Knowledge Discovery and Data Mining

CS 5228

Project Report :
Movie Database Analysis
Producers and Patrons of the Arts Decision Assistance Tools

April 2024

Cao Zhicheng
E1119322

Chan Wei Cong Andy
E1043412

Guilhem Mathieux
E1154518

Lau Jiatai
E0950555

Raphael Perot
E1154503

Table of contents

1 Introduction **2**
1.1 Dataset and Workflow . . . 2

2 Preprocessing **2**
2.1 Data Cleaning . . . 2
2.2 Numerical Data Visualisation . . . 2
2.3 Categorical Features Processing . . . 3
2.3.1 Text Feature Extraction . . . 3
2.3.2 Feature Transformation . . . 4
2.3.3 Feature Selection . . . 4
2.3.4 K-means ++ clustering . . . 4
2.4 Final Preprocessing . . . 5
2.5 Graph Mining . . . 5

3 Data mining **5**
3.1 Regression . . . 5
3.1.1 Method . . . 5
3.1.2 Hyper-parameters tuning : Grid Search . . . 6
3.1.3 Features importances comparison for tree ensembles . . . 7
3.1.4 Performance comparison . . . 7
3.1.5 Influence of splitting ratio and year range on prediction performances . . . 8
3.2 Classification . . . 9

4 Perspectives, Explored Techniques and Conclusion **10**

1 Introduction

The aim of this project is to guide producers and patrons of the arts in selecting movie projects to support. To reach that purpose, we propose to build a complete and rich analysis of the film panorama utilising the TMBD dataset which contains movies and TV shows from the year 1961 to 2015. In a first step, this dataset should allow us to provide key insights to understand the tendencies and rules that determine the success of a movie both in terms of profitability and popularity. Moreover, our analysis should help identify which movie characteristics are uncorrelated to a movie success and justify the choices in terms of preprocessing of the defining features.

In a second step, models will be built to explain the roles of the different movie features in its degree of success. Furthermore, those models should be able to provide reliable predictions on the profitability and popularity of new movie projects. Both the analysis and models should be targeted to the use of patrons of the arts and producers. For instance, production companies might look for estimation of the profitability of a movie project. On the contrary, patrons might be driven by the artistic dimension and look for estimations of the public reception to guide their choice of movies to produce since they would rather produce a classic than a blockbuster. In summary, the methods should give various insights on sufficiently enough aspects of a movie to help a producer make a decision whatever their background or motivations.

A major challenge of this project is to extract information, rules and build classification and regression models from a limited number of samples. Indeed, we used a dataset with less than 1300 rows movie data. This constraint is interesting as it makes deep learning less pertinent prompting us to explore a range of techniques. This will help us understand the impact of features on the popularity or profitability of a movie. Additionally, it requires a meticulous selection of the model architectures to achieve high performance, which is more challenging than simply using deep learning.

Note: All the code used for this project can be found on the TMDb-Data-Mining github (click to access the repository or use the link given in the bibliography [3]).

1.1 Dataset and Workflow

The training dataset has 1287 rows, each row is a movie or TV show that has 21 attributes that may give clues to the profitability of the movie or TV show.

We based our models on the primary properties of the movie: Budget, Original Title, Cast, Director, Keywords, Overview, Runtime, Genres, Production Companies, Release Date. We will omit factors directly related to profit or popularity like revenue and Vote average.

The workflow diagram in the appendix (figure 17) illustrates the workflow to analyse the attributes and tune the different models of prediction. It gives a good idea of how we organised the main steps of this data mining process from exploratory data analysis to modelling.

2 Preprocessing

2.1 Data Cleaning

In this part, the data type of each attribute is determined. Missing values are identified, with 1 row having missing value for the attribute "popularity". We will simply remove the entry as 1 row out of 1287 entries should not be significant. We also spotted some erogenous entries for "release_year" where the release year is 2062. This record is corrected based on "release_date" attribute.

2.2 Numerical Data Visualisation

Data visualisation provides insight into the data distribution which can help us in subsequent feature extraction and preprocessing. Outliers may also be surfaced out by data visualisation and will need to be handled by a case-by-case scenario. Data visualisation may also give clues to whether an attribute is relevant or not.

First, we looked at the distributions of the numerical features of the dataset (Figure 18 in the appendix). This visualisation shows that for some of the attributes such as popularity, budget, revenue and profit, there are outliers data points with one or two movies having very large numbers. We can choose to discretise them into groups based on quantiles. Doing so will remove the effect of outlier points skewing the model, however it will also cause some loss of information as we expect that budget is closely linked to profit. Therefore, at this stage, we will keep both original and discretised attributes and test which one performs better in the modelling phase.

Secondly, we looked at the correlation with profit (figure 19 in the appendix). This visualisation shows that while most of the numerical attributes are correlated to profit, release year doesn't. Indeed release date has a similar distribution if we divide our movies in four profit levels (0 indicates movie have negative profits, with 1, 2 and 3 generated by discretising profits based on quantile):

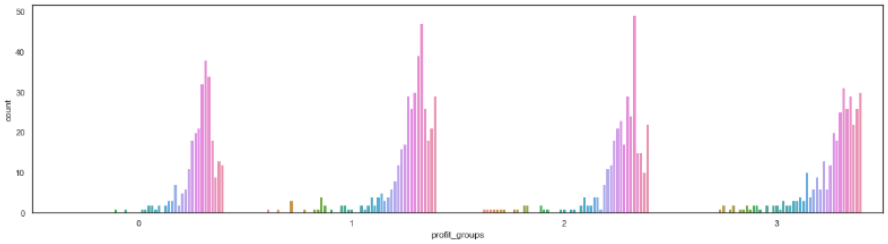


Figure 1: Release year distributions for four profit groups

This would seem to make release date not relevant for our goal, however this feature might still be useful by only taking the month rather than whole date:

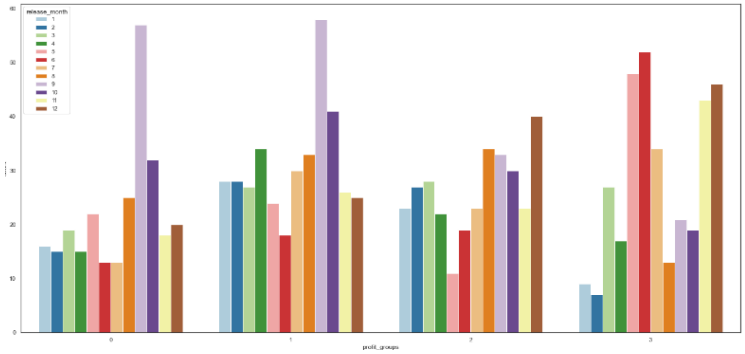


Figure 2: Release month (extracted from release date) distribution for four profit groups

We see some months have abnormally low /medium ratings such as September and December, while some have higher ratings such as May, this may be due to many unknown but chronic factors hence this could be used as a training feature. To encode that, we have created an ordinal attribute, release_period, with a value of -1 for months with disadvantageous period (September/October), 1 for advantageous periods (May/June) and 0 for the other months.

2.3 Categorical Features Processing

2.3.1 Text Feature Extraction

Some categorical features can't be visualised directly, and they may not be useful in the prediction model if used directly. These attributes include movie title, tagline, and overview. However, we can extract some potentially useful information from them. The most straightforward feature to extract is the word count, hereafter called length.

