Data Mining: Fundamentals

A.Y. 2024/2025

Group 12

Bruno Barbieri, Noemi Dalmasso, Gaia Federica Francesca Ferrara

The aim of this report is to display an analysis carried out on the IMDb dataset; the analysis has been conducted making use of data mining methodologies. After the data understanding and preparation phase, clustering, classification, regression and pattern mining techniques have been applied.

Contents

1. Data Understanding and Preparation

The dataset *train.csv* contains 16431 titles of different forms of visual entertainment that have been rated on IMDb, an online database of information related to films, television series etc. Each record is described by 23 attributes, both numerical and non-numerical.

1.1 Discrete attributes

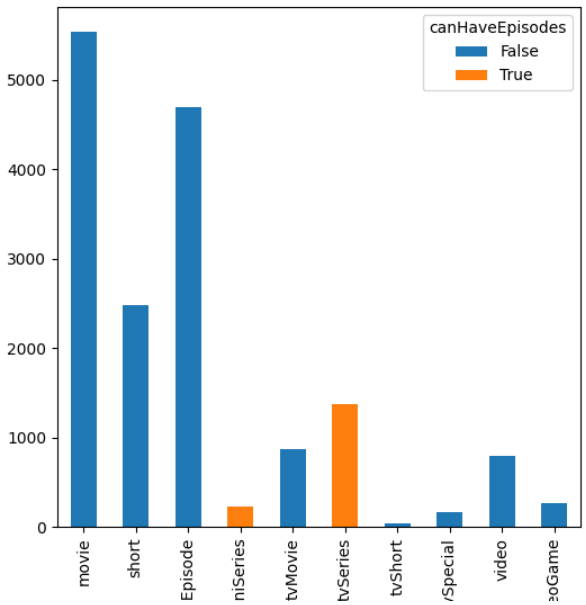
Table 1.1 shows the discrete attributes of the dataset, their types and a brief description of each attribute.

Attribute	**Type**	**Description**
`originalTitle`	Categorical	Title in its original language
`rating`	Ordinal	IMDB title rating class The range is from `(0,1]` to `(9,10]`
`titleType`	Categorical	The format of the title
`canHaveEpisodes`	Binary	Whether or not the title can have episodes `True`: can have episodes; `False`: cannot have episodes
`isRatable`	Binary	Whether or not the title can be rated by users `True`: it can be rated; `False`: cannot be rated
`isAdult`	Binary	Whether or not the title is for adults `0`: non-adult title; `1`: adult title
`countryOfOrigin`	List	The country(ies) where the title was produced
`genres`	List	The genre(s) associated with the title (3 at most)

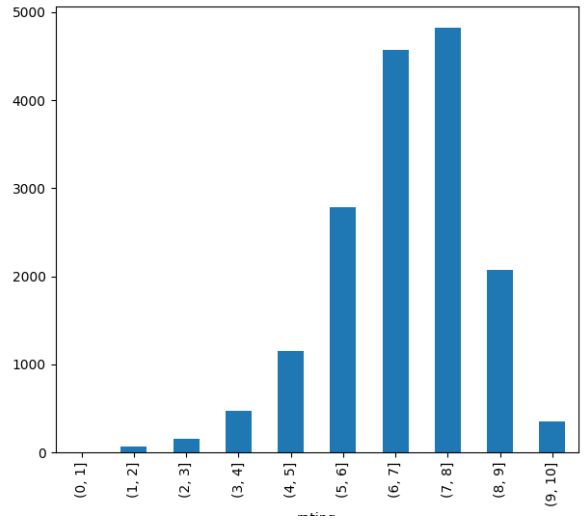
Table 1.1: Description of discrete attributes

