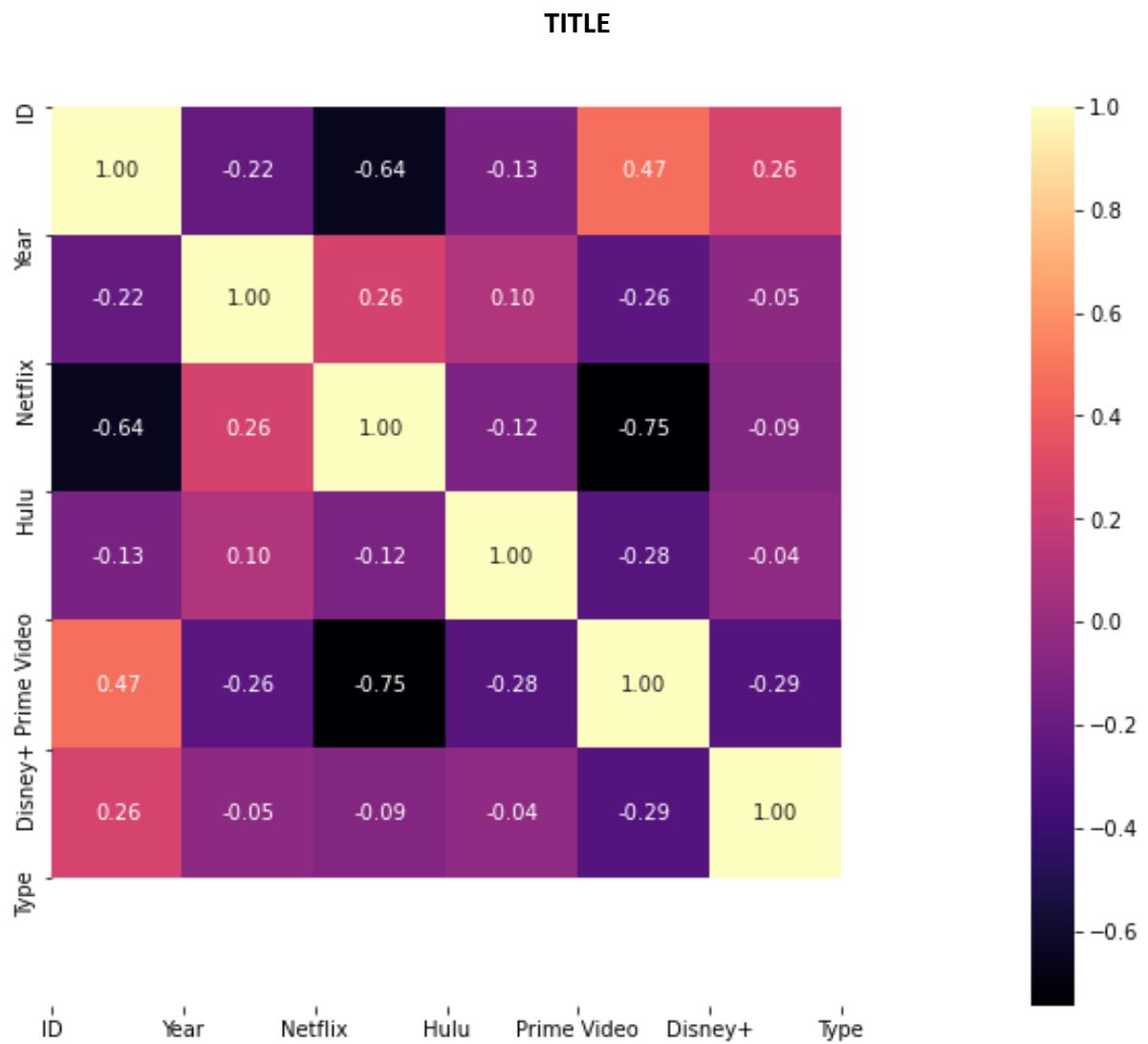




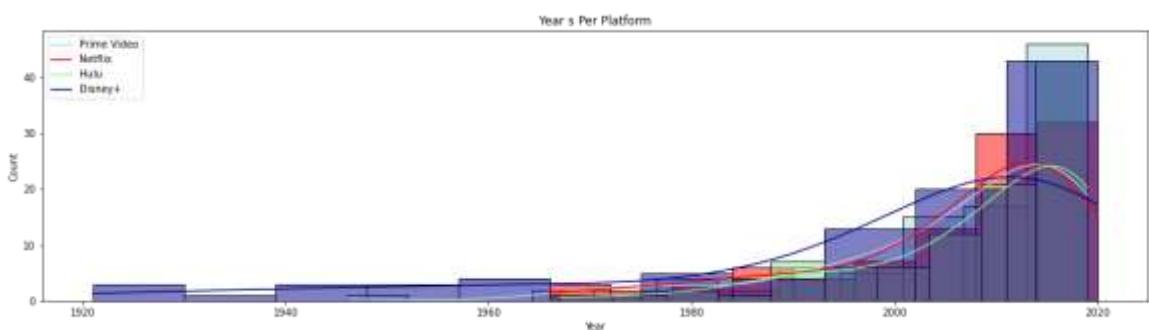
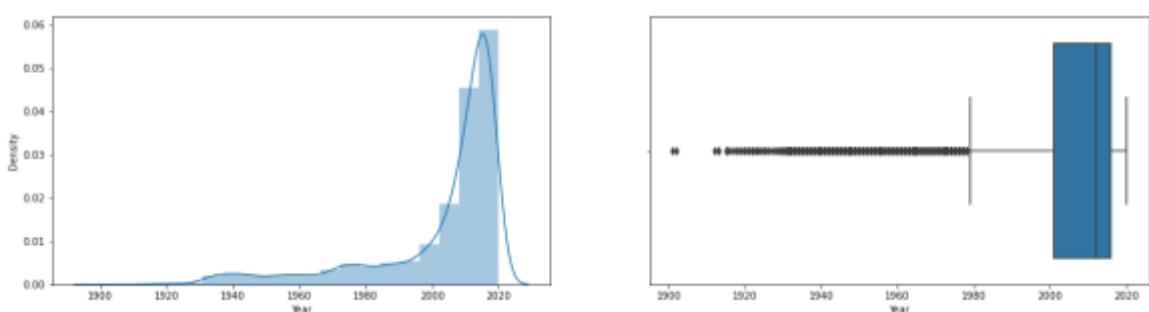
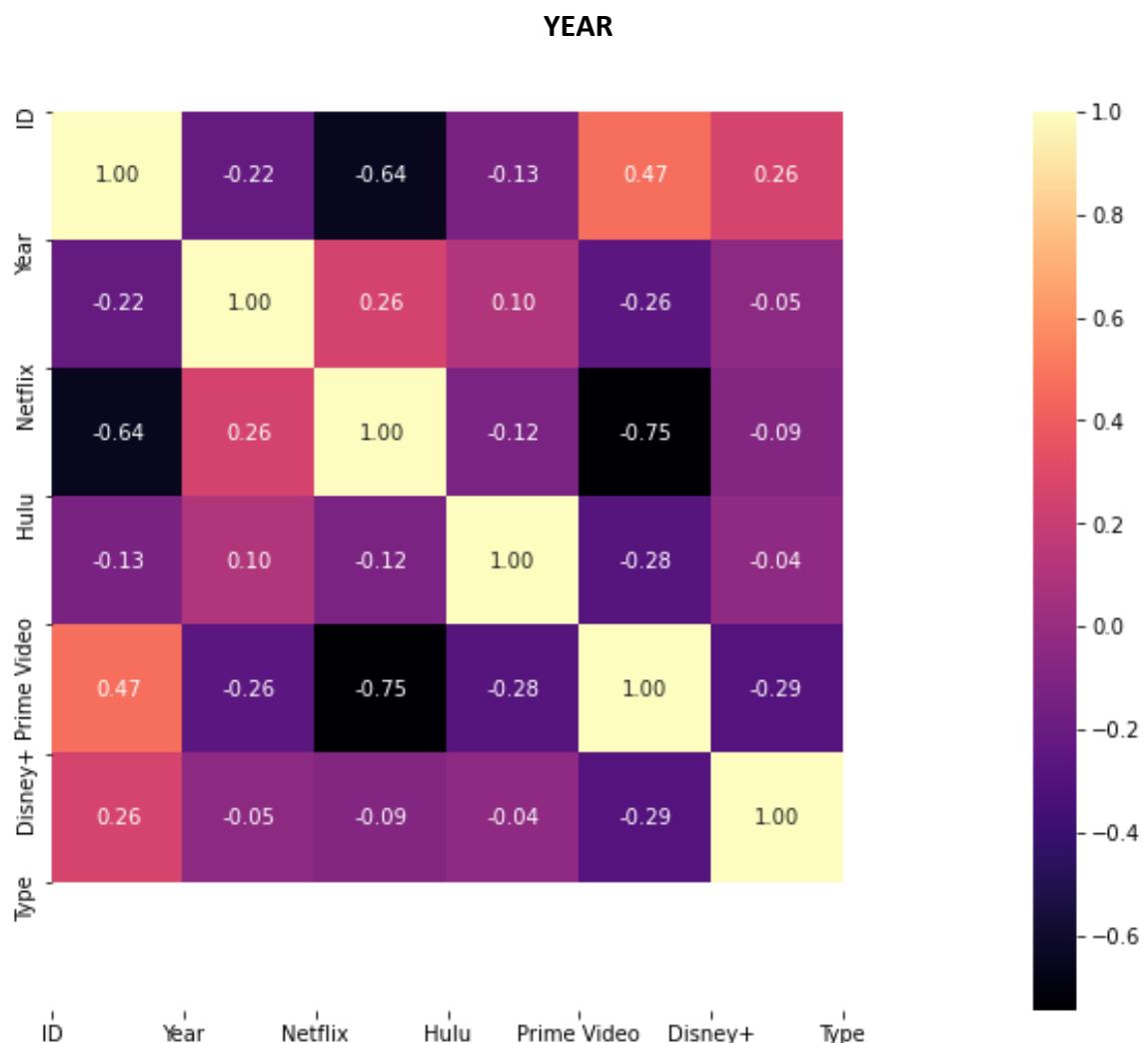


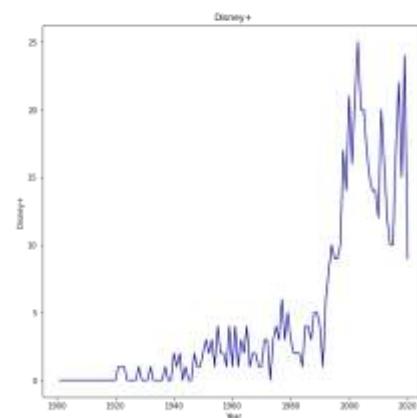
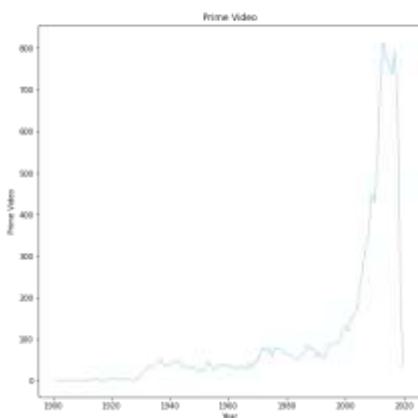
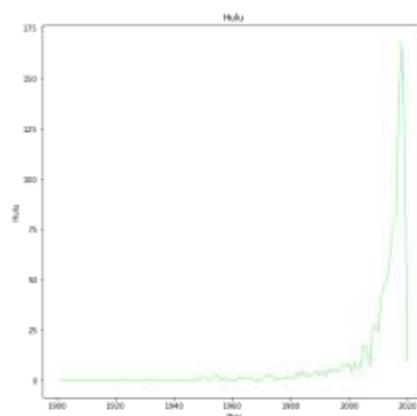
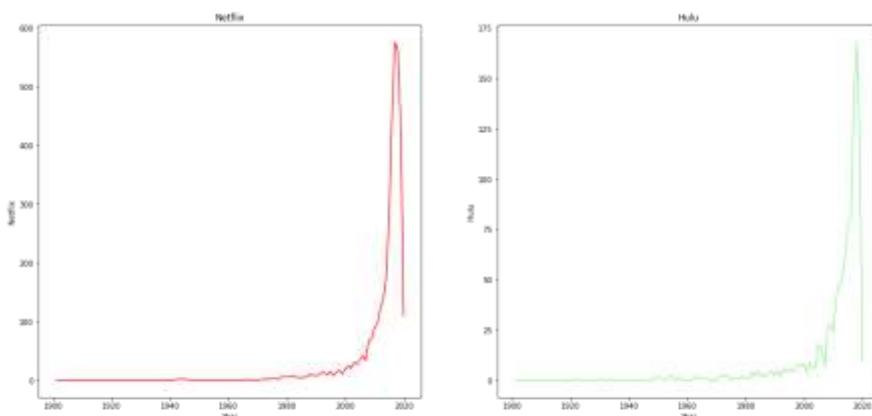
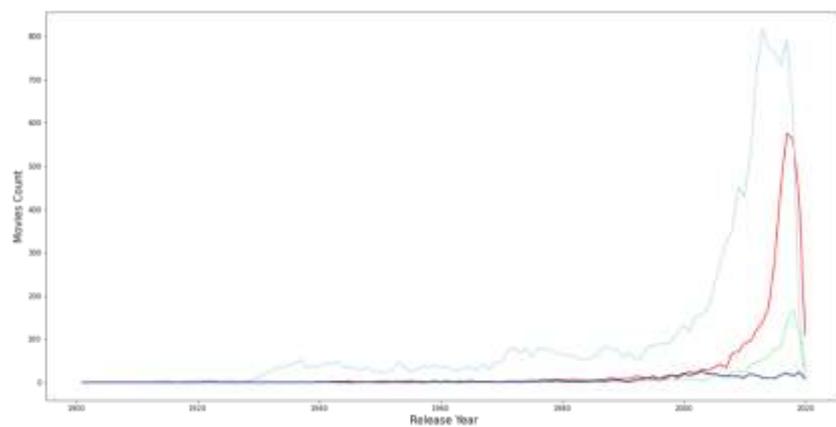
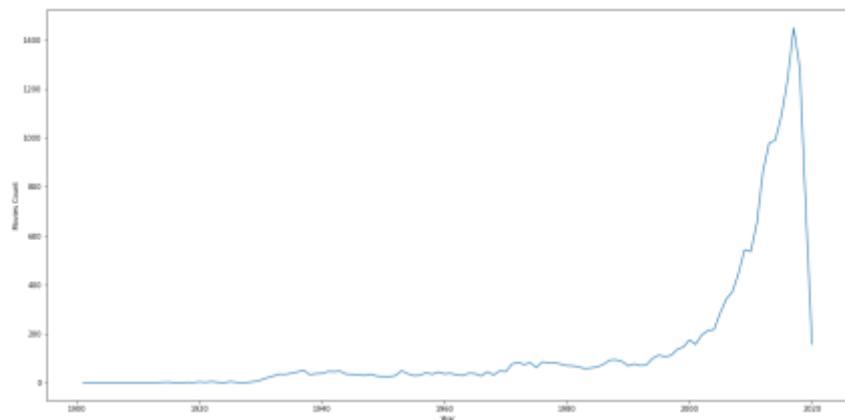


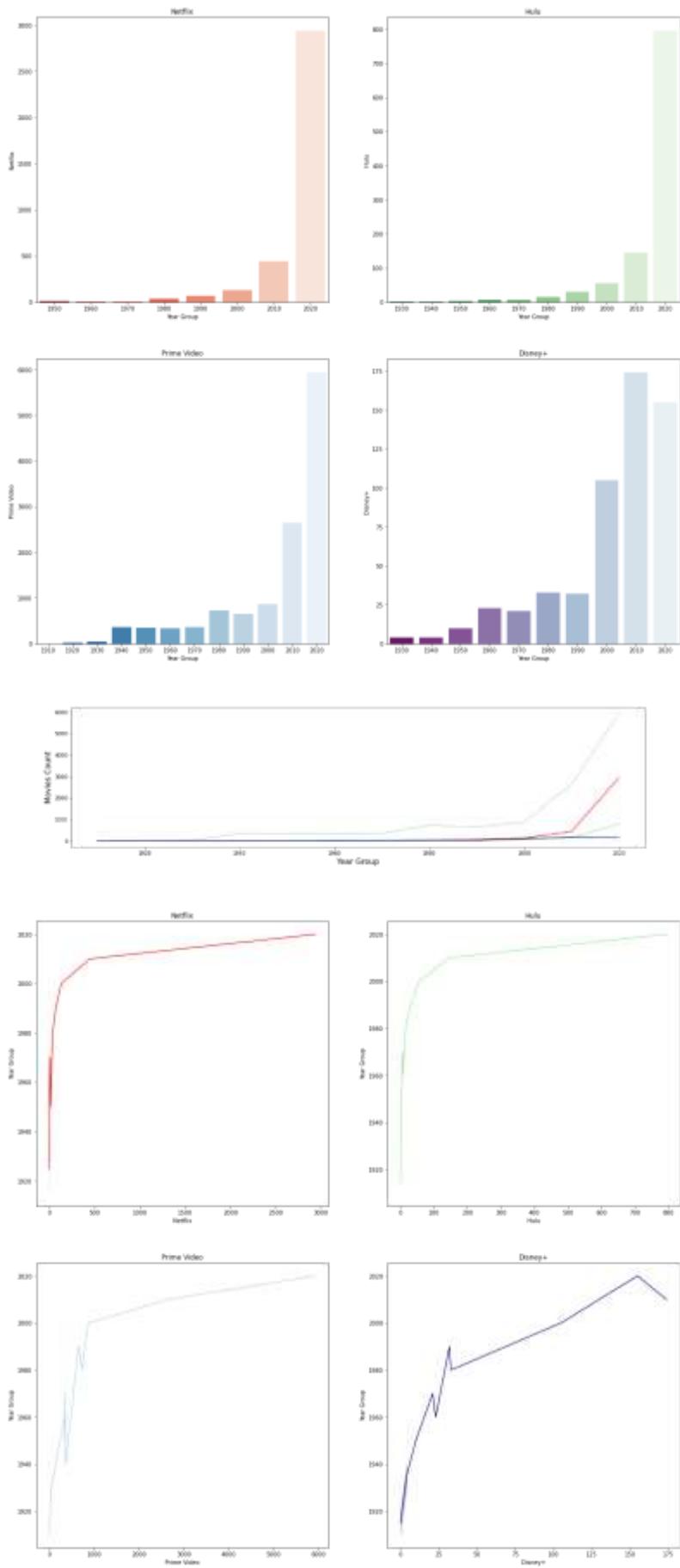


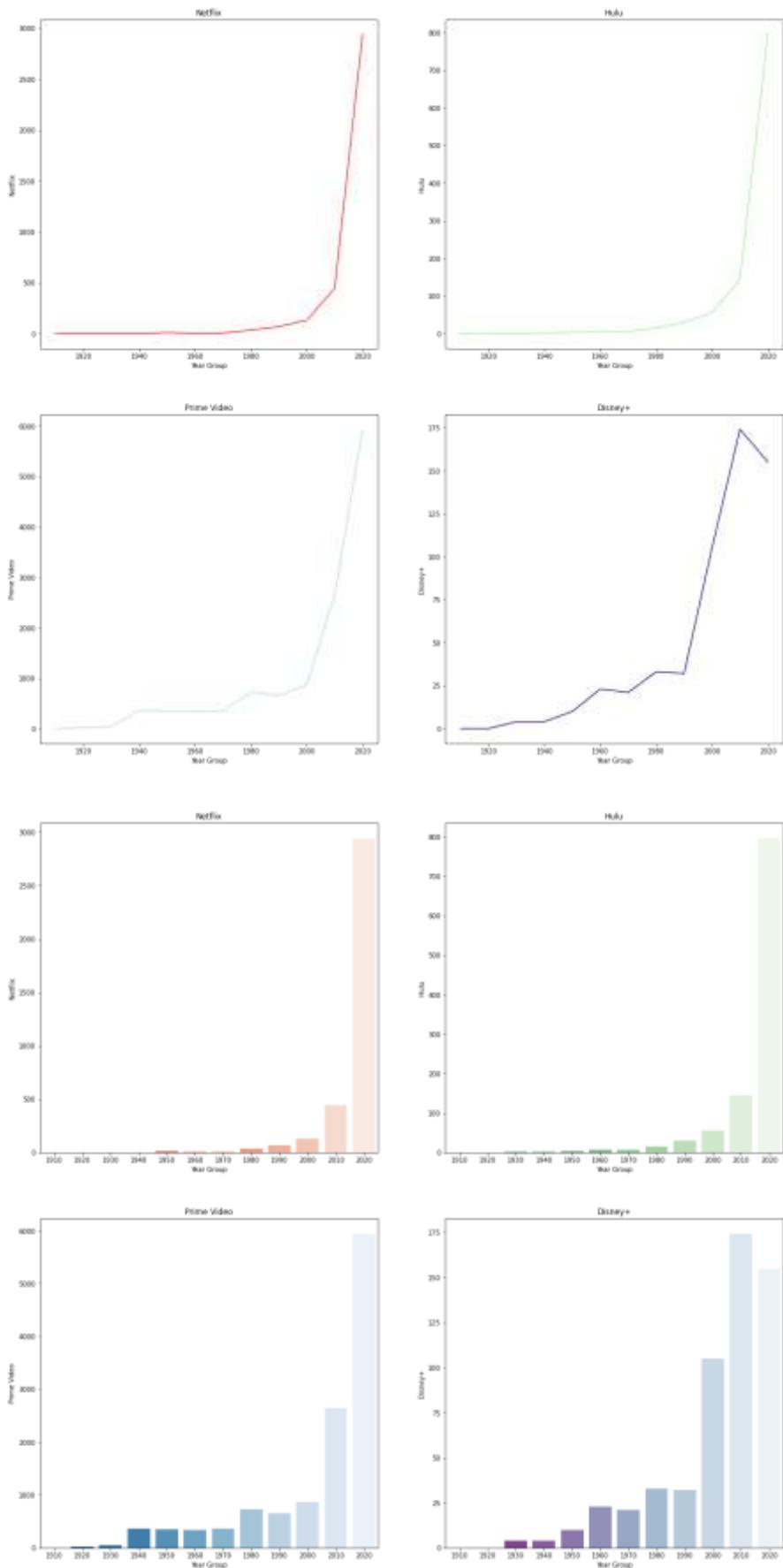






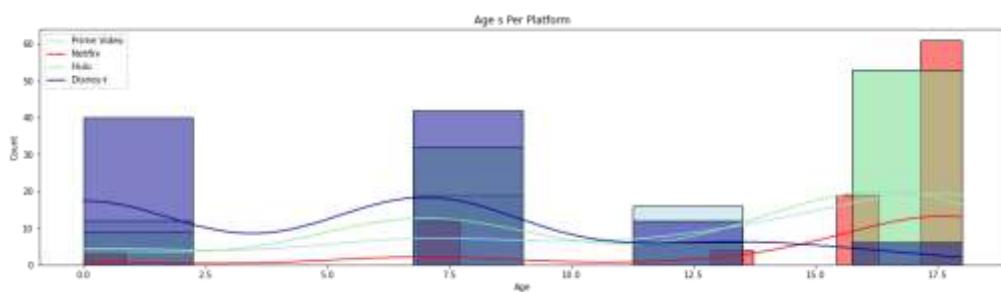
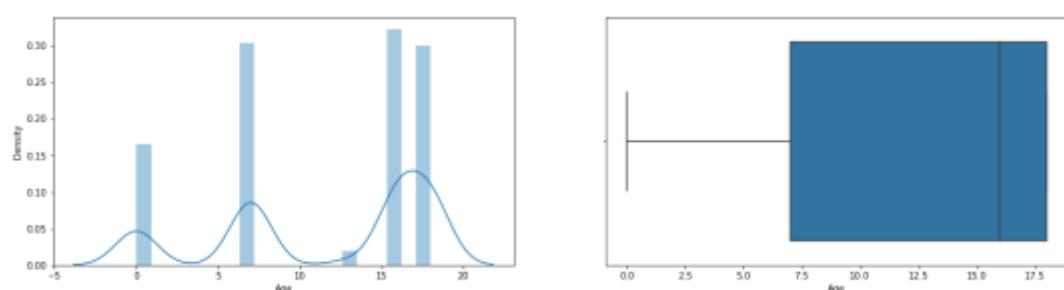


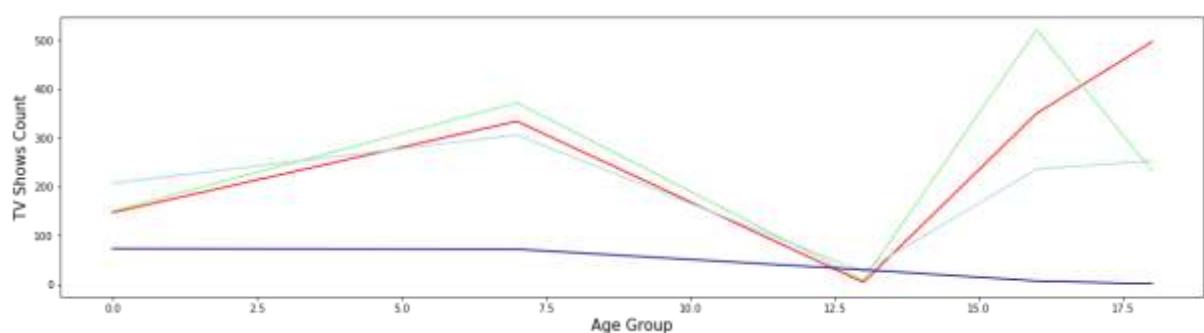
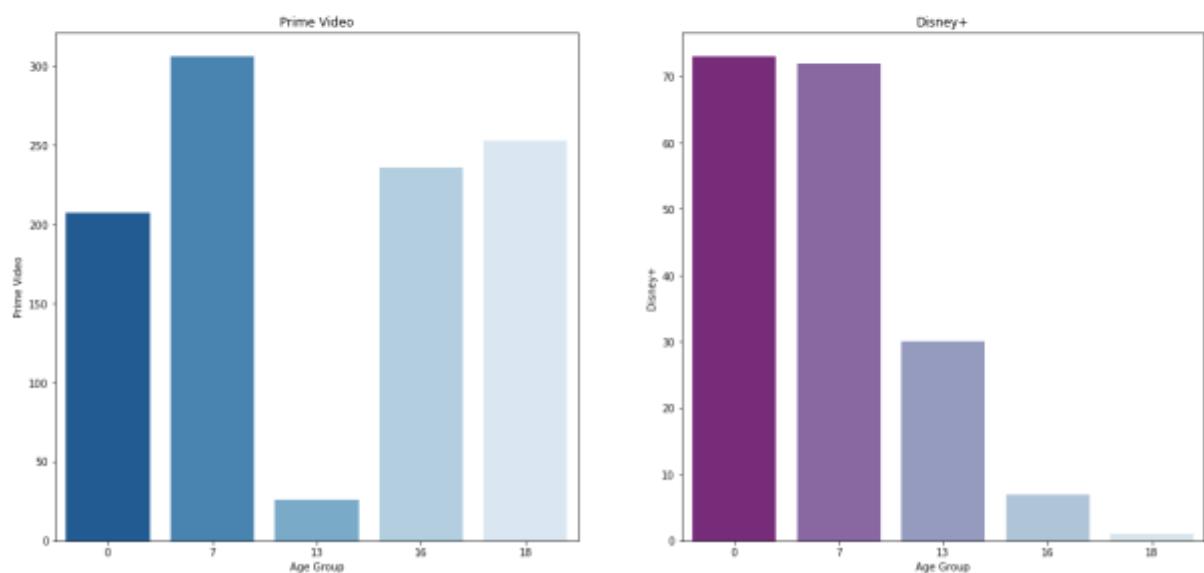
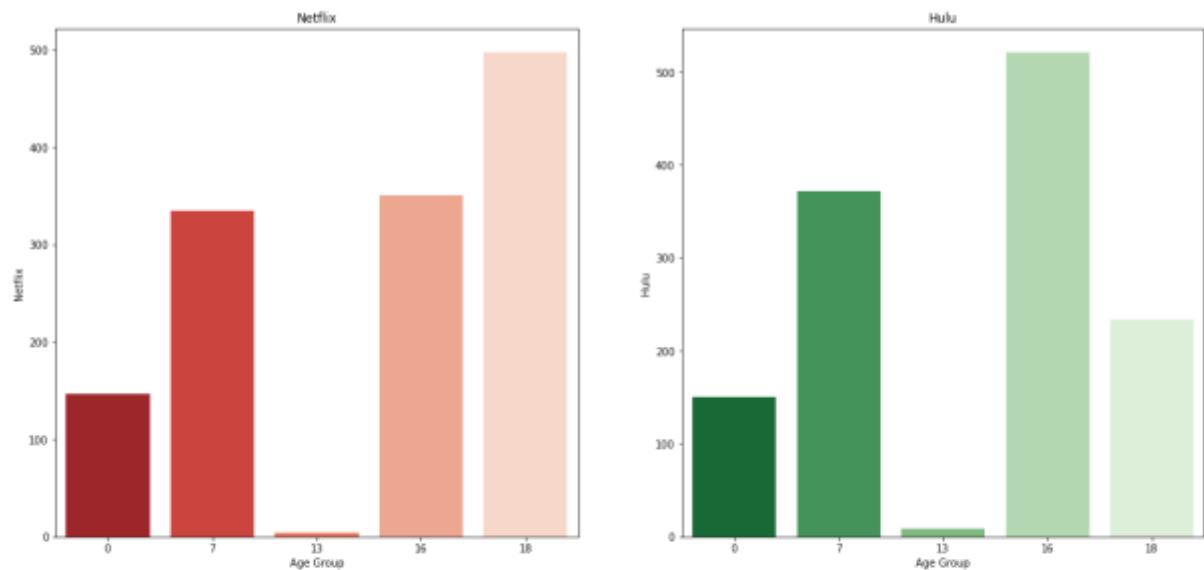


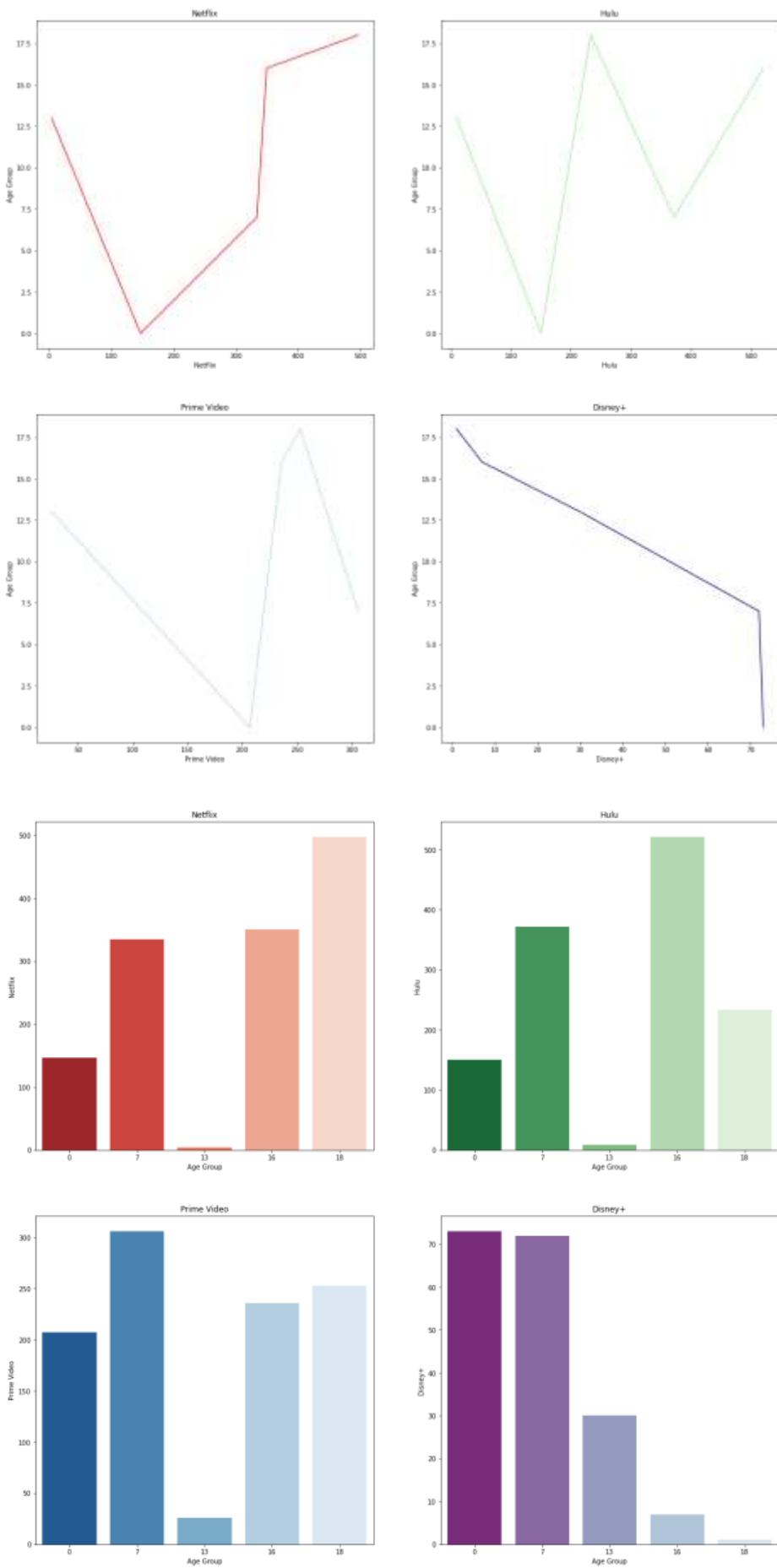


Output: TV Shows

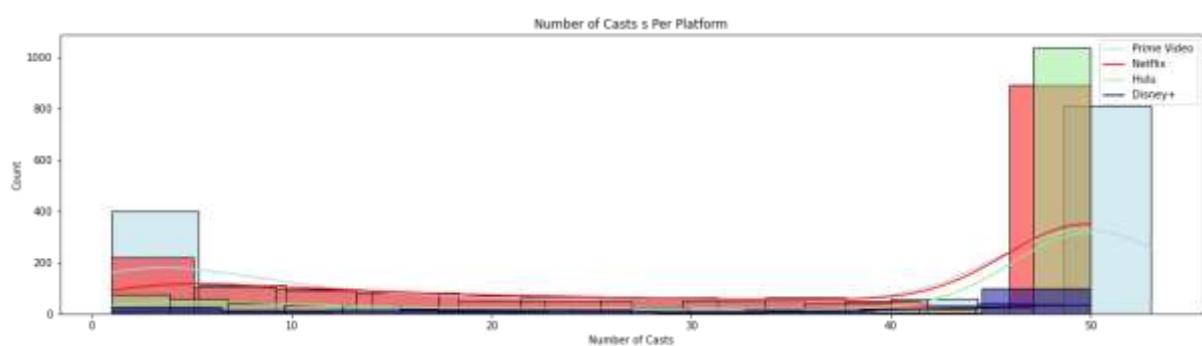
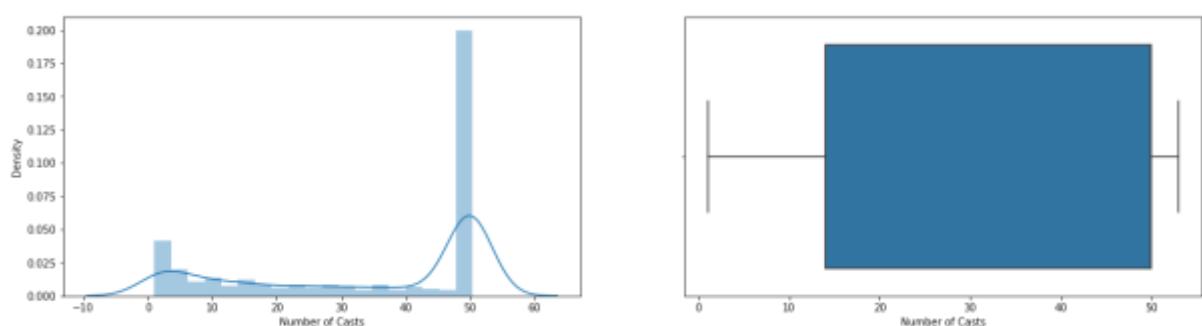
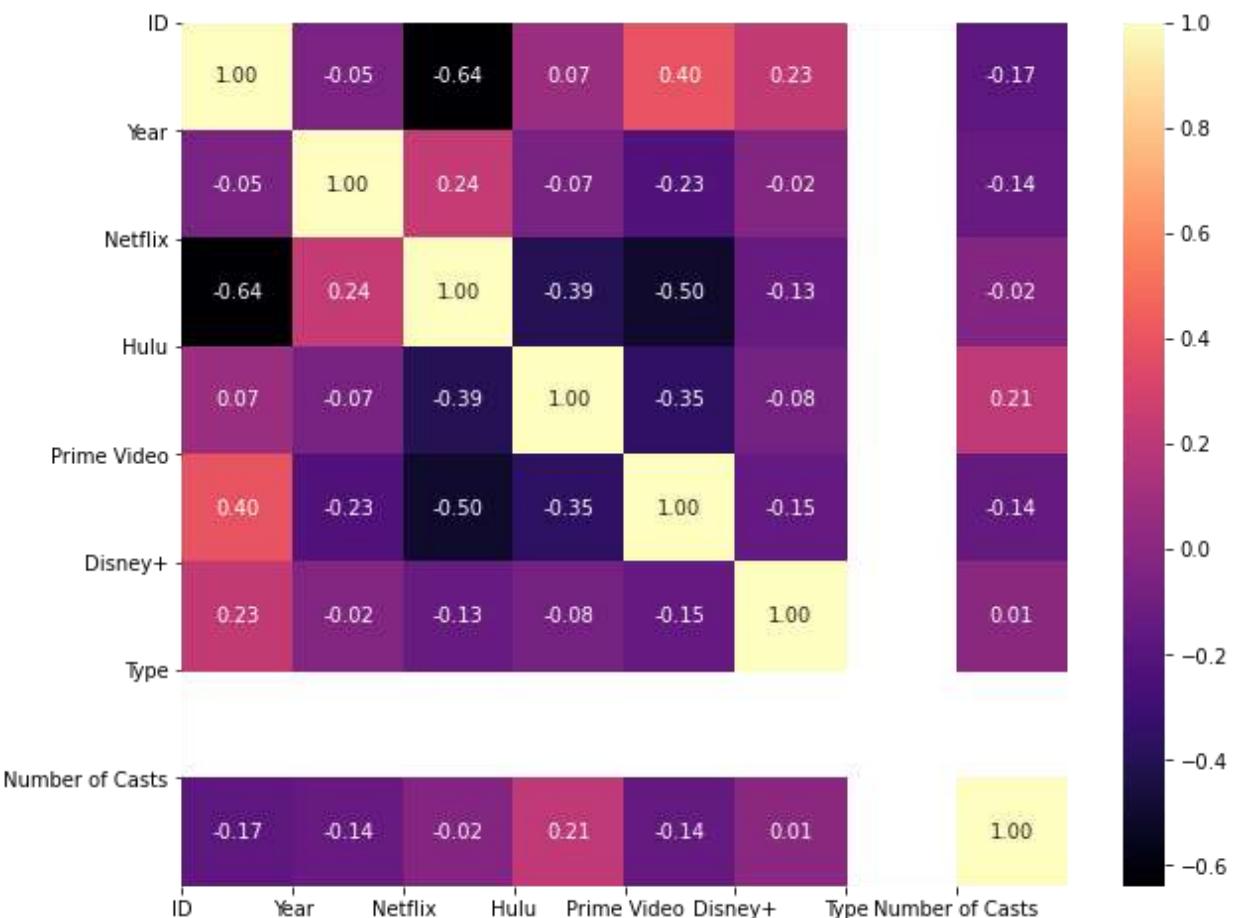
AGE

