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

1.1 Data Semantics

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. All the variables of the dataset are introduced and explained in Table 1.1 and Table 1.2.

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]
worstRating	Ordinal	Worst title rating
bestRating	Ordinal	Best title rating
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	Categorical	The country(ies) where the title was produced
genres	Categorical	The genre(s) associated with the title

Table 1.1: Description of discrete attributes

1.2 Distribution of the variables and statistics

This section will give an overview about the distribution of variables that has been carried on to understand patterns, detect meaningful statistics and assess their relevance to the project.

1.2.1 Discrete attributes

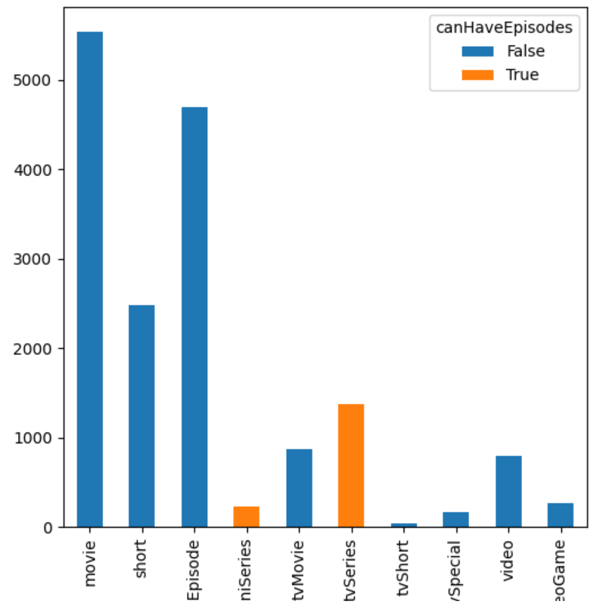
DA CAMBIARE IN BASE AI GRAFICI CHE DECIDIAMO DI TENERE In this paragraph, the most informative discrete attributes of the dataset are examined to provide an overview of their statistics and frequencies.

From figure 1.1a 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

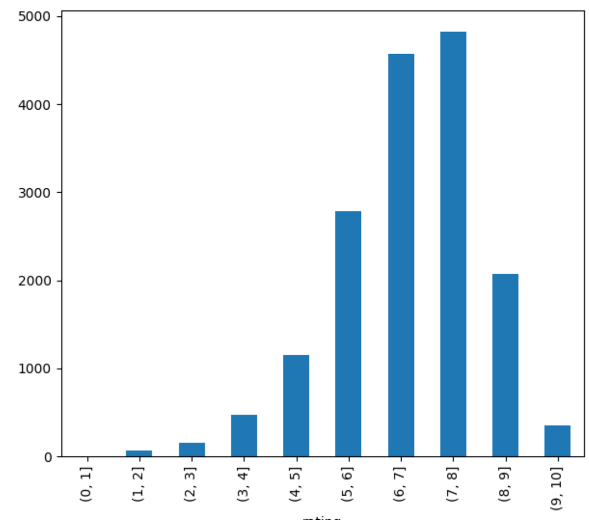
Attribute	Type	Description
runtimeMinutes	Numeric	Runtime of the title expressed in minutes
startYear	Interval	Release/start year of a title
endYear	Interval	TV Series end year
awardWins	Numeric	Number of awards the title won
numVotes	Numeric	Number of votes the title has received
totalImages	Numeric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

Table 1.2: Description of continuous attributes

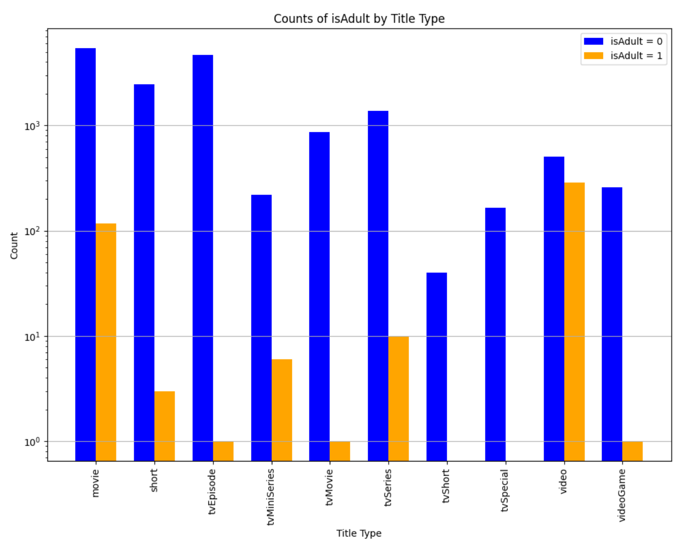
`canHaveEpisodes` attribute within these `titleType` values, it is found that only *tvSeries* and *tvMiniSeries* can have episodes, as expected. As shown in figure 1.1b, the frequency of rating classes is slighly skewed toward higher values, with the most frequent rating class being (7, 8], which is 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.



