*The Development of a Unified Content Recommendation system based on Netflix, Hulu, Disney Plus, and Amazon Prime Video*

**Rafeed Ullah**

***Abstract***

*Creating a unified machine learning algorithm that helps users find new recommendations based on their total movie and tv watching or like history, without the borders of various platforms, would be a very useful tool in the age of streaming. These services have fantastic machine learning algorithms to recommend you that next show to watch. However, as streaming becomes more popular and various production studios partner or create with new streaming services, it becomes hard to have your recommendation software to point you to your best matches if you watch on multiple platforms. I plan to create a Machine Learning model that takes content data from Hulu, Netflix, Disney Plus, and Amazon Prime Video to help create suggestions for your next watch based on your previously watched or liked content. This is in efforts to increase revenue by companies by increasing rates of renewal of subscriptions and for ad-based platforms to increase watch time which increase number of ads watched which will increase revenue streams.*

*According to Statista, the increase in streaming subscriptions is due to a high demand for original content [18]. They also note that projected revenue in the Video Streaming market in the United States will amount to about $39.25 billion during the year 2023, and average revenue per user will be $249.50 in the same time period. To ensure customers continue subscriptions for optimal growth, streaming corporations will need to increase their ability to not only lure in new customers, but to maintain their current ones as subscription prices continue to rise. This is particularly important as Statista notes that the user penetration rate will only increase by 6.4%. To help consumers in their cost-benefit analysis when renewing their subscription, recommendation algorithms will be key to help find the best content for users to stay satisfied with their subscription.*

*Today companies like Disney, Netflix, HBO, and more use a multitude of machine learning algorithms and advanced deep learning techniques to best recommend content to users. By applying a similar tactic to an amalgamation of data from various streaming services, we can better promote the best content for the user so that user satisfaction in these services could increase and the market would continue to grow.*

*As of the update of this dataset which was in 2021, movies outnumbered TV shows almost 3 to 1 in the combined dataset. As more and more original content comes out it brings up the question, is original content something that keeps users on the platform and increases watch time. Data from the Harvard Business Journal suggests that users do not care where their content comes from as long as the content is of good quality [17].*

*When finding a machine learning model to work with on this project I eventually settled on a Content-Based Recommender system due to my struggles with my other attempted algorithms. This program was influenced heavily by Kaggle user’s Erin Ward’s algorithm and converts all features of interest into binary values. The matches then are ranked based on cosine similarity and the top 5 results are returned. The algorithm could take any number of inputs but due to device and time constraints, I was limited to only 3 inputs.*

*My results found that the Content-Based filtering was at least partially successful, however it seemed to bias the first input heavily. In future iterations of this project, the goal would be to refine this model, especially in terms of performance. I would also aim to find a way to tie this tool into chat bots so that the user experience with this tool could be improved.*

**Introduction**

During the golden age of streaming, new services were popping up and “brought more TV and film into our homes than ever before” [1]. Companies like Disney were continuously investing in their content portfolios to draw new users away from the old cable system into their more affordable, ad-free, digestible content. However, with so many new services consumers ended up getting multiple streaming services rather than just “cutting the cord”. When watching different content on different platforms, you lose one of the best features of streaming services, or at least limit it’s benefit: the content recommendation engine.

All these platforms will first ask what you are interested in and allow you to select a few shows to base its algorithm around. According to Netflix, titles that are consumed on the service then supersede the previously inputted preferences [2]. Additionally, recently consumed titles take precedence over older ones in terms of weight in the recommendation algorithm. While all 4 of these services stand to gain a lot from good content recommendation systems and have very sufficient systems, all of them continue to work to better their algorithms.

One such way of doing this is the enhanced usage of Neural Networks and the continued research in this field. From at least 2017 to 2020, papers on neural networks have led other algorithms based on frequency of publications [3]. Netflix and Amazon, both market leaders, both implement deep learning among other various algorithms to complete different tasks to optimize their model’s efficiency and increase time spent on the app therefore leading to better customer retention and stable profits [8]. This comes as streaming prices continue to rise with Netflix currently being the only profitable service on the market while others work to increase their prices to become more profitable [9].

I investigated 2 papers to provide more insight into the topic prior to delving into creating a custom model for this project. The first paper [8] talks about the impact deep learning have when used for machine learning and recommendation systems. It uses Netflix’s famous competence in that field to better communicate its effectiveness. The paper begins by reviewing various recommendation tasks on Netflix and in their case study they find that different algorithms excel at different tasks. Deep Learning was believed not to have a significant improvement in performance when compared to highly tuned non-deep learning methods, however it was found to have significant advantages when dealing with a diverse and heterogeneous set of attributes in a dataset.