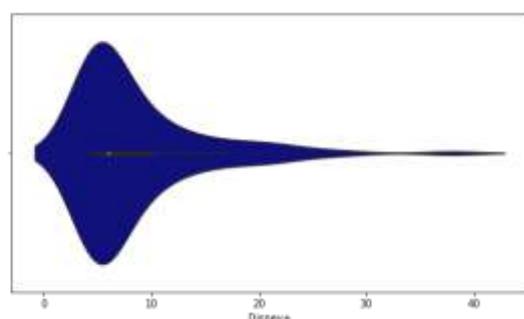
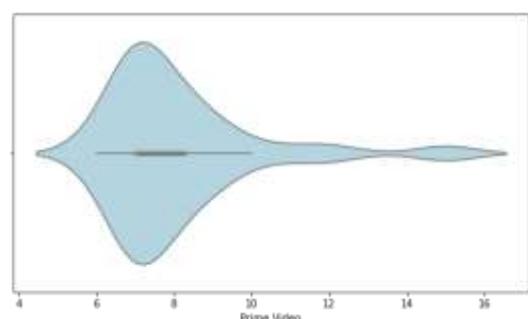
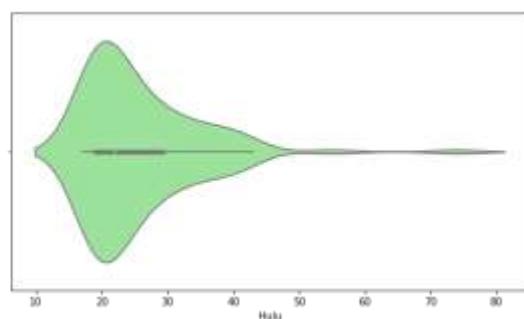
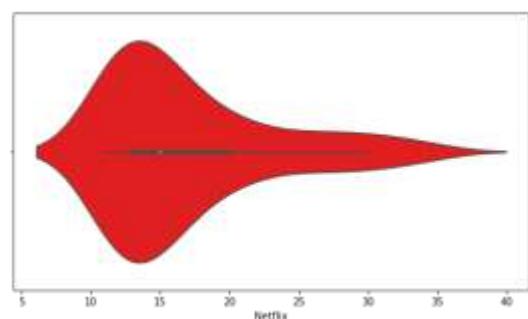
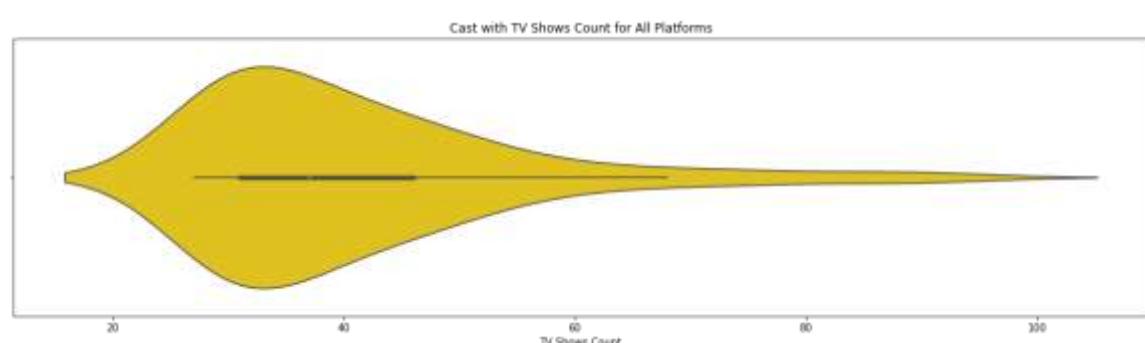
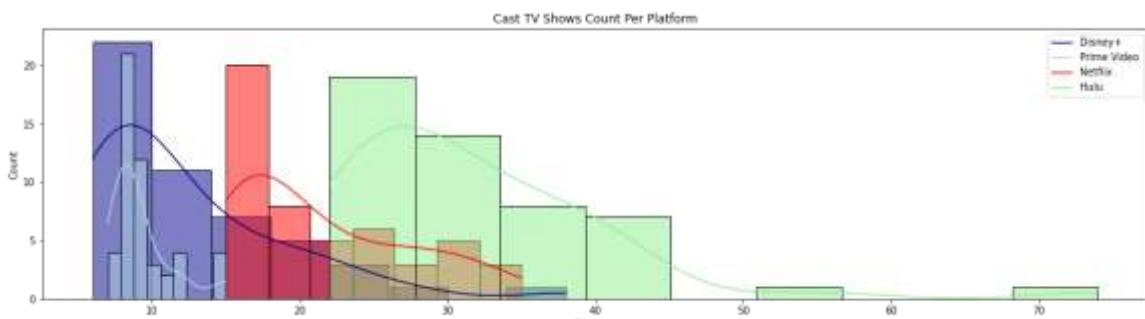
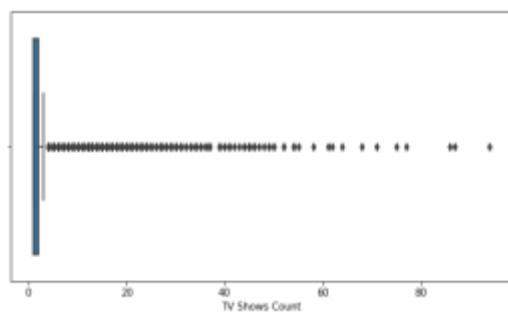
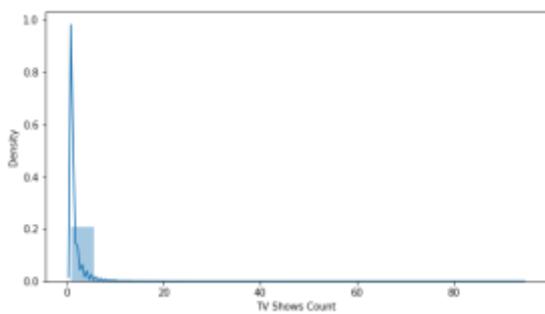


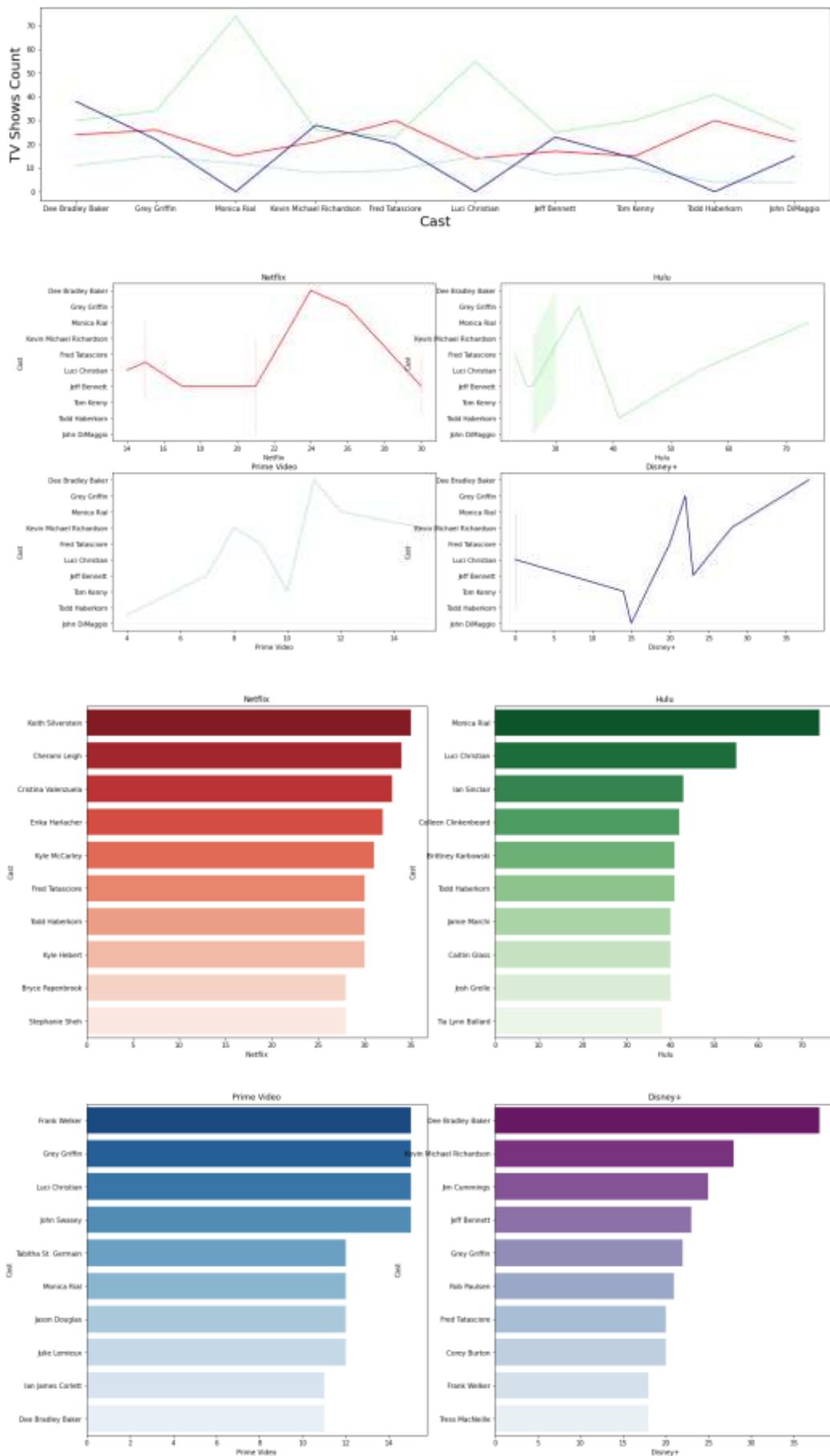


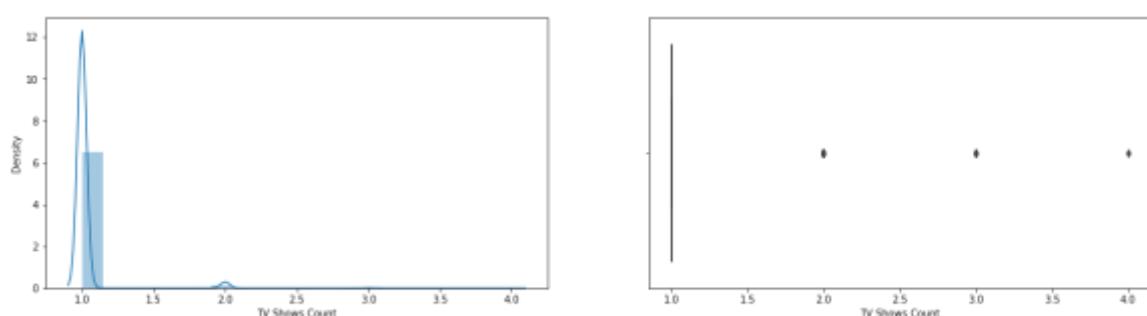
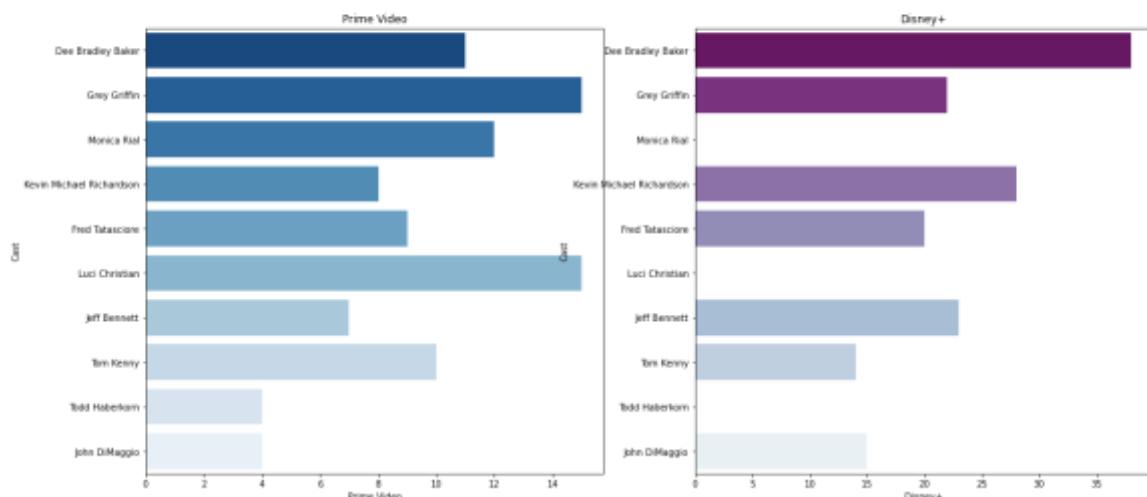
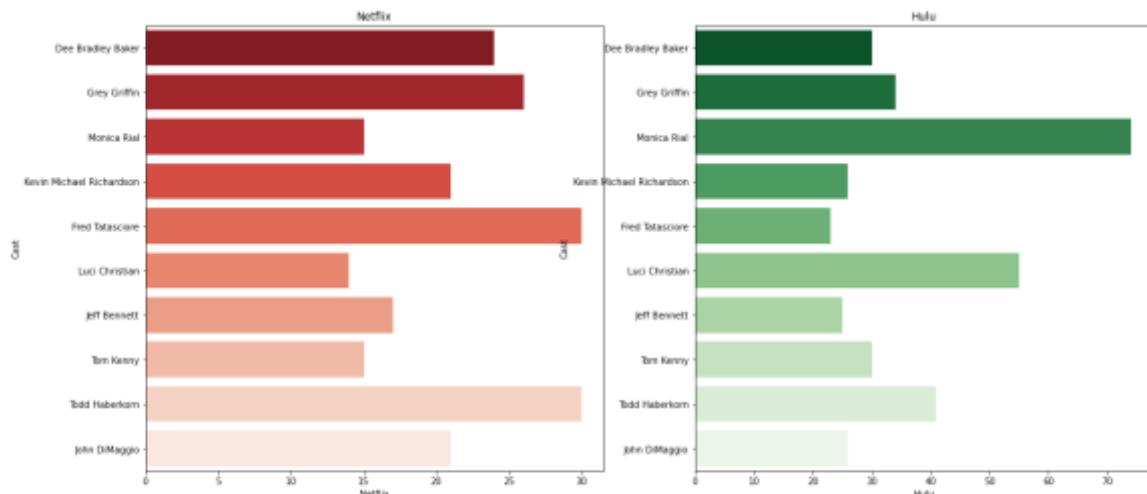
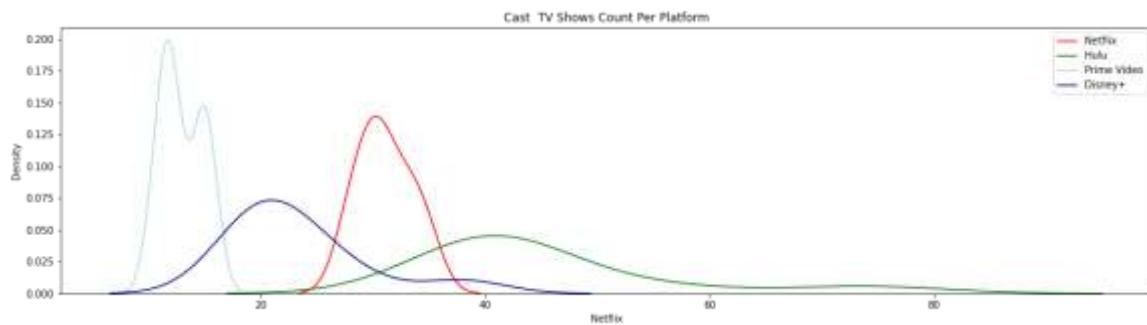


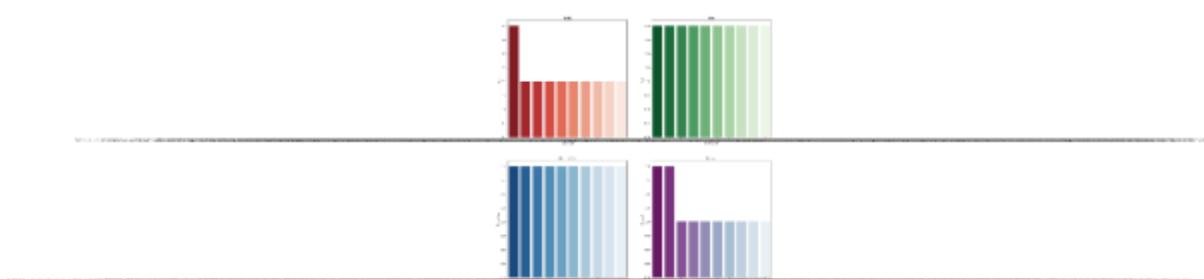
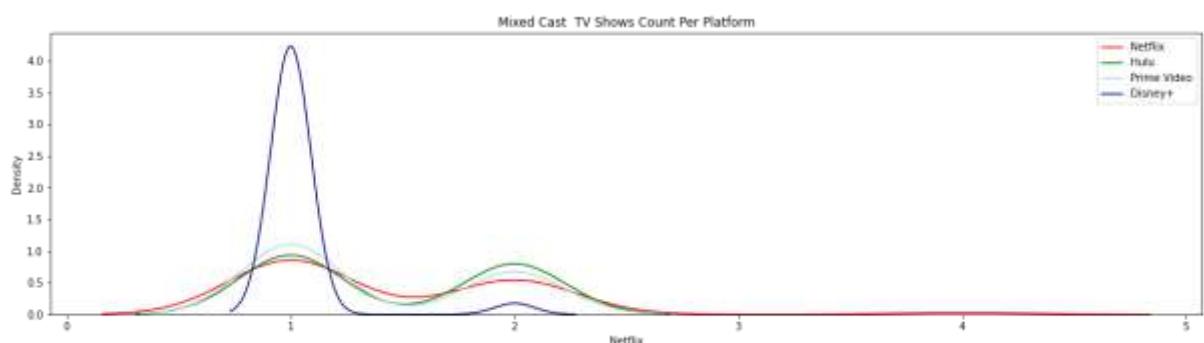
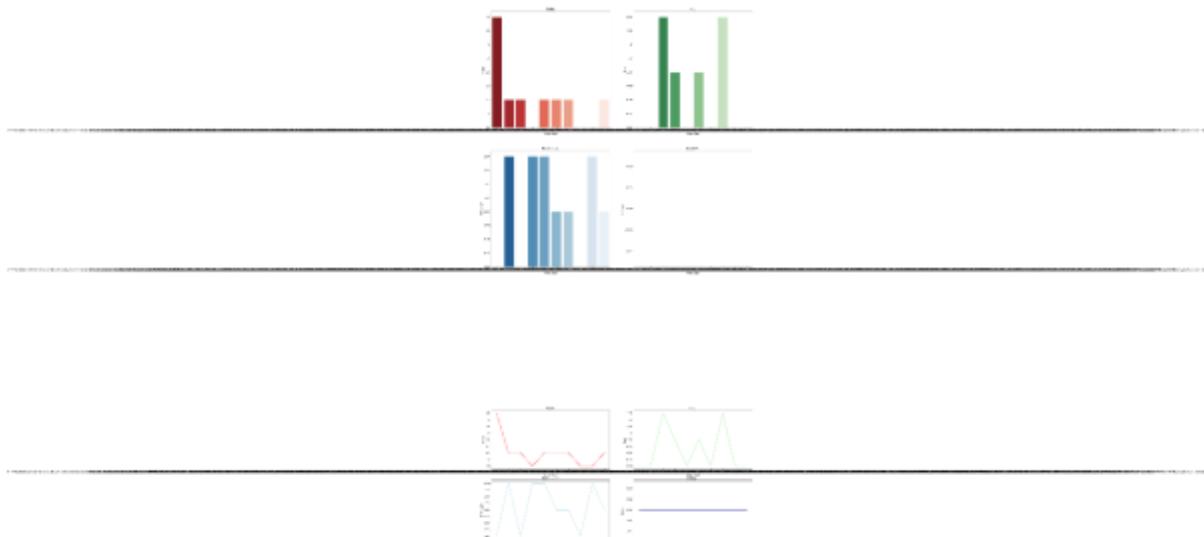
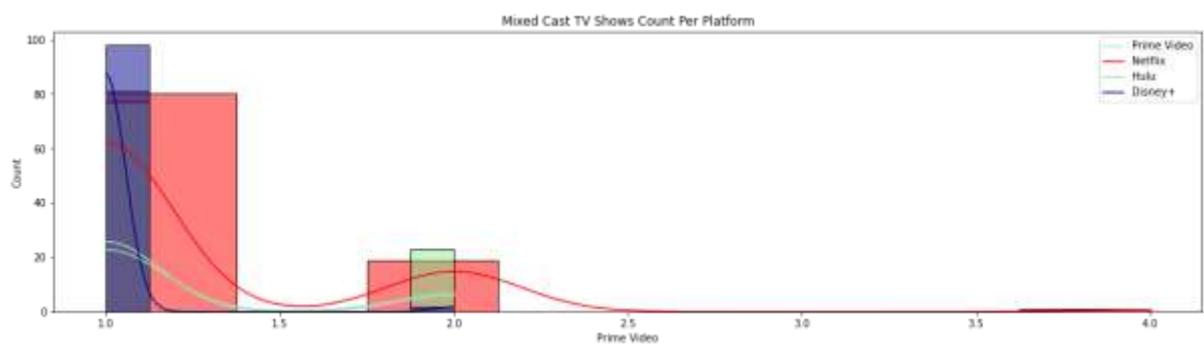
CAST



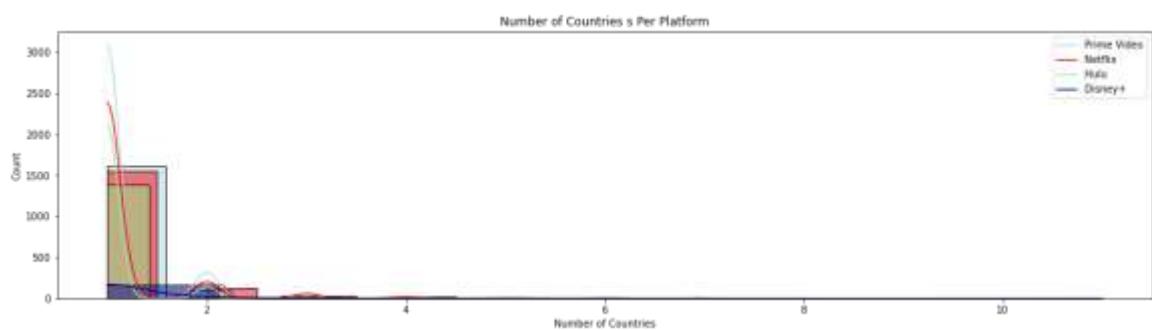
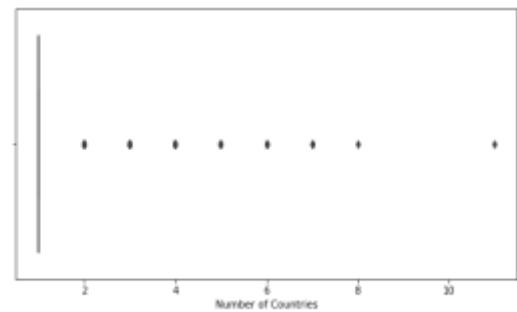
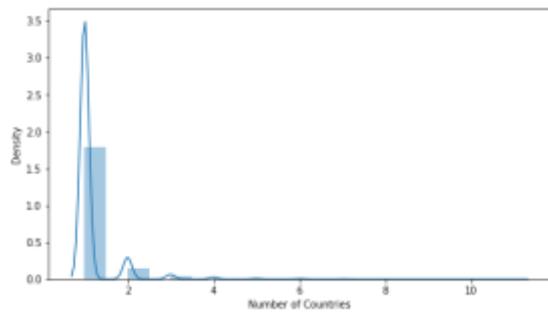
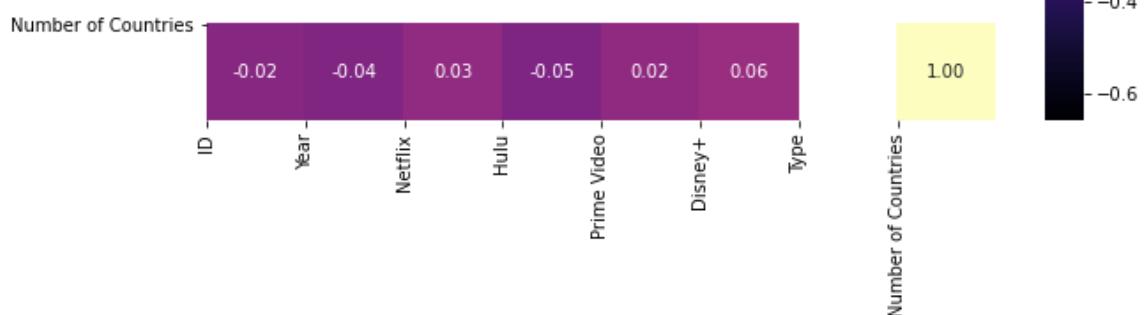




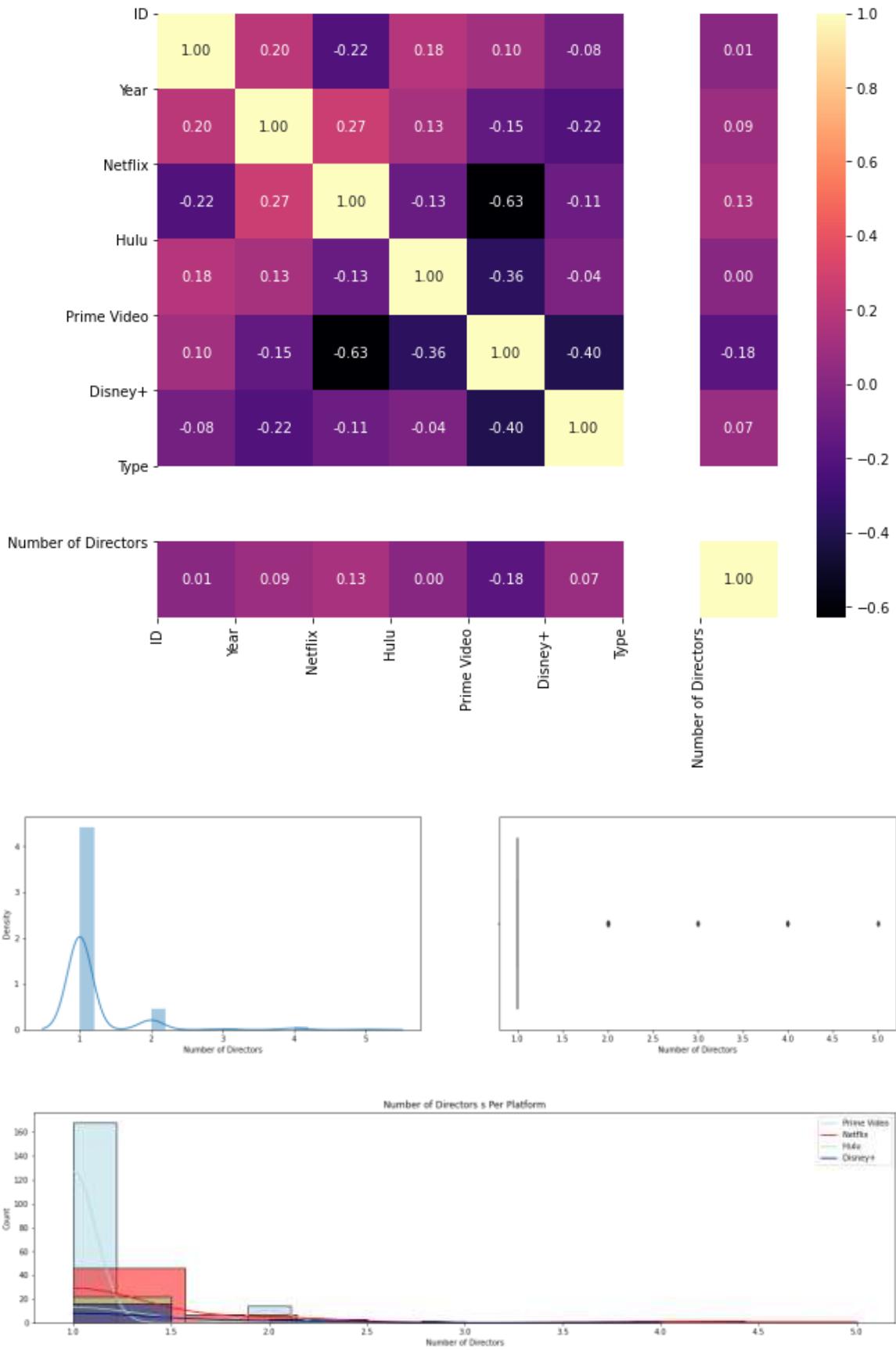


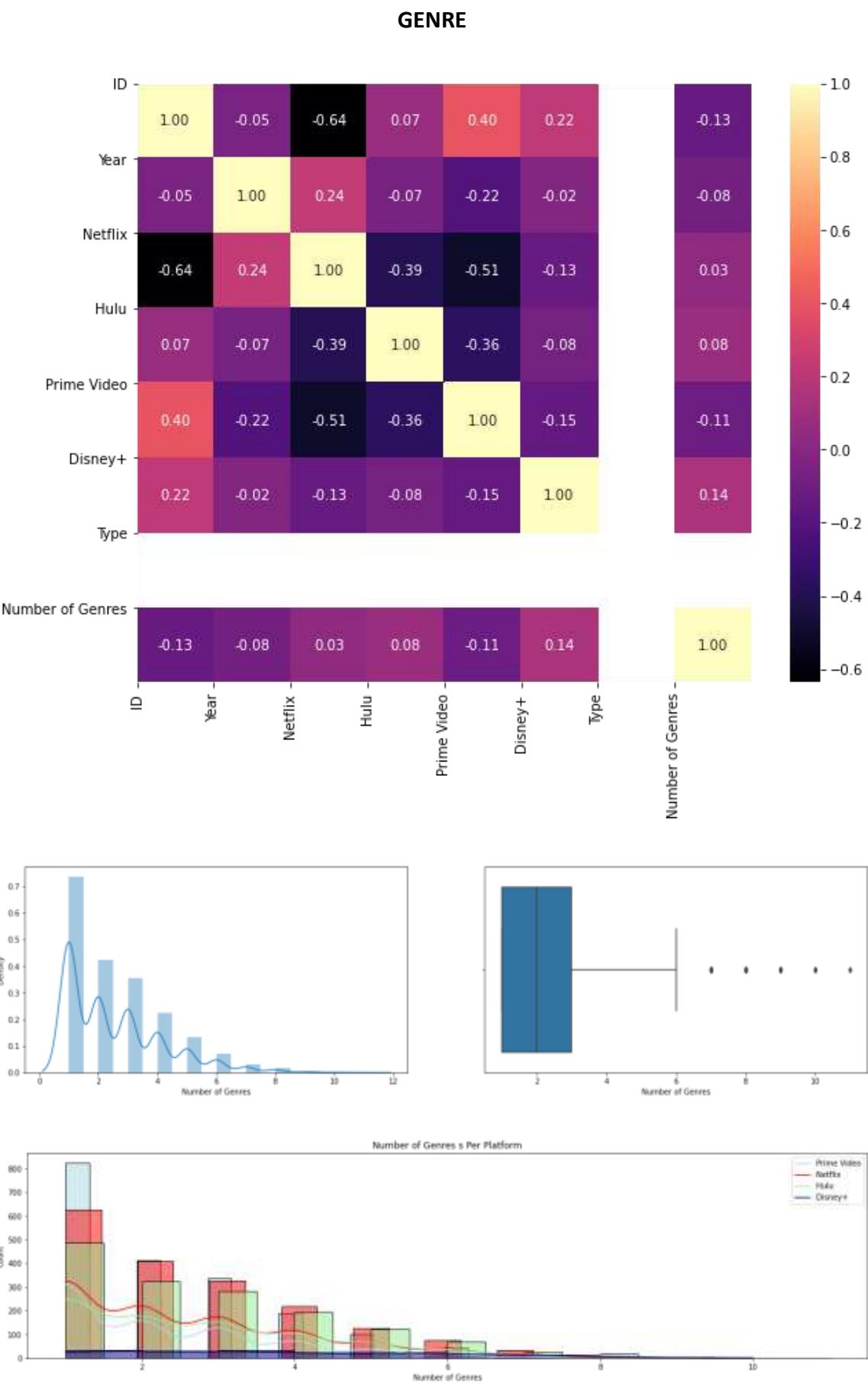


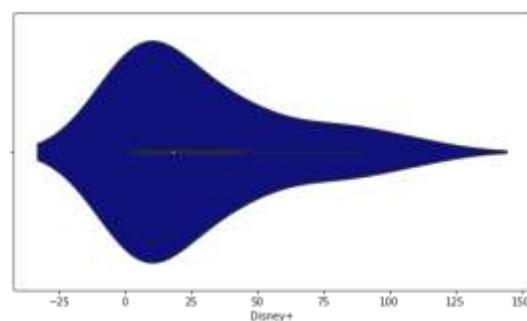
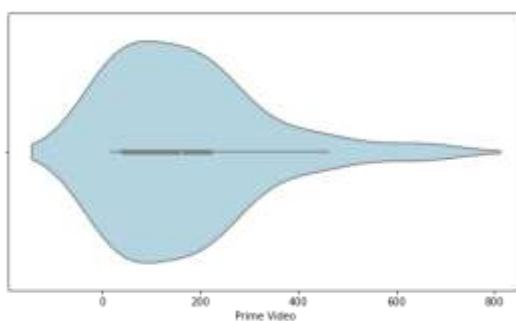
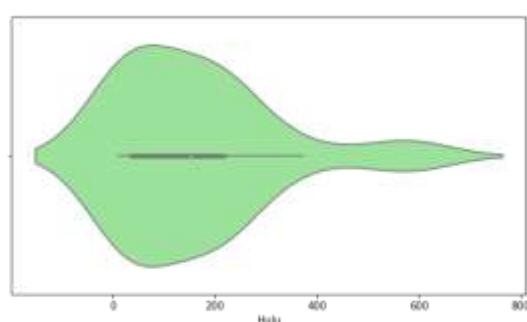
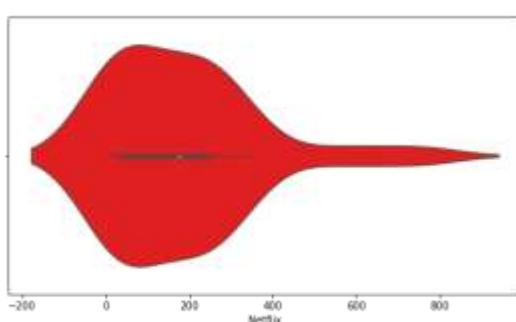
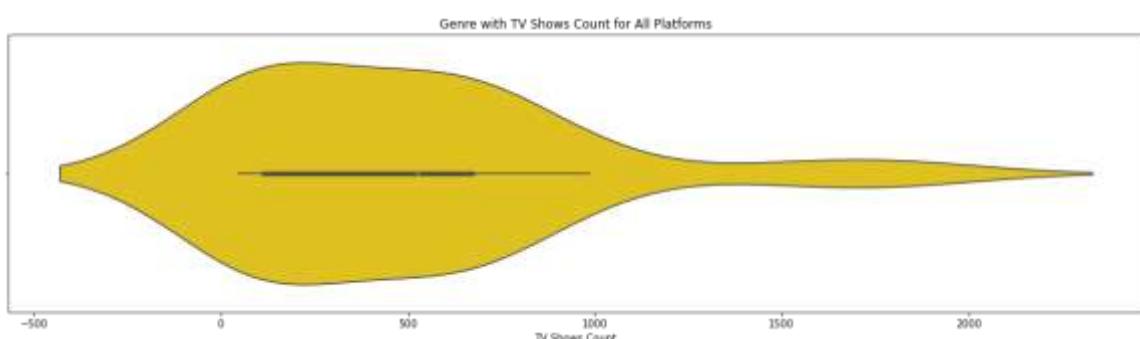
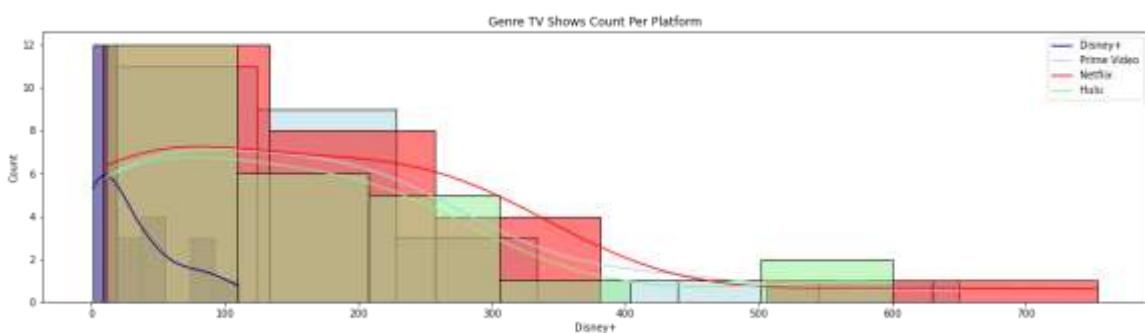
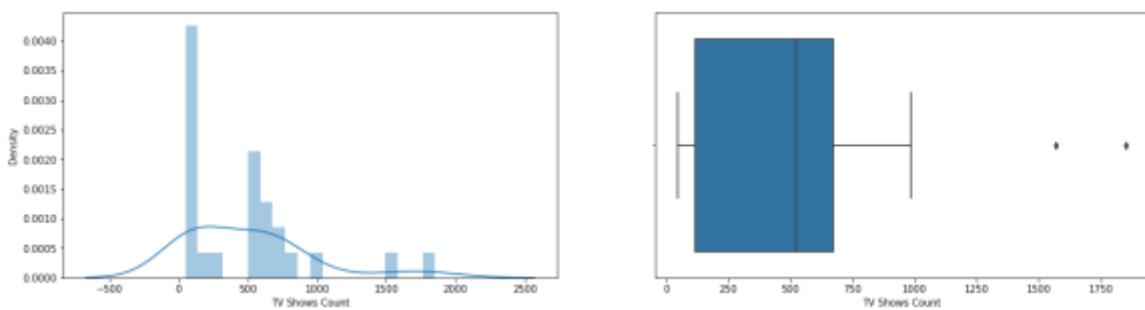
COUNTRY