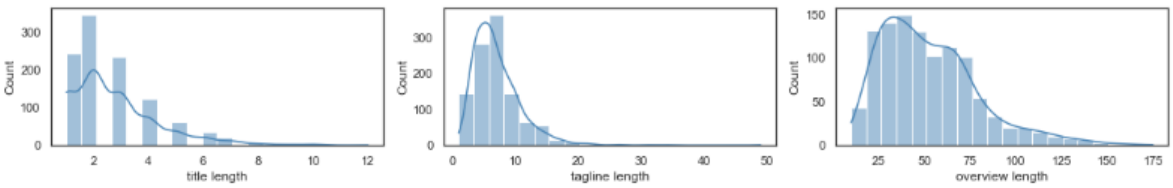


Figure 3: Histograms of title, tagline and overview length

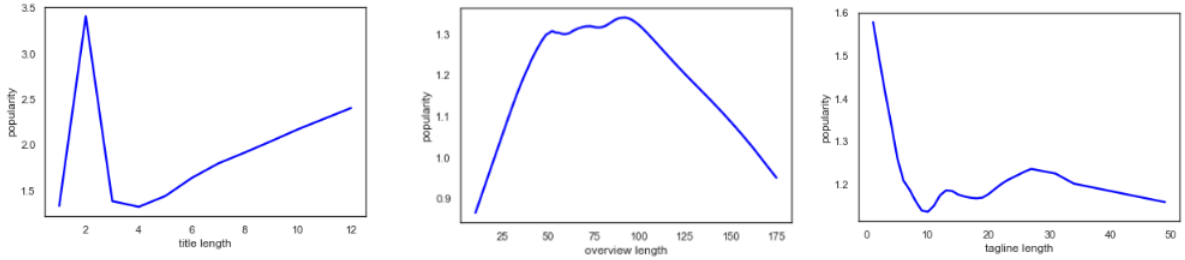


Figure 4: Correlation plots of popularity against title, tagline and overview length

Title length does not seem useful, overview length might be useful as too short overview lengths might not give audience enough information to capture their interest, while too long overview lengths may be boring and cause audiences to lose interest. Tagline length is useful too as we observe that high popularity movies are likely to have shorter taglines.

More features can be extracted from tagline, title, and overview such as the words themselves. However, there is already the "keywords" attribute which should better capture the characteristics of the movie, hence we will not extract words from tagline, title, and overview.

2.3.2 Feature Transformation

Based on the 98-2 rule, where 2% of actors/actresses make 98% of profits we use selectKbest method to select 100 most impactful cast based on movie popularity. We choose popularity instead of profit as the metric for selectKbest because there is a more direct link between actors and movie's popularity instead of movie's profit. We label movies with impactful cast as 1 and movies without impactful cast as 0.

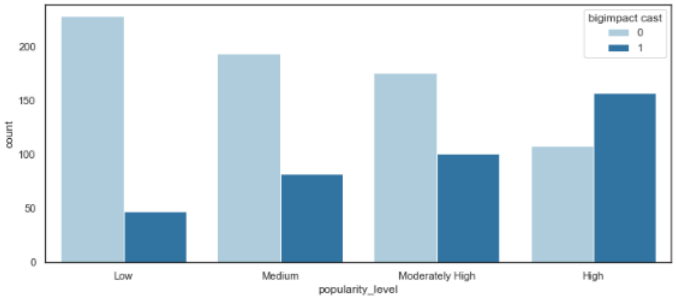


Figure 5: Count plot of movie with or without big impact cast

The significance of these top 100 actors is that they acted in very popular movies or very bad movies. We account for this by doing a filter on these big impact actors based on their popularity score (calculated by averaging all the movies each actor acted in). Our list of good cast is then left with big impact cast with popularity score of higher than the movies' average popularity score. Next, our prior knowledge tells us that profitable actors are likely to act in more movies. Therefore, we add actors with more than 7 movies (around top 2% quantile) to good cast list. Finally, for each movie, we count the number of good cast members involved.

Based on the trending with movie popularity, we observe that this extracted feature from cast is informative and useful in predicting a movie's profitability.

We also checked if bad cast was a useful metric. We obtained bad cast from big impact cast list with cast having lower popularity score than movies' average. Using similar techniques, we observed that bad cast is not a useful feature as no correlation can be seen: moderately high popularity movies have worse cast than low popularity movies. This may be because having bad cast does not mean movie cannot be positively impacted by good cast. Hence, this feature will not be used on subsequent model training.

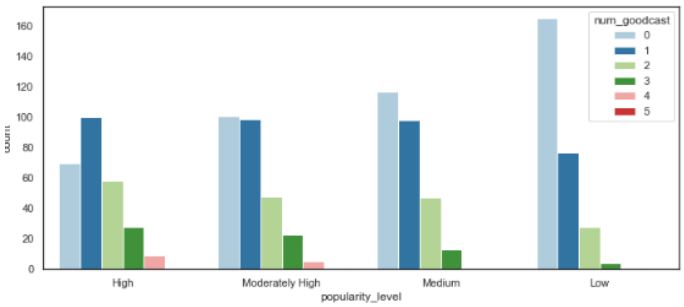


Figure 6: Count plot of movie with number of good cast members

2.3.3 Feature Selection

We follow the previously mentioned steps to transform the directors, keywords, genre, and production companies features. This allowed us to select three additional transformed features:

1. bad director: directors with only one movie or with an average popularity lower than mean popularity.
2. production quality: between 0 and 2, based on movie number and average popularity.
3. high impact genres: 1 for SelectKbest genres (action, adventure, fantasy, and science fiction).

Count plots for those features by popularity levels can be seen in the appendix figures figs. 20 to 22.

2.3.4 K-means ++ clustering

To obtain additional features, we have performed a K-means ++ clustering on an intermediate processed database where the prepossessing is identical to the one in 2.4 except from the presence of the "cluster" attribute. The clustering model was trained on the training dataset and the cluster labels used as cluster attributes were then computed for each sample of the train and test datasets. The number of neighbours was selected in order to find a trade-off between a low Within-Cluster Sum of Square (WCSS) and a reasonable number of clusters. On the one hand the goal is to not add too much columns to the dataset (since that feature will be one-hot encoded), and on the other one we want to avoid over-fitting. At the end, following the indications from A. Gupta [2], we chose a number of clusters of 8 since there is no "elbow point" in the WCSS curve but with 8, we prevent reaching the elbow point from the minimum number of samples by cluster that is around 10. The WCSS values by number of clusters curve as well as the minimum number of samples by cluster by number of clusters curve are given in appendix on figure 23 and 24.

2.4 Final Preprocessing

To complete the preprocessing, two more steps are needed:

- Standardisation of numeric attributes : budget_adj, runtime, vote_count, vote_average, tagline length, overview length, num_goodcast, production quality, keyword quality, popularity, release_period
- One hot encode of category attributes: director_quality, highimpact_genre, cluster

Then, we have split the dataset into training and test datasets. To do so, we have first sorted the data by release_date (increasing order), and then, we have kept the 85% oldest movies for the training dataset and the 15% most recent ones for the test dataset. Indeed, the goal of the project is to be able to predict the reception of new movies. We finally used as training features all the attributes cited above post processing (which involves multiple columns for the cluster attribute, the two other categorical attributes being binary).