When first implementing a deep learning approach, there may be some difficulties in actually building the engine, however there are existing toolboxes that make it very easy to develop and provide flexibility to tune your model for what task is needed. When using a “bag-of-items” model, the team noticed that these models ignored temporal information. They then tried to move onto sequential models and learned that both these models were viable solutions for various tasks.

Their goal for their recommendation system was to ensure the best long-term satisfaction for their members. This is important to help make customers feel like they should continue renewal of their membership to continue watching more content. They mention that some of the challenges of this is that everyone individual is unique in term of their interest, taste, and context. One of the problems they encountered in development was that offline performance (which is driven by historical data) was not the same as online performance. Deep-Learning models also struggled with short-term actions as these could have noisy data, which can cause changes in the training objectives. These small changes in the training-objective would sometimes lead to big changes in the recommendations. Deep Learning models also may work for the majority of users but may fail for the minority.

Deep Learning tools are abundant and powerful, which is another benefit to it’s use. Tools such as TensorFlow, Keras, or PyTorch can be used to address recommendation problems. The use of multiple customized models working in unison in a greater ecosystem at Netflix created an unparalleled level of efficiency as they noted. Applying deep learning technologies can help solve problems that traditional methods struggled with such as images, text, and videos and their various modalities. Deep Learning can also amplify weaknesses in recommendation systems which will help developers better tune their models when addressing those issues.

The second paper [3] examines how and which advanced recommendation systems trended between 2010 and 2021, and what were the business aspects of those services. They used papers and articles published on Google scholar to investigate the topic. They then took this data, systemized them as well as analyzed research trends by year. This paper also discusses diving into each of these recommendation systems and classified the application service fields the systems would be used for as well as the techniques used were analyzed. They aimed to do this because they note that the spread of the internet, smart devices, and Social Network Services have led to an increase in number and development of web and application-based services. These services often rely on various recommendation systems to receive data then make or assist in the decision-making process of the user. As more and more data is introduced, this process needs to make decisions quicker and recommendation systems have improved to combat this problem.

The paper found a correlation between the increase in research/papers published of recommendation systems, and the business growth in the applied fields related to the recommendation system. The researchers note that recommendation systems that utilize real-time data were able to drive better outcomes oftentimes. For example, wearable data that constantly gathered information helped aid in better diagnostic and treatment outcomes for patients even if clinical data provided more accurate information. It is important to note that the data here did not work alone, but instead it was used as supplemental to clinical data and provided real-time data.

The recommendation systems studied generally had 2 parts where there was a data mining segment as well as a recommendation filtering model. Each one of these systems were highly customized to perform the best at whatever task it was assigned to. In terms of model usage, the Content-Based filtering model has moved to be used alone. Collaborative Filtering models meanwhile seemed to be used in a hybrid model to complement it’s strength and weaknesses.

When looking into the filtering models of recommendation systems, text filtering was a method that had been used and studied over an extended period of time. However, Neural Network interest has increased recently, as more organizations and individuals look to implement this system.

This article also noted that real world companies such as Netflix, Amazon, and Yahoo are actively using these various systems in their recommendation algorithms according to papers published by representatives of their companies.

**Methods**

The data for this project will be taken from the online data community, Kaggle. Shivam Bamsal, a user in the Kaggle community, provided 4 separate datasets with data from Amazon Prime, Netflix, Disney Plus, and Hulu [4, 5, 6, 7]. According to Phil Nickinson, these services were among the most popular streaming services in the world [22]. Netflix ranked 1st, Amazon Prime Video 2nd, Disney+ 3rd, and Hulu at 6th. This is impressive considering Hulu is only available in the United States [23]. Disney, who owns Disney+ and Hulu, is considered the biggest media company in the world by Statista with a market value of $183 billion U.S. dollars [24]. These datasets has everything someone might want to use in terms of associating different shows such as release date, title, cast, and most importantly: category tags. I adjusted the IDs of each piece of content to ensure no overlap then append all the data into a single sheet before adding the file to the data frame. Since all the category tags and cast tags are all split based on the delimiter “,”, I needed to split the data based on that, so my algorithm can read all the various tags separately.

Scraping and/or distributing data directly from Netflix, Disney Plus, and Hulu are expressly denied based on the terms and conditions of these various platforms [11 , 12, 13]. Because of this I decided to use an already public database to ensure no legal repercussions regarding this data. None of the data used has personal information or information not available to the public. I created a GitHub Repository to host my code as well as other project deliverables such as this project update. The public repository can be found in the following link: https://github.com/RafeedU/Unified-Content-Recommendation-System.