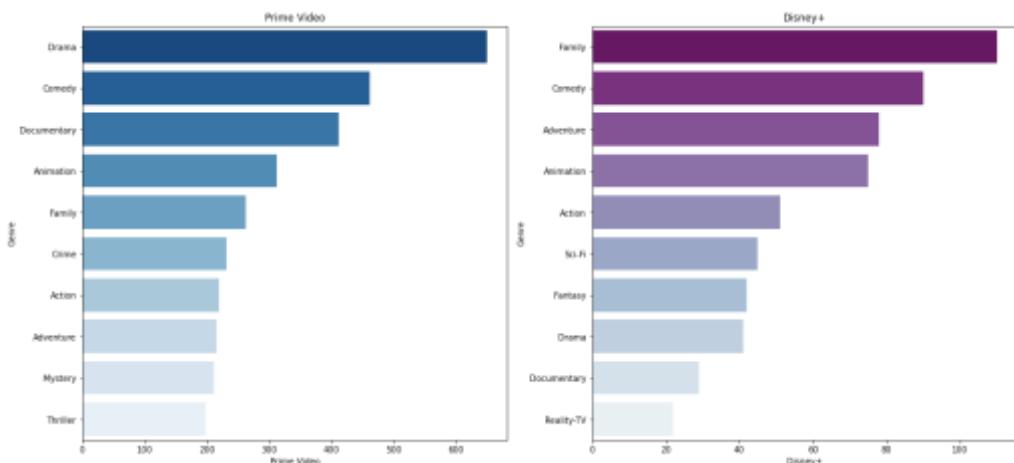
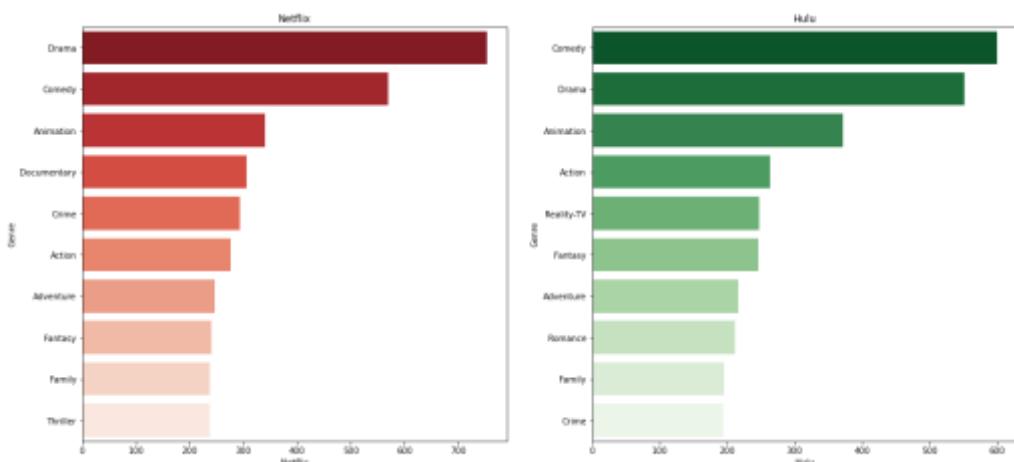
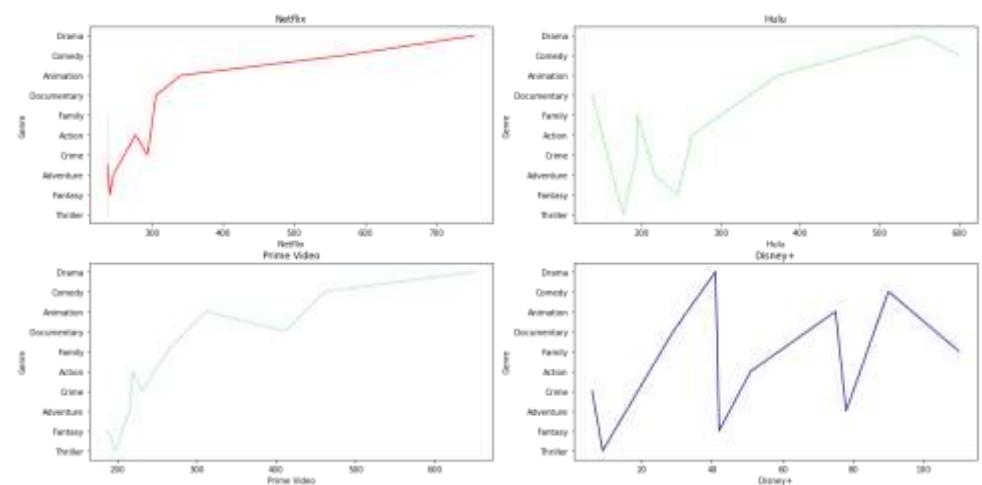
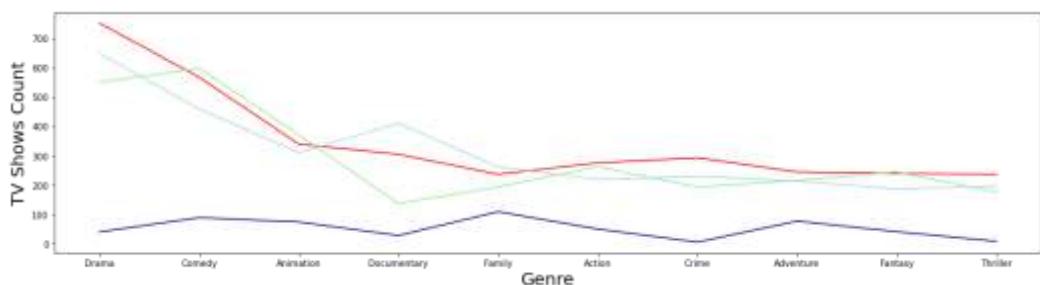


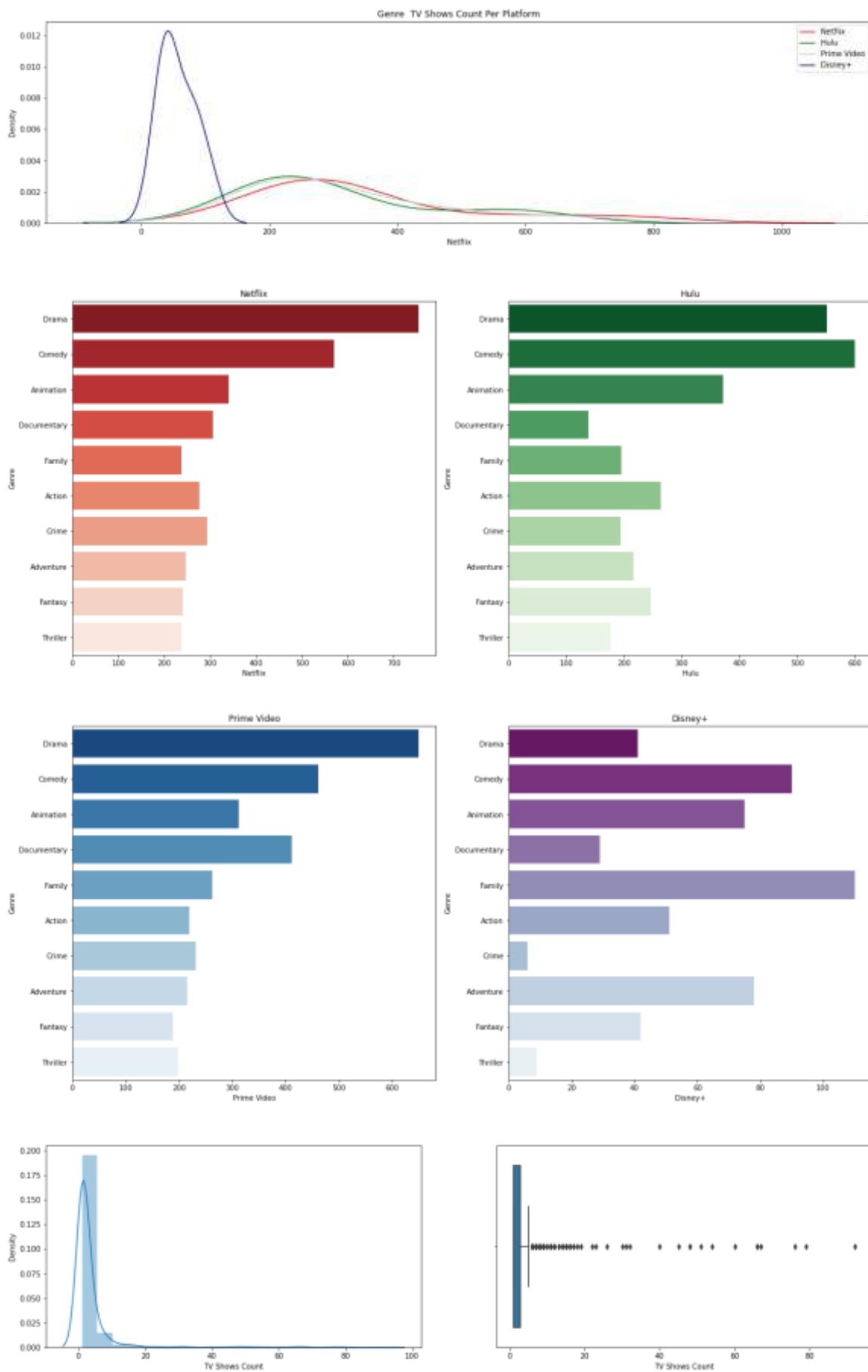
DIRECTOR

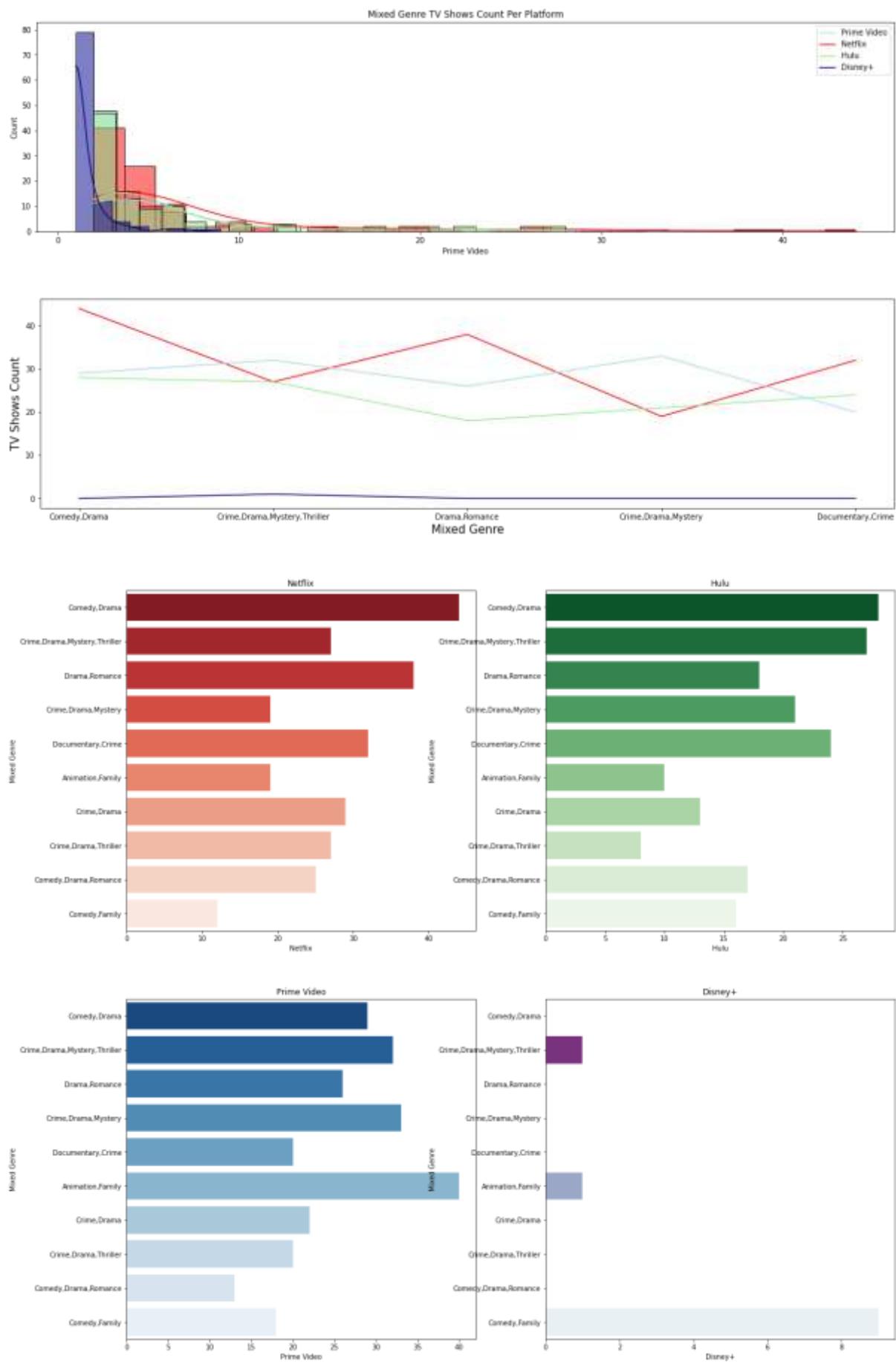


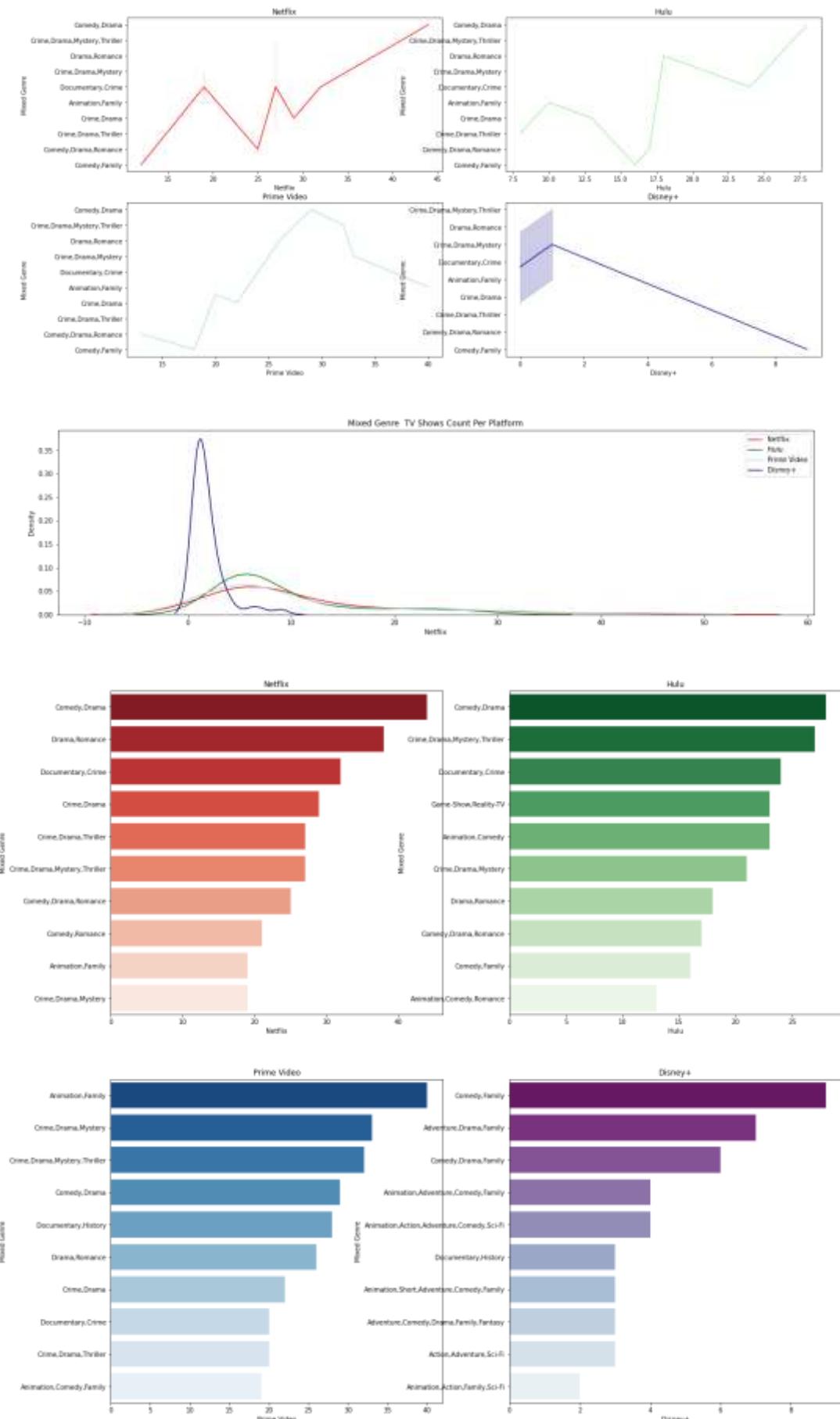




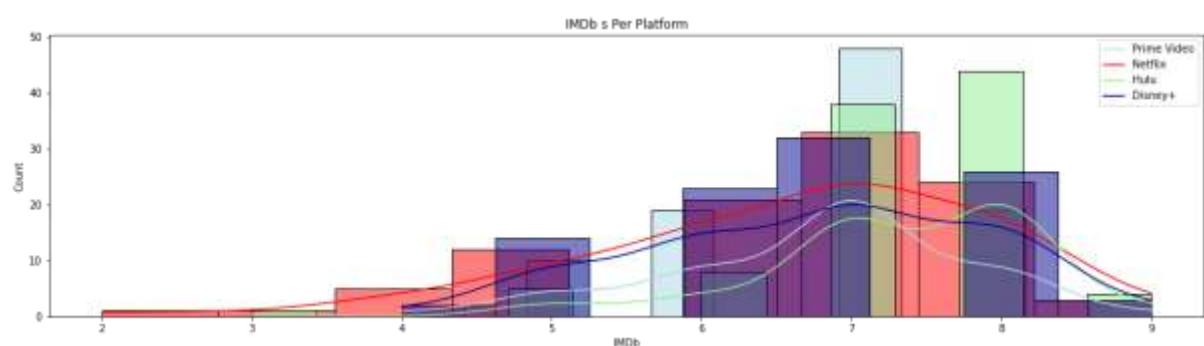
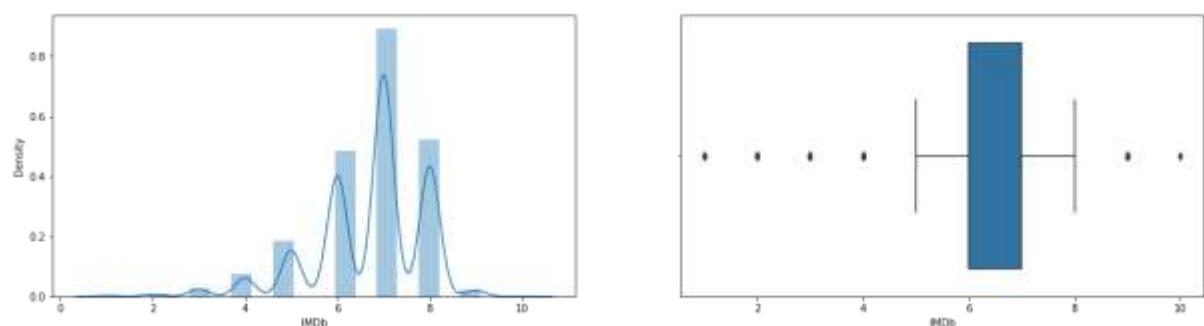


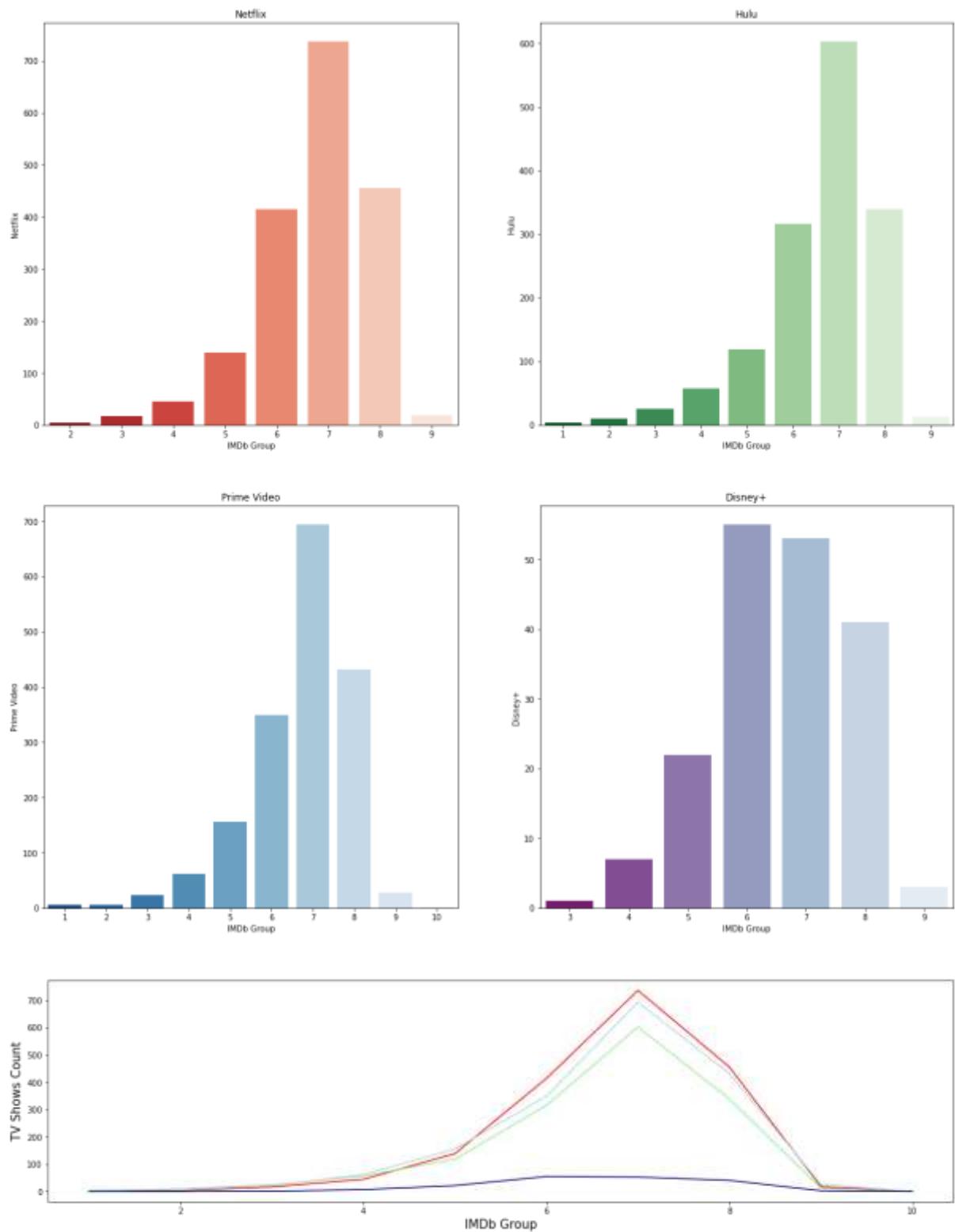


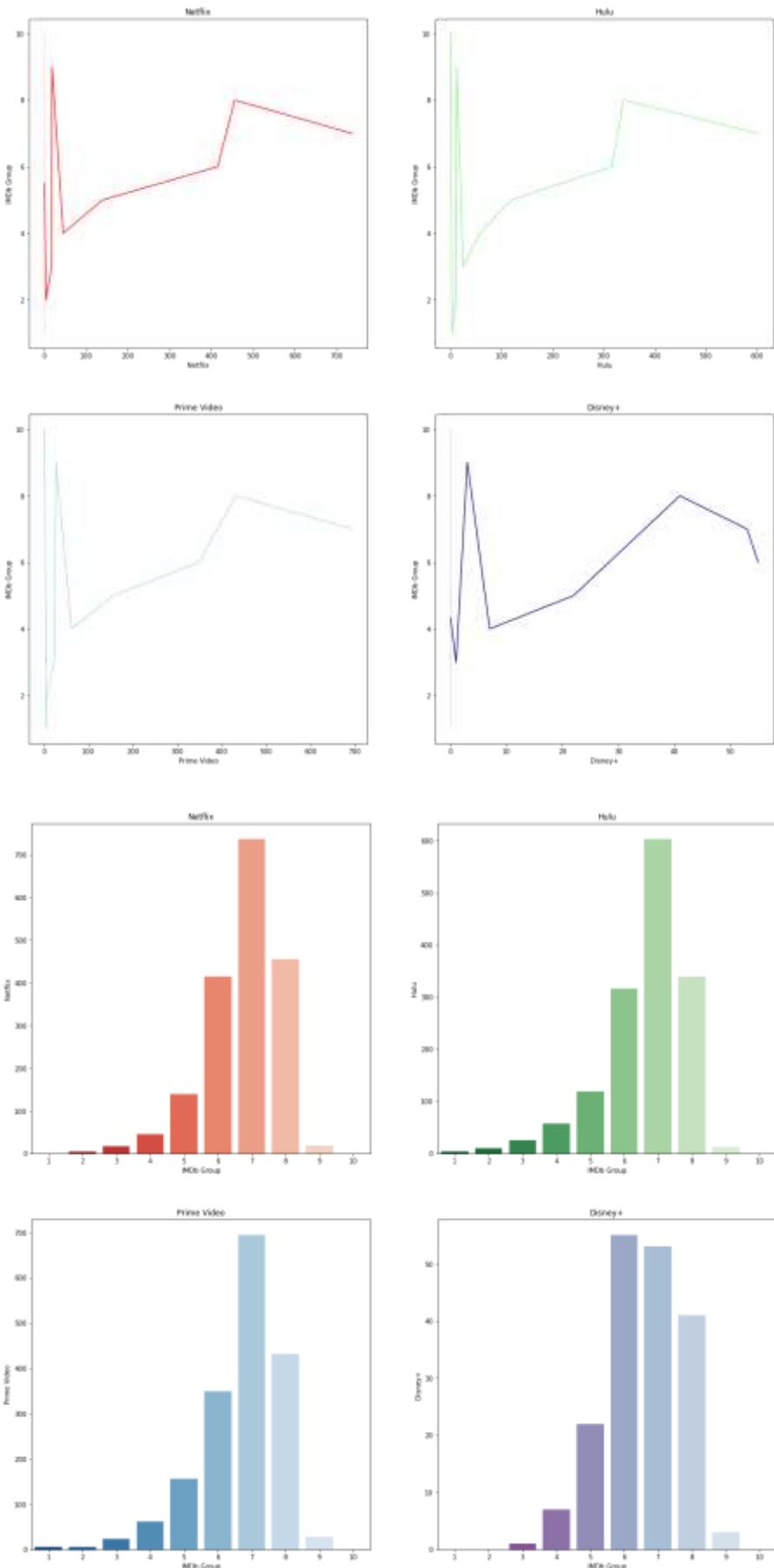




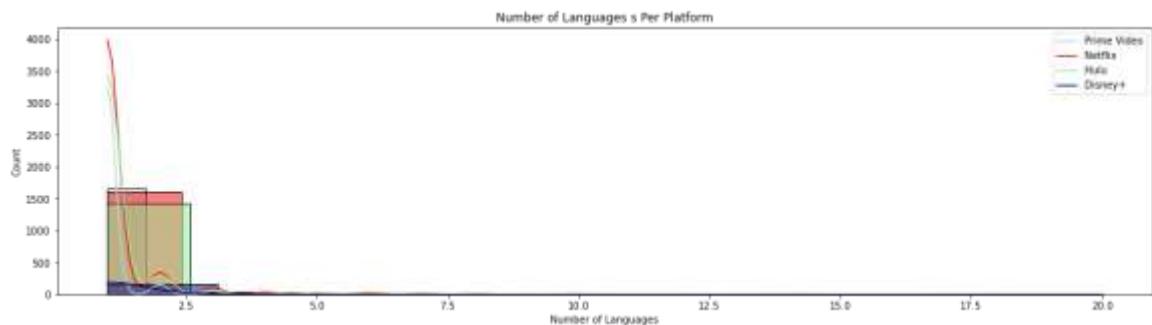
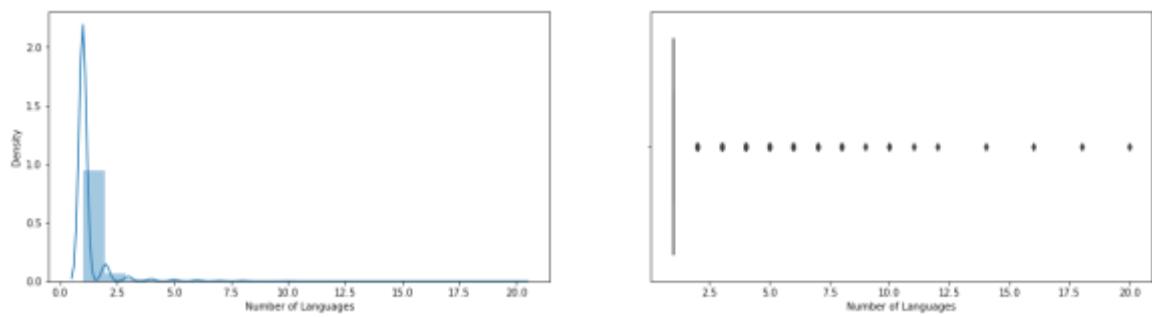
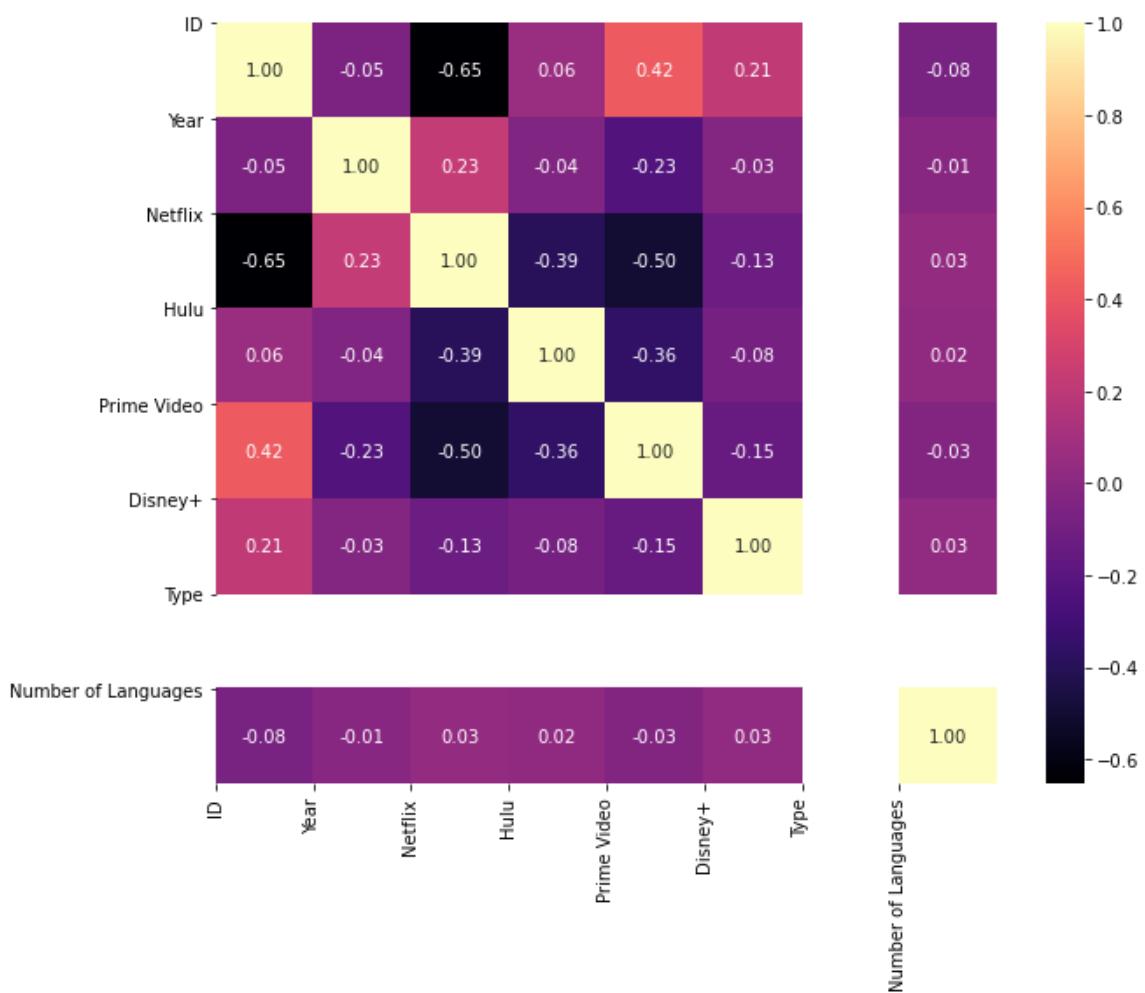
IMDB

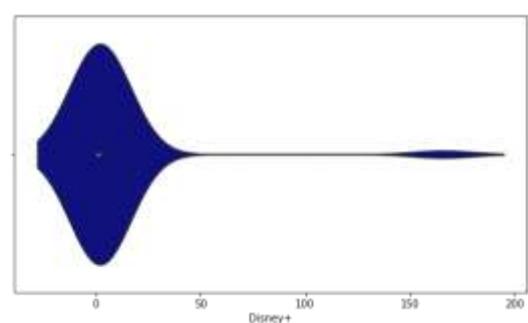
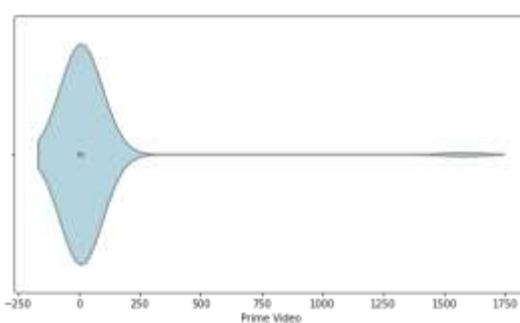
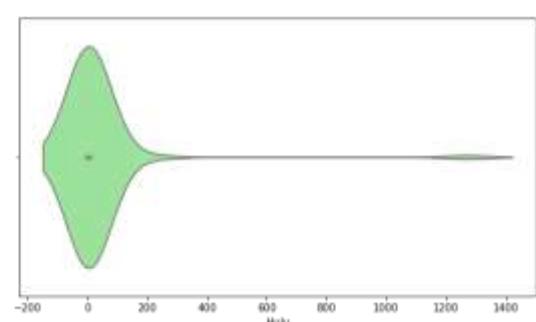
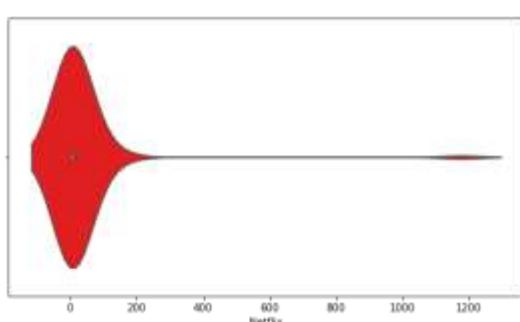
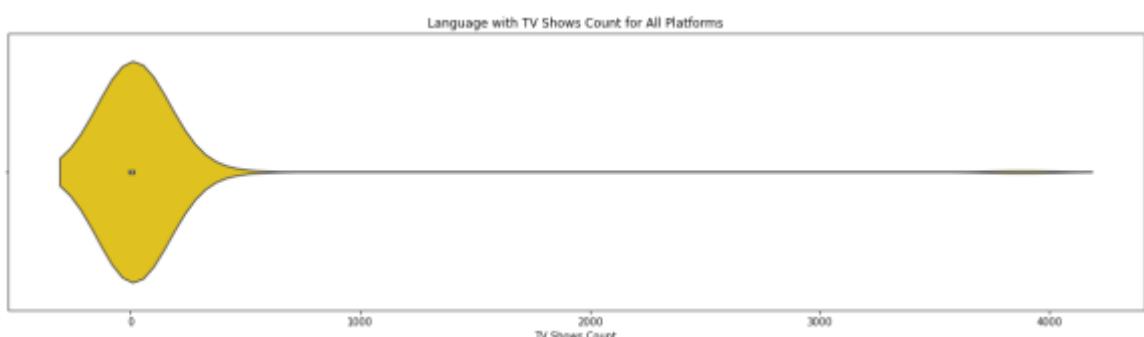
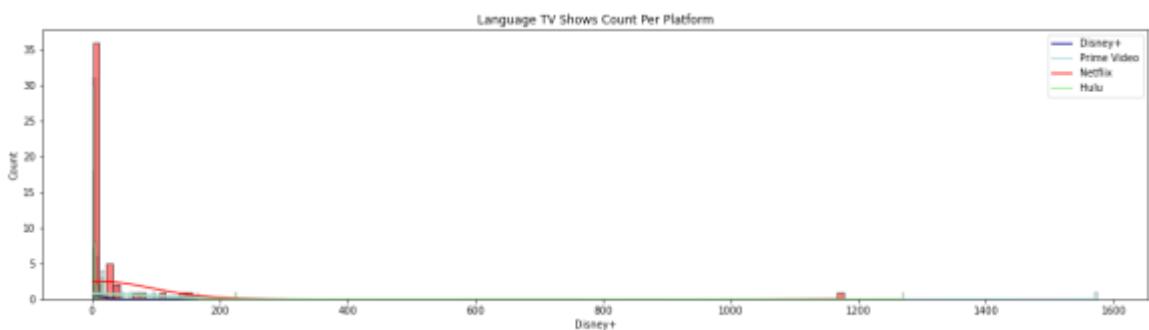
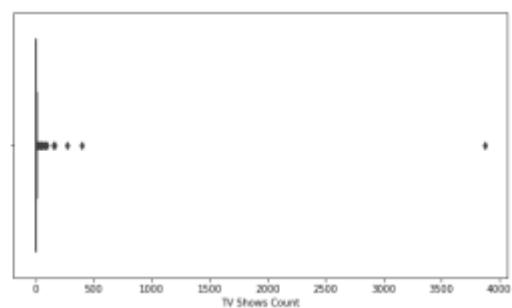
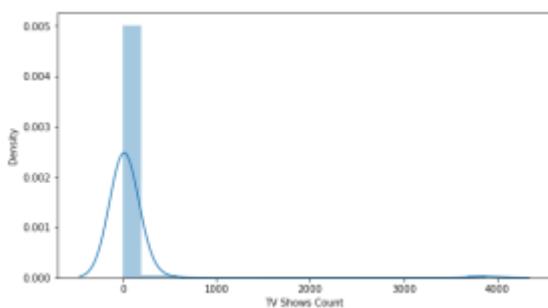