1.1.1 Discrete attributes analysis

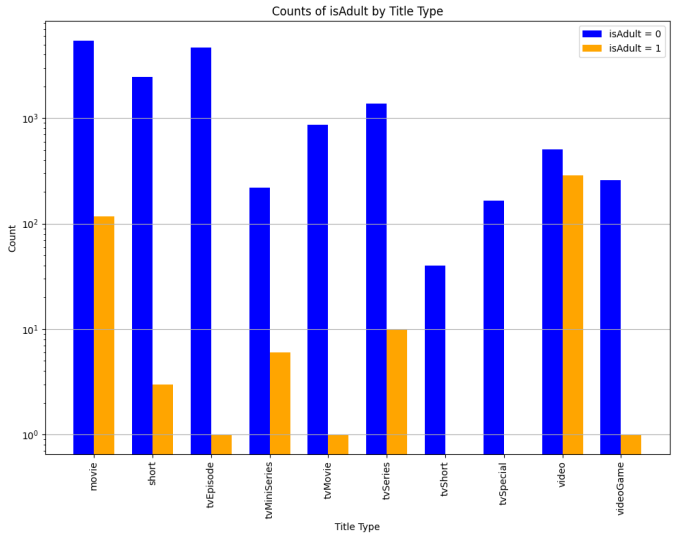
In this paragraph, the most informative discrete attributes of the dataset are examined to provide an overview of their statistics and frequencies.
From figure 1.3a it is observed that the classes of the `titleType` attribute are unbalanced, with *movie* being the most frequent class (5535 records) and *tvShort* the least frequent (40 records). By analyzing the `canHaveEpisodes` attribute within these `titleType` values, it is found that only *tvSeries* and *tvMiniSeries* can have episodes, as expected. As shown in figure 1.3b, the frequency of rating classes is slighly skewed toward higher values, with the most frequent rating class being (7, 8], which is the rating of 4822 titles. Another important aspect is that all 16341 titles are ratable and the vast majority of them (16005) are non-adults contents, as shown in figure 1.1c Finally, as indicated in figure 1.1d, an analysis of the `genres` variable across different `titleType` values reveals that *Drama* and *Comedy* are the most common genres, as they appear in the top 3 genres of nearly every `titleType` category.



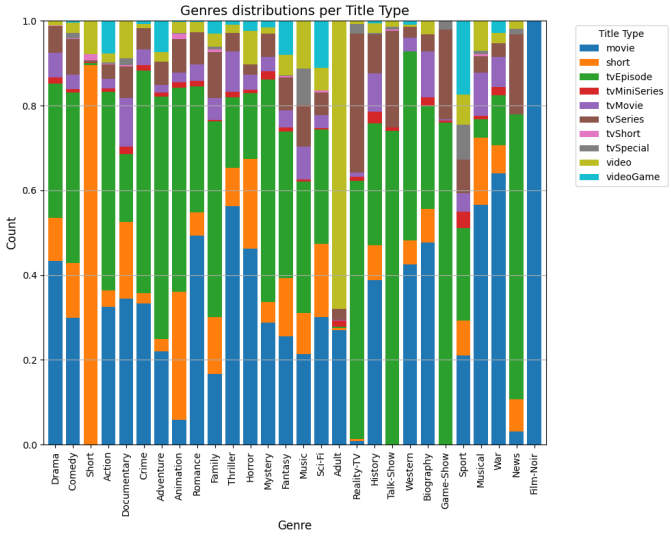
(a) Counting of the title types frequencies



(b) Counting of ratings frequencies



(c) Counting of the adult and non-adult per type



(d) types per genre

Figure 1.1: Bar chart of the discrete attributes.

1.1.2 Variable elimination

The following categorical variables were removed from the dataset:

- `originalTitle` was removed because it is not relevant for the analysis;
- `isRatable` variable was removed because all the titles in the dataset are ratable;

Additionally, the `isAdult` feature is highly correlated with the presence or absence of *Adult* as a genre (16 records differ in the train set, 1 in the test set), so the two were merged with a logical OR operation.

1.1.3 Feature transformations

The attribute `rating` was converted into an ordinal variable by taking the upper bound of each rating interval's string representation. This approach was chosen because the minimum rating is 1, meaning the lowest interval corresponds only to ratings of 1. For consistency, the same transformation was applied to all other intervals.

The attribute `genre` was represented through one-hot encoding. This generated 28 new attributes. Rows with no genres were assigned a vector of all zeros, indicating the absence of any genres.

The attribute `countryOfOrigin` was represented by grouping the countries by continent. The following variables have been created:

- `countryOfOrigin_AF` (Africa);
- `countryOfOrigin_AS` (Asia);
- `countryOfOrigin_EU` (Europe);
- `countryOfOrigin_NA` (North America);
- `countryOfOrigin_SA` (South America);
- `countryOfOrigin_OC` (Oceania);
- `countryOfOrigin_UNK` (Unknown country);
- textttcountryOfOrigin_freq_enc (Frequency encoding of the original list).

Each of the first six features provides the number of countries in the corresponding continent. `countryOfOrigin_UNK` is used to represent the strings that are not chategorized as being part of a continent, by again counting the strings that are not recognized. The feature `countryOfOrigin_freq_enc` provides the frequency encoding of the original list as a whole.

In summary, the original feature is represented by the seven attributes regarding the continents, plus 1 representing the frequency encoding. This representation was chosen as it allows to keep a lot of the original information, while limiting the number of new features.

1.2 Continuous attributes

Table 1.2 shows the continuous attributes of the dataset, a brief description of each attribute and their type.

1.2.1 Variable elimination, merge and creation

The `worstRating` and `bestRating` attributes were removed from the dataset because they assume the same values for all records of the dataset (1 and 10 respectively).