To complete this project I’ve had to employ the use of various tools and plugins for Python. These include pandas, numpy, nltk, re, seaborn, tensorflow, plotly, sklearn, rake, and keyBERT so far. Most of these are common tools in the data science community and are used for machine learning, however some like seaborn and plotly are used for graphing/plotting data. They also allow for custom color patterns and detailed tweaking to your charts. I planned to use Vikas Singh’s outline [15] for data cleaning in an attempt to prepare my data for the machine learning methods I would employ as the dataset he used for his project was similar in structure and complexity to mine. However, due to the different natures of our datasets, I will be engineering a solution unique to my dataset.

A major issue I encountered was that the ratings field in the data which noted parental ratings for many of the titles in the source data file. However, when performing the exploratory data analysis (EDA), I noticed that there were gaps in the tagging of this data. On further analysis, I had discovered that the tagging for rating was non-existent for a vast majority of the data. Additionally, there was an error in tagging for other fields such as release data which had to be corrected. Upon correction, it was decided that the rating column would be dropped this application.

The description field in the data will also need to be disseminated to find common themes between content. This is a very important step to help the model find related content. To do this I will need to implement a form of NLP (Natural Language Processing). According to at least 1 paper, a good method to approach this would be using RAKE (Rapid Automatic Keyword Extraction). They note that based on their benchmarking, RAKE was found to be “more computationally efﬁcient than TextRank while achieving higher precision and comparable recall scores” [14]. I then found a repository in Kaggle that attempt a similar type of problem I was facing and used part of their code to help attempt to perform keyword extraction [15].

After many hours of struggle, I found that RAKE had issues with the object types in the dataset and none of the keywords were being extracted. In search of an alternative I learned about a method called KeyBERT. KeyBERT was touted as being “minimal and easy-to-use” [16], so in the interest of time, I attempted to leverage the tool as a solution for my keyword extraction woes. By tokenizing the words and using KeyBERT’s ranking function, I found that this method was able to resolve this issue. I also broke apart the cast and category columns to allow for the machine learning algorithm to parse through a list of multiple attributes to find a match. This process involved removing punctuation, and stop words as well, and left me with 3 new columns with the top 3 keywords from each description. Once we had this, we could drop the description column from out of the Recommendation data frame.

For the cast and category columns, I used a more simple approach by splitting the list on the comma delimiter. The lambda function allowed me to take many arguments and produce a controlled result with only the top 3 values.

Unfortunately, I also ran into more issues, as extracting director information in particular seems to have given me a multitude of errors as my attempts to tokenize and standardize names from lists have been unsuccessful. My plan was to try using various machine learning methods to attempt to create a well-functioning model, however I began to focus on creating a single well-functioning model.

I then had to work to encode my potential features so that I would be able to feed the data into my machine learning algorithms. I attempted to use K-Means Clustering as it was considered an easy to use tool, that only “required a kd-tree as the only major data structure” [19]. This method also efficient, and would hopefully run smoothly on my personal laptop which is limited by about 16 GB of RAM. K-means clustering is an unsupervised model, which would it to use clustering to find the relationships between the shows you currently like and similarly tagged shows. Unfortunately, this algorithm required data dense data. Since, too much of my cast and keyword data was sparse, this prevented me from using this algorithm, and converting the data to dense and encoding it proved too much of a challenge in the time I had.

I tried to remove the keywords from my algorithms, and attempt to solve the issue, however due to my cast encoding issue, I remained unsuccessful. I tried to use various encoding methods, from manually performing encoding on data such as streaming service and content type, to using One Hot Encoding and TF-IDF (Term Frequency – Inverse Document Frequency) vectorization. At this point I had read about or attempted multiple methods machine learning including K nearest neighbor, Support Vector Machines (SVM), and Deep Learning. With limited time remaining, I decided to focus my efforts on finding a solution that better fit my data.

I then tried to pivot to Agglomerative Hierarchical Clustering, which also failed when attempting to featurize the cast column. I attempted to work around this issue and concatenate whatever features I could I had an issue where zero-dimensional arrays were being created. It turned out many of my features were having issues because of something having to do with their data types, even though they were all objects. Sometimes modifying the data converted these into uniterable tuples or lists that could not be read or converted to a uniform lowercase string attribute. When modifying these into strings, the encoders could not find uniform data, or claim that the strings being passed “had no attribute ‘lower’”.

When no method of encoding seemed to be a solution I decided to utilize a combination of binary encoding and Content-Based filtering, which Kaggle-user Erin Ward employed [21]. I decided to use her strategy as she used her formula for the Netflix dataset by Shivam Bansal which was one of my data sources. This meant that our data would be in a similar format and that her method should be plug-and-play. I tried to create an additional encoding field to create binaries for all the countries that content release, however I was having an error where the data could not be manipulated to prepare for the algorithm. Python claimed that there was “no attribute lower” on the country field even though the contents of this value only included simple string objects. This was similar to other errors I had throughout this project that I had to leave unresolved due to time constraints.