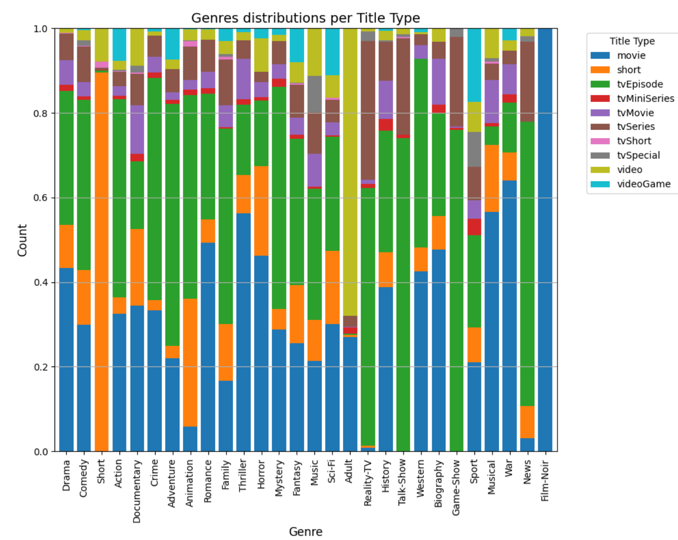
(a) Counting of the title types frequencies



(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.

1.2.2 Continuous attributes

ANCORA DA FARE!!! plot + descrizione ...content...content...content...content...content...content...

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

In the exploration of the dataset it has been noticed that `awardWins` was the only feature having missing values marked with `NaN`. However, there were missing values also in other columns (`endYear`, `runtimeMinutes` and `genres`), but they were indicated with the string "\N". To avoid this inconsistency that might have caused problems during data preparation, these values have been replaced with `NaN`. By doing so, any cell in the `endYear`, `runtimeMinutes` and `genres` columns that previously contained the string "\N" is now considered a proper missing value, detectable and manageable using Pandas' functions.

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%). For this reason and due to the lack of reliable imputation methods, the entire feature has been removed;
- `runtimeMinutes`: it has 4852 missing values (29.5%) that have been handled in two different ways according to the data mining task in hand. More details are provided in the specific sections **SPECIFICARE NEI PARAGRAFI DOVE ABBIAMO USATO _Bruno e dove _notitletype**;
- `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

While examining the dataset, it became apparent that some attributes have outliers. The important aspect to highlight is that since `awardWins`, `totalVideos` and `awardNominationsExcludeWins` have many values as 0 (respectively 11971, 14821, 14427), these might be considered variables with many outliers (as seen in Figure **METTERE GRAFICO che rappresenti in qualche modo il result di DETECT_OULIERS_MULTI_ATTRIBUTE in data_quality noemi**) but they actually have less outliers compared to the other variables. 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.

1.4 Variable Transformation

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 (such as ”NA” for North America, ”EU” for Europe, ”AS” for Asia etc.). 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. **DECIDERE SE SPECIFICARE EFFETTIVAMENTE IN QUALE DM METHOD L'ABBIAMO APPLICATA O SE RIMANERE VAGHI**. 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.