2.5 Graph Mining

Graph are a powerful visualisation tool for datasets. We produced an undirected and weighted graph to represent the actors and the links they share (I.E. movies they both act in).

To make the graph more readable only actors with more than two movies and more than 2 connections are shown. This filtering is done using the one-hot encoding discussed earlier.

The graph properties are as follows:

- The size of each node represents the sum of an actor's movies revenues.
- The links represent shared movies between two actors (weight of an arc is exponentially linked to number of co-stared movies)
- The colour of each node represents the most common genre of an actor's filmography. We computed the six most important genres: SF, Adventure, Thriller, Comedy, Action and Drama and assigned to each a colour as seen in figure 2.5.

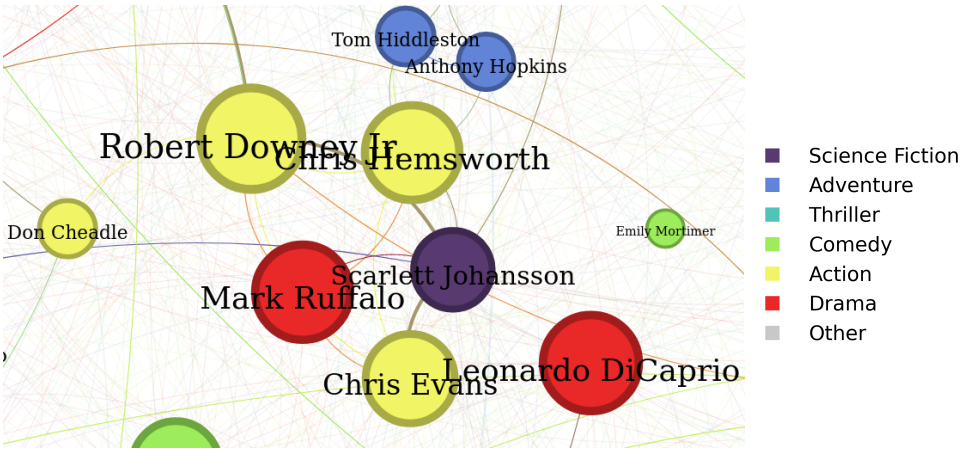


Figure 7: The Marvel Avengers cluster

The structure of the graph is computed in python and the network spatialisation was done using Gephi, a graph visualisation tool that includes many spatialisation algorithms.

The one that seemed the most promising for our use case is the force atlas 2 algorithm. It gives a few useful parameters to change the general shape of the network and does succeed in producing groups of strongly connected actors.

Using the *LinLog mode* and the *Prevent overlap* options to respectively increase overall distance between groups and respect the size of the nodes, we managed to produced the graph that can be found in svg format in the github. We can see some clusters forming, usually because of a saga like Harry Potter or the Lord of the Rings.

3 Data mining

3.1 Regression

3.1.1 Method

For the regression, we chose as targets 7 different attributes from the preprocessed database. 4 of those concern the profitability of the movie : the profit group ("profit_group"), the profit as part of the budget ("profit_part_budget) that can be negative as it is defined as the profit divided by the budget, the revenue and the revenue in terms of 2010s dollars ("revenue_adj"). The three remaining targets concern the popularity with the "vote_count", "vote_average" and "popularity" attributes. They are all numerical attributes apart from "profit_groups". However, this attribute is ordinal, so we have decided to perform a regression and determine the predicted profit group of a given test movie as the rounded predicted value.

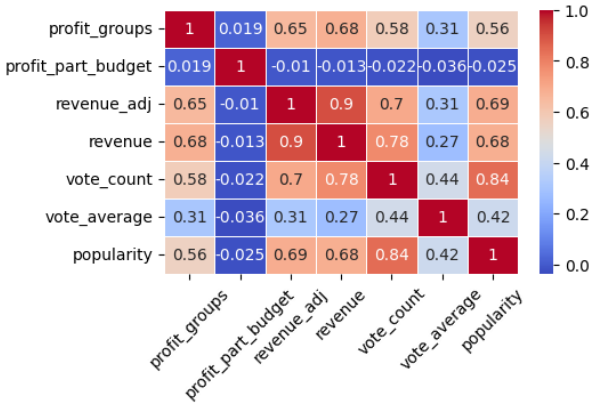
The "profit_groups","revenue_adj","revenue","vote_count" and "popularity" are positively correlated while the "vote_average" and "profit_part_budget" stand clearly apart. Thus, we can foresee that percentage of the budget generated as profit by a movie, as well as its public reception are not linked, directly at least, to the raw amount of entries or money it generated. Therefore, those two targets might show different behaviours from the other ones in the regressions.

To maximise our chances of obtaining good performances, we have used different types of models as regressors:

- Decision Tree regressors.
- Random Forest regressors
- Gradient Boosting regressors
- Adaboost regressors
- K-Neighbours regressors

It is important to note that scikit-learn can build both decision trees and random forests regressors with multiple targets. Thus, for those two types of tree ensembles, we have built both types (single and multiple targets).

Figure 8: Correlation between the different targets



To have a reference in terms of scores, we have used two types of baseline models for each target : Ordinary least square regressors and Mean predictors that consist in giving as a prediction the average target value in the training dataset for each test movie and target.

In order to have a more advanced regressor as reference in addition to the baseline regressors to evaluate whether the models are adapted for such a dataset, we will also use neural networks.

Moreover, we have chosen two metrics to evaluate the models : The Mean Squared Error (MSE) gives importance to significant errors, which is crucial for avoiding financial disasters for producers, while the Median Absolute Error (MedAE), though less common, compensates for MSE's drawbacks by being less sensitive to outliers and providing investors with a typical uncertainty estimate in predictions.

At the end, we have nine different types of regressors We will use the following aliasing to make reading more fluid in the rest of the report : {Decision Tree single target : DTs, Decision Tree multiple targets: DTm, Random Forest single target : RFs, Random Forest multiple target : RFm, Gradient Boosting : GB, Adaboost : Ada, K-Neighbours : KN, Ordinary Least Squares : OLS, Mean baseline : BL, Neural Network : NN}.

3.1.2 Hyper-parameters tuning : Grid Search

The baseline regressors are straightforward and do not require any fine-tuning. However, it is a necessary step for tree ensembles and K-Neighbours to reach great performances. For that purpose, we have performed multiple grid search using the negative MSE as selecting metric (since MSE is one of the metric used to compare the models at the end), with between 10 and 100 combinations of parameters for each regressor. This grid search used a k-fold cross-validation with $n = 5$ folds. Since there are $\#targets \times \#types\_of\_regressors + \#multi - targets\_regressors = 7 \times 5 + 2 = 37$ different regressors, we will not list every combinations selected that lead to the best performances within its category. However, the hyper-parameters optimised through the grid search are the following :

- Decision Tree : maximum depth, minimum number of samples to split, minimum number of samples by leaf, maximum number of features (None, square root)
- Random Forest : same as Decision Tree plus the number of trees
- Gradient Boosting : same as Decision Tree plus number of trees and learning rate
- Adaboost : number of trees, learning rate and loss
- K-Neighbours : type of weights (uniform or distance) and number of neighbours

Moreover, for the Decision Tree, Random Forest and Gradient Boosting regressors, we chose the Friedman MSE criterion given that preliminary tests showed that it was the best splitting criterion in general, but also because it was aligned with the metrics used to evaluate our regressors.

