Clément FORNAGE
Clément EDOUARD
Quentin DENIS
Marin HEROGUER
Romain ACHARD
Pierre-Antoine HIGNARD NAUDEAU

FUNDAMENTALS OF MACHINE LEARNING

Unsupervised Learning Challenge : *Netflix*

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Outline

Introduction

For this project, we focused on the practical application of the methods covered in the Fundamentals of Machine Learning course. To do so, our group chose to analyze the Netflix Titles dataset, which compiles information about all movies and TV shows available on the platform.

The aim of this report is to present, in a structured manner, the different steps undertaken to explore and understand this dataset, as well as to illustrate the application of various unsupervised learning techniques (such as clustering, factorial analyses and movie recommendation system) with the goal of uncovering patterns and segmenting content effectively.

1. Data Cleaning & Feature Engineering

Initial Dataset Description

The initial step of this project involved a raw display of the dataset to understand its structure and quality. The dataset is composed of 8,807 observations and 12 variables describing the content available on Netflix. The majority of the variables are textual or categorial, with the exception of release_year which is numerical.

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	nan	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmmaker Kirsten Johnson stages his death in inventive and comical ways to help them both face the inevitable.
1	s2	TV Show	Blood & Water	nan	Ama Qamata, Khosi Ngema, Gail Mabalane, Thabang Molaba, Dillon Windvogel, Natasha Thahane, Arno Greeff, Xolile Tshabalala, Getmore Sithole, Cindy Mahlangu, Ryle De Morny, Greteli Fincham, Sello Maake Ka-Ncube, Odwa Gwanya, Mekaila Mathys, Sandi Schultz, Duane Williams, Shamilla Miller, Patrick Mofokeng	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town teen sets out to prove whether a private-school swimming star is her sister who was abducted at birth.
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabiha Akkari, Sofia Lesaffre, Salim Kechiouche, Noureddine Farihi, Geert Van Rampelberg, Bakary Diombera	nan	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Action & Adventure	To protect his family from a powerful drug lord, skilled thief Mehdi and his expert team of robbers are pulled into a violent and deadly turf war.
3	s4	TV Show	Jailbirds New Orleans	nan	nan	nan	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Feuds, flirtations and toilet talk go down among the incarcerated women at the Orleans Justice Center in New Orleans on this gritty reality series.
4	s5	TV Show	Kota Factory	nan	Mayur More, Jitendra Kumar, Ranjan Raj, Alam Khan, Ahsaas Channa, Revathi Pillai, Urvi Singh, Arun Kumar	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV Comedies	In a city of coaching centers known to train India's finest collegiate minds, an earnest but unexceptional student and his friends navigate campus life.

Features Choice

At this stage of the project, we decided to keep all variables except for ‘show_id’ because we already have IDs given by the row indexes and ‘data_added’, which we did not feel had a clear relevance to grouping films by similarity.

Missing Values Management Strategy

The analysis of data completeness revealed a significant heterogeneity in the distribution of missing values. We identified two categories of variables:

Variables with a nearly null rate of missing values : ‘rating’ and ‘duration’ columns contain only a few missing values (3 to 4 rows). For simplicity and to minimize impact, these few rows were dropped from the dataset, resulting in a negligible loss of information.

Variables with a high rate of missing values (5% to 30%): ‘director’, ‘cast’ and ‘country’ variables show a missing rate greater than 5%, reaching close to 30%. Deleting the corresponding rows was not feasible, as it would have led to an excessive loss of information for subsequent analyses.

For these categorical variables, we opted for the technique of imputation with a specific modality. Missing values were replaced by ‘*no_feature’s name’*.

This approach is favored for non-numerical data because it allows observations to be retained while transforming the absence of information into an explicit category, thus avoiding biasing the model with imputation by the most frequent value.

Furthermore, a systematic check was performed to identify and address any duplicate entries. No duplicate values were found across the observations, confirming the uniqueness of each movie in the dataset.

2. Exploratory Data Analysis

General Structure

The dataset distinguishes two types of content: ‘movies’ and ‘TV shows’. Movies account for the majority of records, while series make up a smaller proportion. This composition provides a clear structural basis and reflects how information such as duration and production format differ between the two categories. The overall number of observations and the diversity of features make the dataset suitable for exploratory and clustering approaches.

Quantitative & Categorical Variables

Several variables provide structured, interpretable information. The *release_year* column spans a wide range, covering titles from older productions to very recent ones, with a strong concentration after 2010.

The *duration* variable is expressed in two different units depending on the type of content—minutes for movies and seasons for TV shows. After separating the numeric value and its corresponding unit, the data appear consistent, with most movies lasting between 80 and 120 minutes and the majority of series composed of one or two seasons. We can also observe a strong correlation between the *duration* variable and the *type* variable.