1.5 Pairwise correlations and elimination of variables

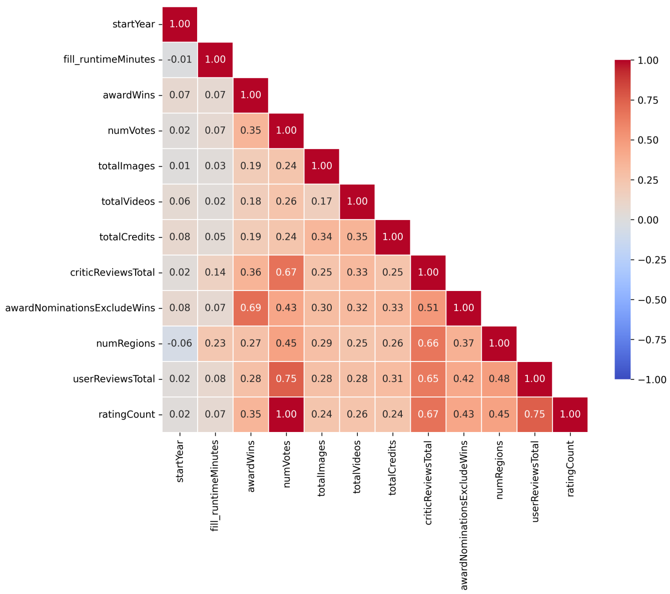


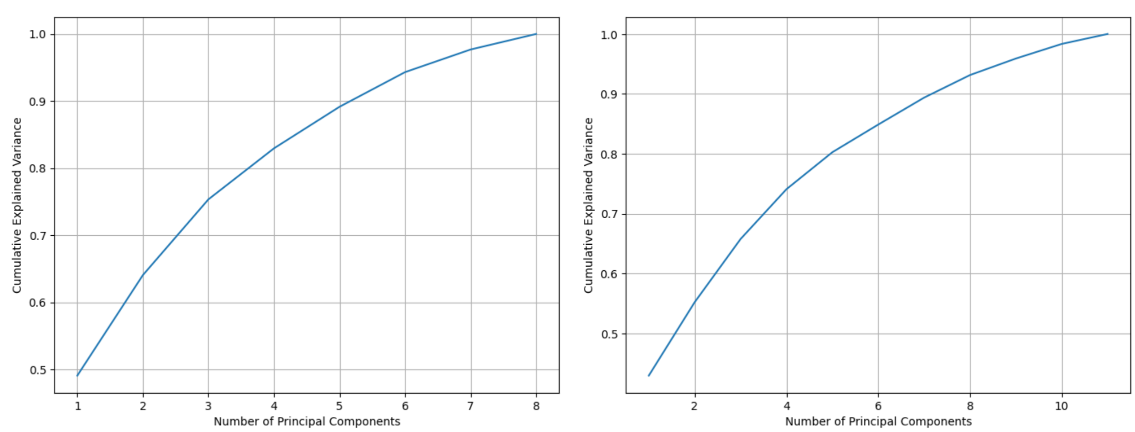
Figure 1.2: Correlation matrix

The plot in figure 1.2 is a Pearson's correlation matrix that takes into account the continuous variables of the dataset. For what can be observed, `numVotes` and `ratingCount` have a perfect positive correlation, making it redundant to keep both of them. For this reason it has been decided to drop `ratingCount`. Another variable that was removed, but for different reasons, is `endYear`; as mentioned in the *Missing Values* subsection, this variable was discarded due to the lack of reliable imputation methods for its numerous missing values.

On the other hand, regarding categorial attributes, after having analized all of them, `bestRating`, `worstRating`, and `isRatable` were discarded because they were found to have, as unique values, respectively 10, 1 and True, resulting in variables with a limited contribution based on their distributions.

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 `StandardScaler`. For the K-means algorithm, `awardWins`, `awardNominationsExcludeWins`, and `totalCredits` 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.
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. **SE EFFETTIVAMENTE CAMBI-**



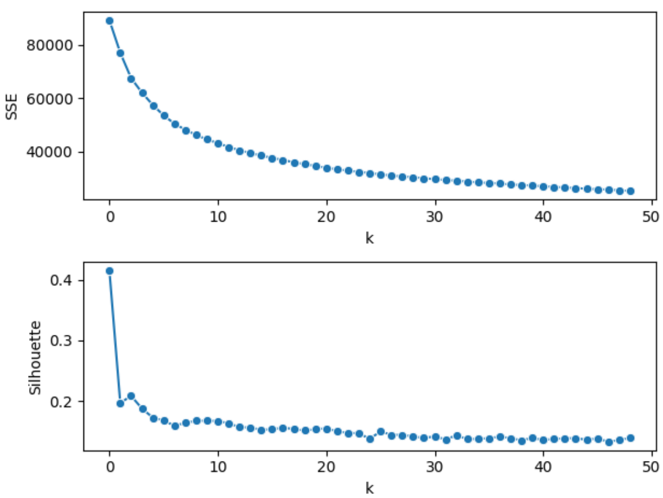
(a) PCA excluding variables (b) PCA with all variables