Attribute	Type	Description
worstRating	Float	Worst title rating
bestRating	Float	Best title rating
runtimeMinutes	Integer	Runtime of the title expressed in minutes
startYear	Integer	Release/start year of a title
endYear	Integer	TV Series end year
awardWins	Integer	Number of awards the title won
numVotes	Integer	Number of votes the title has received
totalImages	Integer	Number of Images on the IMDb title page
totalVideos	Integer	Number of Videos on the IMDb title page
totalCredits	Integer	Number of Credits for the title
criticReviewsTotal	Integer	Total Number of Critic Reviews
awardNominationsExcludeWins	Integer	Number of award nominations excluding wins
numRegions	Integer	The regions number for this version of the title
userReviewsTotal	Integer	Number of User Reviews
ratingCount	Integer	The total number of user ratings for the title

Table 1.2: Description of continuous attributes

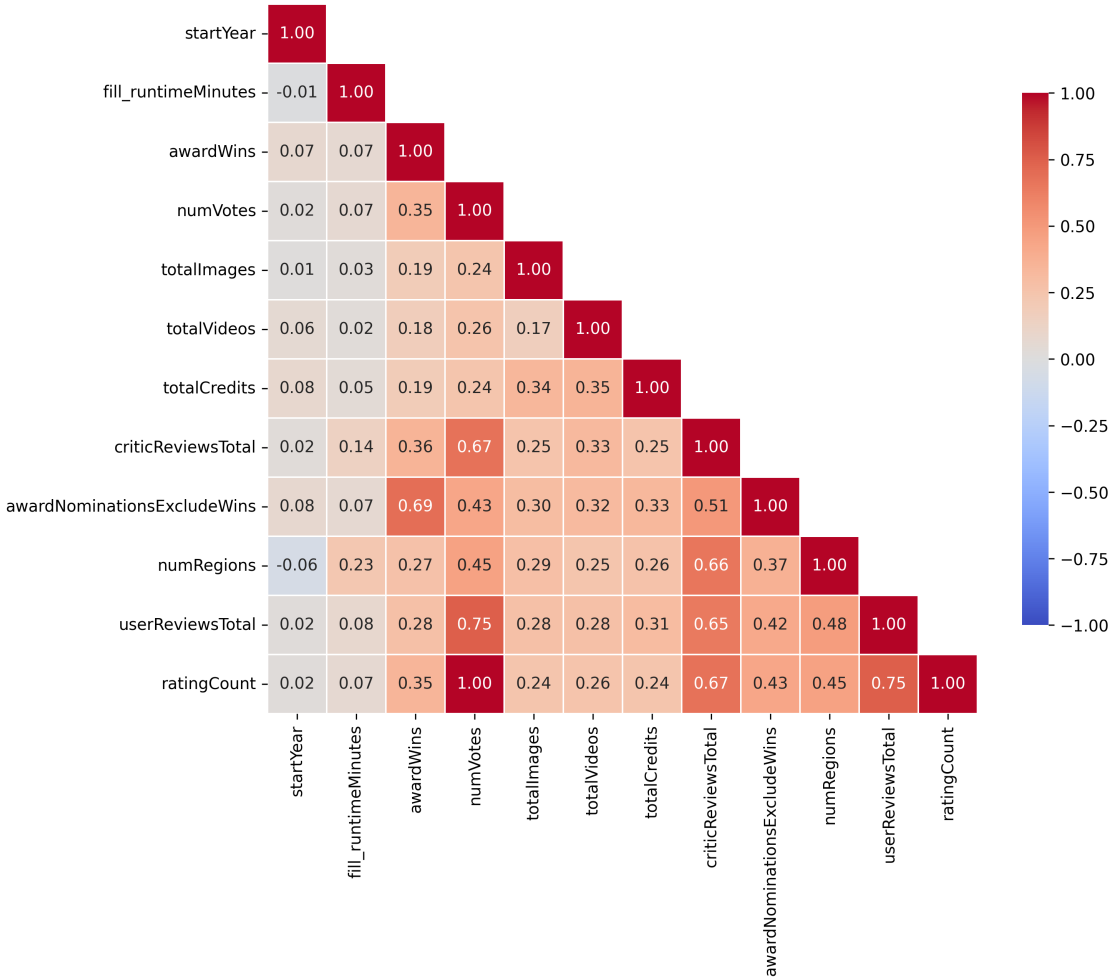


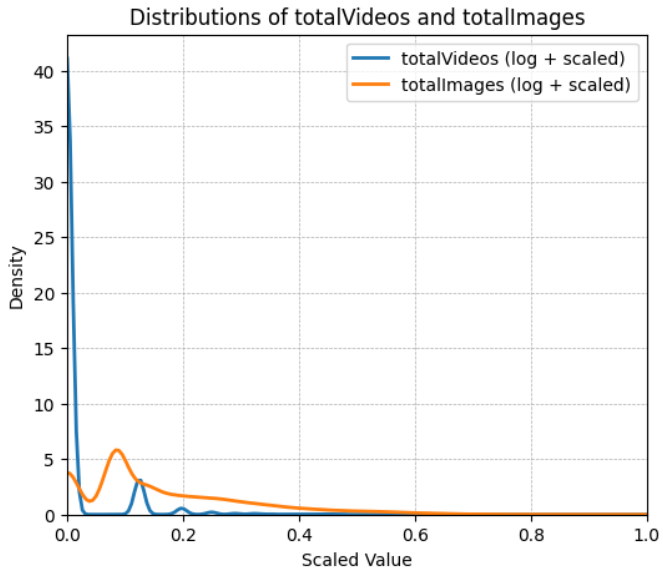
Figure 1.2: Correlation matrix of the numerical features

The plot in figure 1.2 is a Pearson's correlation matrix that takes into account the continuous variables of the dataset.

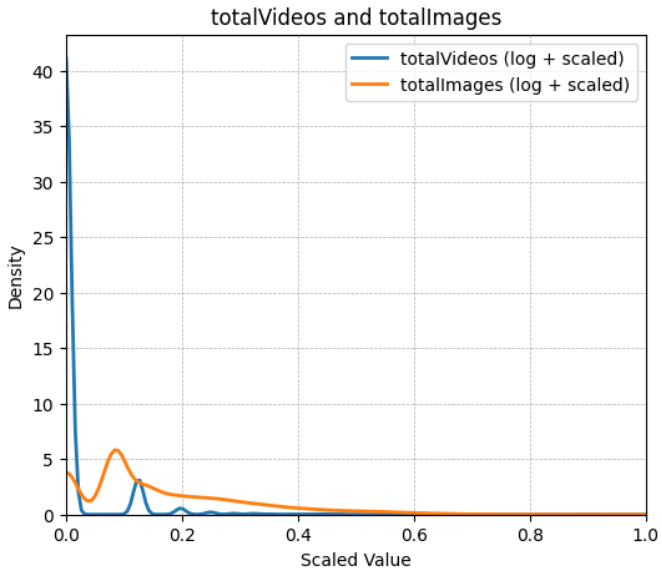
The correlation matrix shows that `ratingCount` and `numVotes` are perfectly correlated; for their redundancy, `ratingCount` was discarded.

The `awardWins` and `awardNominationsExcludeWins` attributes were combined into a single attribute, `totalNominations` due to their strong semantic similarity and high correlation (0.95). The new attribute represents the sum of the two original variables. This transformation also helps mitigate the impact of their heavy left skew (shown in figure 1.3a), resulting in a more meaningful and interpretable feature.

Similarly, the `totalVideos` and `totalImages` attributes were combined into a single attribute, `totalMedia`, representing the total number of media items associated with a title. Although the original attributes are not highly correlated, both exhibit skewed distributions (as shown in figure 1.3b), the first in particular. Due to this, and their similar semantic meaning, they were merged to form a more consolidated and interpretable feature.