For the neural network regressor, we have selected the best combination by experimenting and iteratively fixing each hyper-parameter, to avoid a long grid search. We also performed a k-fold cross-validation with 5 folds to evaluate which model was the best. At the end, the structure of the chosen model can be described as follows:

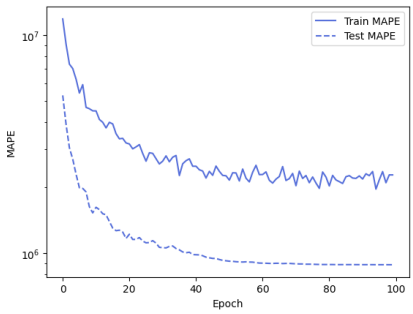
- a dense layers with 32 neurons, a ReLU activation function, and a dropout layers with a dropout rate of 0.2,
- 100 epochs,
- a learning rate scheduler with an initial rate of 0.0005 that is divided by 2 every ten epochs,
- a batch size of 16
- the Mean Absolute Percentage of Error as loss function.

We have maintained the number of layers low as well as the number of neurons in order to respect the machine learning "rule of thumb", explained by H. C. Donker in article [1], that prescribes to have less parameters (weights, biases) than the number of training samples.

The choice of the loss function is particularly important since MSE or $MedAE$ provide unbalanced performance among the target attributes since the latter have very different average values and the loss is common to all targets. The Mean Absolute Percentage of Error (MAPE) was the most appropriate because it enabled to normalise the loss among targets to provide balanced performances.

We have also plotted for each configuration the train and test loss curves to avoid selecting an over- or under- fitting model. Notably, we can see on figure 9 that the selected neural network avoids both of those problems with close training and testing MAPE curves at the end of the training (note that the scale is logarithmic).

Figure 9: Evolution of the train and test MAPE values during the training of the optimal neural network



3.1.3 Features importances comparison for tree ensembles

The features importances of the best estimator for each type of regressor are given on figure 10.

Clearly, what appears is that for all regressors and targets, the most important features are cluster_4, budget_adj, budget, runtime and production_quality.

First, the K-means ++ clustering enabled to create 8 binary attributes, 7 of which having negligible importances. That means that the within-cluster homogeneity in terms of target attributes is limited for all clusters except cluster 4. Indeed, for target attributes such as the revenue, the belonging or not to cluster 4 is nonetheless a relevant information.

Moreover, the tagline and overview lengths play a significant role especially for target profit_part_budget. Similarly, the keyword quality demonstrates a great importance for the prediction of the popularity. Thus, for the film advertising campaign, the choice of the tagline and overview is crucial. As figure 4 shows, the link between the popularity and the tagline length is not linear and even if shorter taglines are particularly catchy, the local maximum at 30 of length might be a good option too.

If some features importances values are expected, like the high values observed for the budget, adjusted budget and production quality for the prediction of pecuniary target attributes, some more curious patterns deserve to be highlighted. Indeed, the feature importance of the runtime is particularly high for target vote_average, while we might have thought that the director, cast and production qualities would play a more significant role in the audience appreciation. Thus, it seems more interesting for producers to choose a project with a wisely optimised runtime if they want to create a classic.

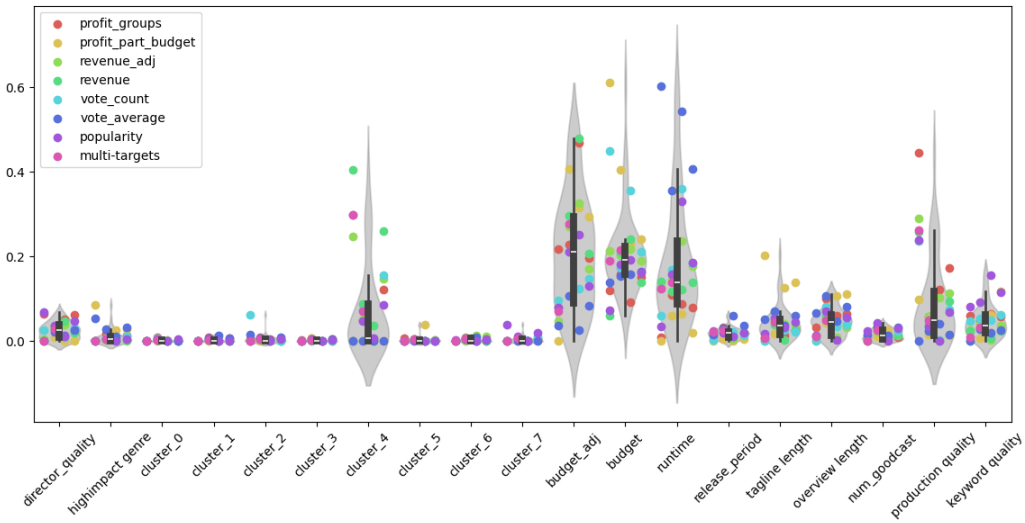


Figure 10: Comparison of the features importances among the different regressors by target (colour). For a given feature, the position of the dots corresponds to the type of tree ensemble, respectively from left to right Decision Tree, Random Forest, Gradient Boosting and Adaboost.

3.1.4 Performance comparison

The type and scores of the best regressor for each target and metric (MSE and $MedAE$) are given in table S1, as well as OLS and baseline scores. Clearly, the force of tree ensembles models in comparison to the baseline is the robustness to outliers. The best examples are the scores for target "profit_part_budget". Indeed, the baseline models provided extremely poor performances due

to outliers in the training set (e.g. "Paranormal activity" that generated 194 million dollars with a budget of 15,000 dollars). K-Neighbours regressors showed great performances in terms of revenue. Overall, we can see that tree ensembles and K-Neighbours show better performances than baseline models and neural networks except for vote_count. Notably, popularity and vote_average can be well predicted. Besides, the MSE and MedAE difference can be a factor of 10 between the optimal regressor and the baselines. For instance, with a decision tree, we can reach a great MedAE for profit_part_budget with 1.54. That means a median error of 154% of the budget when predicting the profit which is reasonable, especially when considering that the average percentage is 442%. Furthermore, neural networks showed extremely poor performances, and even when violating the rule of thumb [1], better performances could not be reached.

In general, the same type of models show the best performances for both metrics, but there is a wide variety of model types depending on the target attribute. Thus, we suggest to follow the two-steps model optimisation method we presented (1. grid search for different model types and 2. Selection by target among the best estimators) when working with new targets.