Figure 2.1: Principal Component Analysis

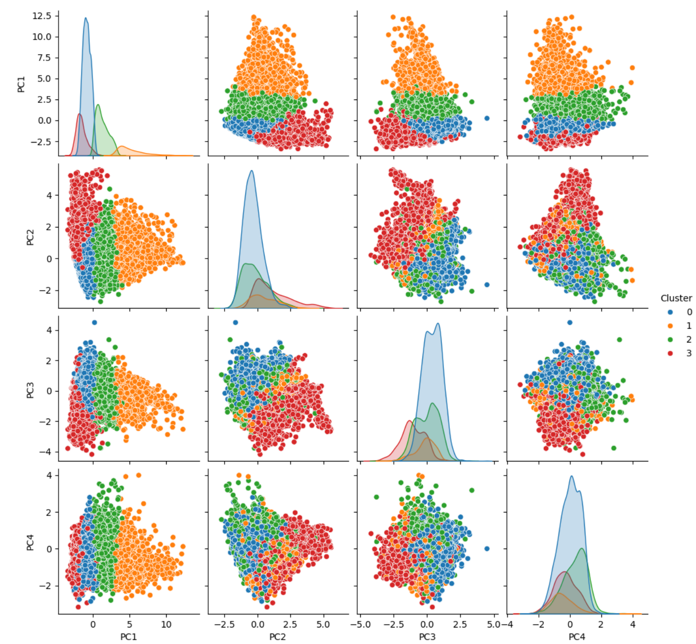
AMO NAN ENDYEAR CON IL RISPETTIVO STARTYEAR GIUSTIFICARE IL PERCHE' NON L'ABBIAMO POI INCLUSA COME VAR NELLA CLUSTERING ANALYSIS

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. The plots in figure 2.2a demonstrate that $k = 4$ provides the optimal balance between these metrics. Choosing $k = 4$ returns a SSE score of 67496 and Silhouette score of 0.21.
The cluster results are presented in figure 2.2b.



(a) SSE and Silhouette scores



(b) K-means Visualization

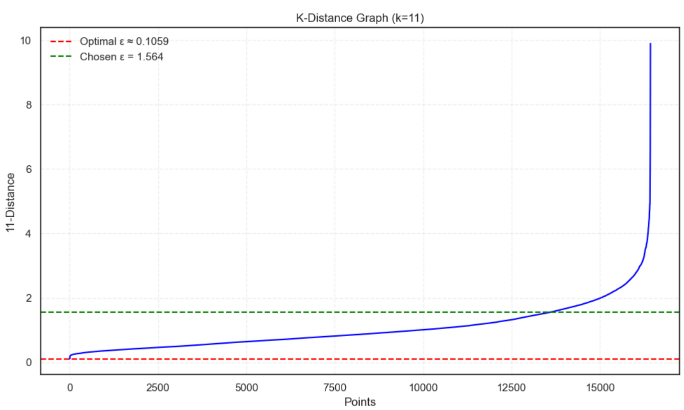
Figure 2.2: K-means clustering analysis

The distribution of data points across the four clusters is as follows (shown in percentage of data points per cluster): Red (0): 51.68%, Blue (1): 7.42%, Green (2): 24.22%, Orange (3): 16.68%. The clusters are not as well-separated in most Principal Component combinations as they are with PC1. In fact, in the other combinations, the clusters tend to overlap and their boundaries are not always clearly distinct. This might be an indication that the true clusters have irregular shapes or different densities, resulting in boundaries between them being not clearly defined.

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. 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 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 **SCRIVERE COME E' STATA IMPUTATA NO_TT**. This approach prevents a significant error, as it would be methodologically incorrect to use `titleType`-based imputation when `titleType` itself is the target variable to predict.

3.1.1 K-NN

3.1.2 Naïve Bayes

3.1.3 Decision Trees

3.2 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.

3.2.1 K-NN

3.2.2 Naïve Bayes

3.2.3 Decision Trees

4. Regression

5. Pattern Mining

5.1 Extraction of frequent patterns

5.2 Extraction of rules

5.3 Exploiting rules for target prediction