(a) Logarithmic distribution of the `totalNominations` attribute



(b) Logarithmic distribution of the `totalNominations` attribute

Figure 1.3: Distribution of the `totalNominations` and `totalMedia` attributes

1.3 Data Quality

In this phase, a proper evaluation of the observed data was conducted in preparation for the analysis. Once having checked that there are no duplicates and no incomplete rows in the dataset, attention was given at identifying missing values and outliers within the columns.

1.3.1 Syntactic Inconsistencies

1.3.2 Missing Values

Once having solved the above-mentioned inconsistency, the resulting total amount of the missing values are the following, also represented in percentages for a better understanding:

- `endYear`: it is the feature with the highest number of NaN values (15617; about 95%). Although the feature is only relevant for *TVSeries* and *TVMiniSeries* titles, it still had approximately 50% missing values within those categories, limiting its usefulness even in the appropriate context. For this reason, the feature was discarded;
- `runtimeMinutes`: it has 4852 missing values (29.5%). Two imputation strategies were used: one leverages the titleType feature to guide the imputation, while the other imputes the missing values independently of any other features. Depending on the task at hand, one or the other was chosen;
- `awardWins`: this feature has 2618 NaN values (about 16%). Since the mode associated with this variable is 0, it has been decided to substitute the missing values with 0;
- `genres`: it has 382 missing values (2.3%). Having dealt this variable with a multi-label one-hot encoding process (as will be described in the *Variable Transformation* section), a vector of all zeros is assigned to record with missing genres values.