The *rating* variable contains audience classification codes such as “TV-MA,” “TV-14,” or “PG-13.” These categorical values describe the intended viewing audience for each title. The codes are heterogeneous but well formatted so, we decided to group them into three categories – kids, teens and adults – in order to simplify the classification and implementation of our model. This process allows them to be used in a better way for grouping or comparative analysis.

The *country* feature provides geographical information, though it sometimes contains multiple entries per title or missing values. Despite this partial incompleteness, it remains informative for identifying broad geographical trends once properly cleaned or aggregated. We also observe within this variable a strong dominance of the United States, but also of India, compared to other countries.

Textual Information & Data Quality

Two columns contain main unstructured text: *listed_in* and *description*. The *listed_in* variable corresponds to the genres associated with each title and multiple genres can be assigned simultaneously, resulting in a large number of unique combinations. This field captures thematic diversity but also introduces high dimensionality.

The *description* field provides short textual summaries, typically composed of 20 to 30 words. These entries are relatively uniform in structure and length, which facilitates future textual processing. Together, these variables represent the qualitative dimension of the dataset and offer complementary perspectives to the categorical information.

We could also describe the variables “director” and “cast”, which contain the names of directors and actors in text format, but with many missing values and unique entries, which will prevent us from using them as criteria for similarity between movies.

3. Dimensionality Reduction & Clustering

Clustering Data Preparation

For all clustering algorithms, we made the following choices:

- We transform categorical and numerical data into a format suitable for machine learning algorithms like KMeans clustering.
- Genres, countries and types are converted to a numerical (binary) format and numerical features are standardized to have similar scales.
- We don't take "director" and "cast" variables because they are too unique for each movie so it does not help a lot for our model.
- The "duration_value" and "duration_unit" variables are also very correlated with the "type" variable and most of the values are very similar, not allowing us to make differences with this criteria. Then, we don't keep these 2 variables.
- Finally, the "description" variable does not fit our model in this case so we do not take it too.

Dimensionality Reduction

A reduction in dimensions to two variables was also achieved using the PCA algorithm in order to simplify data containing too many variables, especially with hot encoding. This technique allows us to improve clustering algorithms. It facilitates the visualisation and understanding of underlying structures in the data. By retaining most of the variance, PCA helps to identify the most explanatory axes and simplify analyses while limiting information loss. In addition, having also tried to apply clustering algorithms without PCA, we quickly realised that the silhouette scores were better with PCA than without it.

K-Means Clustering

For K-Means clustering, we first sought to perform classic clustering to see what results it could give us. Using the silhouette score, we found an optimal k of 3, allowing us to then visualise our data in 2D (PCA) with a different colour for each cluster. The graph shows us clearly distinct clusters, even though they are very close to each other. This clear separation between clusters suggests that PCA effectively captures an underlying structure in the data, which validates the relevance of the variables used prior to clustering.

Nevertheless, when studying the similarity of movies in the same cluster for a single variable, it is difficult to interpret the results, whether for country, genre or rating_category, for example: there is no obvious difference between the clusters based on just one of these variables. It is the combination of

all the variables used that must be evaluated in order to clearly differentiate between the different clusters, even if this remains difficult in the context of K-Means Clustering.

Hierarchical Clustering

In an effort to improve our results, we then implemented a hierarchical clustering algorithm. The “ward” method was chosen in order to minimise variance in each cluster created. Once clustering had been applied, the clustering dendrogram was plotted. The number of clusters was chosen visually using a distance/height defined by us, which allowed us to obtain a consistent number of clusters (max_d = 60 in our case). We were thus able to obtain 3 clusters on which we performed the same analyses as before in order to compare the differences between clusters. However, these results follow a very similar trend to those obtained with KMeans, not allowing us to group movies much more accurately.

TF-IDF Clustering

In an effort to try one last way to improve our clustering, we used the TF-IDF method. TF-IDF (Term Frequency–Inverse Document Frequency) is a weighting method used to transform text into numerical values. In fact, it measures the importance of a word in a document relative to the entire corpus. TF (Term Frequency) indicates how many times a word appears in a document. IDF (Inverse Document Frequency) reduces the weight of words that are very common in all documents (such as ‘the’ or ‘and’). Thus, TF-IDF highlights specific and discriminating words, which are useful for tasks such as clustering or content recommendation.

Furthermore, to use this method effectively, we need to prepare our data so that only the essential information is retained. We then start again from the cleaned dataset before preparing it for clustering. Then, we start by putting all the variables we want to use into a single variable (in the form of a character string), which we will use later when implementing the model with the TF-IDF technique.

The decision was made here not to keep the “type” and "country" variables because they were not adapted to TD-IDF. In addition, it was too unbalanced between “TV Show” and "Movie" values or between "United_States" and other countries values in the dataset, which could skew the scores too heavily in favor of the rarer value (in this case, “TV Show” and other countries). The idea of performing a random sample to obtain 50% for each value was considered, but we realized that too many movies/series could then not be recommended later because they would be deleted. In addition, the "type" variable is strongly correlated with “duration_unit” so we can remove it too.

