**NETFLIX MOVIES AND TV SHOWS CLUSTERING**

**UNSUPERVISED ML**

**Sohail Mohammad, Laveti Syam sagar, Bisetty Kodanda Nagesh**

**Data science trainees,**

**AlmaBetter, Bangalore**

**1. Problem Statement:**

This dataset consists of tv shows and movies available on Netflix as of 2019. The dataset is collected from Flexible which is a third-party Netflix search engine.

In 2018, they released an interesting report which shows that the number of TV shows on Netflix has nearly tripled since 2010. The streaming service’s number of movies has decreased by more than 2,000 titles since 2010, while its number of TV shows has nearly tripled. It will be interesting to explore what all other insights can be obtained from the same dataset.

Integrating this dataset with other external datasets such as IMDB ratings, rotten tomatoes can also provide many interesting findings.

In this project, you are required to do

1. Exploratory Data Analysis

2. Understanding what type of content is

available in different countries

3. Is Netflix increasingly focusing on TV

rather than movies in recent years?

4. Clustering similar content by matching

text-based features

Based on the attributes related to the Tv shows or movies, we will be implementing diﬀerent clustering algorithms which come under the unsupervised Machine learning category.

**2. Introduction:**

The dataset includes over 7787 records and 12 attributes.

**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

## **3. Data Cleaning:**

Earlier to EDA, cleaning the data is fundamental since it'll get freed of any ambiguous data that can have an affect on the results.

director, cast, country, date\_added, and rating columns have missing or null values. we have replaced all the null values of the columns with ‘Not Known’.

we removed the show id column because it doesn't offer any useful information.

## **4. Type of shows:**

1. Movies
2. TV shows

There are two kinds of shows that the Netflix dataset contains.

**5. Steps involved:**

* **Exploratory Data Analysis**

Exploratory Data Analysis refers to the basic handle of performing initial investigations on data so as to find designs, to spot irregularities, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

* **Null values Treatment**

director, cast, country, date\_added, and rating columns have missing or null values. We replaced all the null values of the columns with ‘Not Known’.

**6. Algorithms:**

**TF-IDF:** Term Recurrence, Converse Document Frequency. It shows the significance/relevance of a word in a corpus. Term Frequency represents the number of instances of a given word and inverse document frequency tests how relevant the word is.

**PCA:** PCA could be a dimensionality reduction technique. In this principal component are computed, and a lot of data is additionally retained. The graph shows the clarified change value for the different number of PCA components. n\_components 4000 was chosen in this project.

Textual data were combined and changed over into numerical data utilizing TF-IDF. Applied PCA to perform dimensionality reduction. Data were converted to 4000-dimensional data. K-Means is utilized in this extend.

**Some Clustering algorithms:**

1. **K-Means Clustering**

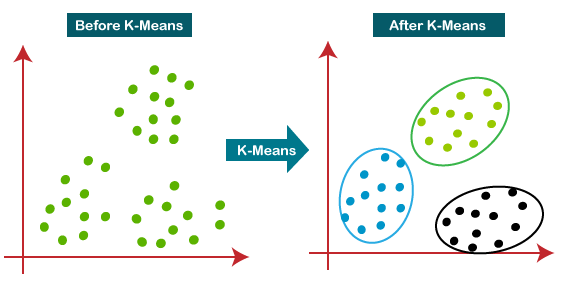
K-means algorithm recognizes k number of centroids and then allocates each data point to the nearest cluster, while keeping the centroids as small as possible.

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids

It halts creating and optimizing clusters when either:

The centroids have stabilized — there is no change in their values because the clustering has been successful.

The characterized number of iterations has been accomplished.



1. **Gaussian Clustering**

Gaussian mixture models can be used to cluster unlabeled data in much the same way as k-means. There are, in any case some of preferences to using Gaussian blend models over k-means.

a Gaussian mixture model includes the blend (i.e. superposition) of multiple Gaussian distributions. Here instead of recognizing clusters by “nearest” centroids, we fit a set of k gaussians to the data. And we estimate gaussian distribution parameters such as mean and Fluctuation for each cluster and weight of a cluster. After learning the parameters for each data point we will calculate the probabilities of it having a place to each of the clusters.

Every distribution is multiplied by a weight ππ(π1+π2+π3=1π1+π2+π3=1) to account for the fact that we do not have an equal number of samples from each category. In other words, we might only have included 1000 people from the red cluster class and 100,000 people from the green cluster class.

**Expectation Maximization**

**Expectation**

The first step, known as the expectation step or EE step, consists of calculating the expectation of the component assignments CkCk for each data point xi∈Xxi∈X given the model parameters πkπk μkμk and σkσk.

**Maximization**

The second step is known as the maximization step or MM step, which consists of maximizing the expectations calculated in the E step with respect to the model parameters. This step consists of updating the values πkπk, μkμk and σkσk.

The entire iterative process repeats until the algorithm converges, giving a maximum likelihood estimate. Intuitively, the algorithm works because knowing the component assignment CkCk for each xixi makes solving for πkπk μkμk and σkσk easy, while knowing πkπk μkμk σkσk makes inferring p(Ck|xi)p(Ck|xi) easy.

The expectation step compares to the last mentioned case whereas the maximization step compares to the previous. Hence, by alternating between which values are accepted fixed, or known, most extreme probability estimates of the non-fixed values can be calculated in an efficient way.

**Algorithm**

* Initialize the mean μkμk, the covariance matrix ΣkΣk and the mixing coefficients πkπk by some random values(or other values).
* Compute the CkCk values for all k.
* Again Estimate all the parameters using the current \ C\_k values.
* Compute log-likelihood function.
* Put some convergence criterion
* In the event that the log-likelihood value converges to some value (or if all the parameters merge to some values) at that point halt, else return to Step 2.

This algorithm as it were ensure that we arrive to a neighborhood ideal point, but it don't ensure that this local optima is also the worldwide. And so, if the algorithm begins from distinctive initialization focuses, in common it lands into different setups.

1. **Agglomerative clustering**

Also known as bottom-up approach or hierarchical agglomerative clustering (HAC). A structure that's more instructive than the unstructured set of clusters returned by flat clustering. This clustering calculation does not require us to prespecify the number of clusters. Bottom-up calculations treat each data as a singleton cluster at the outset and after that successively agglomerates sets of clusters until all clusters have been combined into a single cluster that contains all data given a dataset (d1, d2, d3, ....dN) of size N

# compute the distance matrix

for i=1 to N:

# as the distance matrix is symmetric about

# the primary diagonal so we compute only lower

# part of the primary diagonal

for j=1 to i:

dis\_mat[i][j] = distance[di, dj]

each data point is a singleton cluster

repeat

merge the two cluster having minimum distance

update the distance matrix

until only a single cluster remains

**7. Conclusion:**

1. EDA was used to examine the data.

2. Textual data were converted using

stemming. Data cleansing for the textual

data was done.

3. Textual data were transformed using TF-

IDF.

4. Dimensionality was reduced by using

PCA. The explained variance vs.

components graph was used to choose the

components.

5. Clustering using K-means was used.

6. Using the elbow curve graph and the

Silhouette score, the ideal number of

clusters was discovered.

7. The characteristics of several clusters were

compared.

8. Utilizing a count vectorizer, a content-

based recommendation system was

developed. For any movie name entered, it

suggests 10 other films or TV shows that

are similar

References-

1. MachineL earning Mastery
2. Geeks for Geeks
3. Analytics Vidhya