1.3.3 Outliers detection and variable Transformation

For the other attributes **CONTINUARE..... VALORI OUTLIERS SU TRAIN IN %: 86.7 , 90.2 , 87.8 VALORI OUTLIERS SU DF_PP IN %: 88.7 , 90.2 , 87.8** Even though they are not proper outliers, the records that had "Videogame" as value of the attribute `titleType` were removed because of the fundamentally different titletype compared to the other title types of the dataset. In addition, their value of `runtimeMinutes` seemed erroneous since they have an undefined or irrelevant runtime.
As a first step in the variable transformation process, the `countryOfOrigin` and `genres` variables (datatypes: strings) were converted into lists of strings to facilitate further analysis. This transformation was necessary because some records contain multiple genres or countries as values for these variables. After that, multi-label one-hot encoding was applied to the `genres` column; each unique genre was represented as a binary feature, allowing records that belong to multiple genres simultaneously to maintain this information. A similar approach was taken for the `countryOfOrigin` attribute; however, instead of creating a separate feature for each unique country (as there were many of them), countries were grouped by continent. The following variables have been created:

- `countryOfOrigin_NA` (North America);
- `countryOfOrigin_SA` (South America);
- `countryOfOrigin_AF` (Africa);
- `countryOfOrigin_AS` (Asia);
- `countryOfOrigin_EU` (Europe);
- `countryOfOrigin_OC` (Oceania);
- `countryOfOrigin_UNK` (Unknown continent).

This transformation was implemented using `pycountry_convert`, a Python library that converts between different country and continent codes and names. Its function `country_alpha2_to_continent_code()` was used to process the lists of strings of the `countryOfOrigin` variable; the function takes a country code in the ISO 3166-1 alpha-2 format (e.g., "US", "FR", "IN") and returns the corresponding continent code. If the country code is invalid or there are obsolete country names, the `countryOfOrigin_UNK` variable gets a value according to the number of unknown countries for that record. For each record, these new variables contain counts representing the number of countries from each continent. In addition, `countryOfOrigin_freq_enc` was created to capture frequency-based information. Unlike the continent-based variables that count individual countries, this variable represents how frequently a specific combination of countries appears across the entire dataset.
Furthermore, it has been decided to extract the ceiling value for each entry in the `rating` column, in order to use it as an integer for further analysis.
Looking at the already existing features it has also been decided to aggregate some of them in order to create more stable and meaningful data. In particular, `awardWins` and `awardNominationsExcludeWins` were combined into a new variable called `totalNominations` to take into account which records have received a nominations, independently from the fact that they then won or not. Two other variables, i.e. `totalImages` and `totalVideos` were aggregated into `totalMedia` to sum the number of multimedia elements of each record.

As for the continuous attributes, it was observed that they required a stronger transformation due to their highly positively skewed distributions. Specifically, when required by the data mining method, a log-transformation was applied to all the numeric attributes, since their skewness was highly greater than 1. Following the log-transformation, standard normalization techniques - `MinMaxScaler` and `StandardScaler` - have then been applied (when scaling was necessary); respectively to scale each feature to a given range and to standardize features by removing the mean and scaling to unit variance. The decision to apply one or the other was again made based on the specific requirements of each data mining technique, and so will be specified accordingly in each section. **CONTROLLARE SE IN OGNI SEZIONE C'E' SPECIFICA SULLE DUE NORMALIZZAZIONI USATE!!!**.

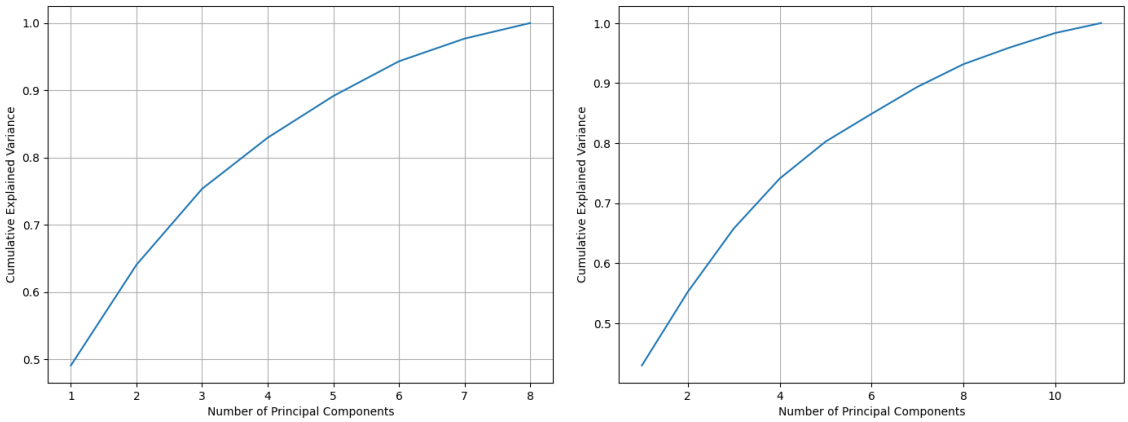
2. Clustering

This chapter of the report aims at illustrating the clustering analysis performed on the dataset at hand. The employed clustering techniques are K-means (Centroid-based), DBSCAN (density-based) and hierarchical clustering.