Table S1: Best Models' type by target and metric (MSE and $MedAE$) and comparison with OLS and Mean baseline models

		MSE				$MedAE$			
		Best Model		OLS	BL	Best Model		OLS	BL
	Avg[a]	Type				Type			
profit_groups	1.56	**Ada**†	7.88e-01	9.95e-01	1.21	**All***	1.00	1.00	1.00
profit_part_budget	4.42e+01	**DTs**†	5.36e+02	8.37e+03	3.03e+03	**DTs**†	1.54	6.57e+01	4.92e+01
revenue_adj	1.99e+08	**KN**†	4.21e+16	5.47e+16	8.92e+16	**KN**†	6.02e+07	6.17e+07	1.58e+08
revenue	1.76e+08	**KN**†	4.86e+16	5.32e+16	1.09e+17	**Ada**†	4.24e+07	5.80e+07	1.33e+08
vote_count	9.48e+02	**OLS**†	1.05e+06	1.05e+06	2.22e+06	**OLS**†	3.83e+02	3.83e+02	6.65e+02
vote_average	6.28	**RFs**†	6.43e-01	6.91e-01	7.82e-01	**DTs**†	5.49e-01	6.22e-01	6.48e-01
popularity	1.79	**Ada**†	1.75e+01	1.85e+01	2.13e+01	**Ada**†	9.09e-01	1.06	1.07

[a] Average value of the target attribute . * All models obtained a MedAE of 1.
† See aliasing in section 3.1.1.

To better compare the relative performance of the different types of models, we have plotted the normalised MSE and $MedAE$ by target (see figure 11). We have excluded the neural network curve due to its too high values. The normalisation $(value_{model} - mean(value_{model\_i}))/std(value_{model\_i})$ has been made along the models for a each target and enables to compensate the difference between targets in terms of orders of magnitude. The normalisation preserves the order so the lower the normalised score, the better the model.

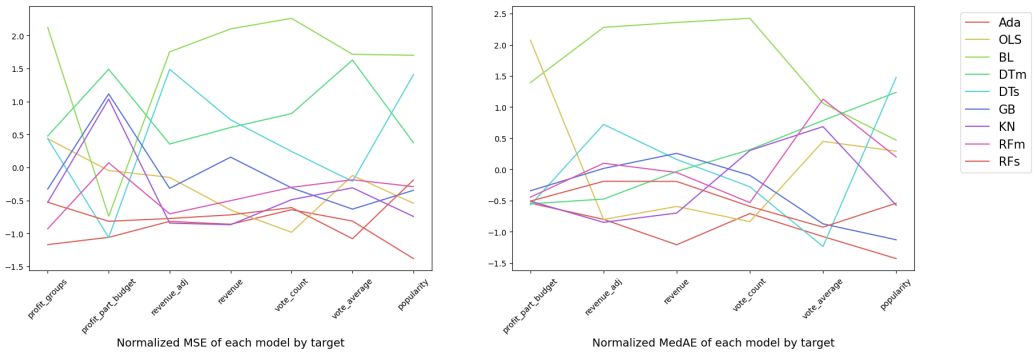


Figure 11: Comparison of the normalised MSE (left) and normalised MedAE (right) of the best models of each type of regressor by target

3.1.5 Influence of splitting ratio and year range on prediction performances

In the pre-processing, we have chosen a split ratio of 85%/15% with data sorted by increasing release date to simulate the prediction of upcoming movies for the test evaluation. We thus propose to try different ratios to evaluate the how the models' performances evolve : 65%/35%,75%/25% and 85%/15%. The data being unbalanced with a skewed distribution towards most recent dates, the date limit between training and test datasets for those ratios are respectively 2011-02-18, 2011-12-21 and 2013-07-18. For each target, we have chosen the parameters and model type that showed the best performance in the previous section. The normalised metric values for each target depending on the ratio are displayed on figure 12.

If some attributes benefit of larger training ratios like the vote_count and profit_part_budget, on the contrary some like popularity and vote_average show poorer performances with larger training datasets. This can be due to over-fitting on samples added to the training dataset when increasing the ratio. It is also interesting to note that the tendency is not always monotonous

with extreme being reached for ratio 75% for some targets. Thus, producers must be careful when uploading the model with new movies : it is not always beneficial to make the training dataset larger.

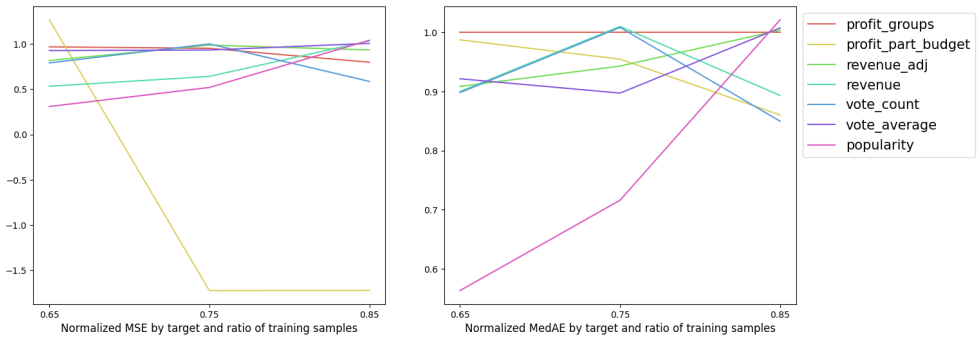


Figure 12: Evolution of the normalised MSE and MedAE of the best models by target with the splitting ratio (with data sorted by date before splitting)

3.2 Classification

We have attempted classification on the "profitability" attribute which value is either "yes" or "no" respectively when the profit is positive and negative. The logistic regression, tree ensembles and neural networks showed poorer performances than a baseline that would select the biggest class systematically. This is due to the fact the dataset is unbalanced with 79% of profitable movies. So, to create a model better than the baseline, the model has to reach 79% of accuracy.
As an alternative, the "profit_groups" ordinal attribute can be used as a classification target. We built a classifier by taking the rounded prediction of the best model for target profit_groups. The confusion matrix and classification report of the so built classifier are given on figures 13 and 15 respectively.

What immediately appears is that the classifier never predicts that a movie would not be profitable (class 0) so we did not solve the previous problem. However, the classifier shows a good performance for class 1 (recall of 83%) and class 3 (precision of 88%). What is striking is the number of misclassified second class movie in class 3. Thus, it might be more reasonable to give only classification in the 0-1 or 2-3 grouped classes. Indeed, as we can see on figures 14 and 16, even if the classification is less precised, we reach great F1, precision and recall scores so it is much more reliable. For a new movie, if it is classified in class 2-3, we have 85% (precision) of chances it was correctly classified and thus that the movie will perform well which makes the classifier better than a baseline for such a situation.

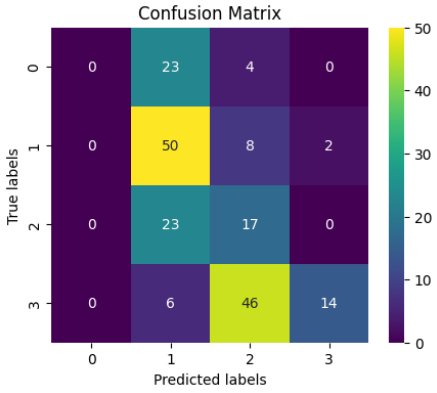


Figure 13: Confusion matrix of the classifier derived from the best regressor for target "profit_group" (Adaboost)

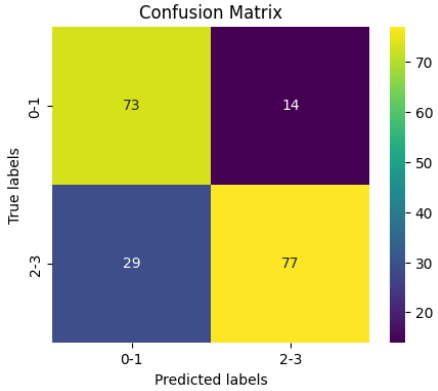


Figure 14: Confusion matrix with grouped classes

Profit group	Precision	Recall	F1-score	Support
0	0.0	0.0	0.0	27
1	0.49	0.83	0.62	60
2	0.23	0.42	0.3	40
3	0.88	0.21	0.34	66

Figure 15: Classification report of the classifier derived from the best regressor for target "profit_group" (Adaboost)

Profit group	Precision	Recall	F1-score	Support
0-1	0.72	0.84	0.77	87
2-3	0.85	0.73	0.78	106

Figure 16: Classification report with grouped classes

4 Perspectives, Explored Techniques and Conclusion

During this project, we have explored many techniques some of which did not lead to decent results. First, we have tested PCA to reduce the training features' dimension but it did not lead to satisfying results.

