**Netflix Movies and TV Shows Clustering**

**Aashish Thakur**

**Data science trainee,**

**Almabetter, Bangalore**

**Abstract**

With a Netflix subscription, users can stream movies and TV shows on any internet-connected device without being interrupted by ads. Additionally, users can download movies and TV episodes on their iOS, Android, or Windows 10 device and view them offline. But users have the option to stop their subscriptions at any moment. As a result, the business must maintain user engagement on the platform and maintain their interest. Systems that make useful suggestions to users are crucial in this situation, which is where they start to play a significant role.

We have a collection of movies and TV shows that were uploaded to Netflix between 2008 and 2021. In order to create a recommender system, we want to group together comparable contents.

**Attribute Information**

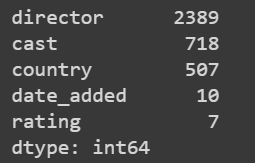
The given dataset contains 7787 observations and 12 columns/variables which are as follows:

* show\_id: Unique ID for every Movie / Tv Show
* type: Identifier - A Movie or TV Show
* title: Title of the Movie / Tv Show
* director: Director of the Movie
* cast: Actors involved in the movie / show
* country: Country where the movie / show was produced
* date\_added: Date it was added on Netflix
* release\_year: Actual Release year of the movie / show
* rating: TV Rating of the movie / show
* duration: Total Duration - in minutes or number of seasons
* listed\_in: Genre
* description: The Summary description

**Exploratory Data Analysis**

1. **Handling Missing Values**

The variables which contains null values are director, cast, country, date\_added and rating. In string data type variables we replaced null values with empty spaces and we removed null values observatuions in date added.

****

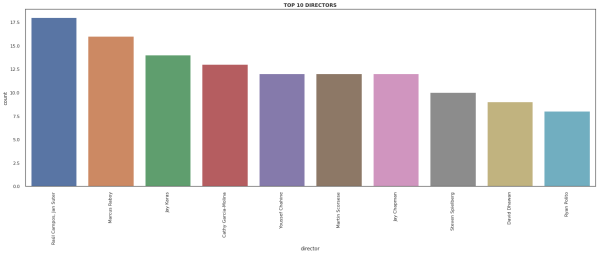
**Checking is there any repetition in TV shows as different seasons**

We look for any recurrence of TV shows throughout multiple seasons. This is accomplished by counting the values in the "type" column to determine the number of films and television programmes in our dataset.

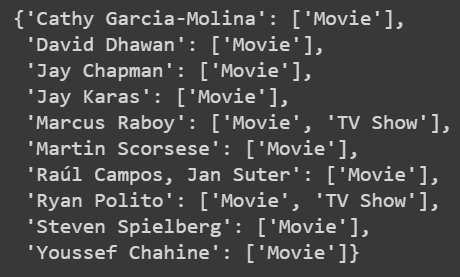
The quantity of original TV-show titles is then determined. We confirmed that there is no repeat of TV Shows across different seasons because the number of TV Shows in the "type" column and the number of unique "titles" in the TV Shows are equal.

**Top 10 directors**

The top 10 directors by the number of contents directed are determined by doing value counts on the director column and then using a seaborn countplot to obtain the top 10 directors' plots.

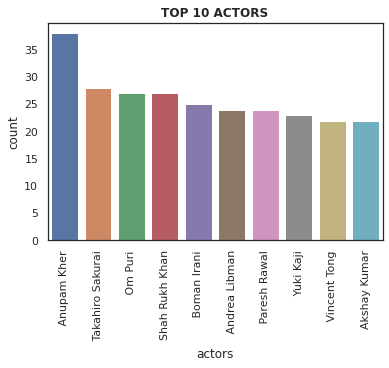


We keep a list of the top filmmakers, and for each entry in this list, we create two dictionaries with the director's names as the keys and the genres of content they have directed as the values

**Top 10 actors with Country**

The actors are listed in the cast columns, and there are multiple entries for each observation, each one separated by a comma. As a result, we create a list of all the actors' values and then run a countplot on it using the top 10 actors.



**When Movies are added on Netflix**

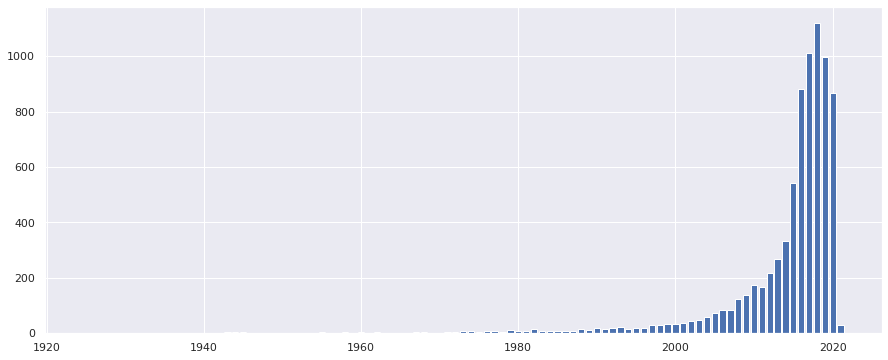
From the datetime column, date\_added we extract the month and append it to a new column called month\_added for each observation.

Next, we perform a countplot on the month added column to get our plot containing months on x-axis and the number of movies released that month on y-axis

**Change in ratings w.r.t time**

First, we display the number of films from each year that are included in our dataset.

For this, we create a graph with the number of movies released on the y-axis and the years on the x-axis.



We then split the dataframe into two sub-datasets, one containing only movies and the other only TV shows, to visualise how ratings of content have evolved over time.

