Data and Information Quality

## ID Project: 3

## Dataset: 1

Luca Mattana – ID Person: 10104067

# Introduction and setup choices

This report outlines the *data preparation* steps undertaken with focus on profiling, assessing, and cleaning the assigned dataset 1: *Movies*. the project pipeline included detailed phases of data quality assessment, data profiling, data wrangling and various cleaning operations such as data transformation, error correction, and deduplication.

The original dataset is composed of 9999 records, 9 columns.

|  |
| --- |
| # Column Non-Null Count Dtype  --- ------ -------------- -----  0 MOVIES 9999 non-null object  1 YEAR 9355 non-null object  2 GENRE 9919 non-null object  3 RATING 8179 non-null float64  4 ONE-LINE 9999 non-null object  5 STARS 9999 non-null object  6 VOTES 8179 non-null object  7 RunTime 7041 non-null float64  8 Gross 460 non-null object |

Considering the original attribute “MOVIES” (titles) there are 6817 unique values and no null values.

Some attribute like YEAR and GROSS contain not homogeneous values (digits + letters + special chars) and they required some actions during data preparation.

For some attributes I have decided to perform a *split*. For example, I have made a split on the original YEAR attribute to manage values containing a range of years like (2010-2020) adding a column with the last year of the range. A split is performed for attribute STARS too to separate the Directors from the list of Stars, originally assembled under the same column.

The *null* values that I can count also with the table above are not a real image of all the null values present in the dataset: some values of the dataset are populated only with’\n’ that can be considered as null equivalent and for the attribute ONE-LINE there is a considerable count of value ‘Add a Plot’ that is equivalent to not a value too. During the data preparation these exceptions are managed and they are explained in detail in the commented code and later in this report.

For the *deduplication* I have decided to not reduce to one title for 2 main reasons

* there could be ad here are for sure movies with same title.
* some episodes of series of TV show with the same title have a lot of populated attributes like the plot, the rating and votes, the different stars and different directors for each episode or season.

To intercept what to deduplicate or not based on these considerations, I have performed multiple checks with *record linkage* with *blocking*. More details in the code and in the specific section *Deduplication* of this report.

# Pipeline implementation

## Data Quality Assessment

The data quality assessment focused on evaluating key dimensions on the original dataset before eany operation:

#### Completeness

#### Accuracy

The accuracy it was evaluated for the original attribute GENRE to check if there were some misspelled word or duplicated genre for each records

#### Consistency

The consistency it was evaluated considering the rule:

*if votes are not null then rate must be not null and viceversa*

All records are consistent on the rule VOTES 🡨🡪 RATING

#### Uniqueness

The uniqueness it was evaluated for the original attribute MOVIES

#### Distinctness

The distinctness for attribute MOVIES is the same of uniqueness cause no null value.  
It is possible to calculate it for any attribute. For example, before any action on dataset I have evaluated it for the attribute GENRE

## Data Profiling

Data profiling was conducted to understand the structure, content, and overall quality of the dataset.

* Schema and Attribute Analysis
* Summary Statistics
* Missing Data Overview and visualization (with *missingno* library)

Key Findings:

* Some columns contained a high proportion of missing values.
* Data formats were inconsistent in certain fields, requiring standardization.

## Data Wrangling

During the data wrangling process of I have transformed and prepared the dataset performing the following tasks:

#### YEAR attribute

Cleaning and splitting the attribute YEAR adding a column Last Year containing an eventual extreme when there is a range like for example 2010-2020. The target is to obtain 2 columns for the attribute: Year and Last Year where the second one could be populated in case of TV Show or series with episodes and seasons.

I have decided to not consider value under attribute YEAR that are not digits so using a transformation, passing for a temporary column extracting digits, I set as *zero* any not digit year value with the objective to obtain a clean and most actionable set for later analysis tasks. The null values under Year and Last Year attributes are for my choice set to *zero* to not change type of data too (for example replacing null with ‘-‘).