Then, to identify recurring collaborations between directors, production companies and actors, we have notably attempted association rule mining. The results were not satisfying which drove us to explore graph mining to provide to targeted users immediate visual insights on strong underlying relationships in the film industry. Regarding classification, we have used the classifiers corresponding to the regressor models but also logistic regression. However, all models were outperformed by the baseline model that always predicts that the movie is profitable. Finally, we have concluded that the performances do not increase linearly with bigger training datasets. Therefore, finding which samples to exclude from the training dataset would be interesting in the future to prevent over-fitting.

In this data mining project we have explored various methods firstly to clean, transform and present the data and secondly to build good prediction models. Standardisation, categorical feature dimension reduction and K-means++ resulted in meaningful features that enabled interesting performances in predicting pecuniary and audience reception attributes. Indeed, all target attributes except vote_count can be better predicted with tree ensembles or K-Neighbours regressors than with baseline models or neural networks (both in terms of MedAE and MSE). Moreover, classification in terms of profitability classes shows interesting results when making larger profit classes.

The constraint of having a small dataset has been a interesting challenge and having already satisfying results is a really promising as we can expect even better performance with a larger dataset.

We used this project to test many concept seen during the lectures from all types of classification & regression to rule and graph mining. This improved our understanding of those techniques by giving us a more practical approach.