The analysis conducted using these methods focused only exclusively on the dataset's numerical attributes, which were appropriately log-transformed (as mentioned in the *Variable Transformation* section) and normalized using `MinMaxScaler`. For the K-means algorithm, `totalNominations` and `totalMedia` were excluded due to their high proportion of zero values, which negatively affected cluster formation.

In addition, an attempt was made to incorporate categorical variables to the analysis with the K-means algorithm by converting them into binary attributes and constructing a mixed-distances matrix. Distances were then calculated using the Euclidean distance for numerical (log-tranformed and scaled) features and the Jaccard similarity for binary ones. However, this approach was computationally expensive and did not lead to any improvement in the results.

RIVEDERE PARTE PCA Principal Component Analysis (PCA) was applied to the preprocessed data just for clusters visualization purposes. Analysis of the numerical attributes reveals that 4 principal components are optimal when excluding variables with many zero values, while 5 components are needed when including all variables. These numbers of components capture the maximum meaningful variance, as shown by the point in the plots where where the line starts to flatten, indicating that adding more components doesn't increase explained variance significantly. The plots in figure 2.1 show the differences between these two approaches. **IN REALTA' NON SI VEDE TROPPO LA DIFFERENZA, QUINDI MAGARI NON FARE PCA MA SOLO VISUALIZZAZIONE CON ISTOGRAMMI??? oppure semplicemente tenere solo uno dei due plot?**



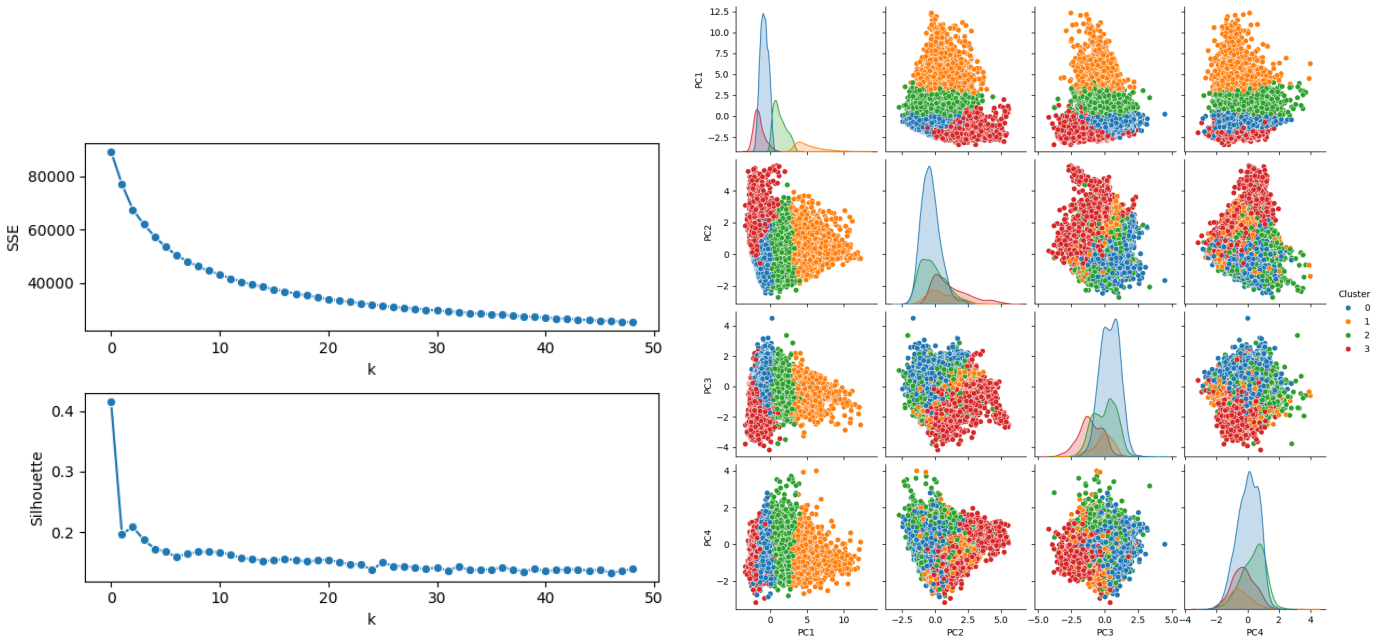
(a) PCA excluding variables

(b) PCA with all variables

Figure 2.1: Principal Component Analysis

2.1 K-means

To identify the optimal number of clusters, both the SSE and Silhouette scores were computed. The goal was to find a configuration that minimizes the SSE while maintaining a robust Silhouette score and a proper k. The plots in figure 2.2a demonstrate that k = ??? provides the optimal balance between these metrics. Choosing k = ??? returns a SSE score of ??? and Silhouette score of ???. **VALORI TROPPO ALTI, VEDERE SE CAMBIANO CAMBIANDO SCALING E/O TOGLIENDO OUTLIER**
The cluster results are presented in figure 2.2b.