#### STAR attribute

I have splitted STARS to obtain two columns: one for Director/Directors and one for the list of Stars. In this case I have opted to replace null value with char ‘–‘ to keep a sort of consistency in the data type.

#### VOTES attribute

Format converted to integer and null replaced with *zero.*

#### GROSS attribute

The attribute has a high number of null. I have decided to change the original format for populated value from $NN.ddM to float without replacing null values with anything to keep the data type. The values obtained as float for this attribute are in millions of dollars.

#### Renaming and Ordering Columns

{'MOVIES':'Title','GENRE':'Genre','RATING':'Rating', 'ONE-LINE':'Plot','VOTES':'Votes', 'All Stars':'Stars', 'Year\_digits':'Year'})

New names and order:

'Title', 'Year','Genre','Director','Stars','Plot','RunTime','Rating','Votes','Last Year','Gross'

## Missing Values Handling

#### Null Equivalent Values

#### /n replaced in all dataset before splitting STARS with empty string

*‘Add a Plot’* replaced for attribute “Plot” where it has a frequency of 13% with ‘-‘

For the attributes RunTime, Rating and Gross I kept the NaN without replacing them:

* Rating 0 (zero) could be a real value of rating, so I prefer to leave NaN if not available. So when Votes are 0 Rating is NaN
* For a similar reasoning I have chosen to not replace NaN for Gross, to leave exposed records for which it is not available Gross, but is not zero (rare event but that could happen)
* For RunTime it was a choice to not have zero minutes, again for similar reasons as above.

## Outlier Detection

Outlier detection was conducted to identify unusual data points, sometime isolating single case or exception to understand if it could be or not a plausible data or an outlier, but without removing any records. Methods included:

#### Statistic-Based Methods (Fitting distributions)

* Z-Score (ZS)
* Median Absolute Deviation (ZSB with MAD)
* Standard Deviation (STD)
* Percentage (PERC)
* Interquartile Range (IQR)

#### Distance-Based Methods

* Local Outlier Factor (LOF)

#### Model-Based Methods

* k-Nearest Neighbors (KNN)
* Clustering (DBSCAN)

## Duplicate Detection

First actions to remove redundant entries and to ensure data uniqueness:

Removing exact match records

To perform an appropriate duplicate detection for the specific dataset I have chosen the Record Linkage method with blocking built on the attribute ‘Title’, applied two times: first to remove based on a wider set of conditions with some relaxed threshold and the second time identifying a complex attribute as plot (that is a long text usually) to find matches with a high threshold.

In detail:

### Comparison #1

Blocking: Title

Exact match: Title, Year

Similarity Measures 1: Director

Method: *jarowinkler*

Threshold: 0.90

Similarity Measures 2: Stars

Method: *levenshtein*

Threshold: 0.50

Limit of Matches: >2

### Comparison #2

Blocking: Title

Exact match: Title

Similarity Measures 1: Plot

Method: *cosine*

Threshold: 0.95

Limit of Matches: =2

Motivation for the use of each method

The method *jarowinkler* is the choice to compare the similarity of Director because perform well for name.

*Levenshtein* is used for Stars: here too we have names, but I have given more importance to the comparison of stars as a list and so this method could give in general a better result.

*Cosine* is used to perform the comparison of Plot because it is in general better when dealing with longer text strings.

# Conclusion

The final cleaned dataset has 6934 records and 11 attributes

|  |
| --- |
| # Column Non-Null Count Dtype  --- ------ -------------- -----  0 Title 6934 non-null object  1 Year 6934 non-null int64  2 Genre 6934 non-null object  3 Director 6934 non-null object  4 Stars 6934 non-null object  5 Plot 6934 non-null object  6 RunTime 5358 non-null float64  7 Rating 5883 non-null float64  8 Votes 6934 non-null int64  9 Last Year 6934 non-null int64  10 Gross 460 non-null float64 |

The Title attribute has duplicate because the list contains also different movies with exact title and TV Show or Series with episodes for different year or different season, exposing the same title.