With binary encoding each data frame feature gets converted to a binary value and each title then checks if it matches with the binary value of directors, cast, and categories. It utilizes dot product or cosine similarity (in my case it was cosine similarity) to indicate which total binary values are the best matches for the input. The higher the product the higher the similarity as this indicates more common features [25] I also removed any rows with Null values in these fields to avoid potential errors and better match content with other appropriate titles. I ended up with a data frame that was over twenty-one thousand rows, and over sixty-nine thousand columns. This resulted in an algorithm that took over 2.5 hours in just running this snipped of code.

**Results and Analysis**

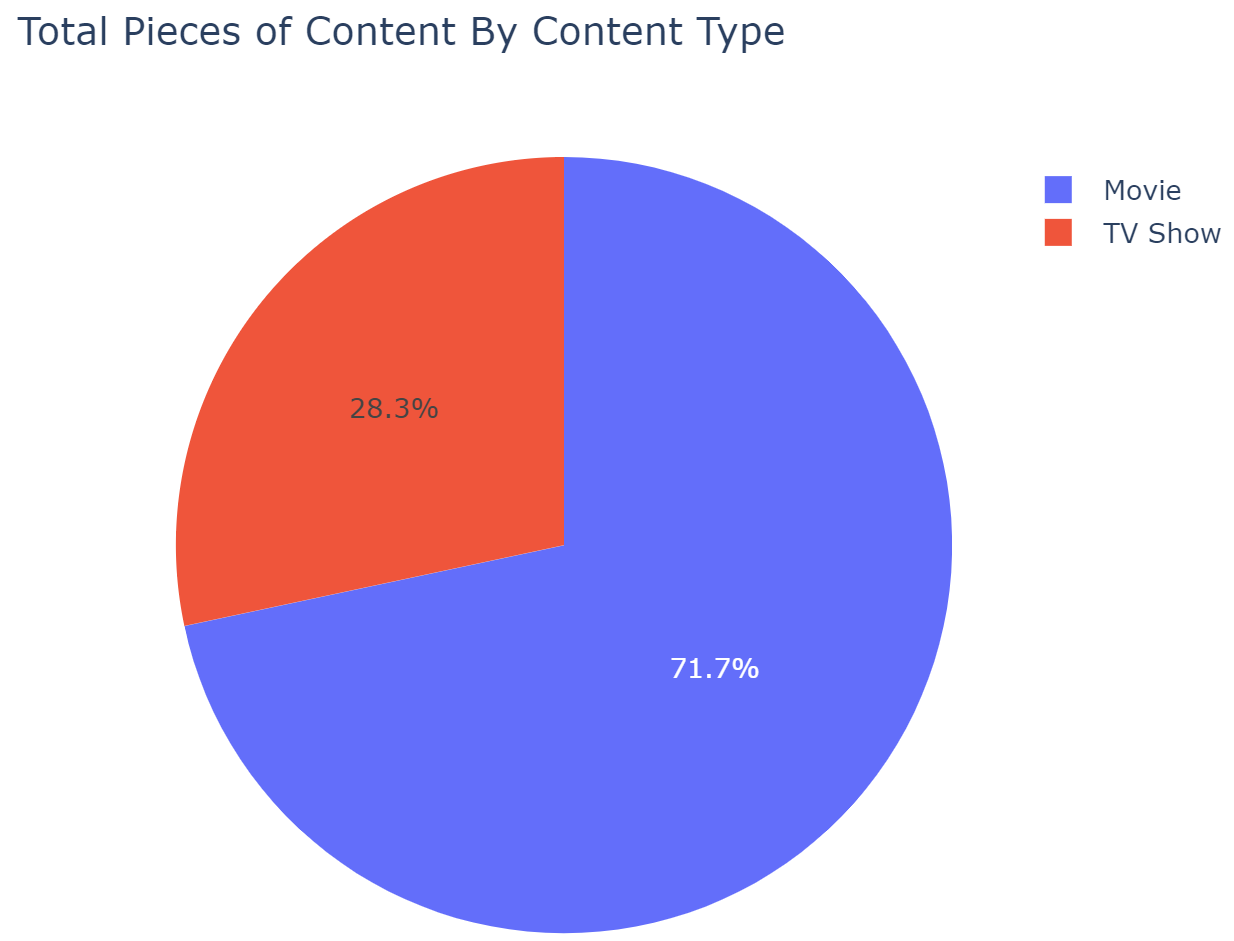
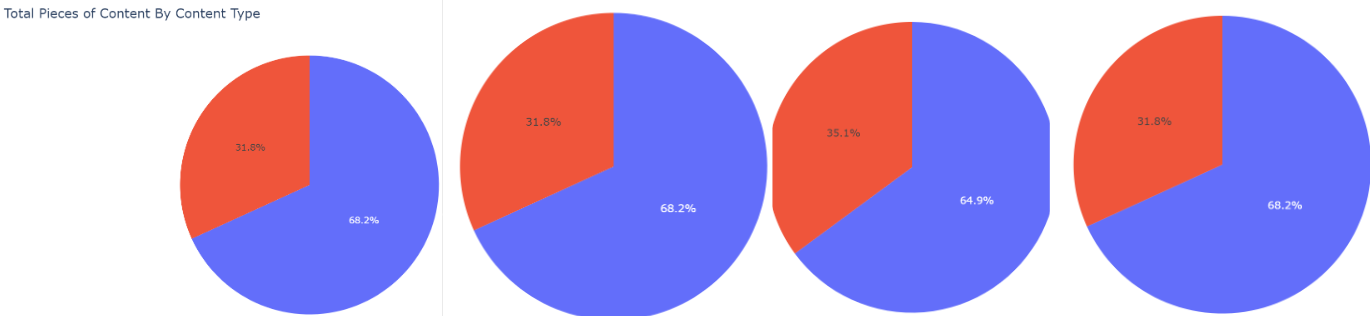