(a) SSE and Silhouette scores

(b) K-means Visualization

Figure 2.2: K-means clustering analysis

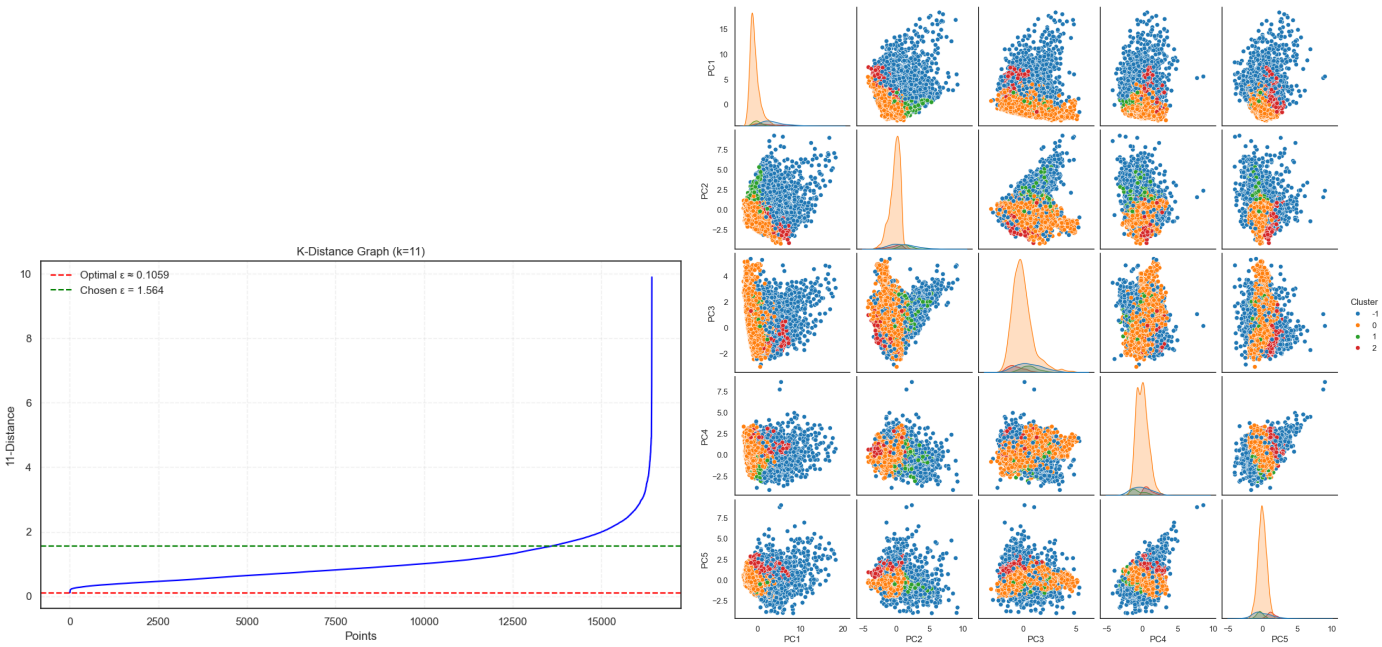
The distribution of data points across the four clusters is as follows (shown in percentage of data points per cluster):

2.2 DBSCAN

To determine the optimal DBSCAN parameters, the k^{th} *nearest neighbors* method was used: this allows to identify *eps* (the maximum distance between two points for them to be considered neighbors) given the value of *Minpts* (minimum number of points in a neighborhood for a point to be considered a core point). Initially, *Minpts* was set to 22, following the rule of setting it above twice the number of dimensions. However, due to the dataset's unbalanced nature and the sparsity of high-dimensional data, reducing *Minpts* to 11 allowed the formation of smaller clusters while preventing the risk of detecting only one dominant cluster and classifying many minority groups as noise instead of distinct clusters. To determine *eps*, the k^{th} nearest neighbors plot with k = 11 was analyzed (figure 2.3a). While the "knee" point suggested an eps of around 0.1, this value would have resulted in excessive noise and a single dominant cluster. To address this, eps was set to 1.564, allowing for meaningful connectivity while preserving the detection of smaller clusters without merging them

into a single entity. The algorithm identified 4 groups in the dataset, including one representing noise (1,753 points). The largest cluster contains 13,198 points, while the smaller clusters consist of 733 and 747 points, respectively.The results are shown in figure 2.3b

To conclude, by adjusting *eps* and *Minpts* appropriately, the clustering results achieved a Silhouette score of 0.139 **(SIL CONTANDO OUTLIERS)**, indicating little improved cluster separation and reduced noise, which is considered good enough for an unbalanced, high-dimensional dataset.