As for the “director” and “cast” variables, their removal is mainly due to the fact that many of their values only appear once in the dataset, which could cause bias in our model.

For "duration_value" and "release_year", we also don't have to consider these variables as they are numerical and TF-IDF is based on the analysis of textual data. So, 2 movies with close "release_year" or "duration_value" will not be considered close on these variables.

So finally, we have decided to keep the variables *listed_in*, *description* and *rating_category* because we believe they are fairly representative and convey similarities in our search (similar genre, category and description). We can see that here, "description" variable can be considered important as we will analyze each word for each value.

We then apply our model's function to create a matrix with:

- for column names, the main words found within the grouping variable created and describing each film with all the variables
- for row names, simply the indices of the movies in the order of the initial dataset
- for the values, it is the weight that each word has in the “identity” of a movie (the more often the word appears, the lower the score because the word has less impact on the identity of the film, and vice versa)

Then, on this matrix, we now use the Truncated SVD method with 100 components, which is more suited to textual or very large data sets, with many zero values and often derived from TF-IDF. This reduces the dimensionality of the data. Each component is a linear combination of words that captures the maximum variance. This improves performance (fewer dimensions = faster computation for models) and reduces noise (very rare or uninformative words have less impact).

Afterwards, we apply the classic K-Means clustering method to our matrix, calculate the silhouette scores, and choose the optimal k.

The raw TF-IDF matrix is very sparse and highly dimensional (thousands of words). Each movie is very distinct: Euclidean distances are often irrelevant. The result is clusters are quite “fuzzy” so silhouette scores stagnate or fluctuate (sometimes with a clear peak at a certain k). From this and since the curve stabilizes more or less from k = 11 onwards, this value can be taken as the optimal k.

For visualizations, we use t-SNE, which better captures nonlinear and local structures (useful for reduced TF-IDF text). It is often more visually “meaningful” than PCA to visualise data in 2D or 3D while respecting the local proximity between points. The graph shows us clusters that are much less separated, overlapping each other, very dense and large in the center and more disparate and small around the edges. However, by combining the graph with a variable-by-variable analysis as previously performed, the clusters seem to capture more information than with previous models. For example, cluster 4 can be clearly identified as a cluster suitable for kids in terms of genres (Kids' TV, Anime Series, etc.) or rating category (Kids = 422 versus Teens = 36 and Adults = 1). Reading the words in the film descriptions certainly helped refine our clusters, allowing us to better group similar films, particularly niche films (small, heterogeneous clusters). This results in finer segmentation that is more representative of real categories, improving the internal consistency and readability of the majority of clusters.

4. Content-Based Recommendation System

Finally, to take things further, we implemented a movie recommendation system based on the TF-IDF matrix created earlier. We calculated the cosine similarity matrix for the raw matrix and the reduced matrix (Truncated SVD) and we add them together, giving each one a weight that allows us to strike a balance between precision and generalisation. The final “mix” similarity matrix gives scores based on the similarity between two films/series. These scores are based on the word weight vectors created in the previous part with matrices and their cosine similarity.

We then created a function that allows a given film to return a number n of films that can be recommended based on the similarity scores of the cosine similarity matrix mix. After that, we test our recommendation system by displaying recommendations for some films/series chosen by us in order to visualize the consistency of the results obtained by adjusting the value of the factors for the similarity matrix and for the SVD decomposition. The recommendations obtained here seem to us to be well suited to the target films.

This system is not easy to evaluate except by looking at whether the recommendations seem correct to us visually. We could also look at whether the recommendations made remain in the same cluster, which was constructed in the previous section.

Once displayed, we can therefore conclude that there are still very diverse movie recommendations that do not necessarily all belong to the same cluster. This is mainly due to the large central clusters (see previous section) that spill over into all the surrounding clusters. Perfect recommendations are therefore difficult to achieve and depend largely on the choices made in the implementation of our models.

Conclusion

This project allowed us to explore the Netflix Titles dataset through several stages — from data cleaning and feature engineering to the application of dimensionality reduction, clustering, and content-based recommendation methods. The different unsupervised learning techniques implemented (PCA, Truncated SVD, K-Means, Hierarchical Clustering and t-SNE) revealed meaningful patterns and helped segment the catalog into coherent groups. The final recommendation system, based on TF-IDF and cosine similarity, demonstrated promising results in suggesting similar content. In future work, the model could be improved by integrating user behavior data (ratings, watch history), applying deep learning embeddings (Word2Vec or BERT) or implementing hybrid approaches combining content-based and collaborative filtering methods to refine personalization and recommendation quality.