Figure 1. A pie chart that that shows the respective proportions of movies versus tv shows in the dataset.

When performing my exploratory data analysis, I learned about more about the data at hand. Over 70% of the content at hand were movies over TV shows as shown in Figure 1 (this is noted by the blue-purple) color in the majority of the histogram). This was interesting, as I would think TV shows may have been higher in quantity to keep users coming back or “binging” content, rather than finishing content in one sitting. I took a deeper look into this to see how this changed over time between 2018 to 2021. In Figure 2 below, we can see how the released content evolved from year to year.

Figure2. This series of pie chars indicate has been a marginal difference in the TV show to Movie content distribution by release date, but primarily indicates little to no active change.



Additionally, the plot that provided the information that our rating system was not sufficient, is shown in Figure 3. As shown, only Netflix had a significant amount of their content tagged. Manually tagging the remaining data would be unfeasible as it would require searching up the data 1 by 1 on IMDB or Common Sense Media, which was done for a few titles before admitting that it would be an unfeasible endeveor.

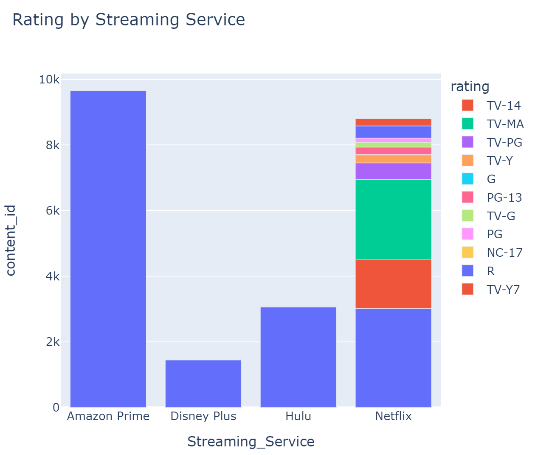
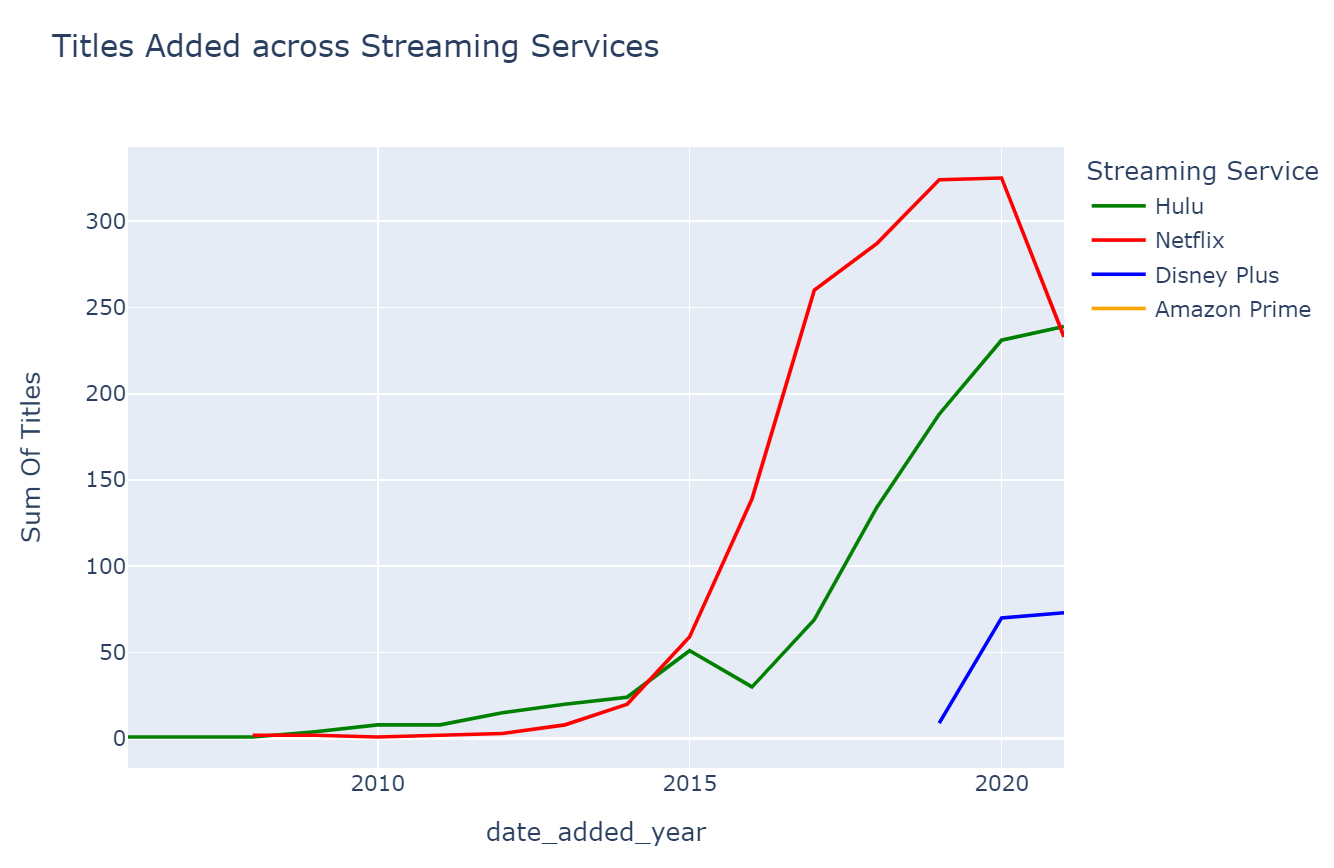


Figure 3. A column chart separated by streaming service and color coded based on parental rating.

A major part of my investigation, is that it is based on the belief that streaming will continue to grow and platforms will continue to add content. This makes content more abundant but harder to choose which title to watch next for the user. As a result I created a line chart, as shown in Figure 4, that shows the number of titles added over time to Netflix, Hulu, and Disney Plus. I wanted to add Amazon, however that data was unavailable. I also had to eliminate a large portion of Disney Plus content. In the graph we can see that all 3 platforms continue to add content at a rapid pace, however it is important to note that this data was made mid-way through 2021. This can account for what seems to be a slowdown in the rate of new content on Netflix.