(a) k^{th} nearest neighbors

(b) DBSCAN Visualization

Figure 2.3: DBSCAN clustering analysis

2.3 Hierarchical clustering

2.4 General considerations

3. Classification

Classification was performed on the available training set using three different algorithms: K-NN (*K-Nearest Neighbours*), Naïve Bayes and Decision Trees. For K-NN **and Naïve Bayes**, a portion of the training set (referred to as the validation set) was used to select the best hyperparameters **each** model. The features used in K-NN and Naïve Bayes were normalized, as these models are sensitive to unscaled values. In particular, a log-transformation and **SCRIVERE SE StandardScaler O MINMAX** were applied to data. After training, the models were evaluated on the test set using standard performance metrics. The target variables chosen for this task are 2: `titleType`, and `has_LowEngagement`. These will be discussed in more detail in the corresponding sections below.

3.1 Binary classification

The binary target variable used in this task, `has_LowEngagement`, was specifically defined for this purpose. It identifies records where the `numVotes` attribute is less than 100.

K-NN

Naïve Bayes

Decision Trees

Table 3.1: Classification report for binary classification (`has_LowEngagement`)

Class	Precision	Recall	F1-score	Support
Low engagement	0.86	0.91	0.88	3686
High engagement	0.78	0.66	0.72	1698
Macro avg	0.82	0.79	0.80	5384
Weighted avg	0.83	0.83	0.83	5384
Accuracy			0.83	5384

3.2 Multiclass classification

Among the multiclass features of the training set, `titleType` was chosen as the target variable for this task due to its relevance in the dataset.
An important aspect to highlight is the usage of `fill_runtimeMinutes_notitleType` as one of the variables to train the models. This feature was created to impute the missing values of the original `runtimeMinutes` variable, but without using the median value according to the titleType. Instead, the missing values were

imputed using the help of two variables: `canHaveEpisodes` and `is_Short` (as one of the resulting variables of the multi-label one-hot encoding process of the `genres` attribute). In particular, **SCRIVERE COME E' STATA IMPUTATA NO_TT - con canhaveepisodes e is_short preso dai generi**. This approach prevents a significant error, as it would be methodologically incorrect to use `titleType`-based imputation for an attribute when `titleType` itself is the target variable to predict.

K-NN

Naïve Bayes

Decision Trees

Table 3.2: Classification report for multiclass classification (`titleType`)

Class	Precision	Recall	F1-score	Support
movie	0.85	0.88	0.87	1877
short	0.92	0.94	0.93	766
tvEpisode	0.89	0.92	0.90	1599
tvMiniSeries	0.51	0.35	0.41	81
tvMovie	0.36	0.29	0.32	299
tvSeries	0.89	0.94	0.91	447
tvShort	0.00	0.00	0.00	16
tvSpecial	0.32	0.12	0.18	49
video	0.55	0.46	0.50	250
Macro avg	0.59	0.54	0.56	5384
Weighted avg	0.82	0.84	0.83	5384
Accuracy			0.84	5384

3.3 General considerations

4. Regression

4.1 General considerations

5. Pattern Mining

The pattern mining technique chosen for this task was Apriori. To perform this task, continuous attributes were discretized according to their distributions. The objective of this process was creating bins that were both meaningful and balanced in terms of number of attributes for each bin. Among all the available attributes, the ones chosen for the pattern mining task are the following (with their corresponding binning): **AGGIUSTARE QUANDO VERRANNO GESTITI OUTLIER e quando avremo var definitive - cambiare nome var runtimebruno**

- `runTimeMinutes_Bruno`: VeryLowRT (0-25), LowRT (26-60), MediumRT (61-120), HighRT (121-180), VeryHighRT (181-3000)
- `numVotes`: VeryLowV (5-15), LowV (16-50), MediumV (51-150), HighV (151-997), VeryHighV (1001-966565)
- `rating`: VeryLowR (1-3), LowR (4-6), MediumR (7), HighR (8), VeryHighR (9-10)
- `userReviewsTotal`: NoUR (0), FewUR (1-2), ManyUR (4-13), VeryManyUR (31-149)
- `countryOfOrigin_EU, _NA, _OC, _AS, _AF, _SA`: not_from_[continent name], is_from_[continent name]

In addition, `titleType` was **come l'abbiamo gestita per PM? non c'è one hot encoding quindi l'abbiamo usata così com'è**. The data on which this task was performed was not normalized.

5.1 Extraction of frequent patterns

5.2 Extraction of rules

5.3 Exploiting rules for target prediction