Appendix

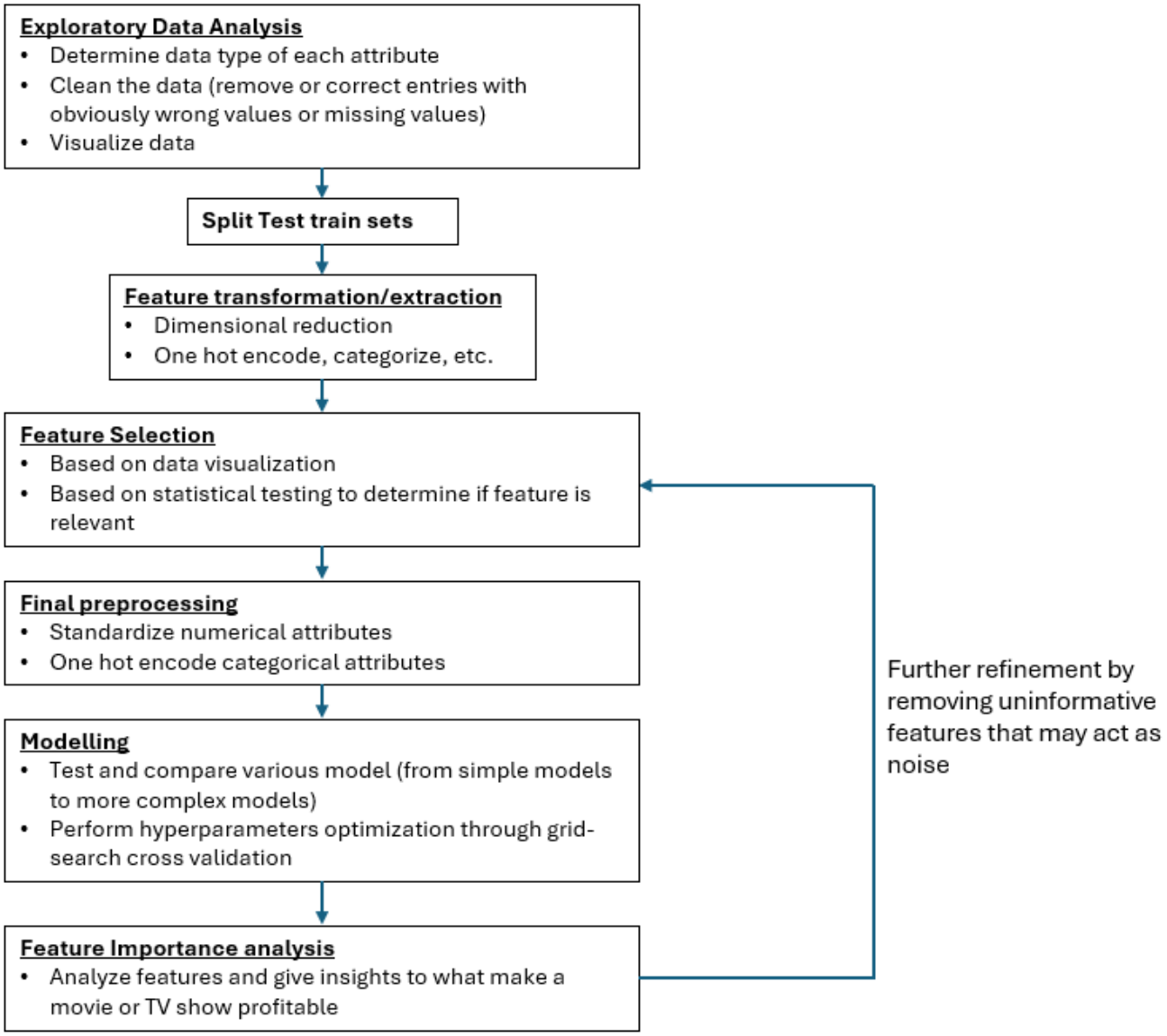


Figure 17: Workflow of data analysis and modelling

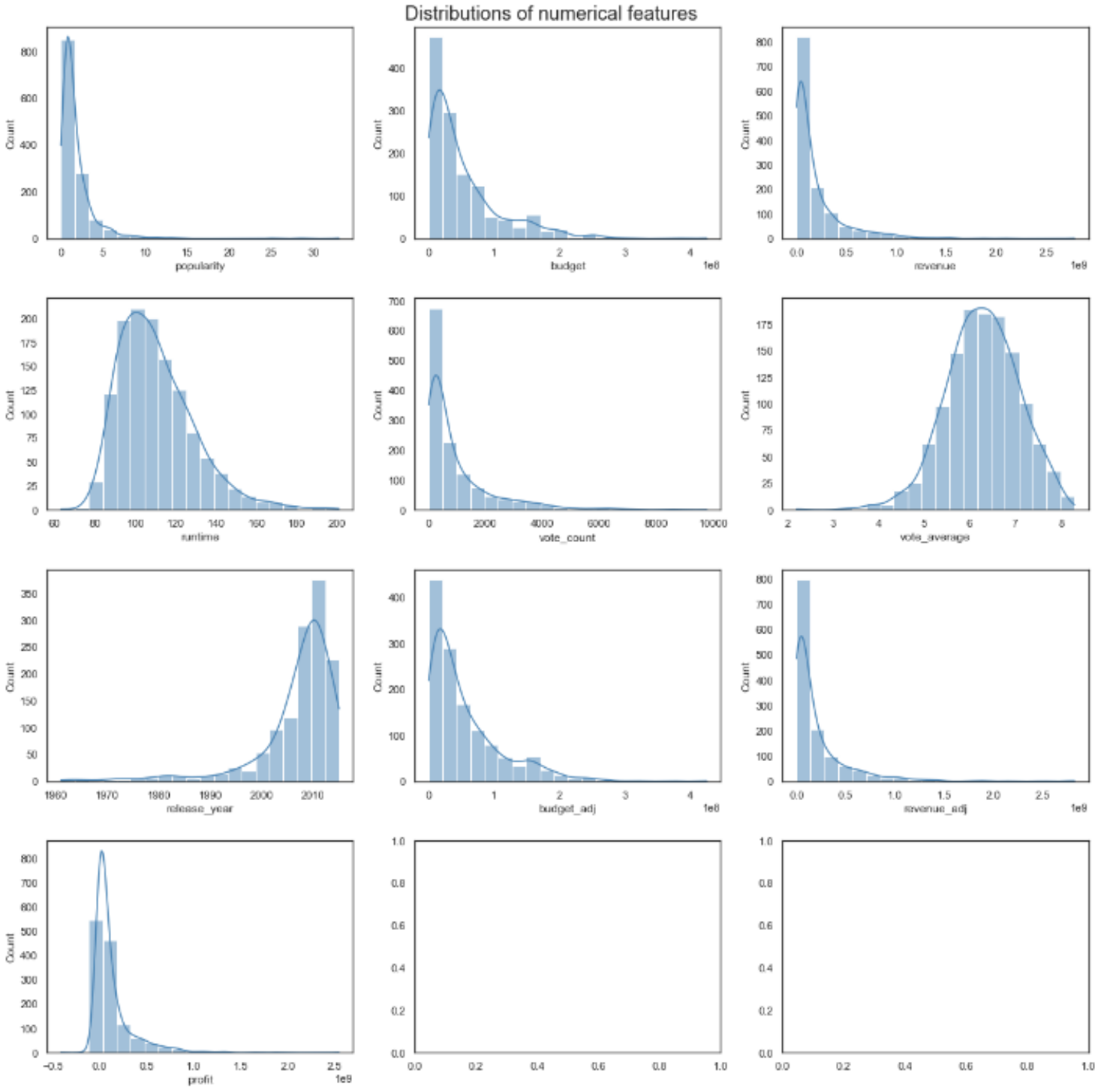


Figure 18: Visualisation of numerical data

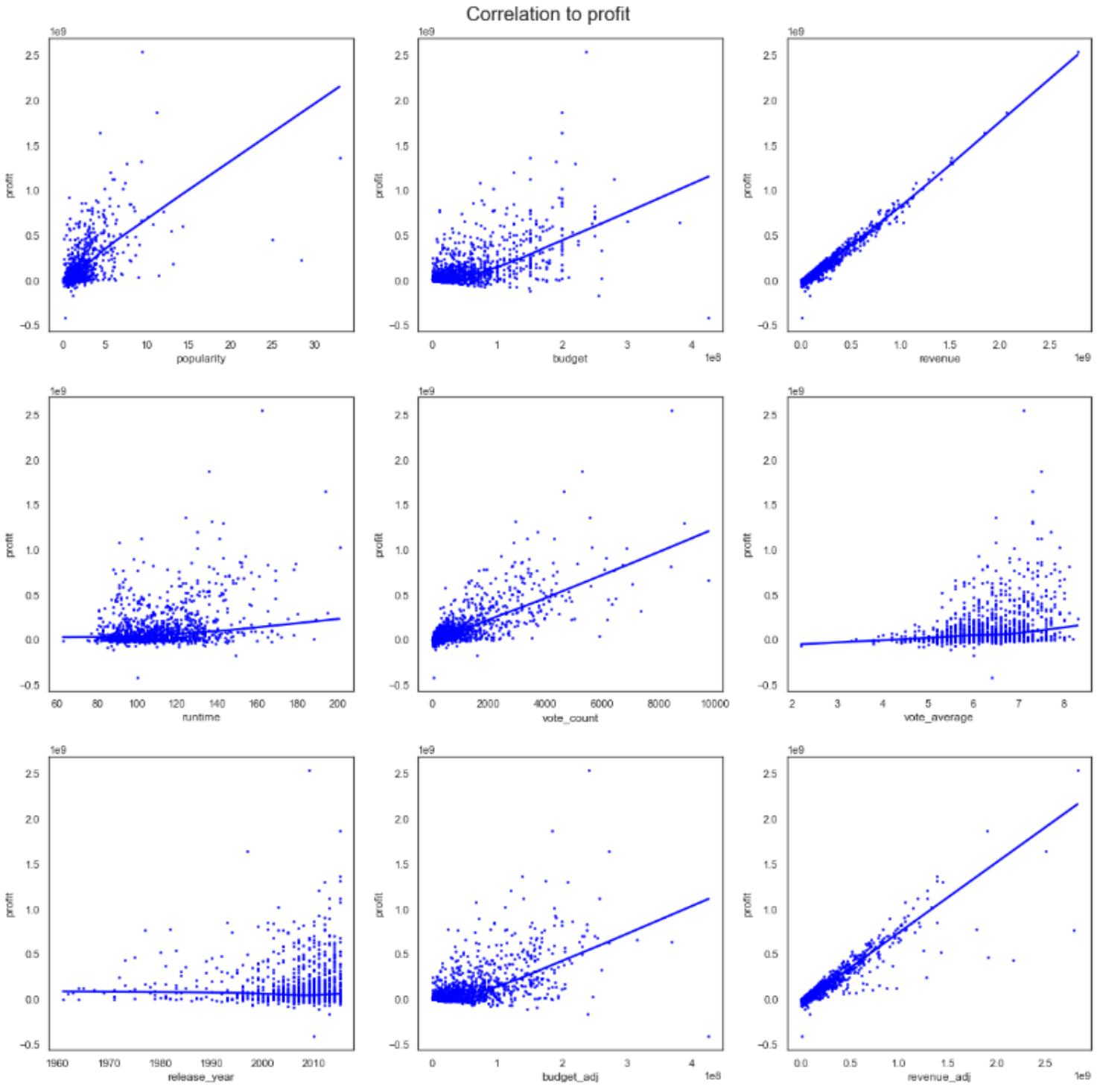


Figure 19: Correlation of each feature against profit

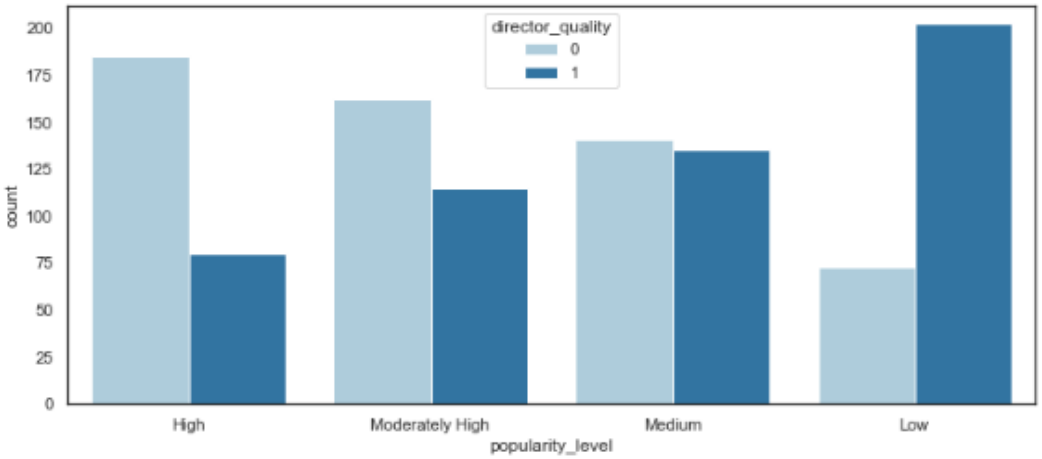


Figure 20: Count plot of movie with bad_directors (made only 1 movie or movies associated to them is on average less popular than population's average)

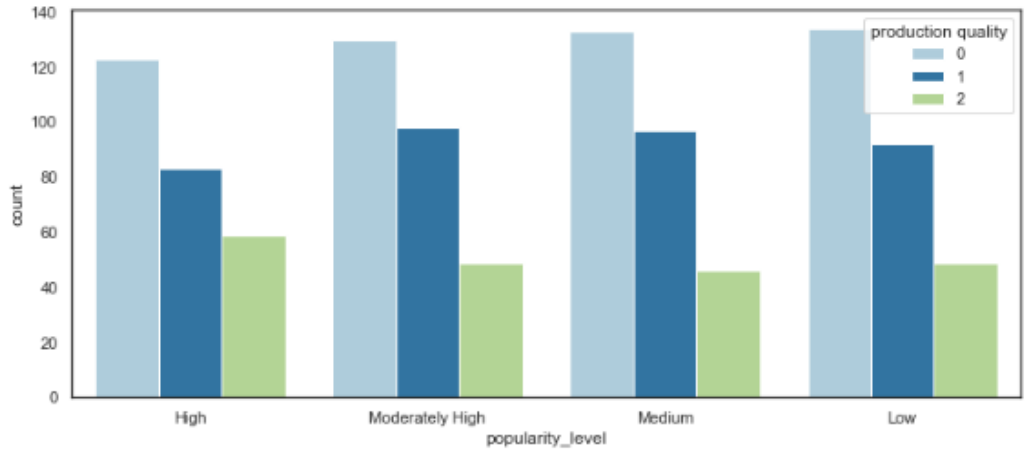


Figure 21: Count plot of movie with production company quality (2 indicates production company have higher than average popularity score and produced more than 60 movies, 1 indicates production company fulfilled 1 of the criteria, 0 indicates production company fulfilled none)

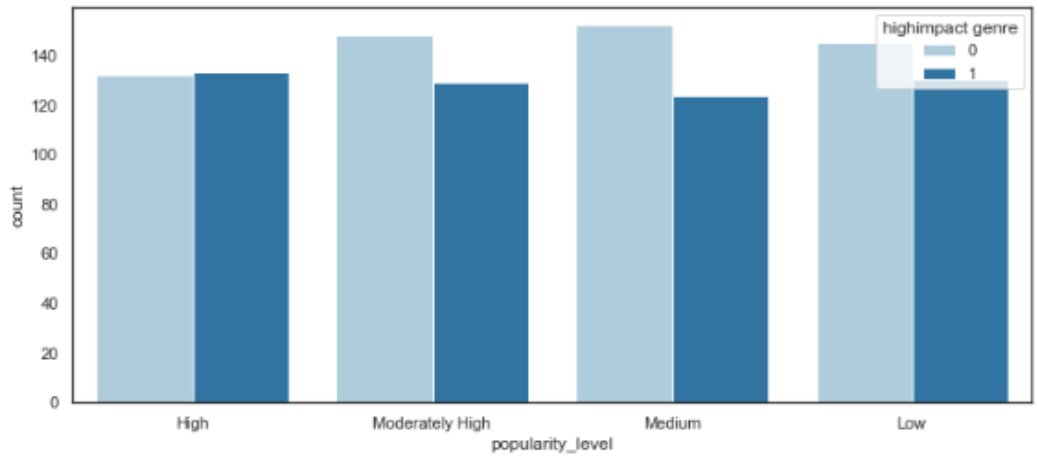