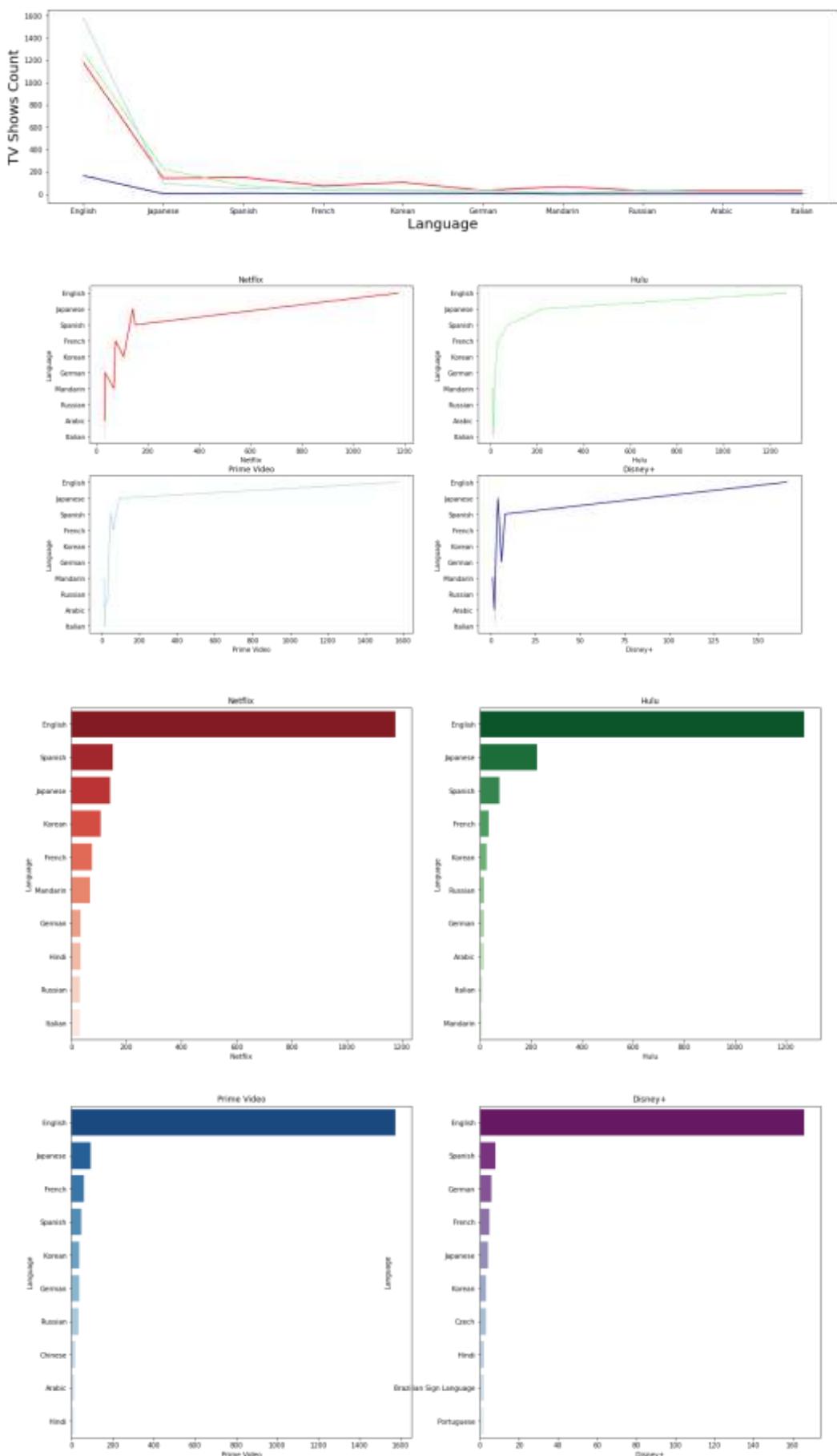


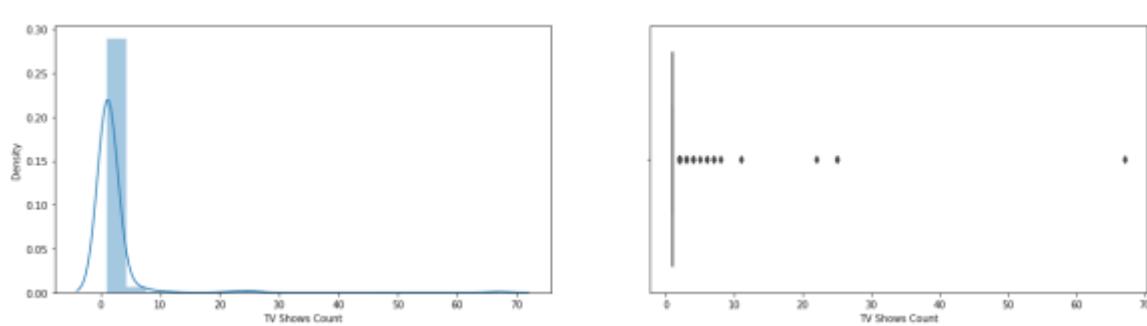
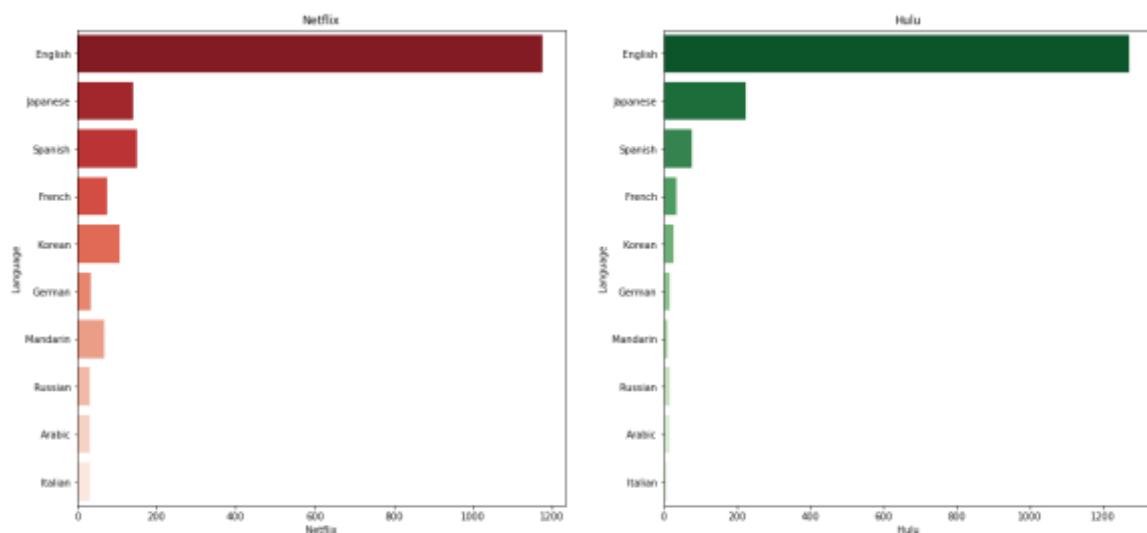
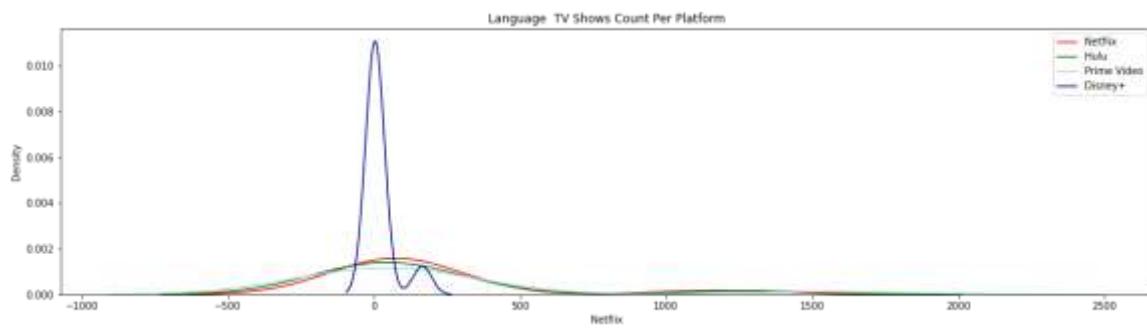


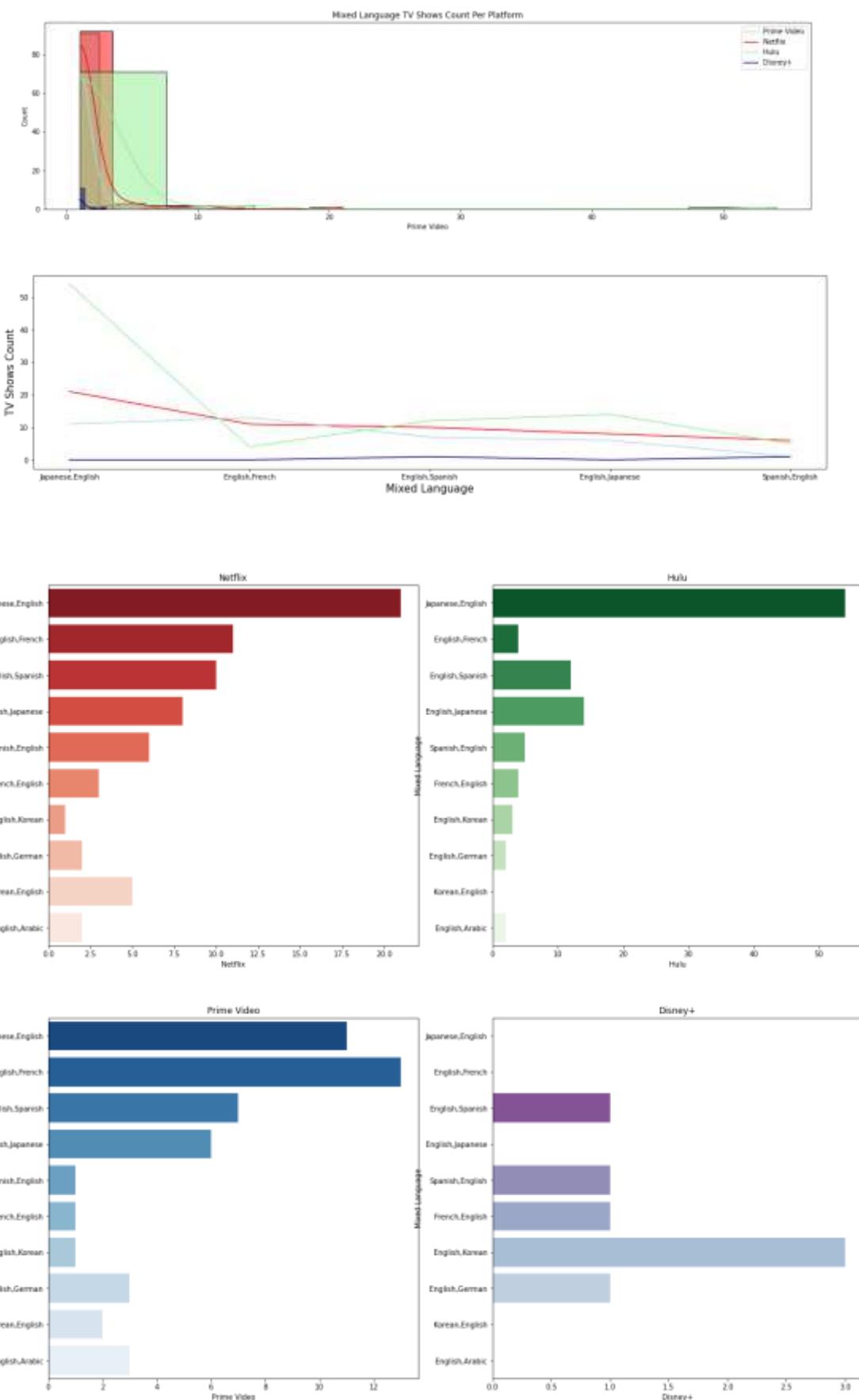
LANGUAGE

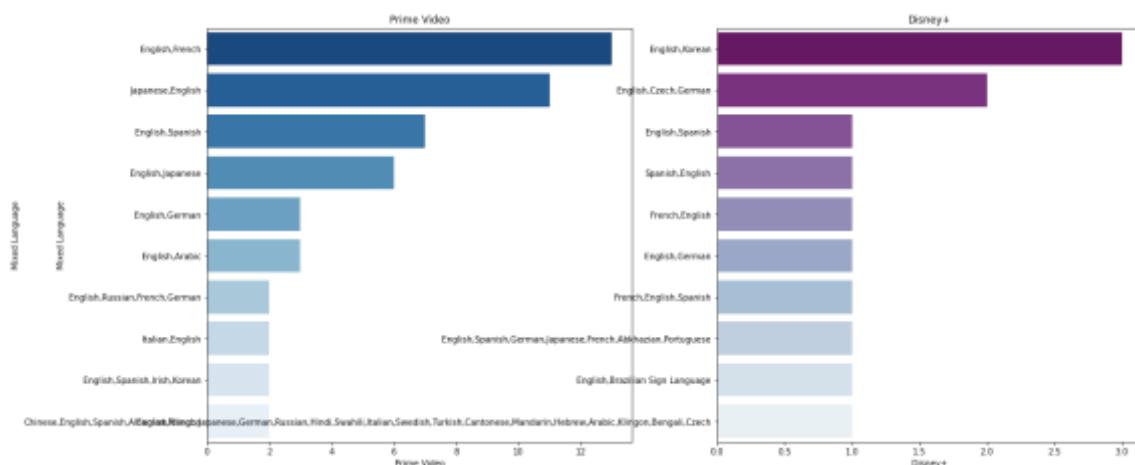
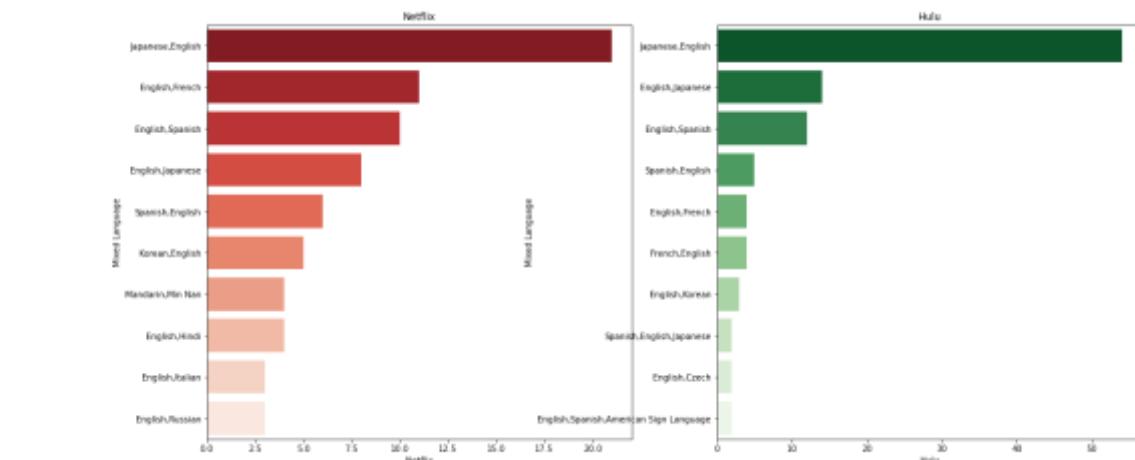
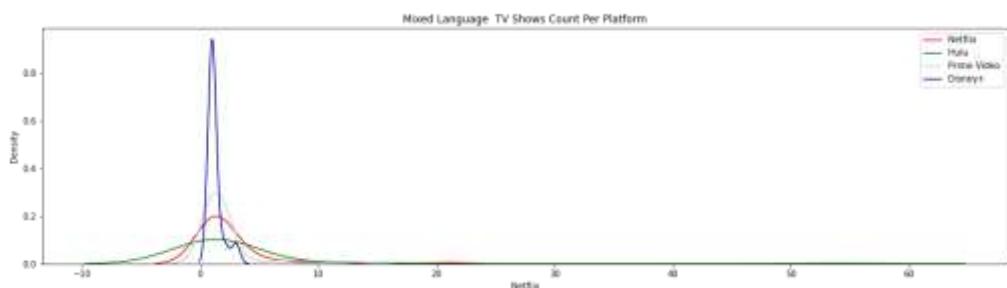
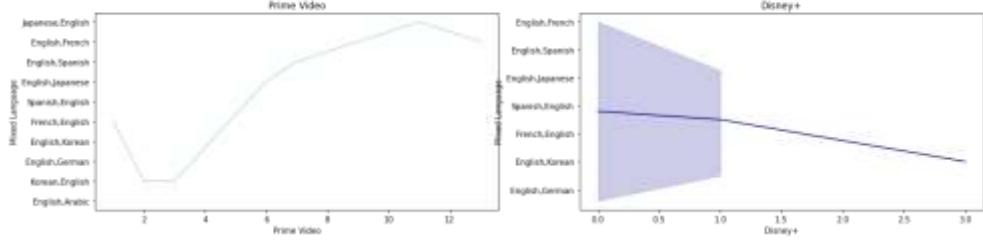
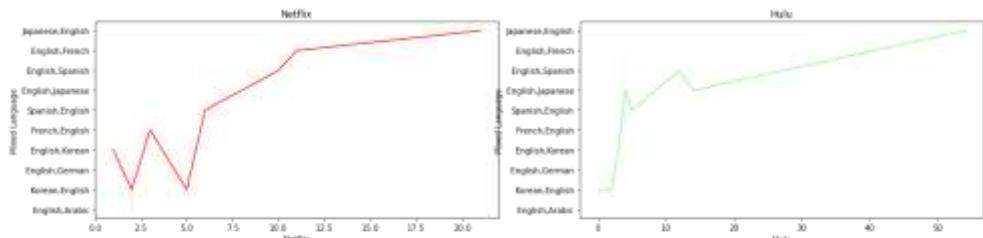






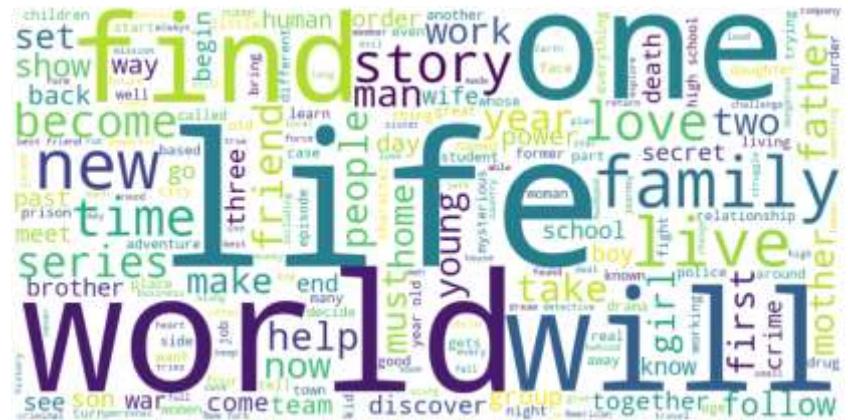




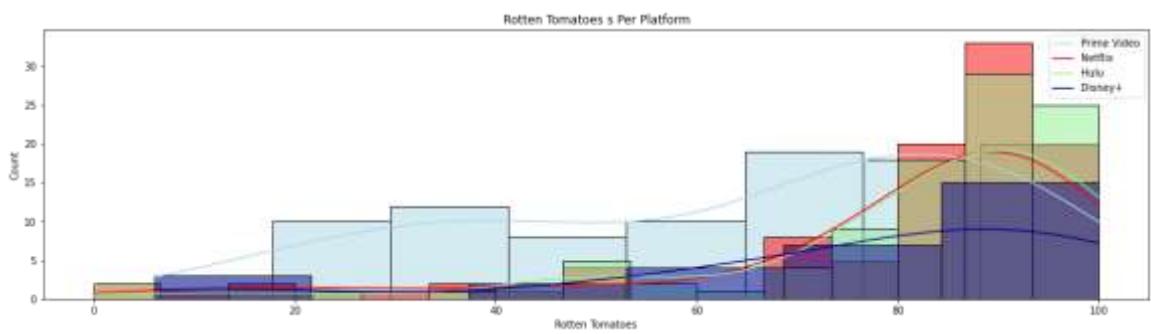
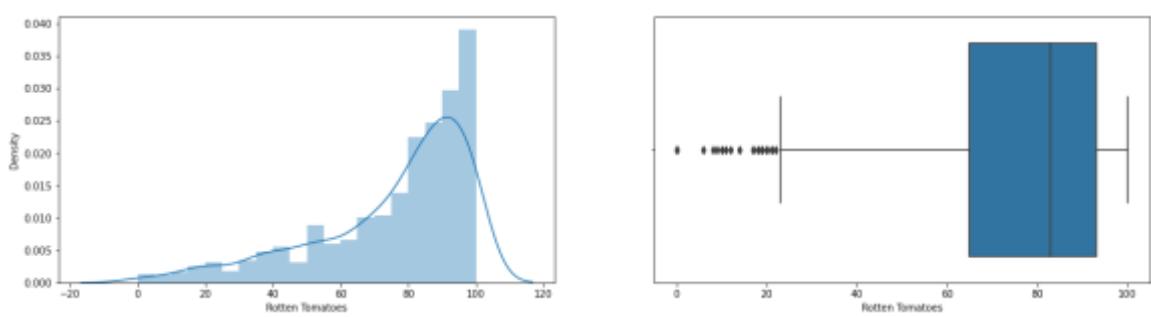
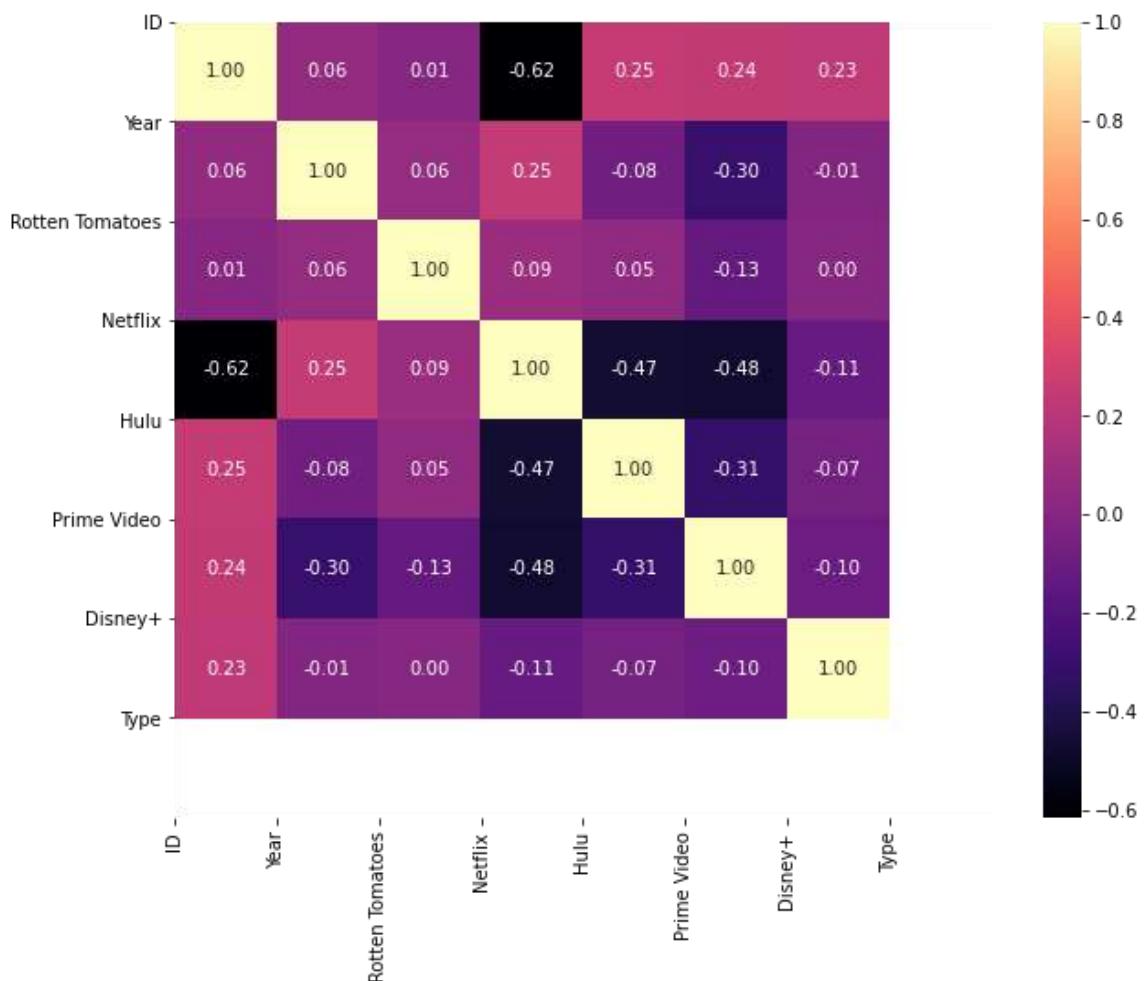


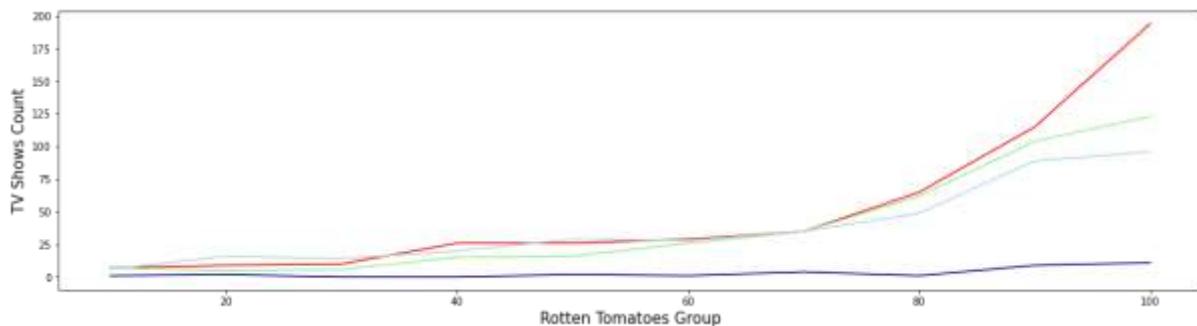
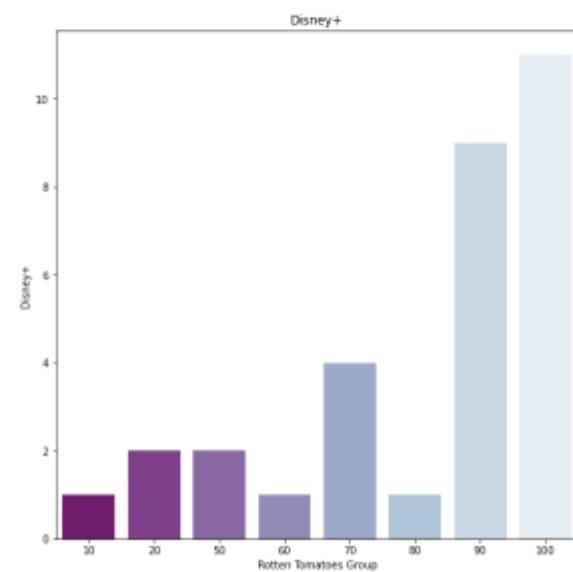
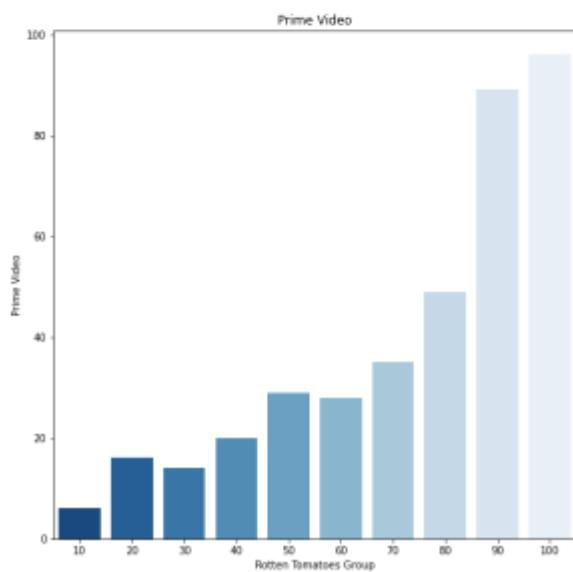
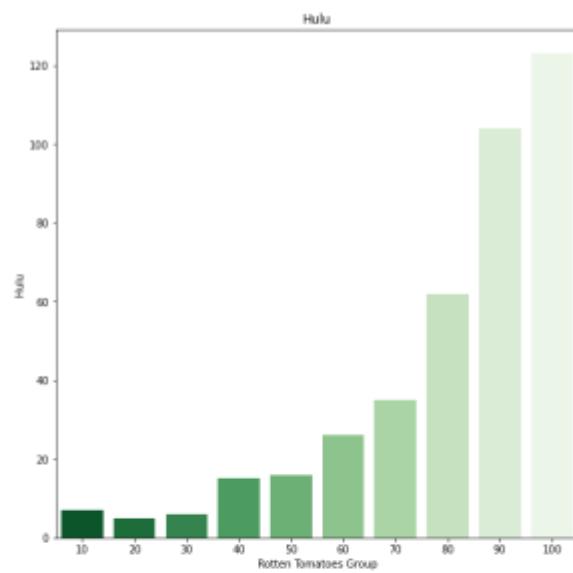
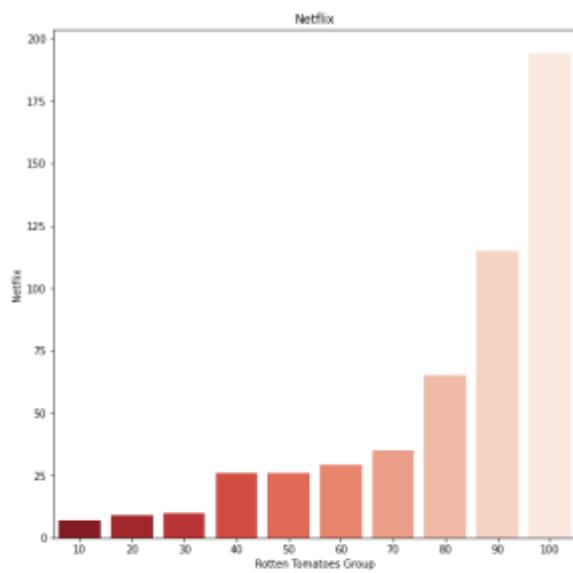
PLOTLINE

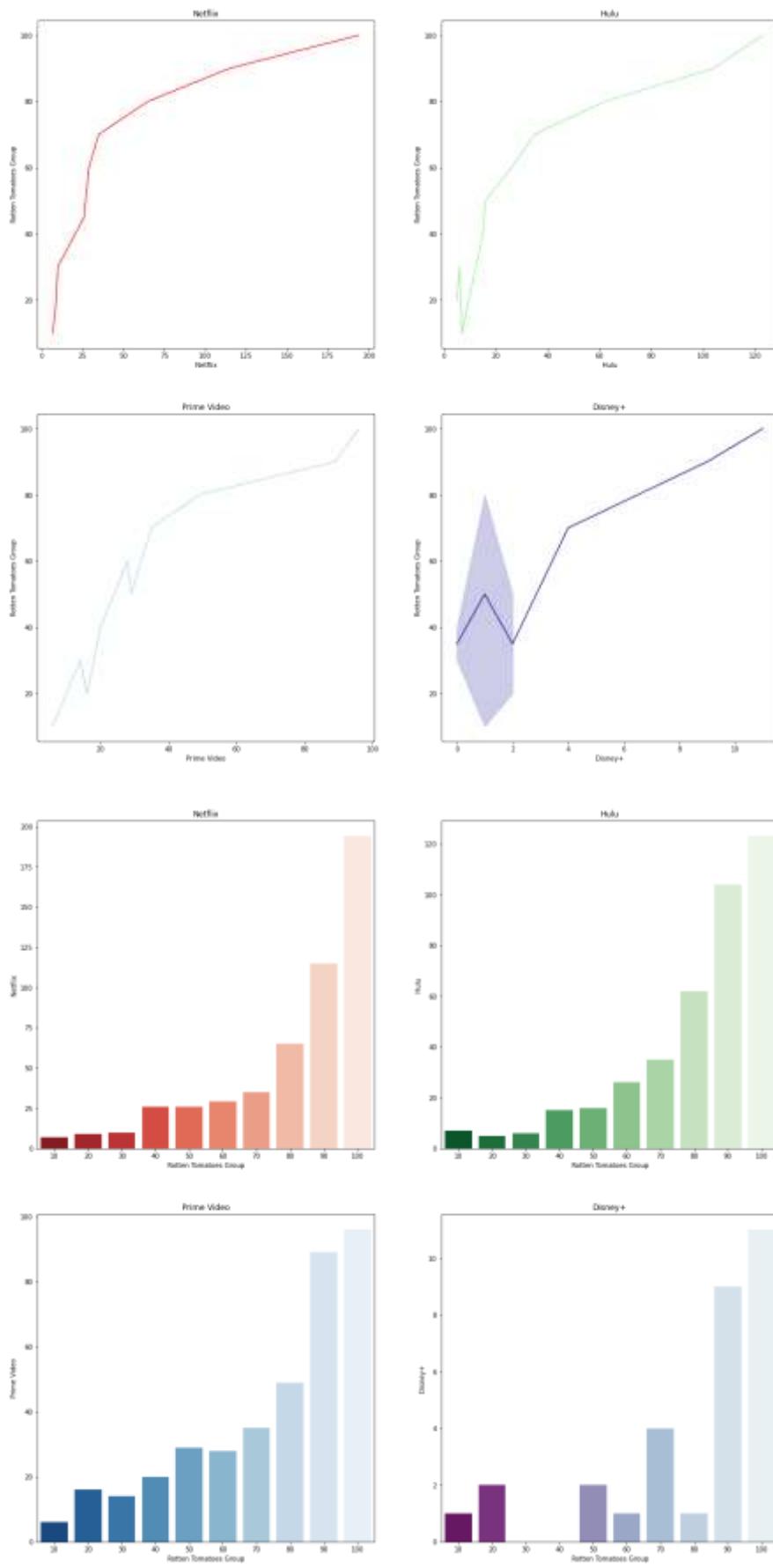




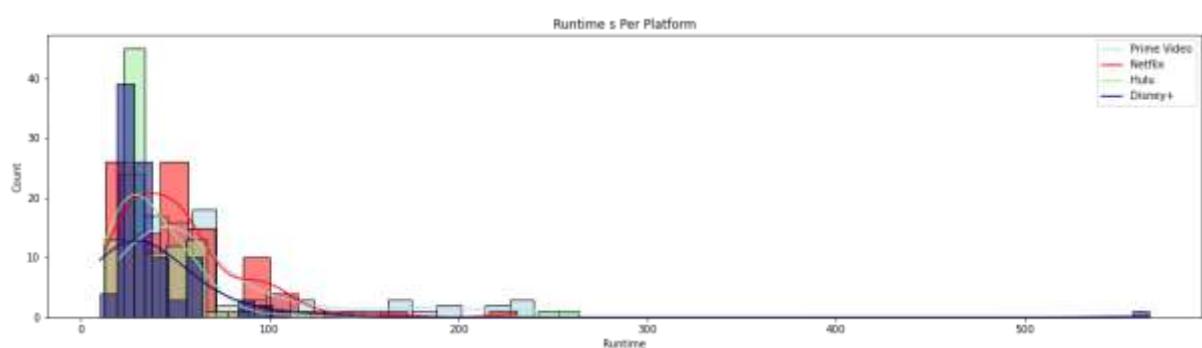
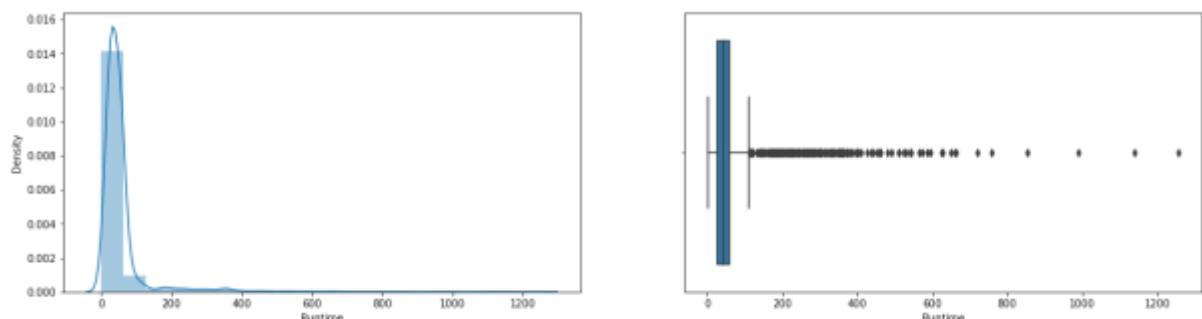
ROTTEN TOMATOES