Disney Plus launched in November of 2019, and on launch it hosted a large catalog of titles. Including these titles as “content added” would skew the data in a way I would not intend, since I wanted to view content added over time rather than content available at launch. I decided to test my modified algorithm with Erin Ward’s Content-Based Data Filtering with binary encoding [21]. Her algorithm utilizes simple and efficient binary encoding to convert every possible feature into a binary then compare the cosine similarities of titles to find the best pieces of content to recommend. I decided to use her algorithm as I knew it would work because the format of out datasets were identical. She also employed the use of Shival Bansal’s Netflix dataset, which was one of my inputs as well. After running a shortened version of her model twice (once for each input), I then compared the results to my modified code. My modifications included the ability to take in multiple inputs then re-run the algorithm, and find a cumulative total of all the cosine similarities and sort them based on their similarities (the highest scores). The 3 titles I used as inputs were “Nevertheless”, “Bridgerton”, and “Itaewon Class”. In both our models we based the encoding on 4 fields: Director, Cast, Category, and Country.

Figure 4. A line chart that shows titles added to each platform (Hulu, Netflix, and Disney Plus) each year not including titles the service launched with.



A screenshot of a computer

Description automatically generatedWhile running a shortened version or Erin’s code for one of these shows, “Nevertheless”. Nevertheless is a South Korean romantic drama lead by Han So-hee and Kang Song. I noticed that it accurately chose a list of International Romantic Dramas, however since my dataset did not have the information of it being a South Korean drama, the algorithm simply picked what seemed to be random International Romantic Dramas as shown in the table below.