The count of ratings is then aggregated for each of these datasets based on grouping by release year and rating. Then, using the groupby operation again on release year and transform to do the percentage operation, each entry is divided by the total number of films/TV shows released that year and multiplied by 100 to obtain the percentage. Plot the dataframe next.

**Type of content available in different countries**

Content can be categorised by type i.e., Movie or TV-Show, rating and genre. We are going to look into all of them.

**1.Content available by rating**

We execute a groupby operation on nation and rating, counting the number of times each rating appears in each country for each rating. Then, for each rating, we depict the number of times it occurs in a nation other than zero.

**2. Content available by type**

We count the type for each country using a groupby operation on country and type. The dataframe is then shown, but first we run a log2 operation because the count values have such wide ranges.

**3.Content available by genre**

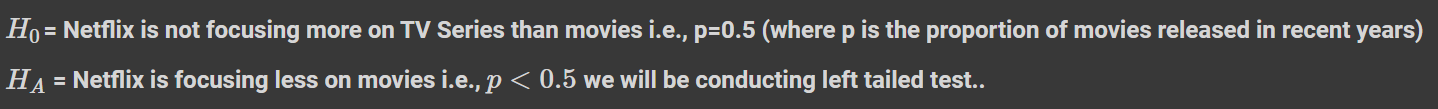
The listed in column contains the genres of each film or television programme, with commas between each genre. The list of all the genres to which a piece of content belongs is therefore first put into a column called genre. Strings in the listed in column are split using the.split(',') function to create a list.

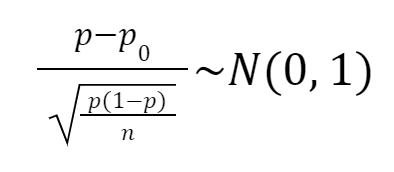
Once the genres column has been retrieved, MultiLabelBinarizer is applied to it, the appropriate dataframe is created, and it is combined with the original one.

The next step is to create a dictionary, where the values are the total of the columns and the keys are the genres or labels of the mutilabelbinrizer. In this manner, we may count the instances of each genre in the dataframe. The intended outcome is then obtained by plotting this data.

**Has Netflix been increasingly focusing on TV rather than movies in recent years**

To draw any conclusions from the 7000 observations spread across 13 years in the supplied dataframe, we use hypothesis testing.



Here, p represents the proportion of movies released in a given year.

Observed proportion =

In our case n=12, because we consider uploads from year 2008 to 2020.

Thus, we get and

Therefore we cannot reject the null hypothesis. Hence, Netflix may not be focusing less on movies

**Clustering and recommender systems**

Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group. In theory, data points that are in the same group should have similar properties and/or features, while data points in different groups should have highly dissimilar properties and/or features. Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields.

K-Means is probably the most well-known clustering algorithm. Each data point is classified by computing the distance between that point and each group center, and then classifying the point to be in the group whose center is closest to it.

**1. k-Nearest neighbours on genre and type only**

To begin, we just use the listed in and type columns to suggest articles of the same category and genre.

We extract all the genres of a certain piece of material from listed in and one-hot encode the type column.

Now, a function based on the k-Nearest Neighbors principle is defined. When a certain id number and the number of contents to recommend are supplied in the function, the function produces the contents of the nearest cosine distances except for the one that was entered. We keep all the columns/variables of an observation on an n-dimensional plane.

Next, we choose to propose movies and group movies using an improved dataset.

**2. Text Pre Processing**

1. Rename the title column with ‘NAME’.

2. Fill null values in text columns with empty spaces.

3. Make a new column called text which contains director, cast, listed\_in and description. *(with space between them)*

4. Next one hot encode the type, rating and country column such that any of them having less than 5 counts is not considered.

5. Remove punctuations, stopwords from the created ‘text’ column and changing all alphaets to lowercase.

6. Lemmatize words from the text column.

7. Apply countvectorizer on the text column with max = 0.9, min = 3, n\_grams=(1,2), max\_features = 5000, binary = True.

**3. DBScan**

On this upgraded dataset, we use DBScan to do a clustering operation, but due to excessive dimensionality, it did not operate as intended.

**4. K-Means clustering**

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. The objective of K-means is simple: group similar data points together and discover underlying patterns. To achieve this objective, K-means looks for a fixed number (k) of clusters in a dataset. A cluster refers to a collection of data points aggregated together because of certain similarities. You’ll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The ‘means’ in the K-means refers to averaging of the data; that is, finding the centroid.

**How the K-means algorithm works:**

Data mining's K-means technique uses an initial set of randomly chosen centroids as the starting points for each cluster to process the learning data, and iterative (repetitive) calculations are then used to optimise the positions of the centroids.

It stops developing and enhancing clusters when either:

• Due to the success of the clustering, the centroids have stabilised; their values have not changed.

• The specified number of iterations was completed.

**K-Means clustering**

The text pre-processed dataframe from type movie through the end of columns is initially created as a 2-dimensional array.

Next, we create methods to calculate the silhouette score, compute the sum of squares within each cluster, and plot the results for each value of k. K-elbow visualizer is essentially.

Now, for equally spaced values between 2 and 703, the distance is 100, we plot the silhouette scores and the k-elbow plot. Then, for the ideal value of k, we selected additional values of k and plotted the elbow plot and silhouette score for values close to the ideal value of k before settling on a k-value of 175 as the final result.

After that, we cluster the pre-processed dataset using k-Means.

Ultimately, employing the same functions outlined in 6.1 With better outcomes, we create clusters and make movie recommendations.

**Conclusion:**

After finishing EDA and clustering, our project has finally come to a close. The created recommender system is very effective and produces outcomes similar to those of the Netflix app itself.

**References:**

1.TowardsDataScinece

2. Medium blogs

3. GeeksforGeeks

4. Javatpoints