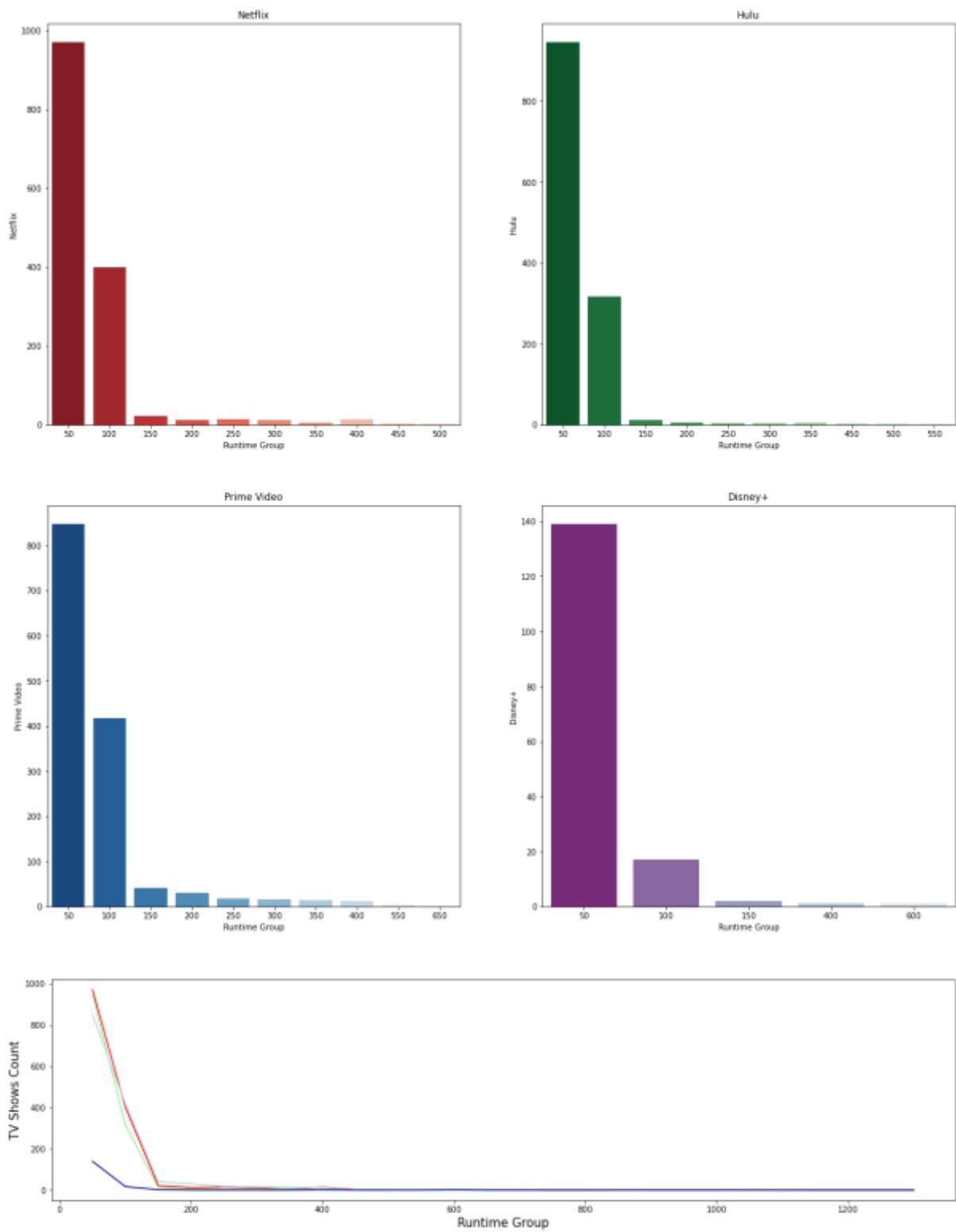


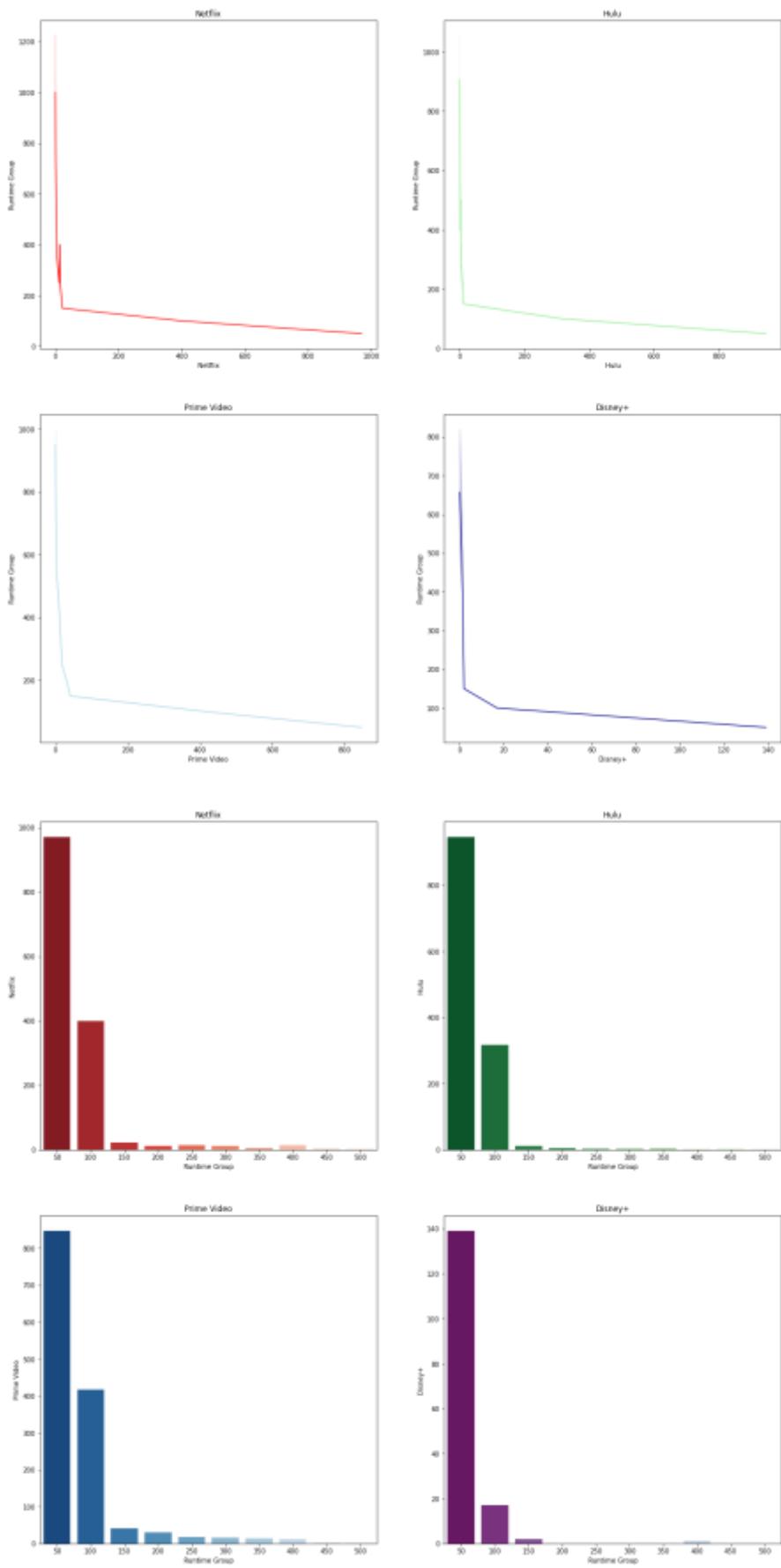


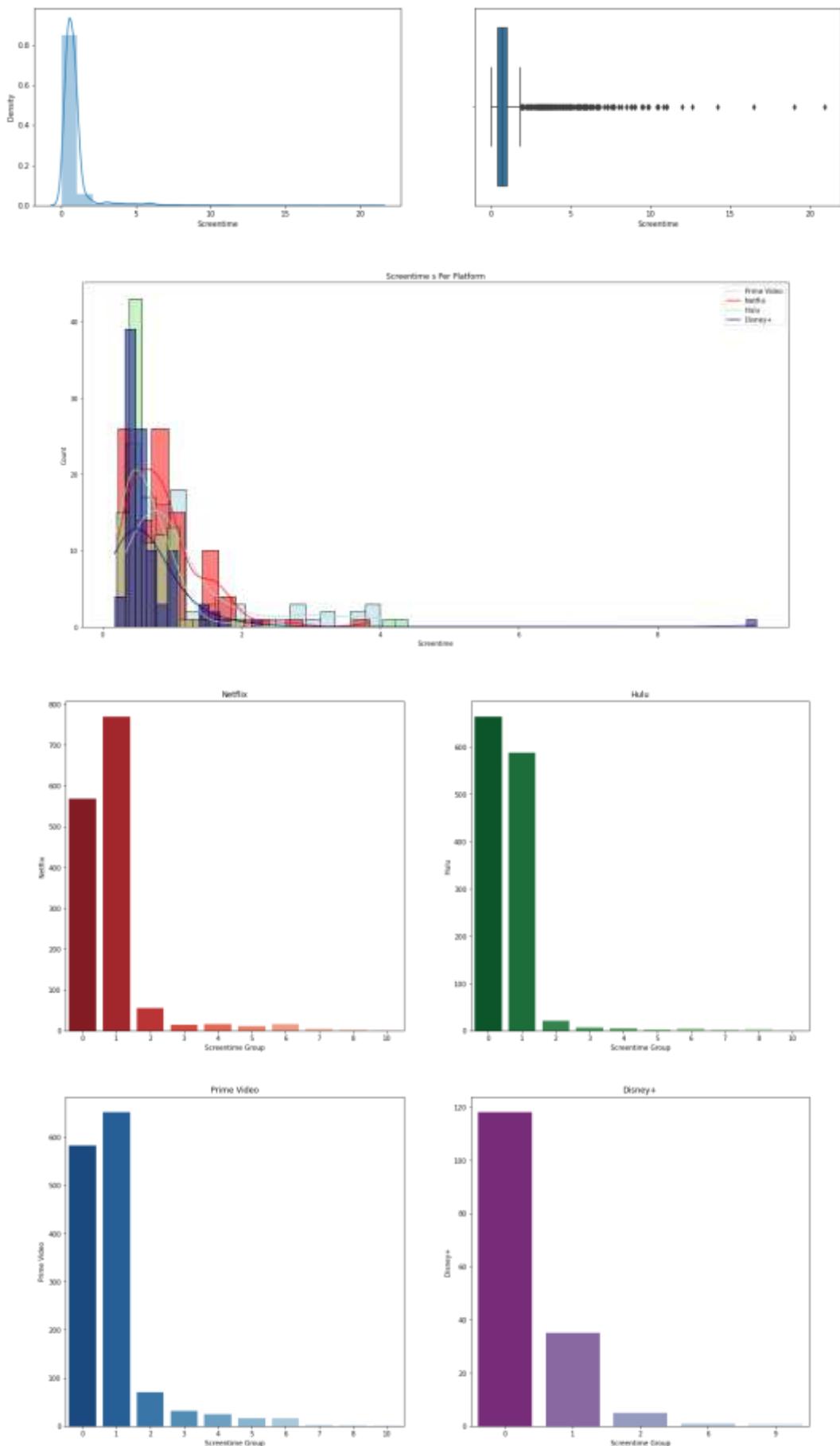


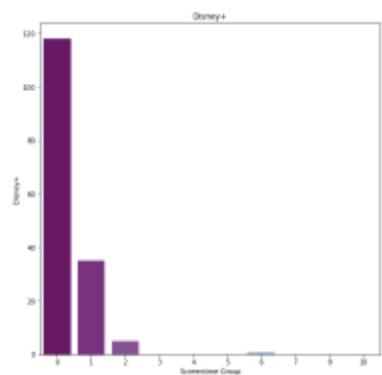
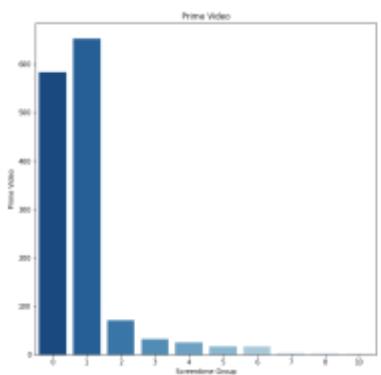
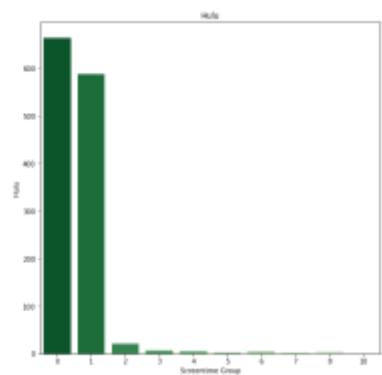
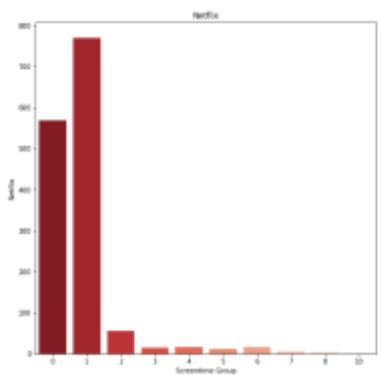
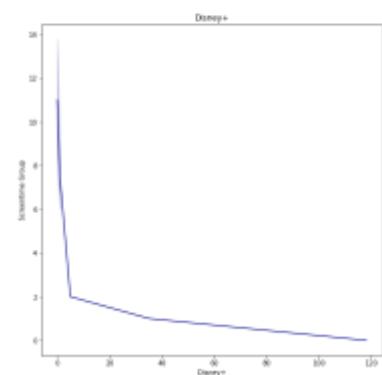
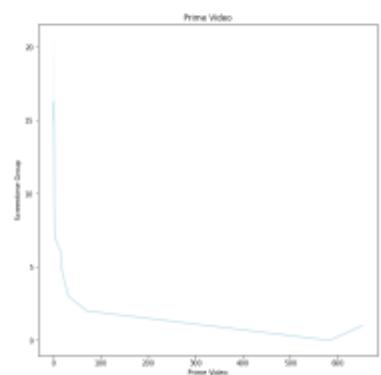
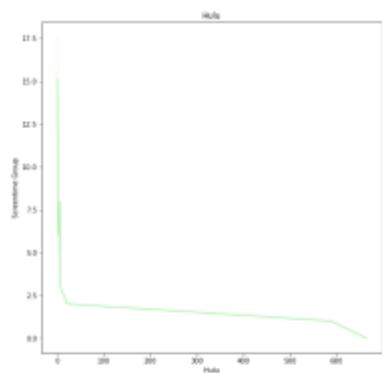
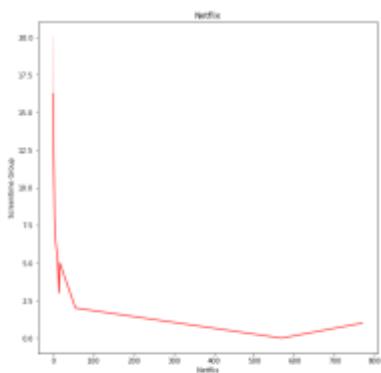
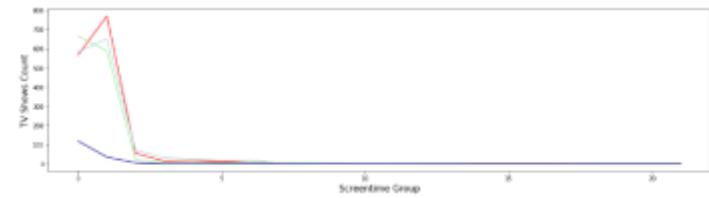
RUNTIME



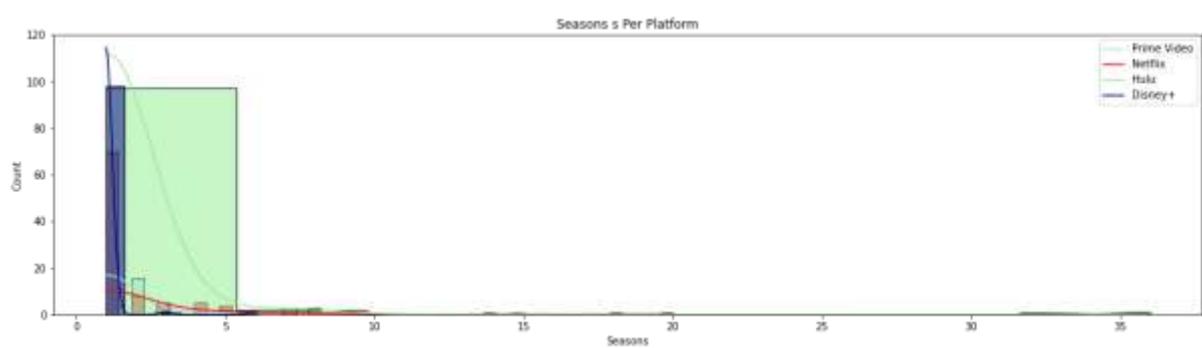
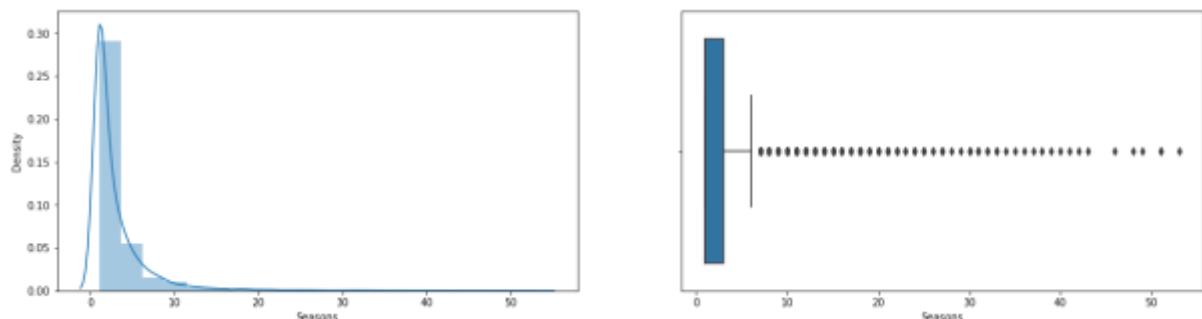


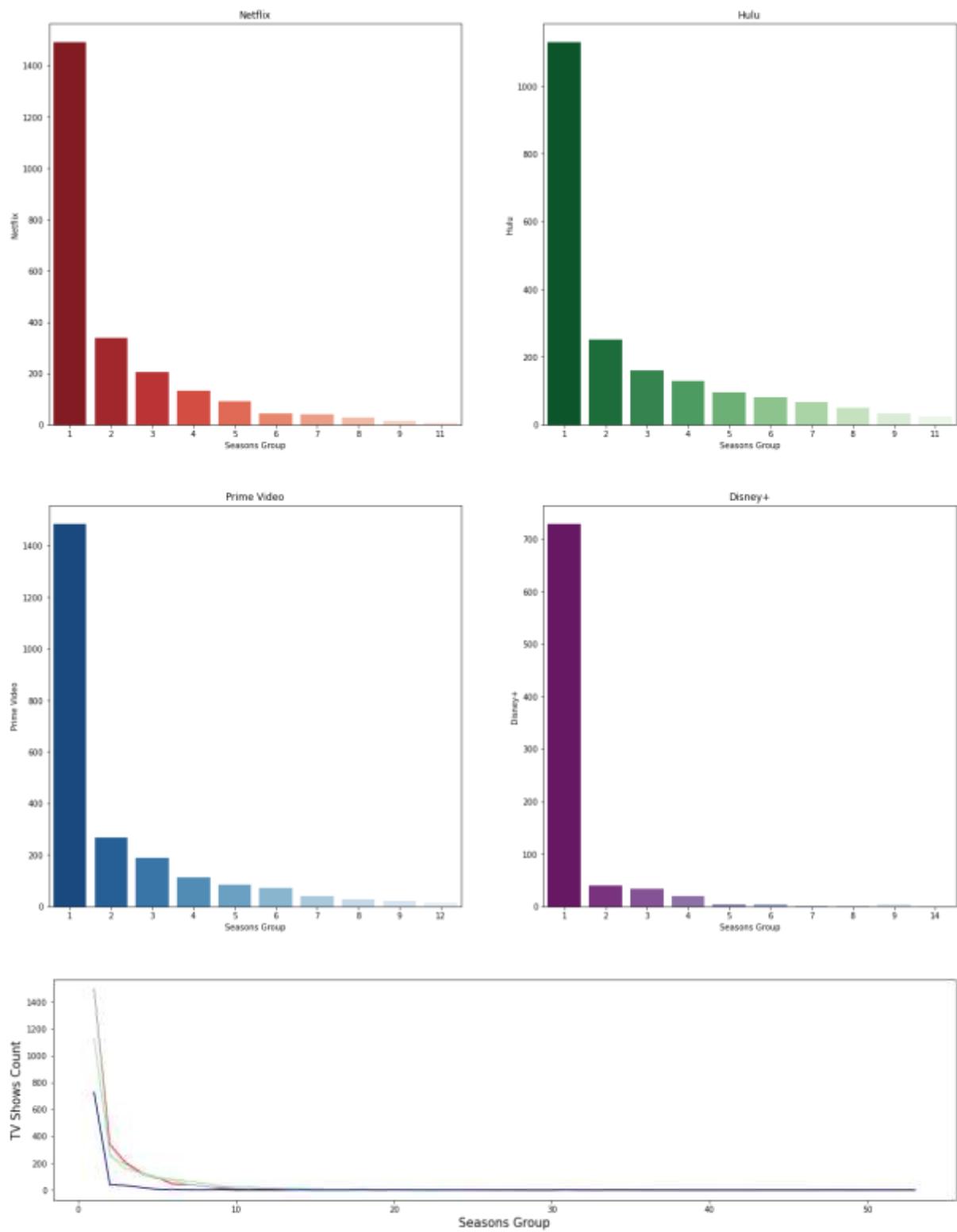


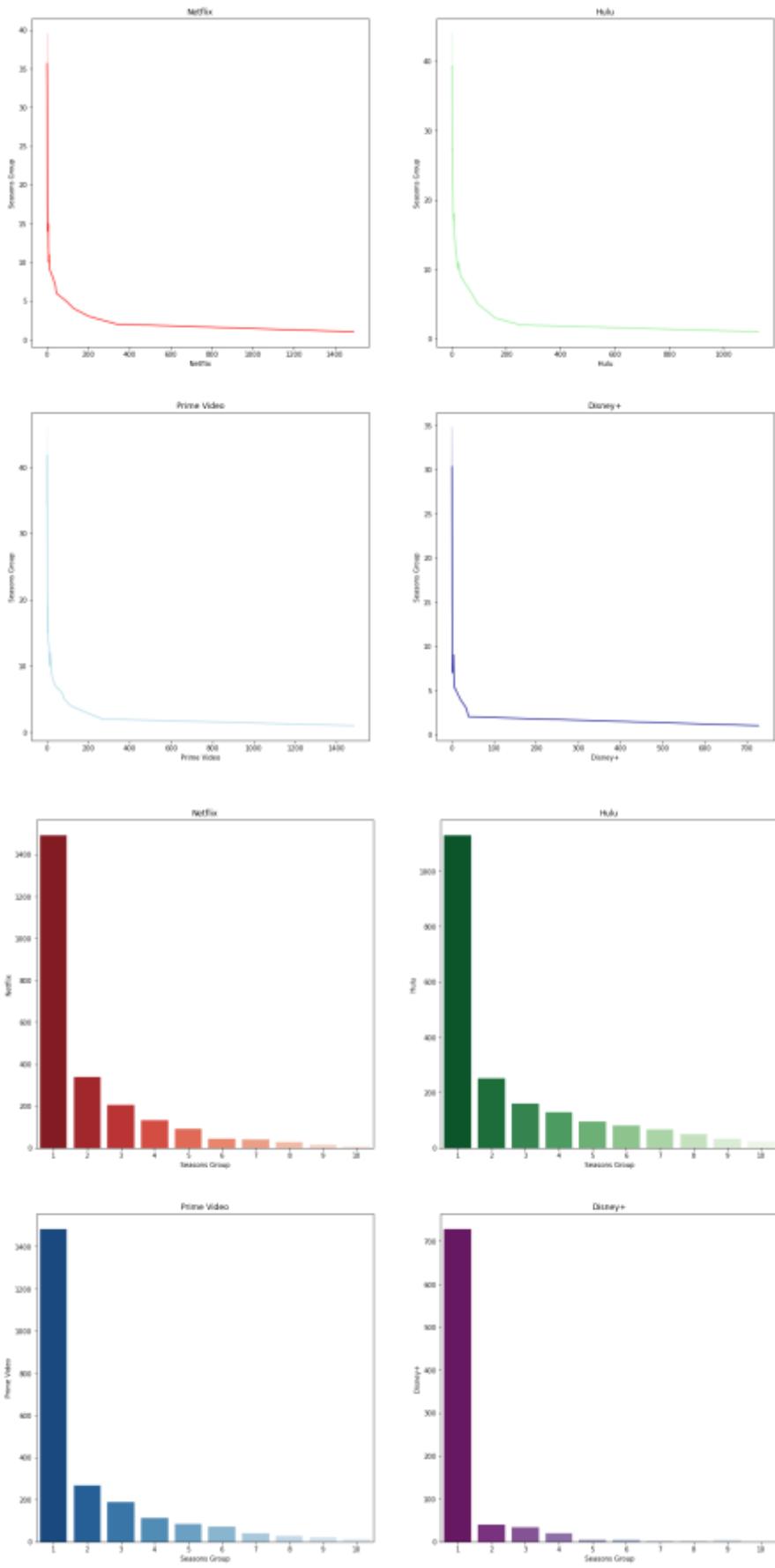


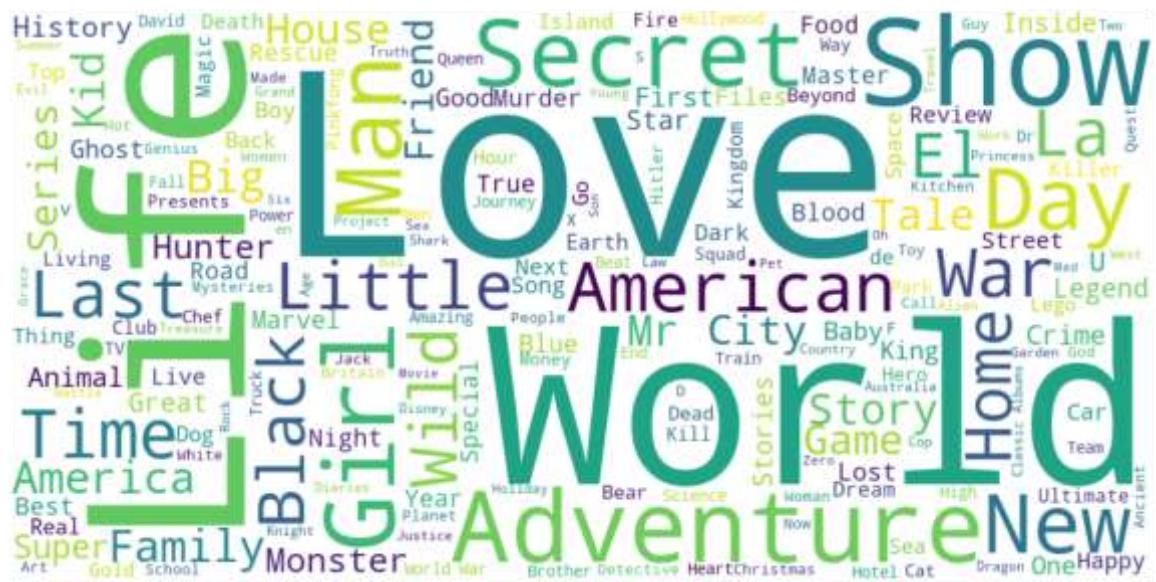
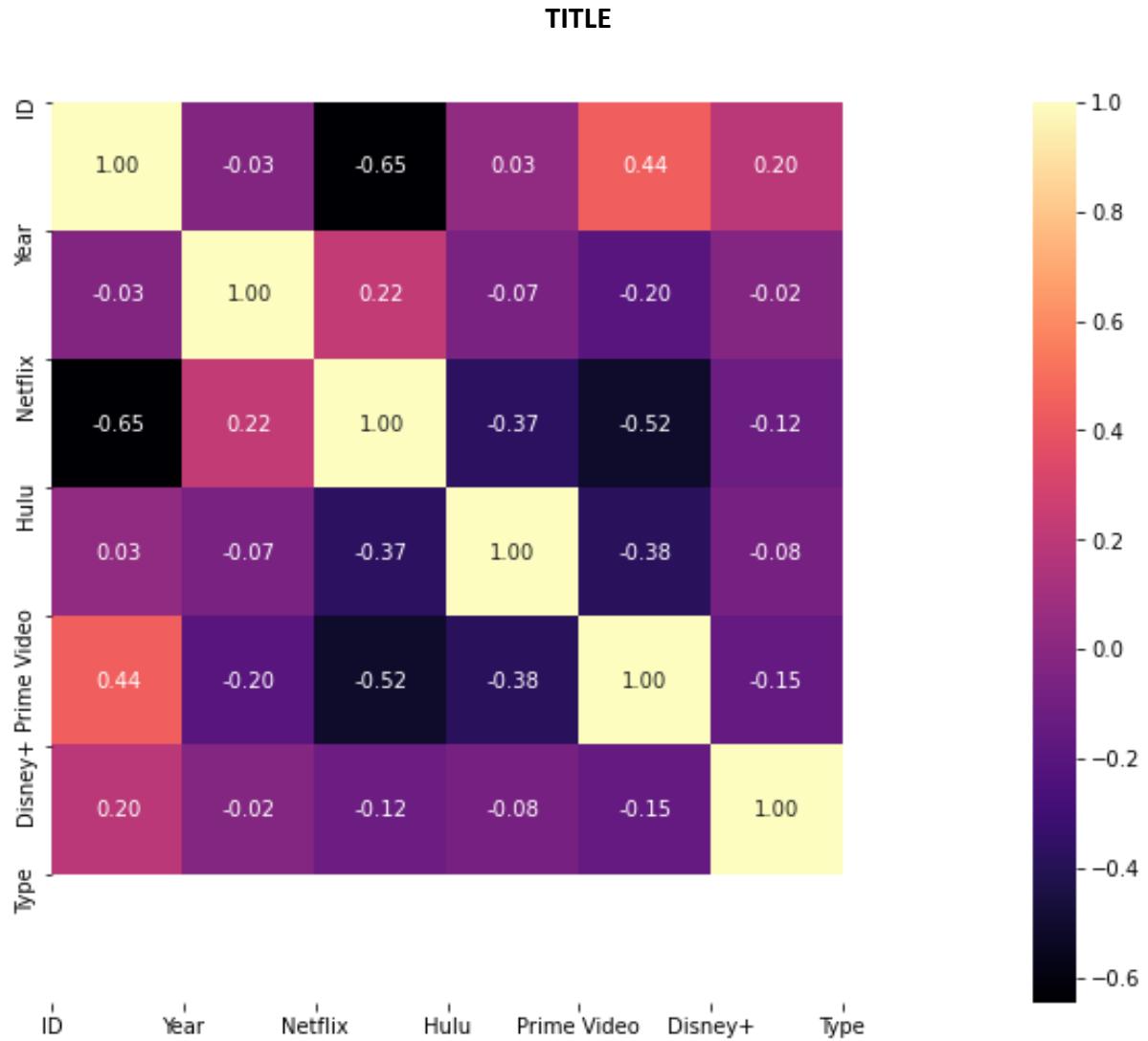


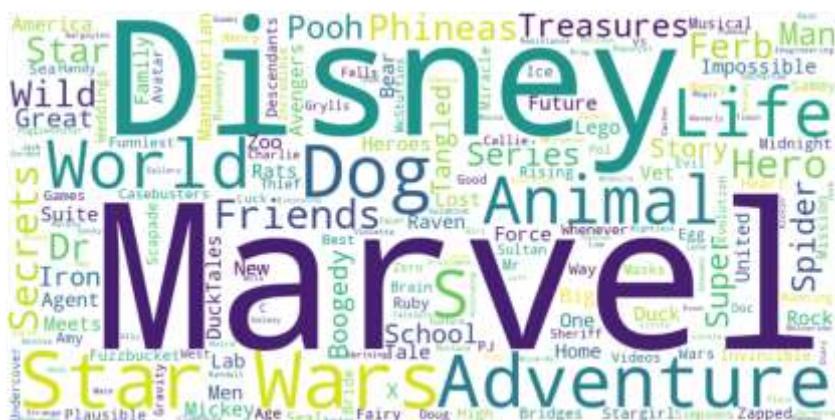
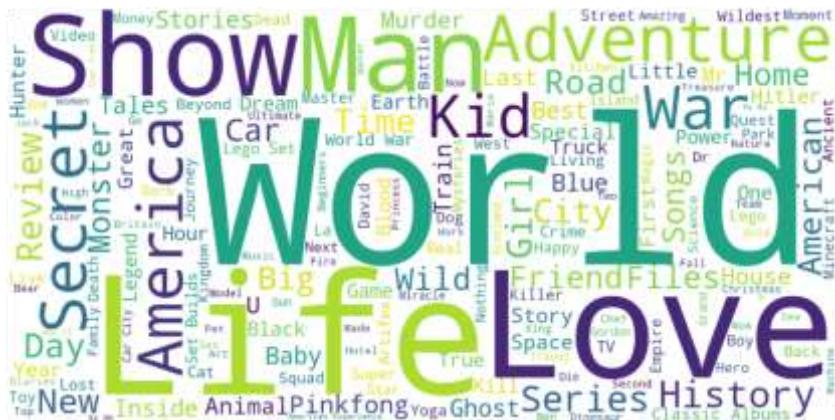
SEASON

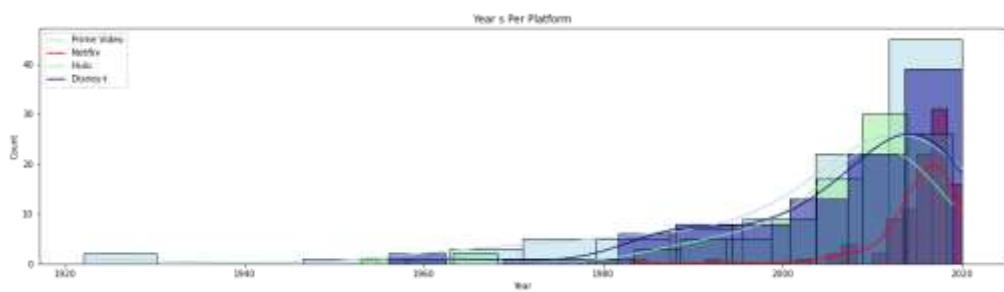
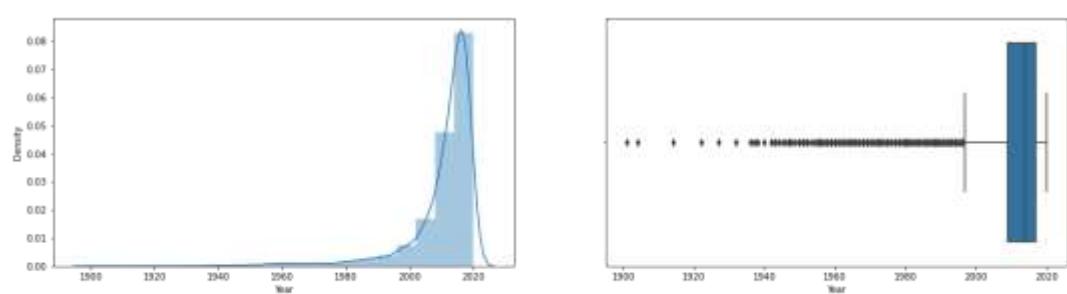
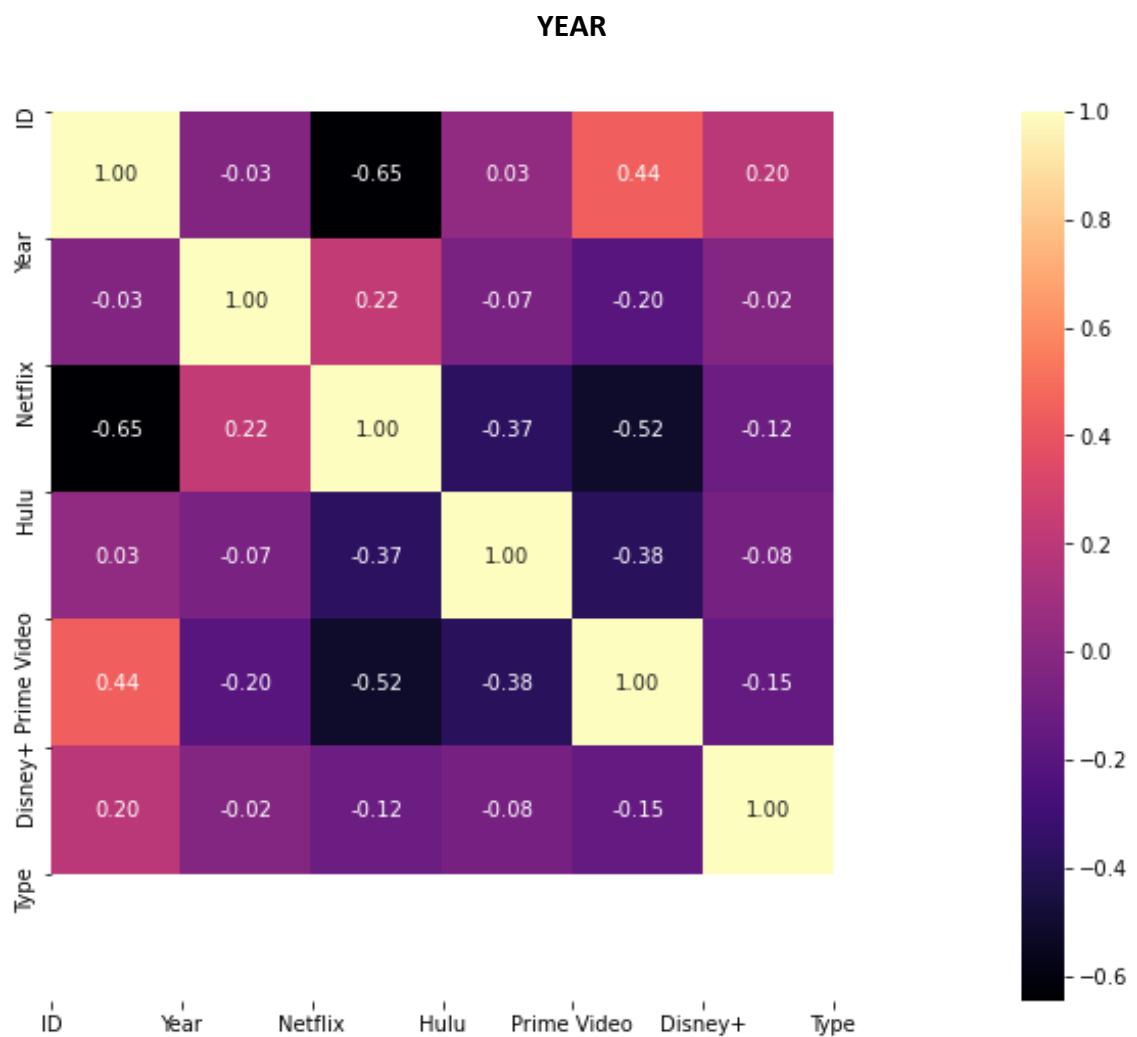