Doing the same for Itaewon class, starring Park Seo-Joon, I got expected results. Since Itaewon class has country data in my data set, the algorithm mostly returned International Korean dramas from South Korea as shown in the table below. I’m not sure on what grounds, it classified “The Adjusters” which is a Chinese drama over other South Korean dramas, but this result seemed to be more promising and a sign the the algorithm is working to an extent.

A screenshot of a computer

Description automatically generated

I then ran my version of the algorithm where I inputted my 3 chose aforementioned titles. The result was similar to that of simply running the algorithm with Nevertheless alone, as shown below. 4 of the top 5 matched the results of Nevertheless. This was alarming to me and I believe hides errors in the design of the program. All 5 of these were also International Romantic Dramas. This was interesting as while Nevertheless filled that criteria, the others titles didn’t necessarily match. Bridgerton was classified as a Romantic Drama, and Itaewon Class was classified as an International Korean Drama.

A screenshot of a computer

Description automatically generated

Following this, I attempted to do this experiment again with a list of 3 pieces of content that had more dissimilar categories. The 3 titles I chose were “Big Hero 6”, “Peaky Blinders”, and “High School Musical 3: Senior Year”. These range from coming of age films to international crime dramas. In the table below, you can see the results. The results seem to indicate that the algorithm is at least biasing its results off of Big Hero 6 due to the Action-Adventure and Family categories.

A screenshot of a computer

Description automatically generated

**Conclusion**

I attempted machine learning algorithm that is able to take at least 3 inputs of previously liked content and use that to create a list with a ranking of most likely to get watched next by the consumer. This service will be important as wages and other production costs increase, costs of content platforms will need to increase for them to be profitable. The only way for these customers to sustain their business is to ensure that despite cost increases or additional advertisements, customers will continue paying for the content that the services provide. The best way to do this is to make them spend more time on the platform and considering this want as a necessity that they would not want to get rid of. We can accomplish this by having a recommendation engine that helps various content platforms by creating a way that allows users to find the best content for them.

Time constraints proved to be a significant issue as by the beginning of November, I began to struggle with roadblocks. The combination of dealing with this capstone’s coding element as well as the project updates proved to fight for my time. I was also working full-time, and leading a project in my one other class which left me with limited time and energy. However, as the last month for the project came around, I found myself working late at night to run, test, and document my code and journey through this project. Unfortunately I had really high expectations for myself during this project, but as time progressed and frustrations increased, I felt myself needing a helping hand. Working solo on such an intensive project proved to be very difficult, however I appreciated the ability to learn about different models and their real world applications.

Due to the similarity of the dataset, I wanted to compare my results to a user named Erin Ward on Kaggle who used the same Netflix dataset as myself [20]. She also had a similar goal in attempting to create a recommendation engine using this data. I have not been able to compare results due my lack of results, but I aim to create a similarly or more effective platform for users with more information available because of my merging of Hulu, Amazon Prime Video, and Disney Plus content.

My final result’s extended runtime length proved that this algorithm was not implementable in the real world. Recommendation algorithms need to work in real time. Additionally, to improve the accuracy of the results found, the combination of algorithms would have proven to be useful. Techniques such as boosting, chaining, or attacking can help provide the best results for the customer.

Future enhancements could be tying this to chat assistant platforms so they can better understand users and help users find the content they would enjoy. By utilizing this algorithm in conjunction with LLMs the user could easily input request new shows to watch based on the previous shows they liked. Additionally, the user could enter specific tags that relate to content they like to watch such as genre or director. This input could be read by the system and create a model that fits only what the user needs to enhance efficiency, and create a more human experience.

**References**

[1] Cranz, A., & Becker, D. (2022, December 14). The golden age of the streaming wars has ended. The Verge. Retrieved October 8, 2023, from <https://www.theverge.com/2022/12/14/23507793/streaming-wars-hbo-max-netflix-ads-residuals-warrior-nune>

[2] Netflix. (n.d.). How Netflix's Recommendations System Works. Netflix Help Center. Retrieved October 8, 2023, from https://help.netflix.com/en/node/100639

[3] Ko, H., Lee, S., Bocanegra, S., & Choi, A. (2022, January 3). A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields. MDPI. Retrieved October 8, 2023, from https://www.mdpi.com/2079-9292/11/1/141.