Figure 22: Count plot of movie with popular genre (1 indicates movies of with genre of Action, Adventure, Fantasy, and Science Fiction (selected through SelectKbest), 0 indicates others)

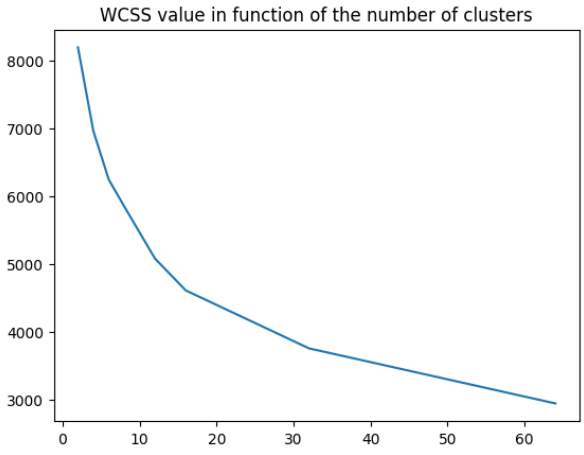


Figure 23: Curve of the WCSS value in function of the number of clusters for k-means++ clustering on the processed dataset

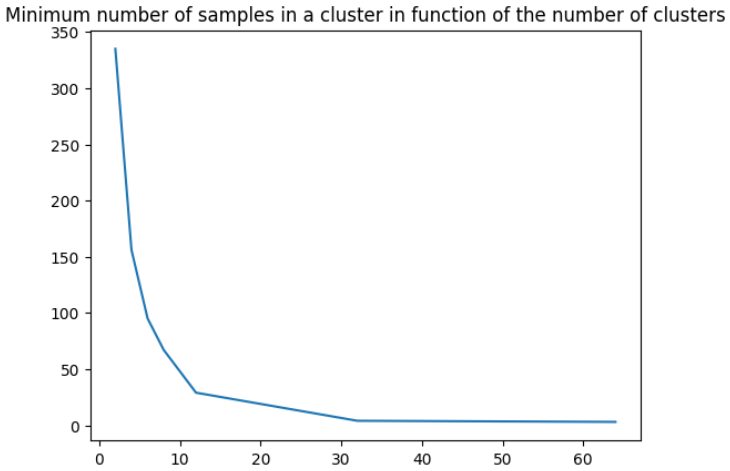


Figure 24: Curve of the minimum number of samples by cluster in function of the number of clusters for k-means++ clustering on the processed dataset

References

[1] H. C. Donker. Machine learning rules of thumb. *Towards Data Science*, 2022.

[2] A. Gupta. Determining the number of clusters: A comprehensive guide. *Medium*, 2023.

[3] TMDb-Data-Mining repository. https://github.com/guimath/tmdb-data-mining. *GitHub*, 2024.