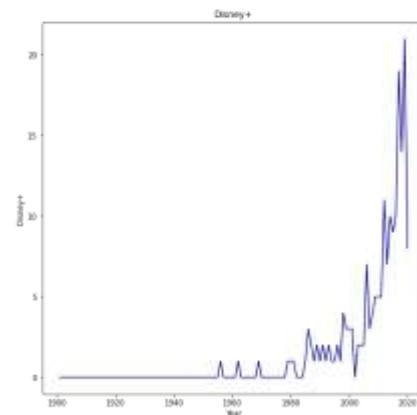
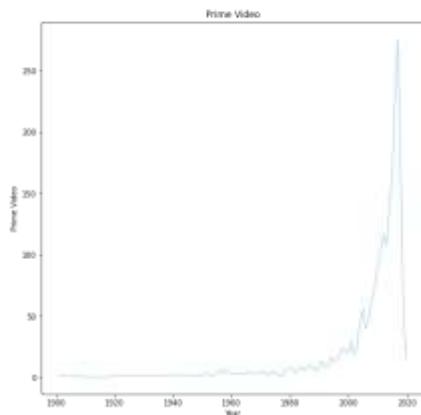
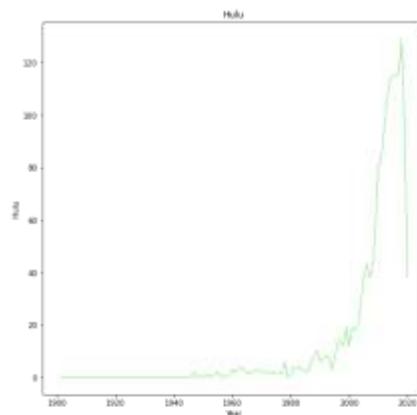
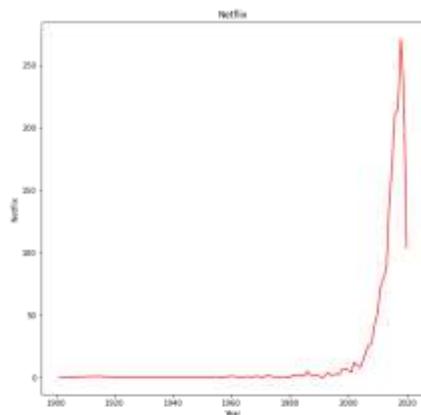
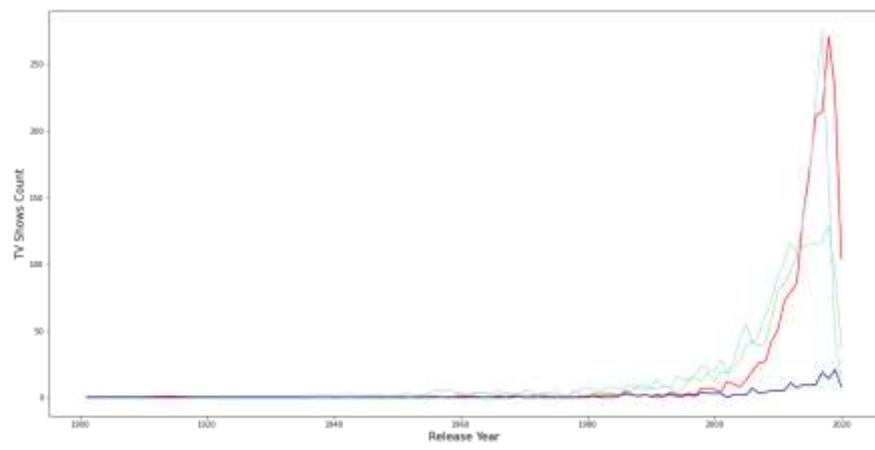
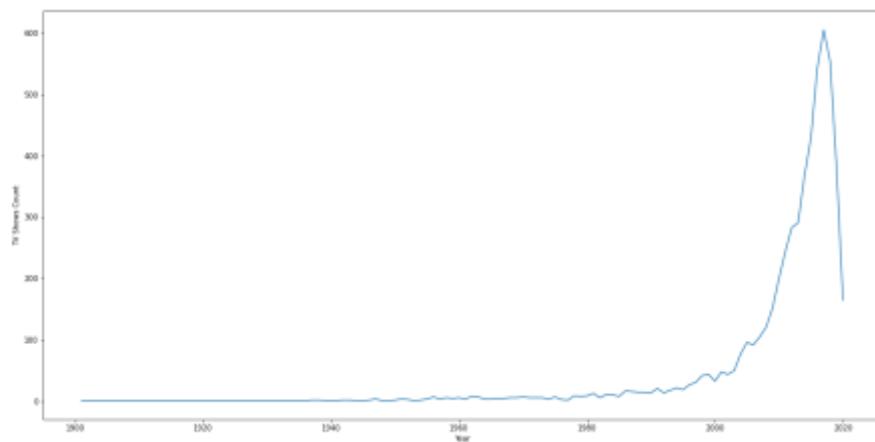


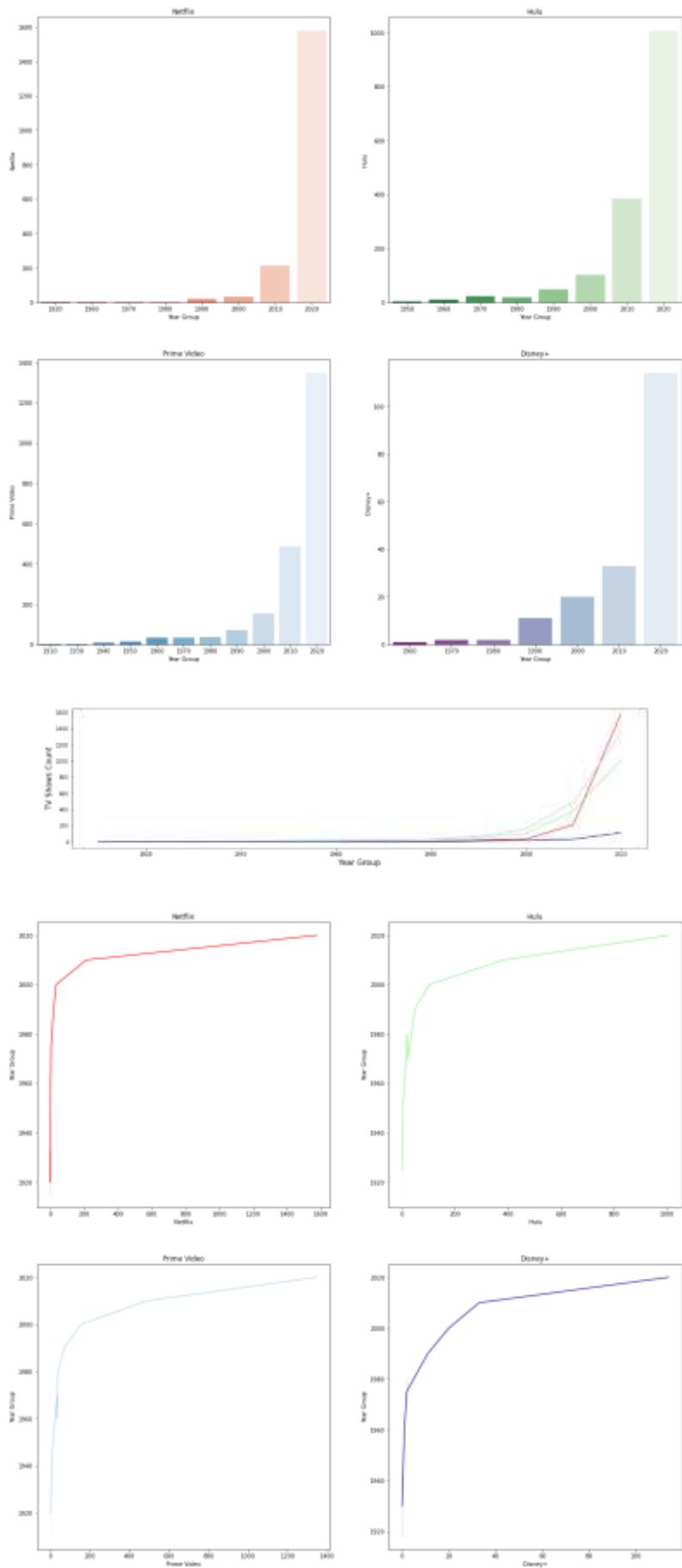


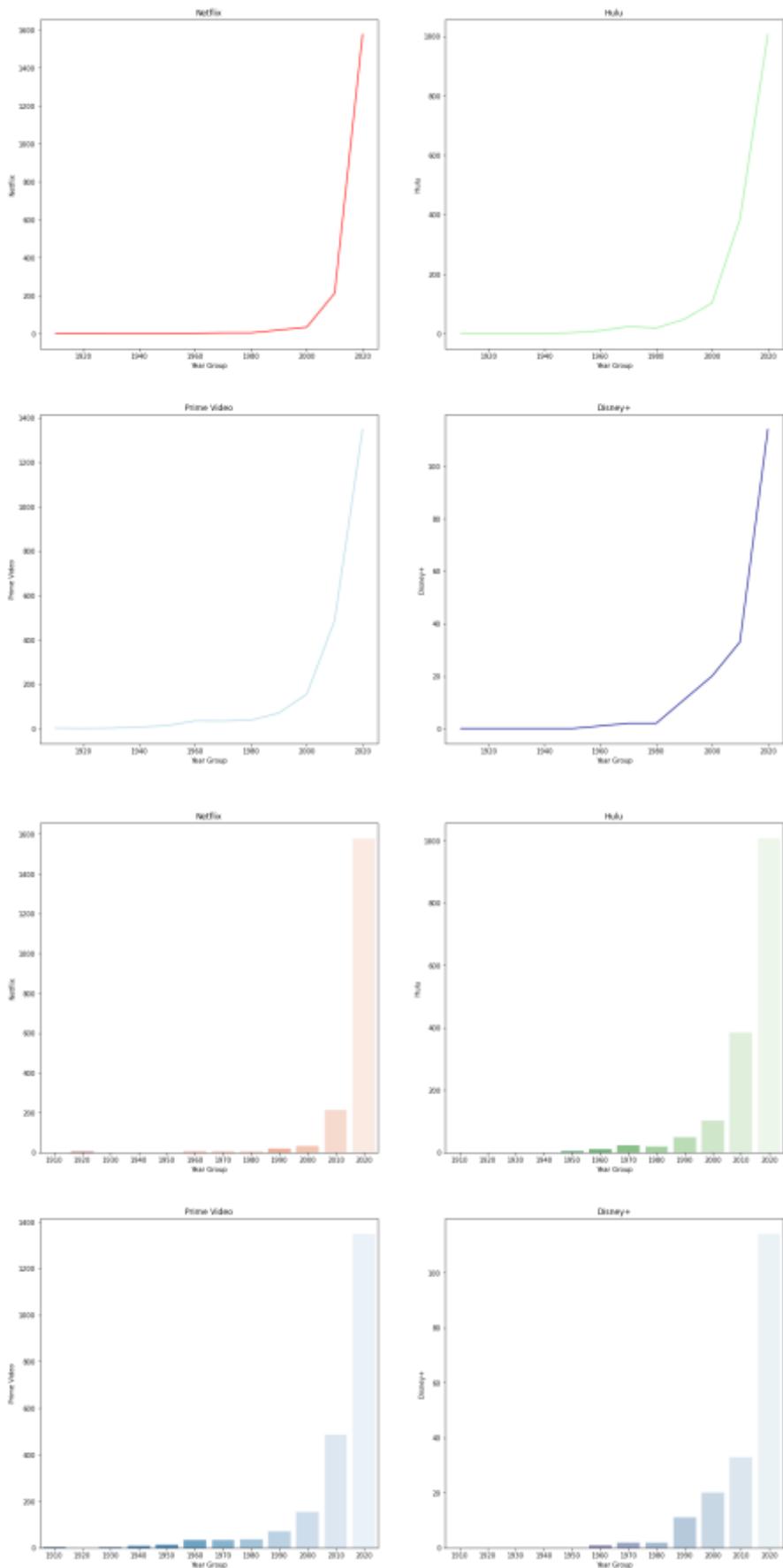












CHAPTER: EIGHT

EXPLORATORY DATA ANALYSIS

Introduction

Movies are considered a significant type of art, a worldwide source of entertainment and a powerful tool for people to be trained or indoctrinated.

OTT channels serve as one of the most amusing causes and a major stress reliever for individuals across the globe as far as the current pandemic situation is concerned.

In order to gain interesting perspectives, this project seeks to explore all the movies on prominent OTT platforms.

This is achieved with the help of a dataset from Netflix, Prime Video, Hulu and Disney+ API, obtained from Kaggle.

The dataset contains complete information about all films, their ratings and the corresponding OTT platforms on which they are available. Detailed information such as release year, genre, IMDb ranking, producer, and the language of each film is given.

```
Data Successfully Loaded!
Total Entries - 16744
Total Columns - 17
[1]:
```

0	0	1	Inception	2010	13+	8.8	87%	1	0	0	0	0	Christopher Nolan	Action,Adventure,Sci-Fi,Thriller		
1	1	2	The Matrix	1999	18+	8.7	87%	1	0	0	0	0	Lana Wachowski,Lilly Wachowski	Action,Sci-Fi		
2	2	3	Avengers: Infinity War	2018	13+	8.5	84%	1	0	0	0	0	Anthony Russo,Joe Russo	Action,Adventure,Sci-Fi		
3	3	4	Back to the Future	1985	7+	8.5	96%	1	0	0	0	0	Robert Zemeckis	Adventure,Comedy,Sci-Fi		
4	4	5	The Good, the Bad and the Ugly	1966	18+	8.8	97%	1	0	1	0	0	Sergio Leone	Western	It	

Data Sets

Movie.csv

MovieLens - Excel																		
File	Home	Insert	Draw	Page Layout	Formulas	Data	Review	View	Help	Tell me what you want to do	Power Query	Format	Filter	Sort & Find	Edit			
Clipboard	Cells	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font	Font		
A1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q		
2	0	1	Inception	2010	13+	8.8	87%	1	0	0	0	0	Stephen	Niemeyer	Sci-Fi	United States	Japanese	148
3	1	2	The Matrix	1999	18+	8.7	87%	1	0	0	0	0	Wachowski	Lilly	Action	Sci-Fi	Fiction	136
4	2	3	Avengers: Infinity War	2018	13+	8.5	84%	1	0	0	0	0	Anthony	Russo	Joey	Adventure	Thriller	149
5	3	4	K-12	1985	7+	8.5	96%	1	0	0	0	0	Robert	Zemeckis	Comedy	Romantic	Science	116
6	4	5	the Bad and the Beautiful	1966	18+	8.8	97%	1	0	1	0	0	Vittorio	De Sica	Western	Crime	War	101
7	5	6	Into the Woods	2015	7+	8.4	97%	1	0	0	0	0	Terence	Malick	Adventured	Family	Fantasy	117
8	6	7	The Pianist	2002	18+	8.5	95%	1	0	1	0	0	Roman	Pawlak	Drama	My	France, Poland, Germany	150
9	7	8	Igor Uryuchik	2012	18+	8.6	87%	1	0	0	0	0	Yuri	Taranovskiy	Comedy	Romantic	Science	165
10	8	9	Life of Pi	2012	7+	8.4	95%	1	0	0	0	0	Ang	Von Trier	Adventure	Thriller	Science	115
11	9	10	Curious Case	2009	18+	8.3	89%	1	0	0	0	0	Tim	Burton	Drama	Thriller	United Kingdom, France	153
12	10	11	Taxi Driver	1976	18+	8.3	95%	1	0	0	0	0	Martin	Scorsese	Dramedy	Thriller	Science	114

TV.csv

	A	B	C	D	E	F	G	H	I	J	K
	Title	Year	Age	IMDb	Rotten Tomatoes	Netflix	Hulu	Prime Video	Disney+	Other	Type
2	0	Breaking Bad	2008	18+	9.5	96%	1	0	0	0	1
3	1	Stranger Things	2016	16+	8.8	93%	1	0	0	0	1
4	2	Money Heist	2017	16+	8.4	91%	1	0	0	0	1
5	3	Sherlock	2010	16+	9.1	78%	1	0	0	0	1
6	4	Better Call Saul	2015	18+	8.7	97%	1	0	0	0	1
7	5	The Office	2005	16+	8.9	81%	1	0	0	0	1
8	6	Black Mirror	2011	18+	8.8	85%	1	0	0	0	1
9	7	Supernatural	2005	16+	8.4	99%	1	0	0	0	1
10	8	Peaky Blinders	2013	18+	8.8	92%	1	0	0	0	1
11	9	Mr. Robot	2015	17+	9.2	100%	1	0	0	0	1
12	10	The Walking Dead	2010	18+	8.1	81%	1	0	0	0	1

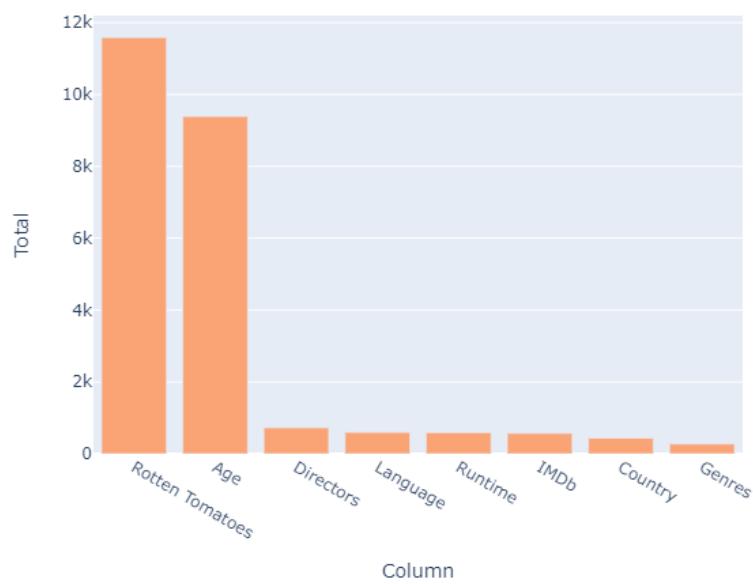
First Key Observations:

- The oldest film dates from 1902.
 - This dataset has a total of 16744 movies.
 - Both of the movies have an average runtime of around 93 minutes.
 - An entry with a runtime of 1256 minutes is available. This looks weird. We need this data to be checked.
 - For the Rotten Tomatoes scores and Age limit columns, more than 50 % of the data is incomplete.

Percentage of missing values

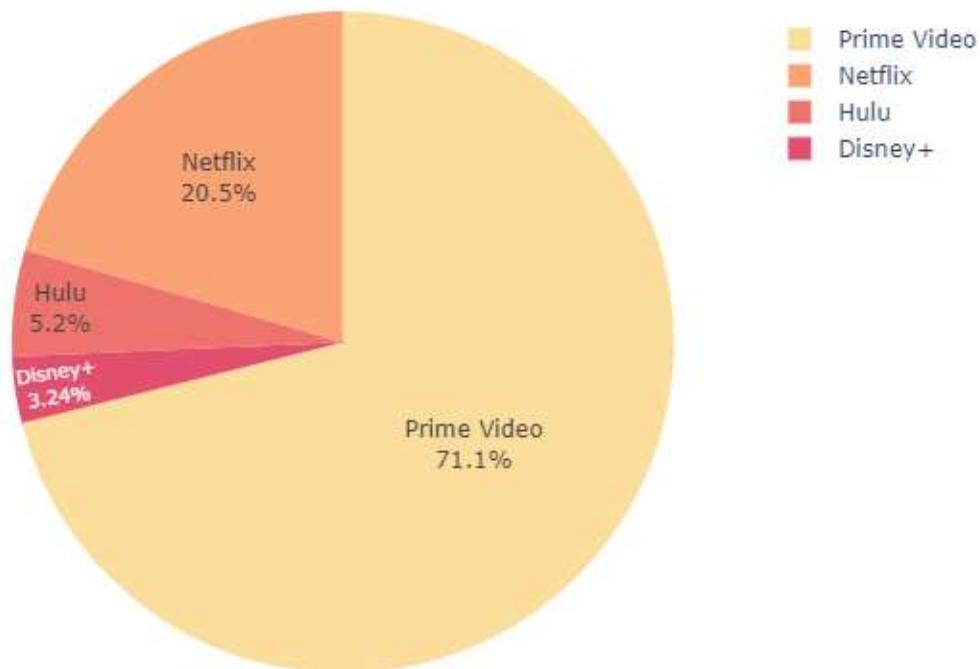
	column name	Percentage missing values
6	Rotten Tomatoes	69.19
4	Age	56.08
12	Directors	4.34
15	Language	3.58
16	Runtime	3.54
5	IMDb	3.41
14	Country	2.60
13	Genres	1.64
10	Disney+	0.00
11	Type	0.00
0	Unnamed: 0	0.00
9	Prime Video	0.00
1	ID	0.00
7	Netflix	0.00
3	Year	0.00
2	Title	0.00
8	Hulu	0.00

MISSING VALUES



PLATFORMS

There are 4 Platforms for this Data.



Prime Video has the majority of movies on its platform.

TITLE

Netflix streams 3560 titles, Hulu streams 903 titles, Prime video streams 12354 titles, Disney + streams 564 titles.

We've got up to 12k titles streaming on Prime Video, while Netflix has 3.5k titles. Compared to the two titles, Hulu and Disney+ are far smaller.

On all the 4 streaming platforms, are there any titles available?

Well there are no movie titles on all 4 streaming sites. Just one of the 4 streaming services accounts for almost 96 percent of the movie titles available in this dataset. On 2 platforms, fewer than 4% of the titles are available, while it is very rare to see a title available on 3 platforms. Let's quickly review the number of titles and other information for the 3 streaming platform titles.

In total, there are 10 movie titles on 3 streaming platforms.

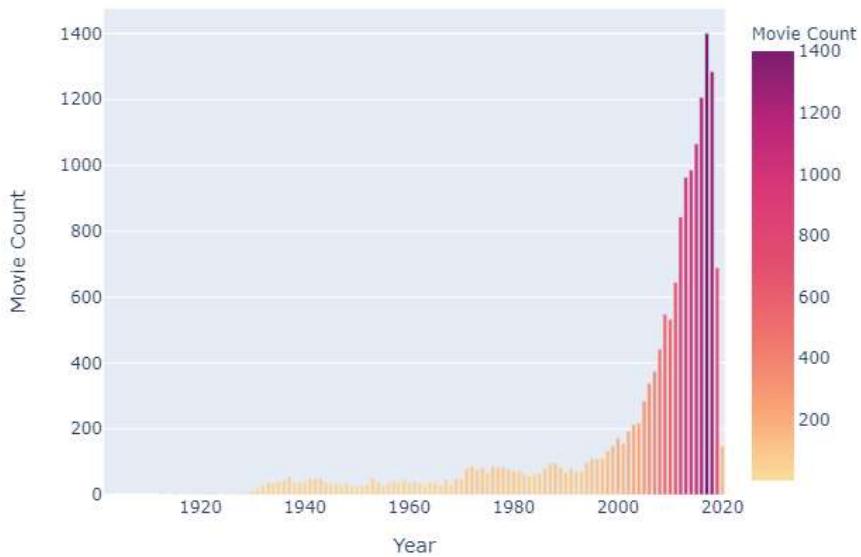
	ID	Title	Year	Age	IMDb	Rotten Tomatoes	Netflix	Hulu	Prime Video	Disney+
103	104	Amy	2015	18+	7.8	95%	1	0	1	1
148	149	The Square	2013	NaN	8.1	100%	1	1	1	0
340	341	The Interview	2014	18+	6.5	52%	1	1	1	0
497	498	Blame!	2017	13+	6.7	82%	1	1	1	0
610	611	Evolution	2001	13+	6.1	43%	1	1	1	0
1133	1134	No Game No Life: Zero	2017	13+	7.5	NaN	1	1	1	0
1776	1777	Zapped	2014	all	5.1	NaN	1	1	0	1
2017	2018	Mother	2016	NaN	5.6	NaN	1	1	1	0
3960	3961	The Kid	2019	18+	5.9	45%	0	1	1	1
4313	4314	Inside Out	2011	13+	4.5	25%	0	1	1	1

Among the four channels, Netflix, Hulu and Prime videos seem to have the most popular names. With each varying only with 1 title, there is a great overlap between Hulu and Prime video here. And two titles, Zapped and Amy, seem to be available on Netflix for streaming.

Compared to Disney+, there are more Hulu titles available in both Netflix and Prime videos.

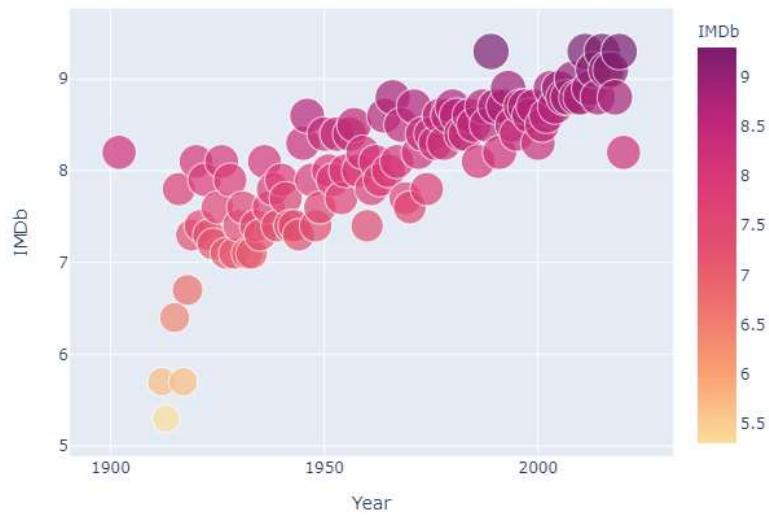
YEAR

Movie Count by Year



Movies have been rising year by year and the biggest movies were released in 2017, with 763 on Prime Video and 569 on Netflix released.

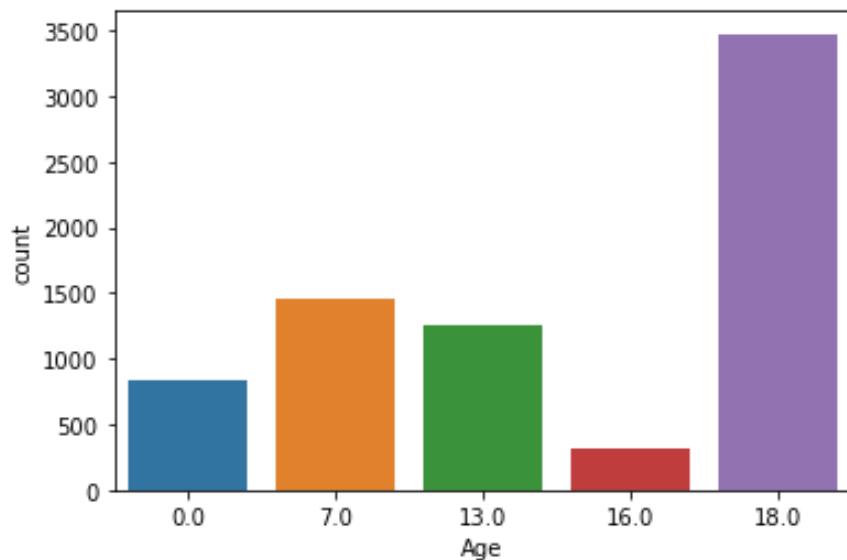
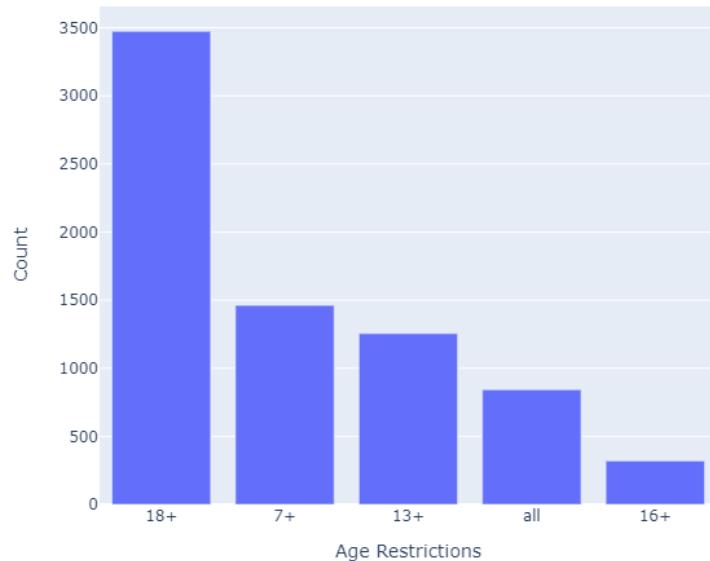
Best Movie Each Year According to IMDB Rating



In terms of IMDB ranking, the best movies in the last few years have been the best in all these years.

AGE

Age Restrictions



In the age group in the above dataset, 0 indicates no restriction.

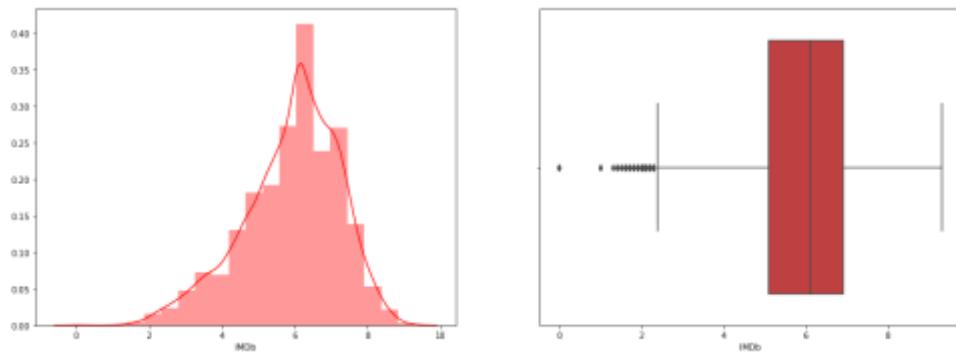
It is evident that most of the OTT platform shows are directed at age groups greater than 18 years of age.

This is accompanied by age categories of 7 and 13 that might include kid-genre movies.

IMDb

IMDb Ratings

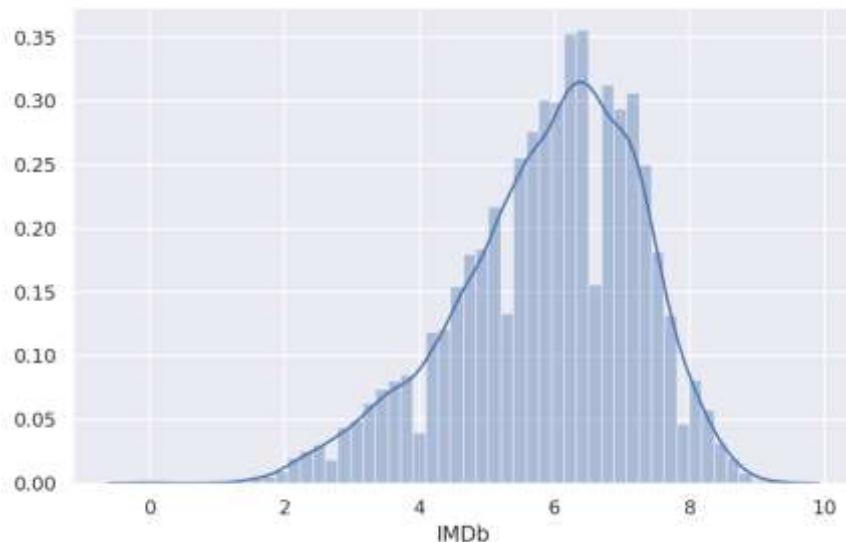
A high IMDb rating is an insignificant contribution to the success of a movie. The rating indicates how good the movie is and decides whether or not it can be viewed.



The average ranking is 5-7. Also, some outliers are present.

If the ranking lies above 8, the movie is considered to be exceptional. If the rating lies between 6-8, the movie is considered to be fine.

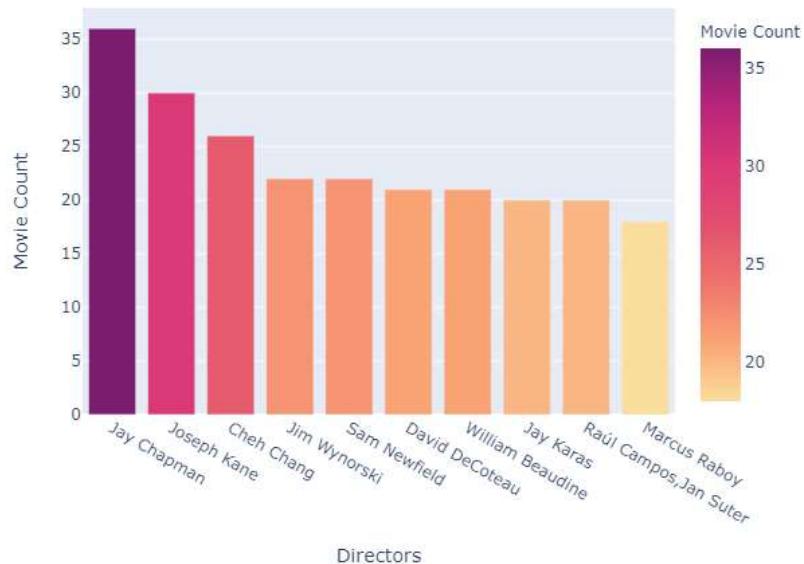
IMDb Ratings Distribution



With an average of around 6.5, IMDb scores of movies approximately match standard distribution.

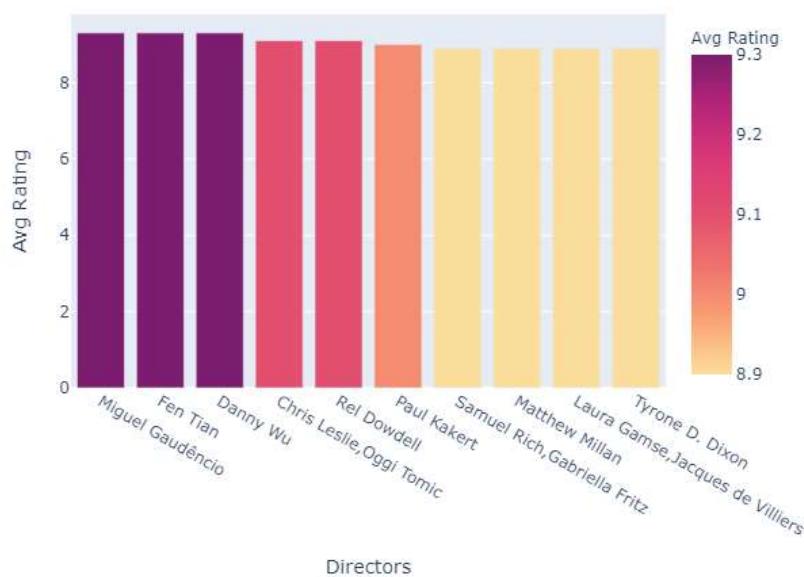
DIRECTORS

Top 10 Directors Movie Count



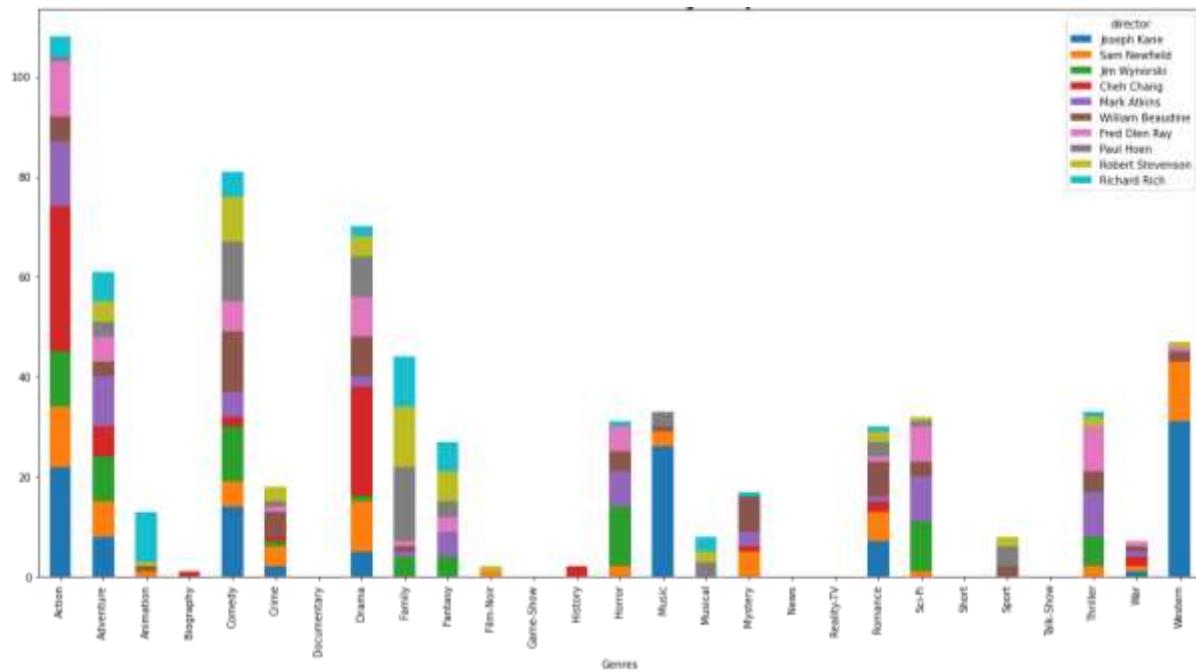
The largest number of films have been directed by Jay Chapman, and most of them are on Prime Video.