[4] Bansal, S. (2021, October 18). Netflix Movies and TV Shows. Kaggle. Retrieved October 8, 2023, from https://www.kaggle.com/datasets/shivamb/netflix-shows

[5] Bansal, S. (n.d.). Amazon Prime Movies and TV Shows. Kaggle. Retrieved October 8, 2023, from https://www.kaggle.com/datasets/shivamb/amazon-prime-movies-and-tv-shows

[6] Bansal, S. (n.d.). Disney+ Movies and TV Shows. Kaggle. Retrieved October 8, 2023, from https://www.kaggle.com/datasets/shivamb/disney-movies-and-tv-shows

[7] Bansal, S. (n.d.). Hulu Movies and TV Shows. Kaggle. Retrieved October 8, 2023, from https://www.kaggle.com/datasets/shivamb/hulu-movies-and-tv-shows

[8] Steck, H., Baltrunas, L., Elahi, E., Liang, D., Raimond, Y., & Basilico, J. (2021). Deep Learning for Recommender Systems: A Netflix Case Study. AI Magazine, 42(3), 7-18. https://doi.org/10.1609/aimag.v42i3.18140

[9] Canal, A. (2019, March 9). Your streaming bill is about to go up even more. yahoo!finance. Retrieved October 8, 2023, from https://finance.yahoo.com/news/your-streaming-bill-is-about-to-go-up-even-more-200938833.html?guccounter=1&guce\_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce\_referrer\_sig=AQAAAFYZ9uhxknmK6RJAOHi33nMu9PyDzuE7Vk3vnrV2rkhtDggUk-dZ-aBzdu29m5zbAurH8vAdo\_ZTQ0O80

[10] Ilmi, M. F. (2022, October 27). Hybrid and Tensorflow Recommender System. Kaggle. Retrieved October 8, 2023, from <https://www.kaggle.com/code/mfaaris/hybrid-and-tensorflow-recommender-system>

[11] Disney+. (2022, September 27). Subscriber Agreement. Legal. Retrieved October 22, 2023, from https://www.disneyplus.com/legal/subscriber-agreement

[12] Hulu. (2022, October 24). Hulu Subscriber Agreement. Hulu. Retrieved October 22, 2023, from https://www.hulu.com/subscriber\_agreement

[13] Netflix Queue. (2023, January 25). Privacy Statement and Terms of Use. Netflix Queue. Retrieved October 22, 2023, from <https://netflixqueue.com/privacy>

[14] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic Keyword Extraction from Individual Documents. In Text Mining: Applications and Theory (pp. 1-20). Wiley. 10.1002/9780470689646.ch1

[15] Singh, V. (2019, March 9). Netflix Movies and Shows: Plotly & Recommender SYS. Kaggle. Retrieved October 22, 2023, from <https://www.kaggle.com/code/vikassingh1996/netflix-movies-and-shows-plotly-recommender-sys/notebook>

[16] Grootendorst, M. (n.d.). KeyBERT. Maarten Grootendorst. Retrieved November 20, 2023, from https://maartengr.github.io/KeyBERT/

[17] Prince, J., & Greenstein, S. (2018, April 24). Does Original Content Help Streaming Services Attract More Subscribers? Harvard Business Review. Retrieved November 20, 2023, from <https://hbr.org/2018/04/does-original-content-help-streaming-services-attract-more-subscribers>

[18] Statista. (2023). *Video Streaming (SVoD) - US*. Statista. Retrieved December 4, 2023, from https://www.statista.com/outlook/dmo/digital-media/video-on-demand/video-streaming-svod/united-states#revenue

[19] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881-892, July 2002, doi: 10.1109/TPAMI.2002.1017616.

[20] Ward, E. (n.d.). Best Movies on Netflix - EDA. Kaggle. Retrieved December 4, 2023, from <https://www.kaggle.com/code/eward96/best-movies-on-netflix-eda>

[21] Ward, E. (n.d.). Netflix Recommendation Engine. Kaggle. Retrieved December 19, 2023, from https://www.kaggle.com/code/eward96/netflix-recommendation-engine

[22] Nickinson, P. (2023, December 13). The 10 most popular streaming services, ranked by subscriber count. Digital Trends. Retrieved December 19, 2023, from https://www.digitaltrends.com/home-theater/most-popular-streaming-services-by-subscribers/#dt-heading-3-disney-1502-million

[23] Getting started with Hulu. (2023, October 16). Hulu Help. Retrieved December 19, 2023, from https://help.hulu.com/s/article/getting-started

[24] Guttmann, A. (2023, August 29). Top media companies by market value worldwide 2023. Statista. Retrieved December 19, 2023, from https://www.statista.com/statistics/1374473/the-worlds-largest-media-companies-by-market-value/

[25] Google. (2022, July 18). Content-based Filtering | Machine Learning. Google for Developers. Retrieved December 19, 2023, from https://developers.google.com/machine-learning/recommendation/content-based/basics