Top 10 Directors with Movies Having Highest IMDB Rating



Just one film count is applicable to the directors who made the top rated IMDB films!

Genres Directed by Top 10 Directors



We can conclude on the basis of the above visualization that most of Joseph Kane's action, comedy, music and western genre movies are directed.

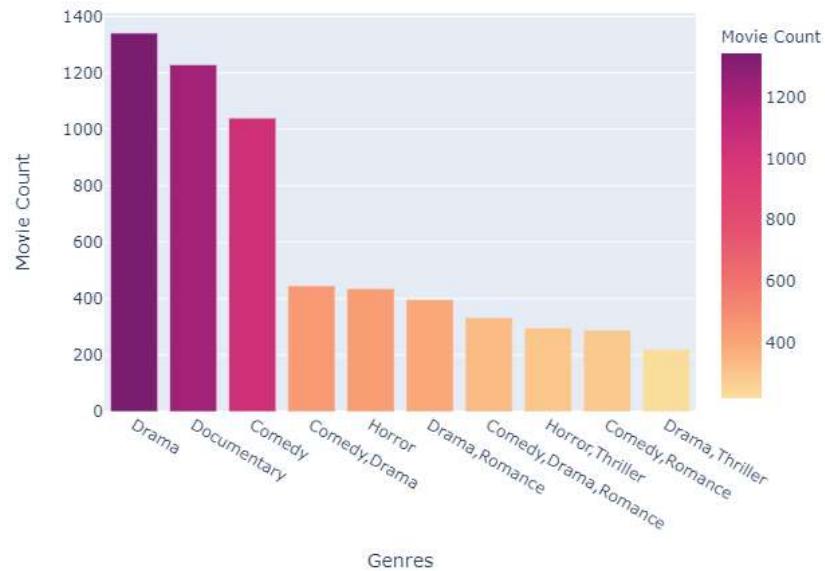
Also, Cheh Chang is directing a wide variety of action and drama genre movies.

If you want to go with suspense movies, then you'll have a wide variety of Fred Olen Ray and Mark Atkins directed movies.

William Beaudine has directed films in nearly all the genres listed above.

GENRES

Top 10 Genres Movie Count



The most popular genres of all are drama, documentary and comedy.

Top Movies of Different Genres based on IMDB Rating



Some movies with 9+ ratings: drama, documentary and comedy.

USING KMEANS CLUSTERING TO PLOT DIFFERENT GENRES

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.

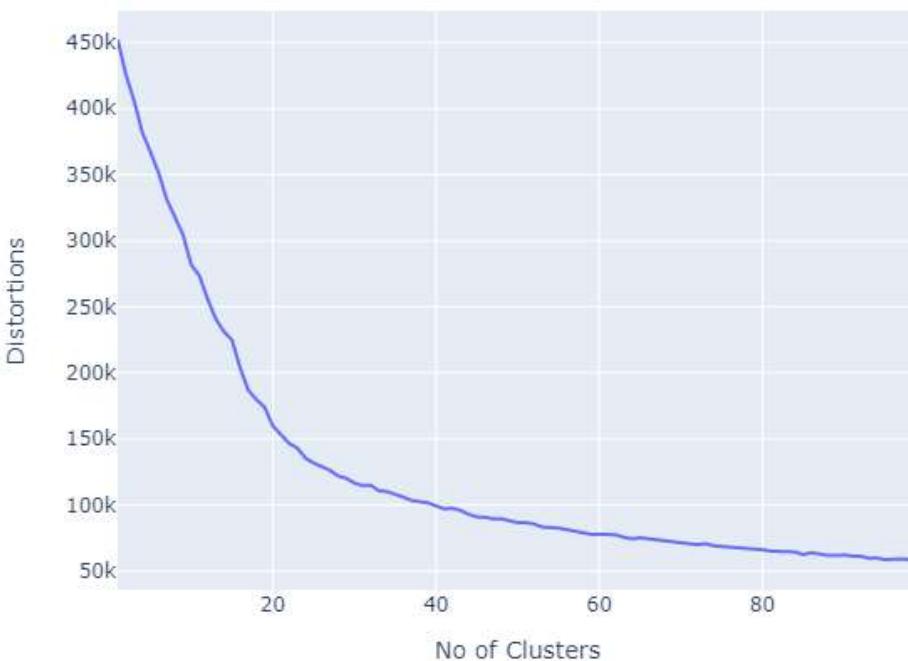
Owing to such similarities, a cluster refers to a set of data points aggregated together. A target number k , which refers to the number of centroids you need in the dataset, will be specified. The imaginary or real location representing the cluster's center is a centroid. By reducing the in-cluster sum of squares, every data point is allocated to each of the clusters. In other words, the K-means algorithm determines the number of centroids in k , and then assigns each data point to the nearest cluster, thus keeping the centroids as small as possible.

StandardScaler - By subtracting the mean, and then scaling to unit variance, it standardizes a function. Unit variance implies that the standard deviation separates all the values.

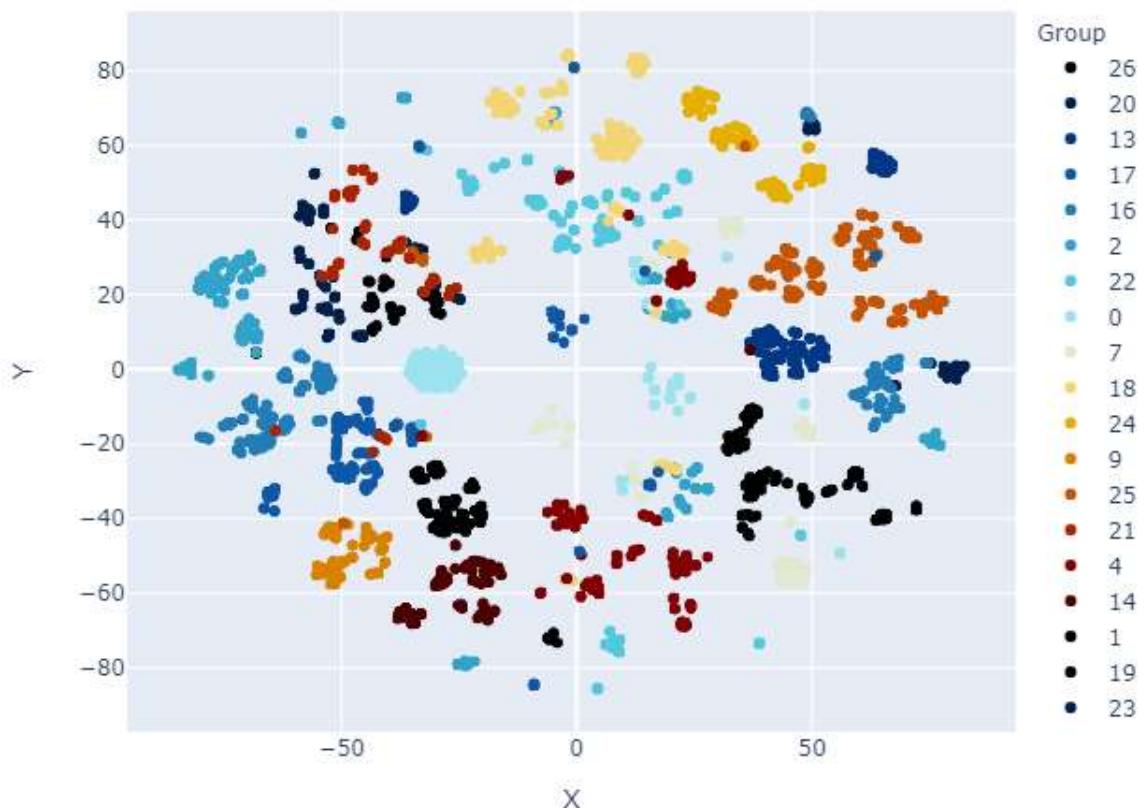
TSNE - t-Distributed Stochastic Neighbour Embedding (t-SNE) is a dimensionality reduction technique that is particularly well suited for high-dimensional dataset visualization.

Elbow Method

By fitting the model with a set of values of ' k ', the elbow approach helps to select the optimal value of k (number of clusters). We will use a 2-dimensional data set here, but for any multivariate data set, the elbow method holds.



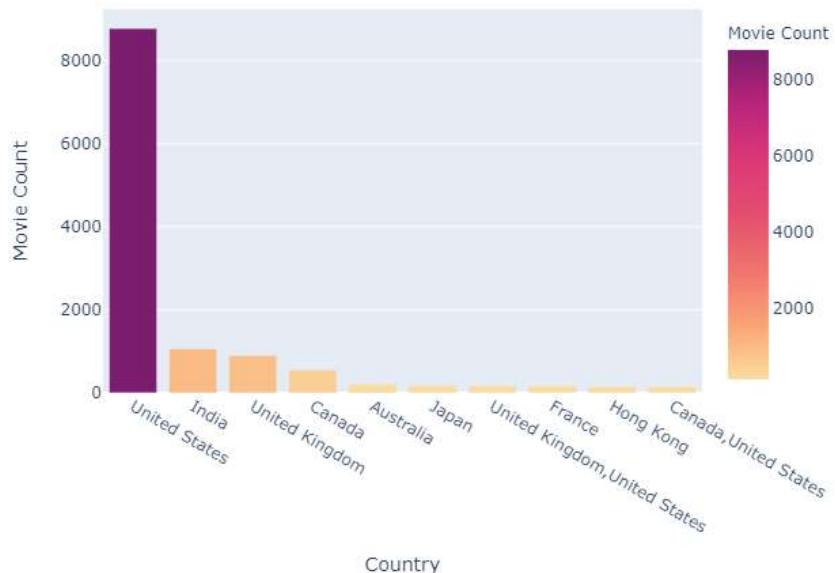
We will observe that the number 27 is the "elbow" that is ideal for this situation. This can also be confirmed since there are 27 different genres, so this outcome was almost predicted.



Terrific! Fantastic! By checking the genres of film with its nearest neighbour, we can also verify that clustering is pretty accurate.

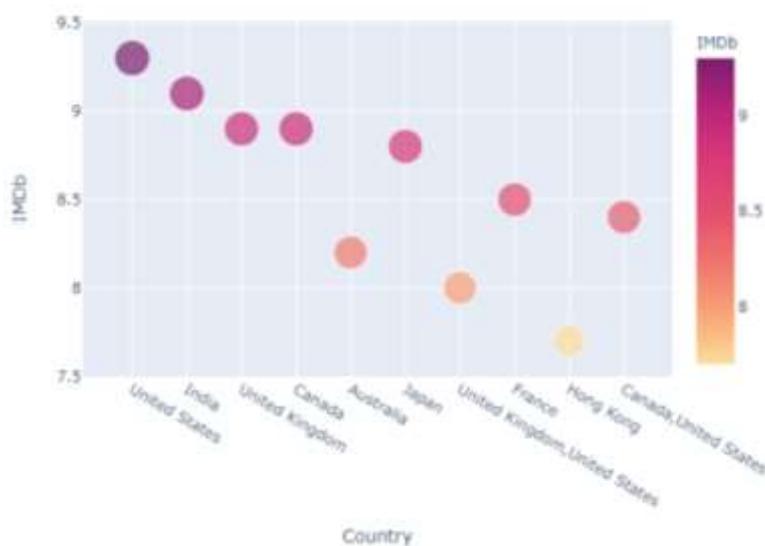
COUNTRY

Movies Count by Country



With 6817 films on Prime Video and 1305 films on Netflix, most of the films are from the United States.

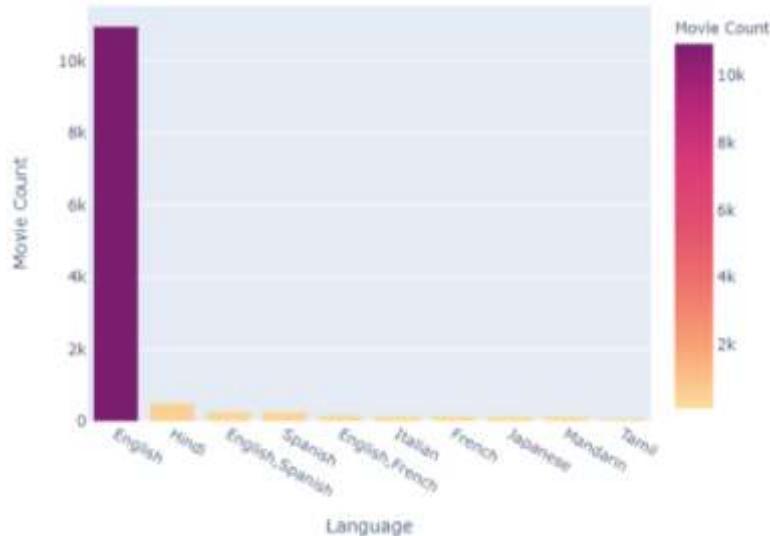
Best Movie in Top 10 Countries According to IMDB Rating



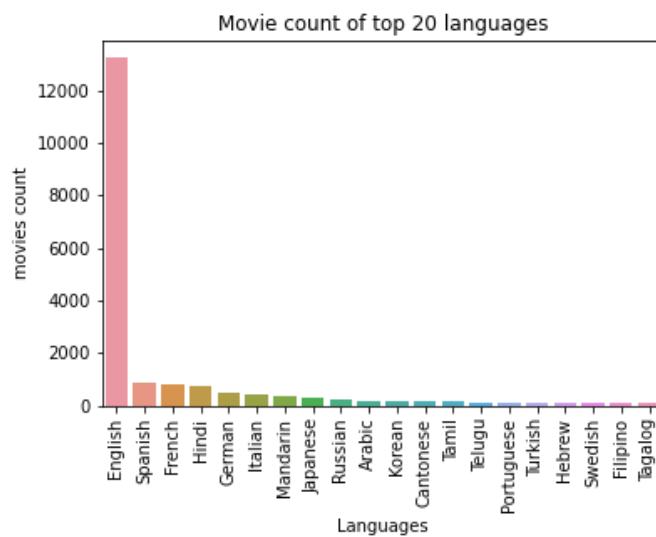
Since most of the movies are from the United States, it was expected that, among others, the best movies from the United States will also be the best.

LANGUAGE

Movie Count by Language

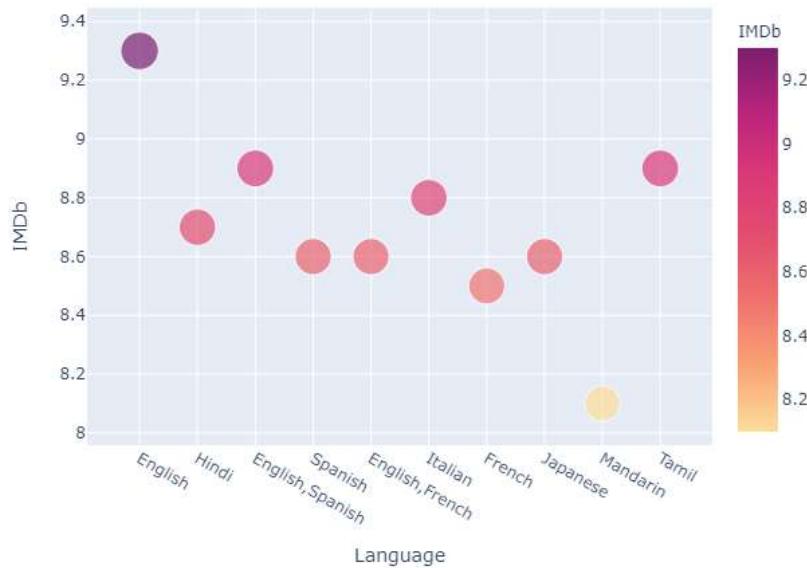


Nearly two thirds of the movies in this dataset are in English. Out of all languages, English is the most popular. On Prime Video, there are 8617 English films and 1624 on Netflix. Most of the films are clearly visible from English, which indicates that all the channels are trying to get viewership across the globe.



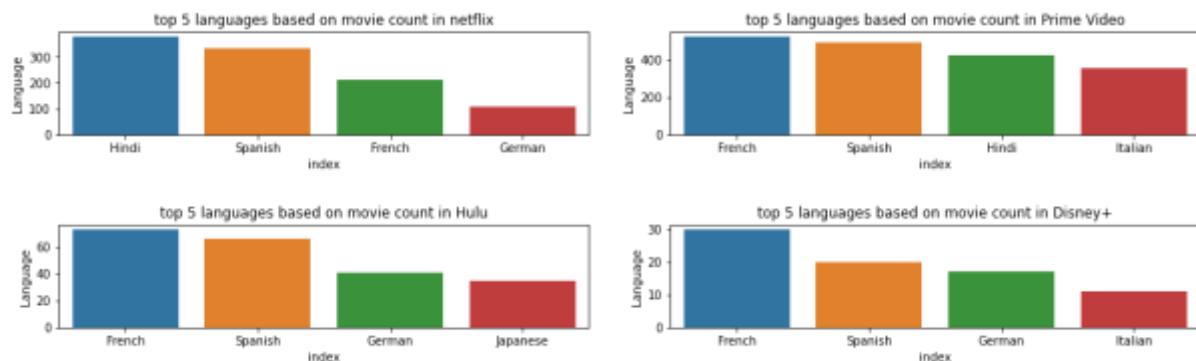
Indian regional languages such as Hindi, Tamil, Telegu, and Punjabi are among the top 20, which shows that India has a strong presence and audience for all OTT platforms. In the top 20, a broad variety of European languages can be seen, which again illustrates the popularity of European nations.

Best Movie in Top 10 Languages According to IMDB Rating



Since most of the films were in English, it was anticipated that, among others, the best English movie would also be the best.

Top 5 languages based on movie count across the OTT Platforms

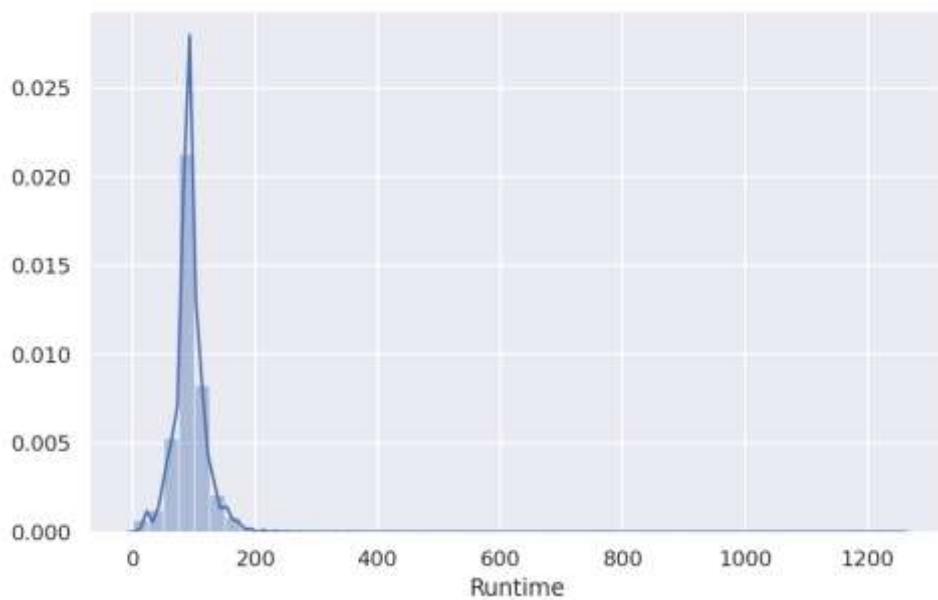


The above display shows the top 5 languages other than English with the largest number of films on various OTT platforms.

As shown earlier, all OTT platforms have a larger number of films in the English language. If you want to watch Hindi movies, you can watch more than 400 movies on Netflix and Prime Video.

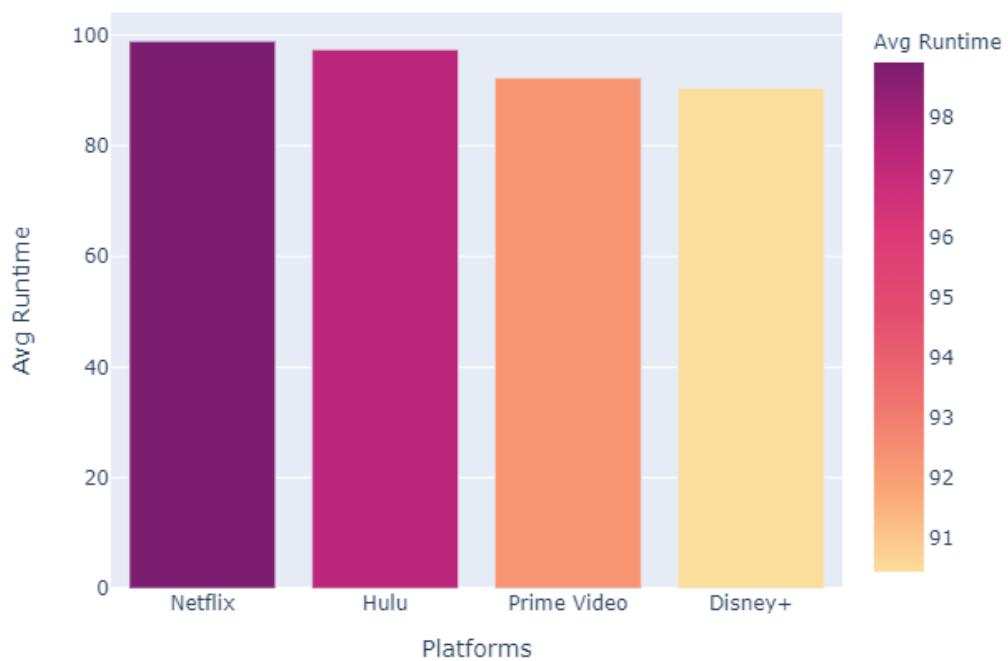
Spanish language movies are distributed similarly through all OTT platforms and have more appeal. German movies, with over 100 movies, can be watched on Netflix.

RUNTIME



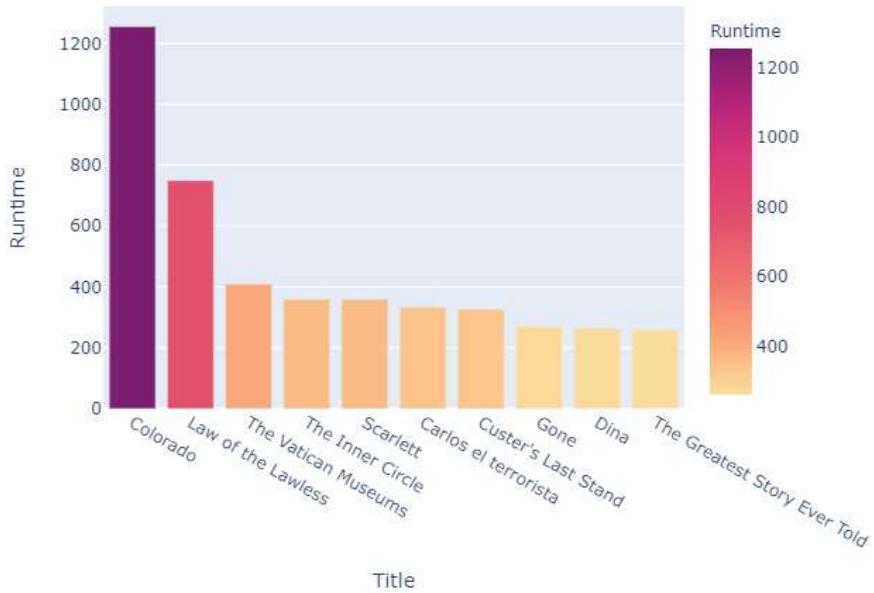
The majority of movies have a runtime of around 90-100 minutes. Since there are some films with very high runtime, runtime is positively distorted.

Average Runtime on different Platforms



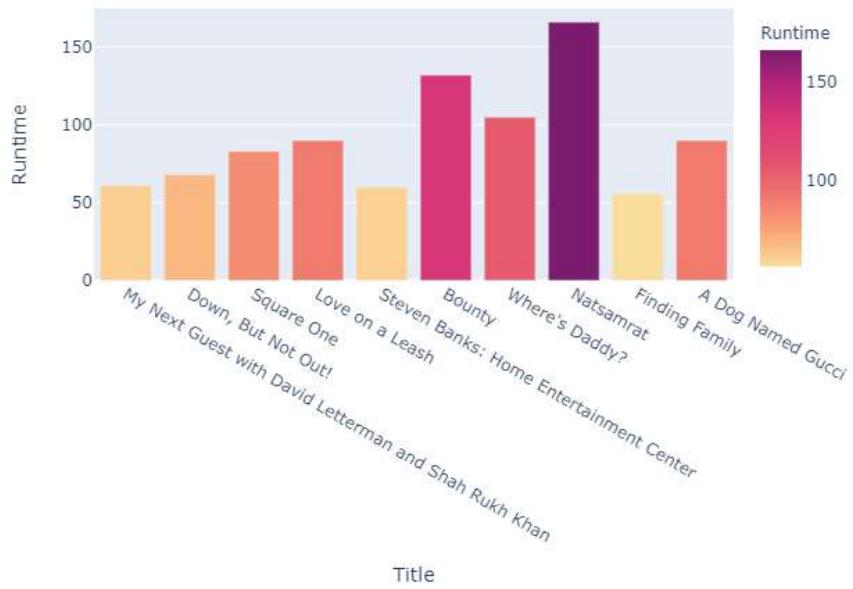
The overall movie runtime is close on all platforms.

Top 10 Longest Movies



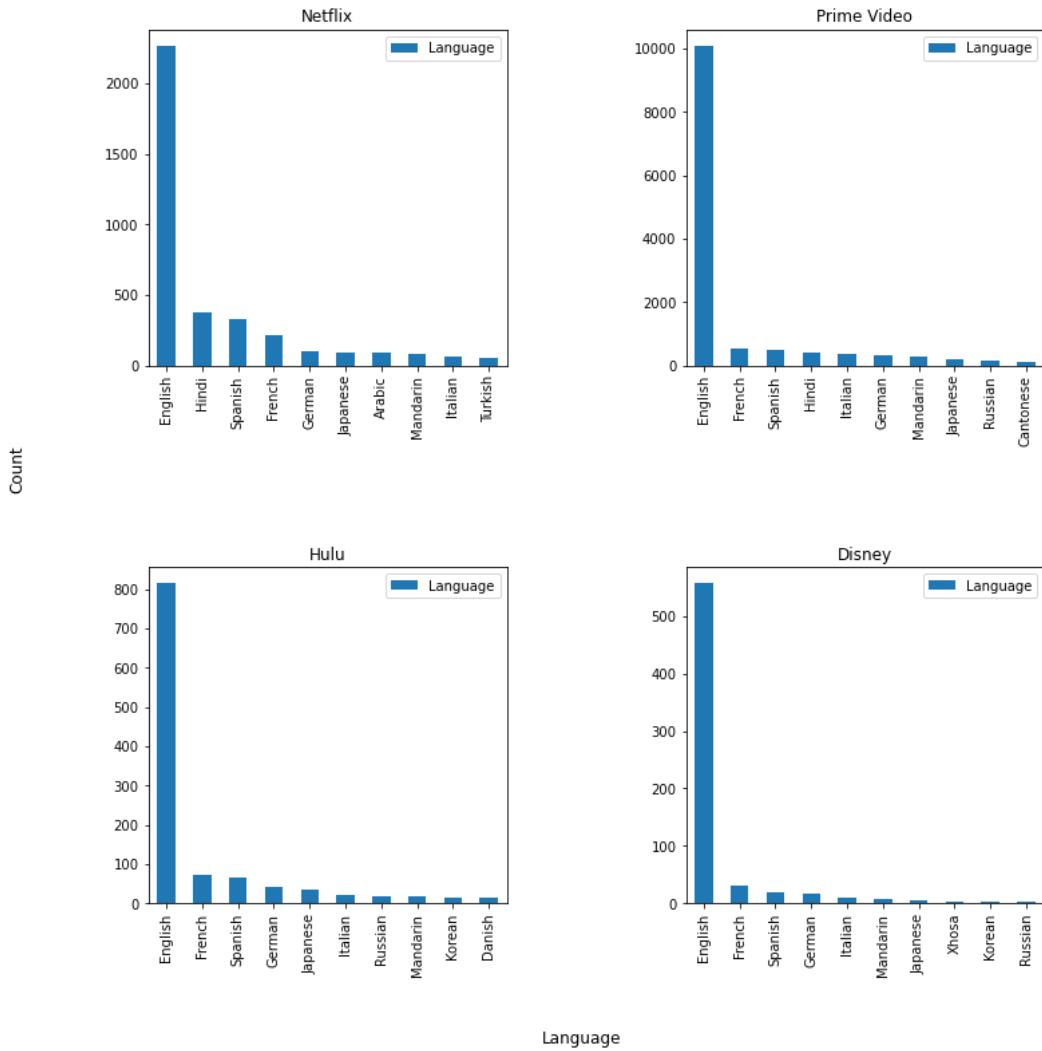
Colorado has a whopping 1256 minutes of overall runtime and is only available on Prime Video.

Top 10 Highest IMDB Rated Movies



Natsamrat has the highest runtime of 166 minutes of all the highest rated movies and is available only on Netflix.

Language Vs Platforms



Both platforms have a higher number of 'English' language films.

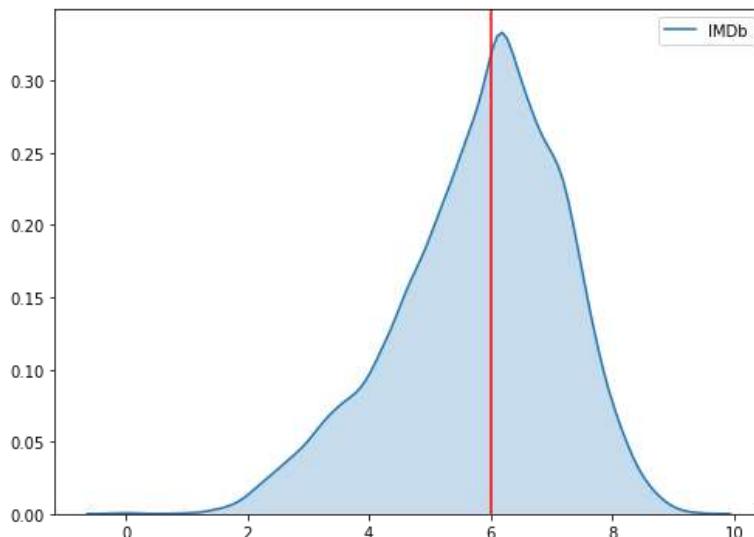
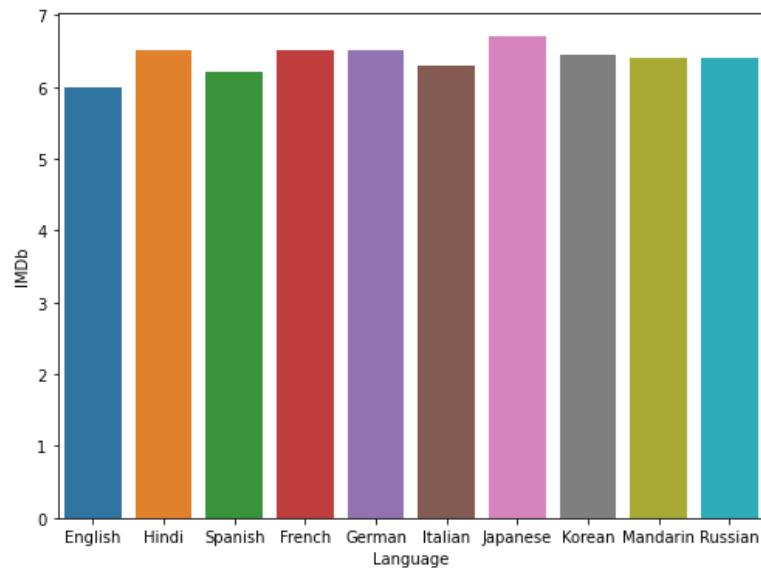
Netflix: Hindi and Spanish films are accompanied by 'English'.

Prime Video: French and Spanish films are in line, followed by 'English'. Behind these three are 'Hindi' films.

Hulu: French and Spanish films are accompanied by 'English'. Standing behind these three are 'German' films.

Disney+: French and Spanish films are accompanied by 'English'. After all this, 'German' comes in.

Language Vs Ratings

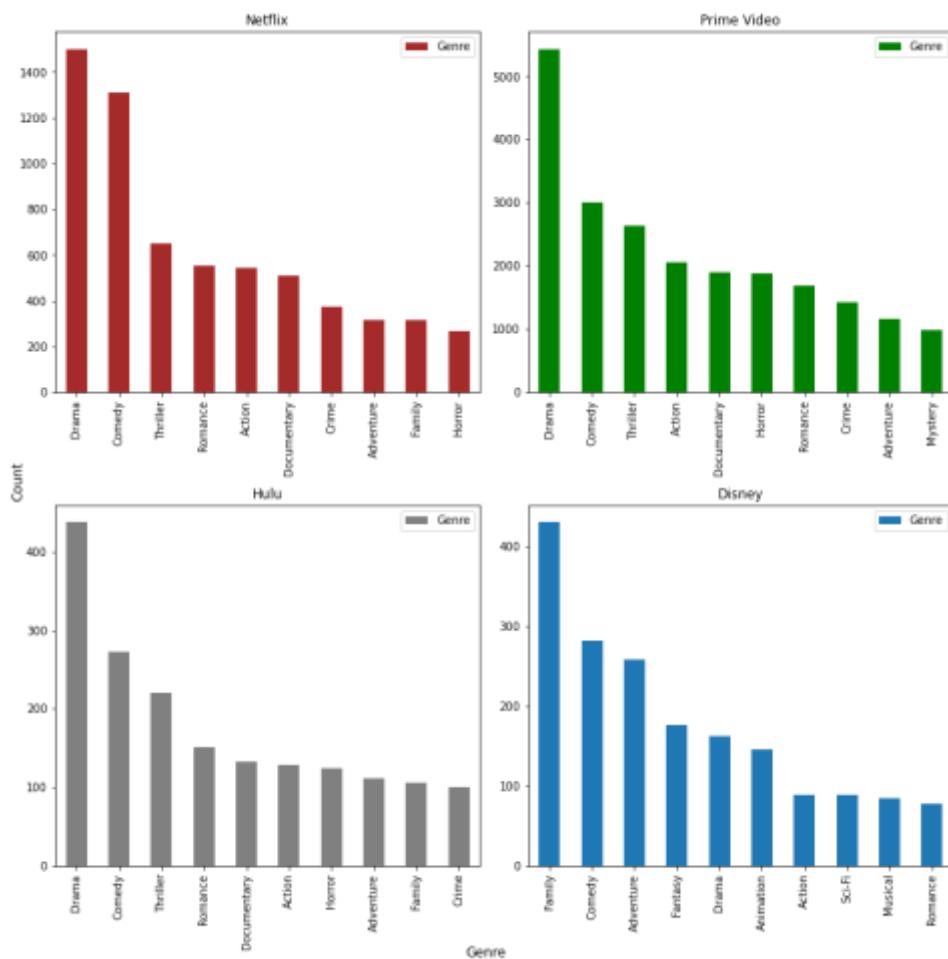


Based on the distribution of movies, the top 10 languages have been chosen and average ratings have been determined.

Since the maximum films are from the 'English' language, the median rating of 'English' films is determined.

Many 'English' movies are said to have a median rating of '6.0'.

Genre Vs Platforms



Netflix: The 'Drama' genre has 1500+ movies. Comedy, which is about 1300 +, follows it. Then 'Thriller', 'Romance', 'Action' and 'Documentary' were distributed similarly.

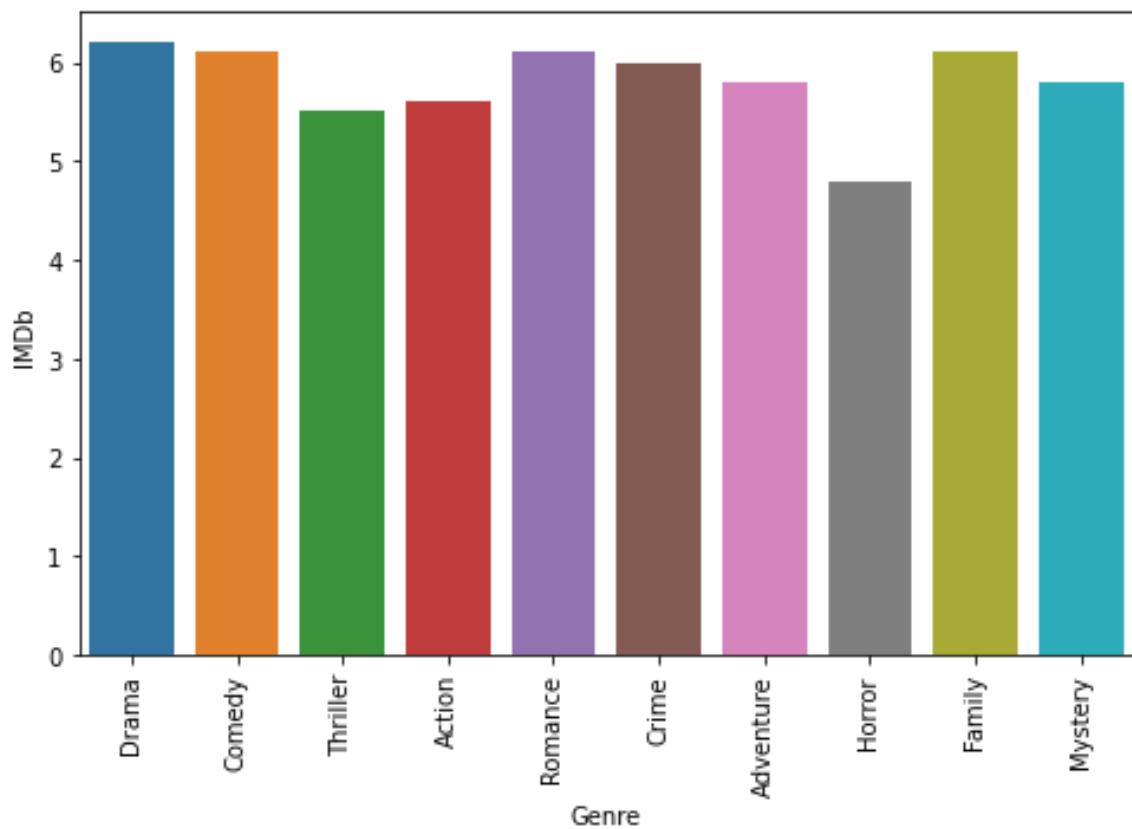
Prime Video: The 'Drama' genre has 5000+ movies. It is followed by comedy and thriller, around 2500+ films. Next comes 'Action'.

Hulu: 'Drama' comes first here too which has around 450+ movies. 'Comedy' follows 'Drama' with 'Thriller'.

Disney: 'Family' has the leading role with 450+ movies, unlike 3 other channels. Then comedy and adventure, followed by fantasy, come into play.

From this, we conclude that 'Disney+' is more appropriate for both children and families. For people who like 'Drama and Comedy', Netflix and Prime Video are more appropriate.

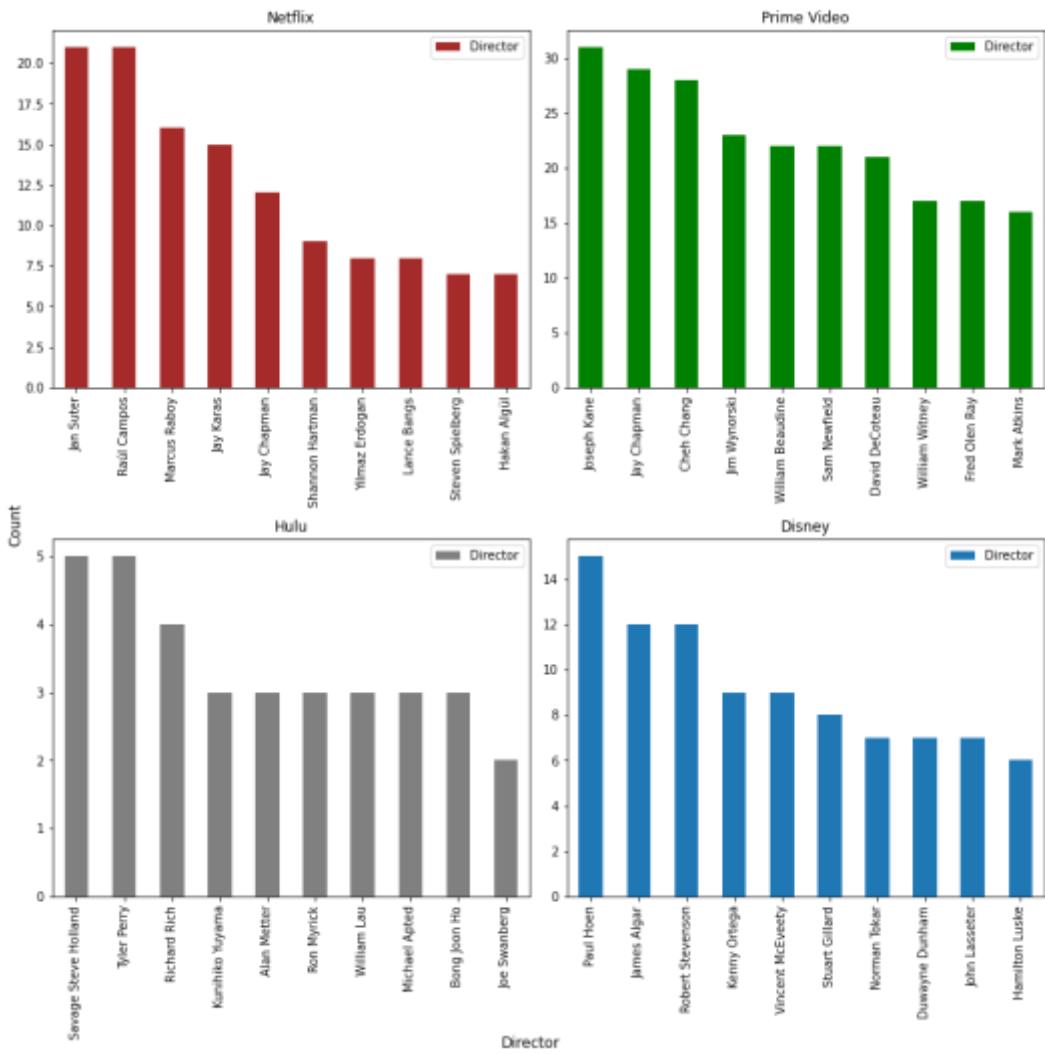
Genre Vs Ratings



The Top 10 genres have been selected based on the film count.

With the help of this we can tell that the median rating is about 5.8 - 6.2 for all genres. The maximum median rating for 'Family' and 'Drama' is 6.2.

Director Vs Platforms



The partnership between the Director and the Platforms is illustrated.

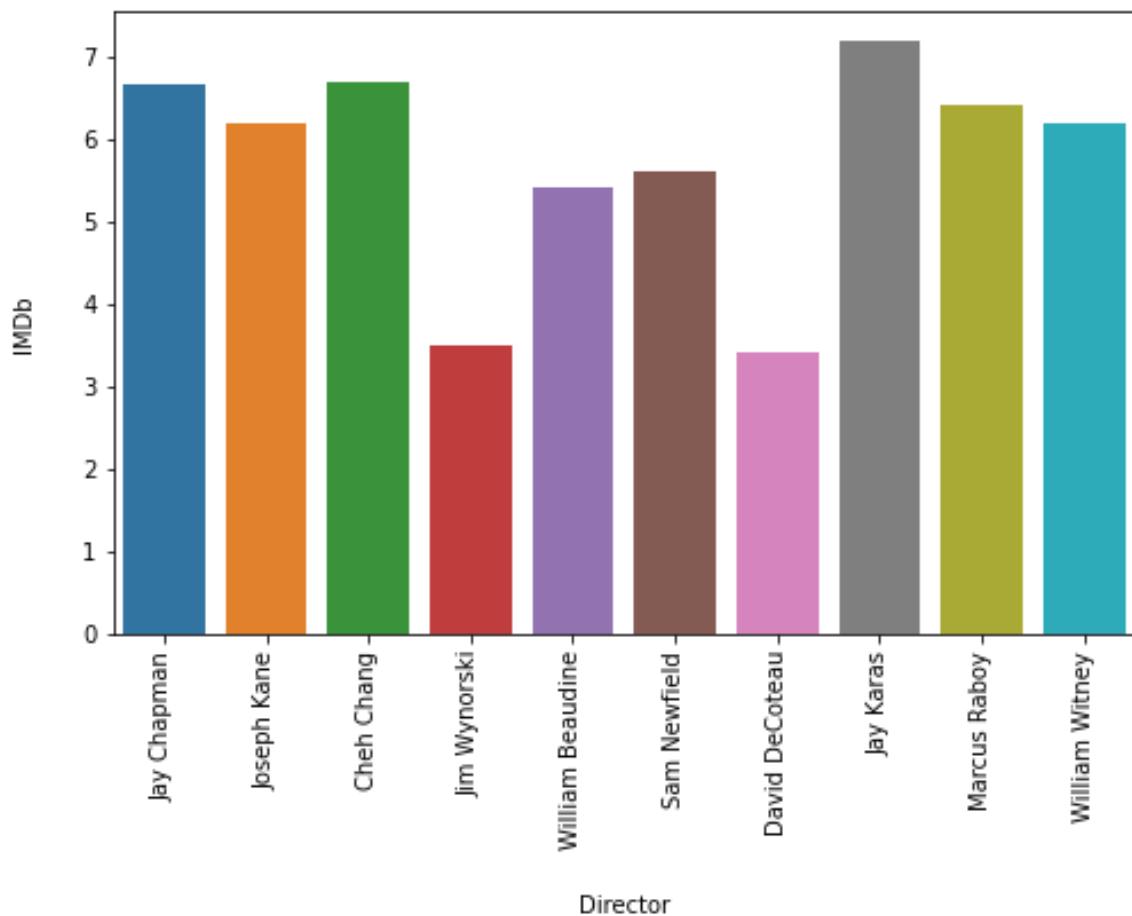
Netflix: 'Jan Suter' and 'Raul Campos' lead the Netflix network with their films, each with 20+ films.

Prime Video: With more than 30+ movies, 'Joseph Kane' leads the prime video network. Then 'Jay Chapman' followed.

Hulu: 'Tyler Perry' and 'Steve Holland' have more than 5 Hulu movies.

Disney+: 'Paul Hoen' leads with 15+ movies on the Disney website. 'James Algar' and 11+ Movies accompany him.

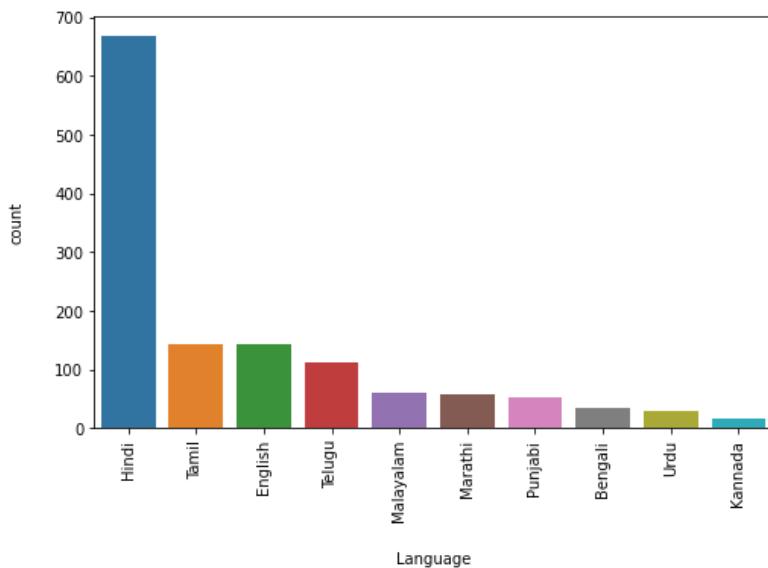
Director Vs Ratings:



Based on the total number of movies, the top 10 directors were selected and their median ratings were shown. Based on the ratings followed by 'jay Chapman',' Jay Karas' takes the first slot.

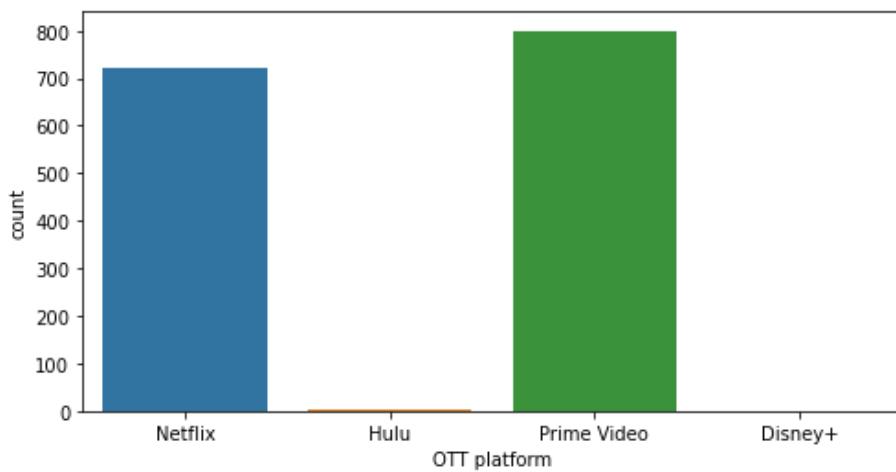
Movies in India

Indian Language Movies



When it comes to Indian movies, 'Hindi' has a film count of 650+ in any other language. Then 'Tamil' and 'English' come with 150+ movies each.

Indian Movies vs OTT platforms



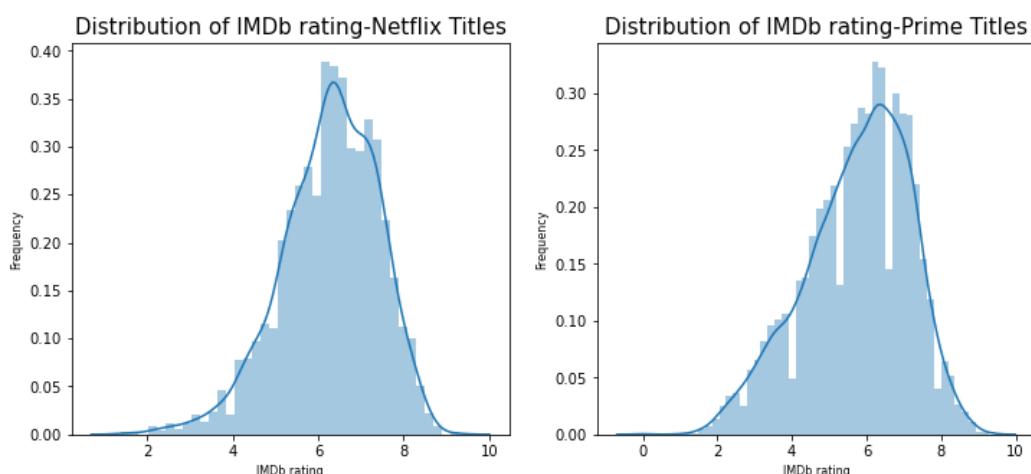
When it comes to Indian movies, Prime Video and Netflix are almost equal. Prime video has 800+ and there is 700+ movies on Netflix. These two outlets, in particular 'Hindi' Language, are preferred for Indian movies.

Netflix Vs Prime Movies

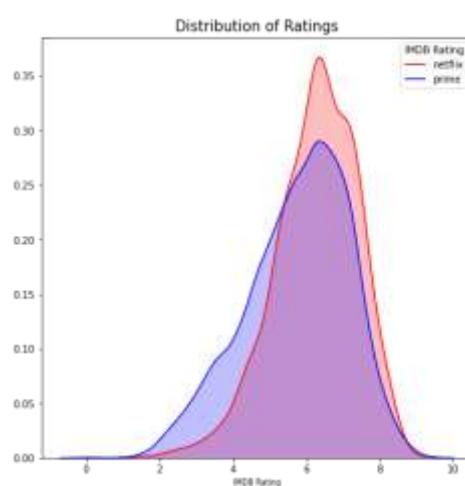
From the previous review, we find that Netflix has 3560 titles available and Prime has 12354 titles available. Let's build a contrast and infer our findings between these two streaming services.

It is shown that both streaming services have 20 percent of their titles in 18+ ratings by examining the age limits for the titles.

Compared to Prime, the proportion of 13+ rated movies on Netflix is higher. While there is a 4% difference in the 13+ titles, there is a 2% difference in the 7+ ratings.

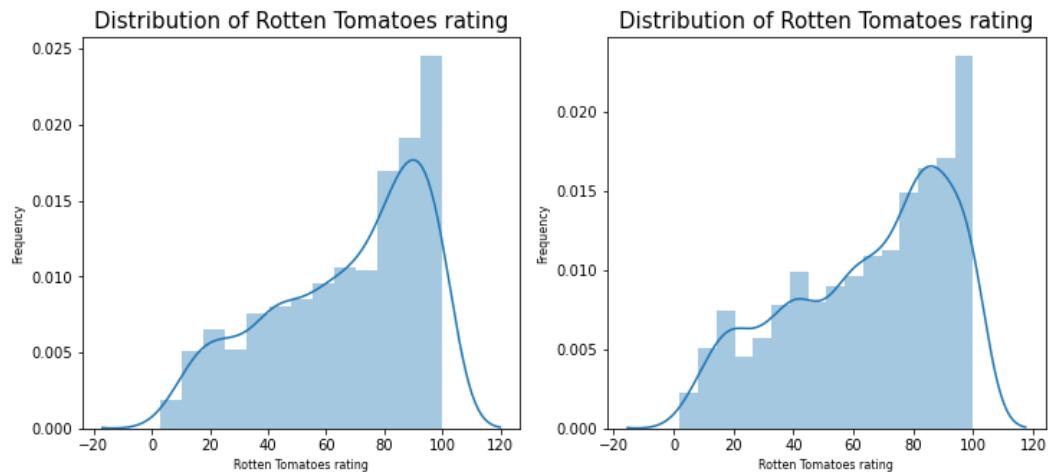


There is a difference in the IMDB ratings between Netflix and Prime.

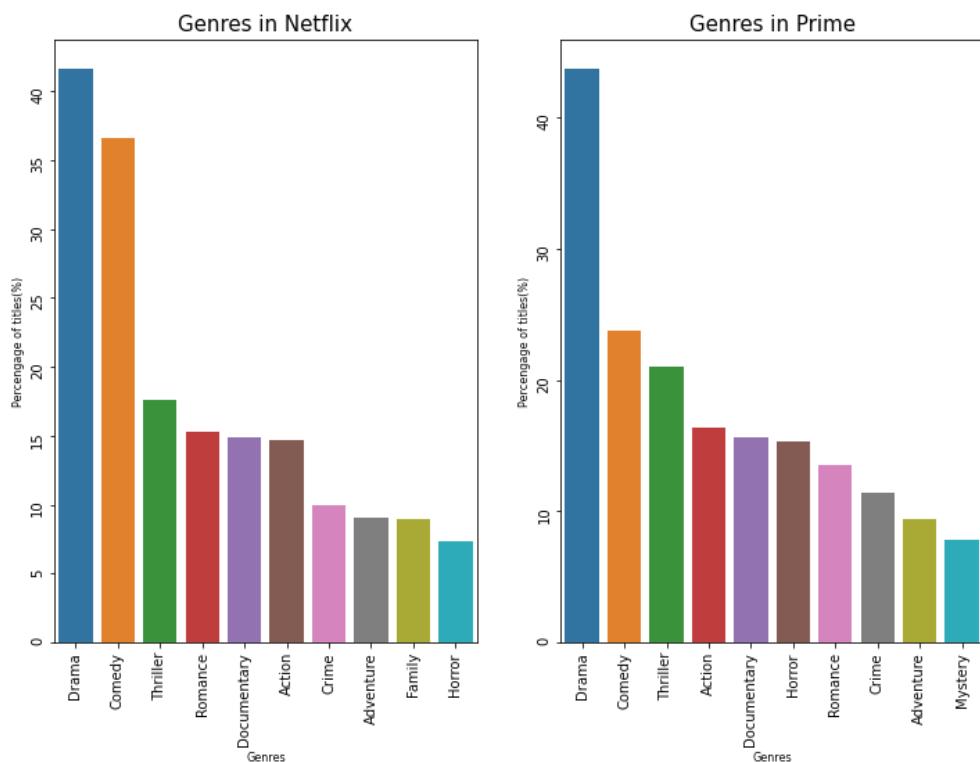


With a peak between 6-7 for Netflix and 5-6 for Prime, the distribution is similar to the bell-shaped curve (not a perfect one).

Few titles are ranked higher than 9. There are more titles with lower IMDB ratings (between 2-5 ranges) available in Prime compared to Netflix, though Netflix tends to be dominant on the higher rating scale.



With titles available in the range of 80-100 for both Netflix and Prime, ratings for rotten tomatoes were on the higher side. When it comes to IMDB ratings, there is a disparity between the two streaming services, but identical ratings are shown for the titles available on Prime and Netflix.



On Netflix and Prime, the top 3 genres available are the same.

While the percentage gap for the first three genres is relatively high, it is less so for the subsequent genres. They seem to be available in equal quantities.

Although the percentage of drama titles between Netflix and Prime is very similar, Netflix has more titles in the comedy genre compared to Prime, while Prime has more titles in the thriller genre compared to Netflix.

Although 16% of film titles belong to Netflix's Romance genre, Prime has an equivalent percentage of titles belonging to the Action genre.

The mystery genre of films seems to hold 10th spot, though Netflix's genre is not in the top 10.

So, both Netflix and Prime have nearly equal coverage of drama, comedy and suspense when it comes to genres.

CHAPTER: NINE

Conclusion

Findings

(There were a lot of missing points in the data set. An analysis has been done excluding those point)

1. 96 % of the titles listed in this dataset are available to view in only one platform. Therefore, each platform has got his own unique catalogue of movies and genres to pull its customers.
2. Netflix has got more movies with higher IMDB rating of 6-7 when compared to Prime. This can be noticed from the histogram plot of genres between Netflix and prime.
3. When it comes to ratings on rotten tomato, both the platforms have got similar ratings for the titles.
4. When it comes to genres, Drama, Comedy and Thriller are the most available genres in both Netflix and Prime.
5. As per the available data, the "Prime video" has the highest number movies followed by "Netflix". "Disney+" has the lowest number of movies casted.
6. Most of the movies produced were "Drama" followed by "Comedy". Almost negligible number of movies produced were of "Game-Show" genre.
7. The number of movies released under the genres "Reality-TV", "Talk-Show", "Film-Noire" and "News" were a lot less as compared to other genres.
8. The number of movies produced throughout under "Musical" and "Short" genres were almost similar.
9. As there were a large number of missing data, after removing those values it is seen that most of the movies produced had "Rotten Tomatoes" rating >81% followed by 61% - 80%. The least number of movies produced had <20%.
10. Most of the movies released were of age group 18+. The least number of movies produced were of age group 16+.

11. Majority of the movies were released between 2000 and 2020. The least number of movies released were between 1900 and 1900.
12. According to the data, it is seen that the frequency of movie production increased along with an increase in Year. They both exhibit a positive correlation.
13. Majority of the movies has an IMDb rating ranging between 5-6 followed by 7-8.
14. The least number of movies got an IMDb rating between 9-10.
15. On the basis of given data, we can say that majority of the movies in these 4 OTT platforms are average or more than average IMDb rated.

Research limitations

The sample reviewed by the author was restricted to Movies.

Data collection was restricted to third Party Availability.

The major limitation of this study is due to time constraint and also a limited data is available for current Year.

Conclusion

This Study Concludes by Saying that, OTT video channels are significantly becoming part of the entertainment time of audiences, according to this article, and they are giving conventional modes tough competition.

The flexibility of time and place, the availability of reliable and affordable data connections, the penetration of smart phones, the availability of cheap and even free access to OTT video platforms, the sheer variety of content to choose from and the quality of content are some of the key factors motivating viewers to migrate to OTT video platforms.

Traditional TV channels, however, will not be totally replaced by OTT video networks, at least in the near future, and they will co-exist.

Traditional TV channels still have a chink of loyal viewers, with some improvements in the quality of programming and tactics that can still draw customers and thrive in the competitive period.

Although we could not prove the hypothesis that OTT will overtake cinema, we can see that there is suddenly an increase in the use of OTT over other media and people have positive responses to movies that will be released at the same time as in cinemas on OTTs.

This shows that even though cinema cannot be replaced by OTT platforms, it definitely creates its own segment. We may assume that there will be very few people in the future who would choose OTT over cinema.

So Yes!

OTT's Will be the New Future!



CHAPTER: TEN

APPLICATIONS OF OTT

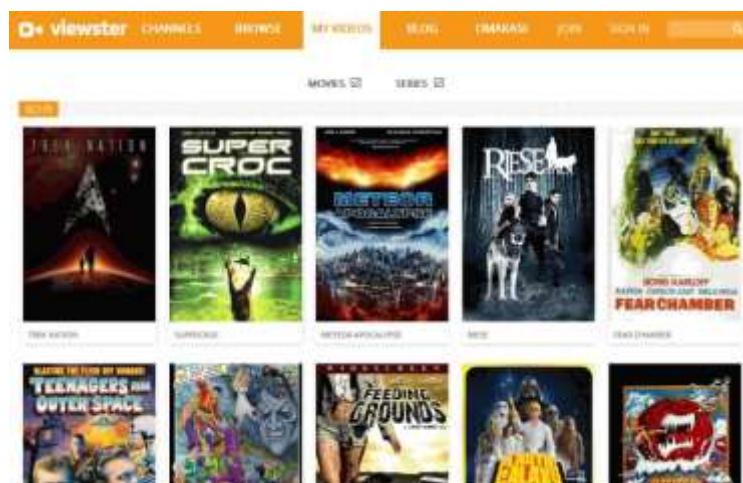
OTT is a term used commonly. The businesses that offer video to end-users across the web also take very distinct approaches. The breadth of thinking about how to monetise online video and where platforms sit in comparison to conventional free and paid TV services is illustrated by five separate case studies.

Viewster, based in Switzerland, has shifted away from traditional' subscription and transactional VOD to a model funded by free-to-view advertising. Voddler, based in Sweden, recently fully redesigned its model to concentrate on offering an open platform with a technology-based appeal. Dailymotion's online video website, whose business model has historically been focused on ads, has added features that enable content partners to launch pay services. Magine provides something more similar to a conventional online pay TV provider, while BSkyB is an instance of an existing pay TV operator that has launched an OTT service to reach a different audience.

Viewster: Breaking Free

Viewster screenshot UKViewster, based in Switzerland, has recently withdrawn from a conventional VOD and SVOD model based on selling a catalogue obtained via specific deals with studios, in favour of a free model primarily supported by advertisements with the option of paying a transactional fee to view content without advertising.

A home screen with weekly programming updates, based on a common theme and aligned to 34 country-specific channels, was also produced by Viewster. The website has also tried to rationalize the amount of content it has put there according to Henniges.



Voddler: Shelving & Sharing

Over the last six months, Sweden-based over-the-top video-on-demand provider Voddler has transformed its model from a classic retailer of high-value premium content, acquired from studio offers, to a more technology-focused pitch based on the venture.

Two utilities are anchored around the new-look Voddler: LiveShelf and ViewShare. LiveShelf enables viewers to assemble a shelf of content based on the cloud that they can access from any device.

ViewShare helps them, for about \$5 a month, to share the material on their shelf with up to 10 friends or relatives.

The concept is based on the idea that content owners can supply their goods to LiveShelf, based on rules that they decide for themselves and customers can then access them on any computer through a combination of free, transactional, subscription and sales.

Over the summer, LiveShelf and ViewShare are being beta-tested with an aim to launch commercially beginning in September.

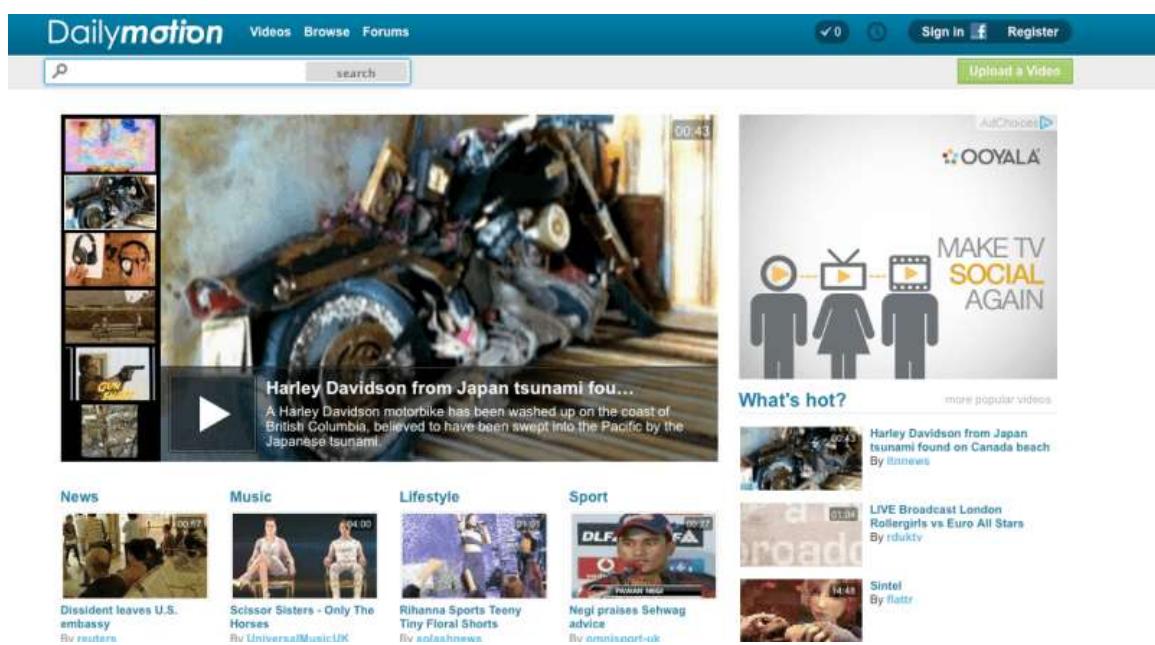
The screenshot shows the Mi Voddler LiveShelf interface. At the top, there's a navigation bar with the 'LiveShelf' logo and links for 'HOME', 'RENTORBUY', 'VIEWSHARE', 'FREE', and 'MYMOVIES'. Below this is a shelf displaying movie posters for 'EL VUELO DEL GLOBO ROJO', 'La Chingona', and 'elBulli'. To the right of the shelf are two film strip icons with '+' and 'x' symbols. The main content area is titled 'Sube tu propia película' (Upload your own movie). It contains a yellow box with instructions: '1. Voy a subir solamente películas de las que soy propietario o he hecho yo mismo.', '2. Su uso será exclusivamente personal y para ViewSharing con mis contactos incluidos en mi familia ViewShare.', and '3. El archivo de tu película debe estar en formato mp4. Aquí tienes [más información sobre el formato de codificación necesaria](#)'. There's also a checkbox for 'Confirma que has leído y aceptado estos términos y condiciones y quieres hacerlo un hogar de [Cargas de archivos](#)' and a 'CONTINUAR' button. To the right of this form is a sidebar titled 'Mi familia ViewShare' with options to 'Subir' (Upload), 'Encuentra' (Find), and 'Invitar' (Invite) family members. It also includes a link to 'Haz clic y añade amigos en tu familia ViewShare'.

Dailymotion: Open House

With 115 million unique visitors and 2.5 billion monthly video views, Dailymotion, the Orange-owned French competitor to YouTube, has a much wider challenge than niche-targeting OTT VOD services.

Dailymotion would introduce a video-on-demand subscription service, Dailymotion+, in collaboration with MDC Digital, a major local media player, in Turkey by the end of the year and said it will consider launching similar services in other territories.

OpenVOD, a component of the site that enables third party content providers to build pay-per-view services, was previously launched by Dailymotion. Dailymotion also has a range of other premium services, including in collaboration with DHX Media, a child SVOD service in France, Belgium and Switzerland, which offers access to more than 1000 children's shows for? A month, 4,49. It has also partnered on new paid channels with the public broadcaster France Televisions and the French-German channel Arte.



CHAPTER: ELEVEN

LIST OF FIGURES

Figure 1.1	OTT Platform
Figure 3.1	Q1 Gender
Figure 3.2	Q2 Age
Figure 3.3	Q3 Subscription
Figure 3.4	Q4 Network
Figure 3.5	Q5 Device
Figure 3.6	Q6 Content
Figure 3.7	Q7 Money Spent
Figure 3.8	Q8 Frequency
Figure 3.9	Q9 Ratings
Figure 3.10	Q10 Best Service
Figure 5.1	Movie.csv
Figure 5.2	TV.csv
Figure 5.3	MISSING VALUES
Figure 5.4	PLATFORMS
Figure 5.5	TITLE
Figure 5.6	Movie Count by Year
Figure 5.7	Best Movie Each Year According to IMDB Rating
Figure 5.8	Age Restrictions
Figure 5.9	IMDb Ratings
Figure 5.10	IMDb Ratings Distribution
Figure 5.11	Top 10 Directors Movie Count
Figure 5.12	Top 10 Directors with Movies Having Highest IMDB Rating
Figure 5.13	Genres Directed by Top 10 Directors
Figure 5.14	Top 10 Genres Movie Count
Figure 5.15	Top Movies of Different Genres based on IMDB Rating
Figure 5.16	Elbow Method
Figure 5.17	K means

Figure 5.18	Movies Count by Country
Figure 5.19	Best Movie in Top 10 Countries According to IMDB Rating
Figure 5.20	Movie Count by Language
Figure 5.21	Best Movie in Top 10 Languages According to IMDB Rating
Figure 5.22	Top 5 languages based on movie count across the OTT Platforms
Figure 5.23	RUNTIME
Figure 5.24	Average Runtime on different Platforms
Figure 5.25	Top 10 Longest Movies
Figure 5.26	Top 10 Highest IMDB Rated Movies
Figure 5.27	Language Vs Platforms
Figure 5.28	Language Vs Ratings
Figure 5.29	Genre Vs Platforms
Figure 5.30	Genre Vs Ratings
Figure 5.31	Director Vs Platforms
Figure 5.32	Director Vs Ratings:
Figure 5.33	Movies in India
Figure 5.34	Indian Language Movies
Figure 5.35	Indian Movies vs OTT platforms
Figure 5.36	Netflix Vs Prime Movies
Figure 6.1	OTT Chill out
Figure 7.1	Viewster: Breaking Free
Figure 7.2	Voddler: Shelving & Sharing
Figure 7.3	Dailymotion: Open House

CHAPTER: TWELVE

REFERENCES

1. Rohit Jacob Jose, Factors influencing the shift from traditional TV to OTT platforms in India, 2020, , <http://sersc.org/journals/index.php/IJAST/article/view/22888/11706>
2. Park, Sungwook, Kwon, Youngsun, Research on the Relationship between the Growth of OTT Service Market and the Change in the Structure of the Pay-Tv Market, 2019, , <https://www.econstor.eu/bitstream/10419/205203/1/Park-Kwon.pdf>
3. Hemant Joshi, Digital Media: Rise of On-demand Content, 2019, ,
<https://www2.deloitte.com/content/dam/Deloitte/in/Documents/technology-media-telecommunications/in-tmt-rise-of-on-demand-content.pdf>
4. Ritu Bhavsar, The Burgeoning Digital Media Consumption: A Challenge for Traditional Television and Advertising Industries, 2018, ,
http://amity.edu/UserFiles/asco/journal/ISSUE68_2.%20Ritu%20Bhavsar%20-%20AJMCS%20Vol%208%20No%201.pdf
5. Apurva Dixit & 17 Others, How OTT platforms can remain ‘on-demand ready’, 2017, ,
<https://assets.kpmg/content/dam/kpmg/in/pdf/2017/10/The-Digital-First-journey.pdf>
6. Ripal Madhani & Vidya Nakhate, Comparative Study of Viewers Behaviour Over Traditional Television Channels and Over Ott Video Platforms in Maharashtra, 2020, ,
<http://sersc.org/journals/index.php/IJAST/article/view/24371/12782>
7. Twentify, The Apple of Digital Television, 2018, ,
<https://www.twentify.com/blog/the-apple-of-digital-television-loyalty-and-satisfaction-research-on-over-the-top-media-services>
8. Tata Consultancy Services, The Rise of Over-the-Top Content: Implications for Television Advertising in a Direct-to-Consumer World, 2017, ,
<https://www.tcs.com/content/dam/tcs/pdf/Industries/communication-media-and-technology/Abstract/Over-d-Top-Content.pdf>
9. MICA, Indian OTT platforms: Streaming the new age narrative, 2018, ,
<https://images.assettype.com/afaqs/2020-09/29d6734e-75b8-44e8-a250-495637c3d0af/c2a7e479ef56b544ad65bd78810aef0fd4446397.pdf>

10. Bhargav Pancholi, The emergence of OTT platforms during the pandemic and its future scope, 2020, , <http://27.109.7.66:8080/xmlui/handle/123456789/619>
11. Sabyasachi Dasgupta & Priya Grover, Understanding adoption factors of Over-The-Top video services among millennial consumers, 2019, ,
http://www.iaeme.com/MasterAdmin/UploadFolder/IJCET_10_01_008/IJCET_10_01_008.pdf
12. Joshi Sujata & 5 others, Impact of Over the Top (OTT) Services on Telecom Service Providers, 2015, ,
[https://www.researchgate.net/publication/276175550 Impact of Over the Top OTT Services on Telecom Service Providers](https://www.researchgate.net/publication/276175550_Impact_of_Over_the_TopOTT_Services_on_Telecom_Service_Providers)
13. Quresh Moochhala, The Future of Online OTT Entertainment Services in India, 2018, ,
<https://actionesque.com/wp-content/uploads/2020/03/QM-OTT-future-2018.pdf>
14. Reshma & Chaithra, Proliferation of OTT apps in India: an empirical study of OTT apps and its impact on college students, 2020, ,
<http://www.ijrar.org/papers/IJRAR2001475.pdf>
15. Manoj Kumar Patel & 2 Others, OTT Viewership in “Lockdown” and Viewer’s Dynamic Watching Experience, 2020, ,
[https://www.researchgate.net/publication/343473867 A Study OF OTT Viewership in Lockdown and Viewers Dynamic Watching Experience](https://www.researchgate.net/publication/343473867_A_Study_OF_OTT_Viewership_in_Lockdown_and_Viewers_Dynamic_Watching_Experience)