How likely is a film directed by Steven Spielberg to be a success?

Machine Learning Project for Prediction supervised by Yannig Goude

Mathieu Even **Pierre Monteiller** **Matthieu Poyer**

February 29, 2020

Contents

1 Introduction

Being a brilliant film director, a famous actor or actress, or working as an amazing stuntman/woman are the dreams of many people on earth. The movie industry appeals and intrigues a lot as success seems obscure and depends on a very large number of causes. For this prediction and data mining project, we have combined machine learning to our passion for movies to understand and demystify the success of a film.

Accordingly, our main task in this project was to design, imagine and code a solution that predicts the economic success of a film regarding various pools of parameters. It is of particular interest for producers and filmmakers to understand the underlying patterns in the different possible choices of the cast, budget or any other variables, that cause and bring success to a film. Hence, this project aims at quantifying the effects and outcomes on the revenue of a film resulting from a comprehensive set of variables. We have chosen this very niche topic for our Machine Learning project because we strongly believe that ML has an important role to play in culture to understand the key points explaining why a piece of art appeals. Furthermore, this study may lead to understand which features enhance the success of a film and therefore may highlight psychological bias on what kind of film people like to watch.

The main source of information was coming from TheMovieDatabase (TMDB) an organization whose main goal is to collect data from all over the world about the film industry. It was found on a *Kaggle* competition. One the competition homepage, it was stated that basic predictors achieve R^2 of about 60 $\%$. Our objective was then to improve such scores. This heavy table of almost 1 Gigabyte raised several challenges. First, it mixes factorial and quantitative variables, the former induced a lot of pre-processing hard work before applying any models on those data. This pre-processing was painful and difficult as the quality and liability in the entries of this data set vary a lot from one film to another. Obvious outliers in the entries, the missing values or difficult encoding of names are some challenges we had to face in this project. As a result, after having pre-processed our data set, the outcome has a very large number of variables ($\sim$ 80000) which is challenging. Also, this data set suffers from different significant biases such as a major sample bias (most of the films in this data set are American movie) and a measurement bias (Revenues are in USD, but their values were not actualized). This leads to particularly challenging issues that we had to address during the whole project.

The global strategy of the whole project was simple and classic, but relatively efficient.

- We first pre-processed and explored in depth the data to understand and highlight the problems we will face in the data.
- Then we divided the variables into three categories. We exploited a first machine learning model on the first category set of variables.
- Afterwards, we developed various models on them to enhance performance of our predictor by trying to capture behaviors of the residuals of this first models. And we iterate over the third category.
- Finally, we tried to take into account the different identified biases in the data to provide different predictors depending on the film you would like to produce.

Code is available here.

2 Data Base Exploration and Features Engineering

The data set was found first on a *Kaggle* competition launched by The Movie Database. This organization reports any film or movie project and files a mine of information about them. For instance, it reports revenues, budgets and cast but also the producing company and the language of the film. This was a dense source of information to reach our goal. However, this data set was messy and quite fuzzy, in the sense that many questions rose from those data. It needed a long work of pre-processing and analysis before undertaking the application of any ML model.

2.1 Data set description

This data set consists in a table of 3000 films with twenty-three columns (Table 1). For every film, we had at our disposal four main quantitative features: budget, revenue, run time and popularity. These features are the only quantitative features we will have in this quantitative prediction task. Thus, analyzing these features were of particular interest.

id	overview	popularity
belongs to collection	poster path	production companies
budget	runtime	spoken languages
genres	status	tagline
homepage	title	Keywords
imdb id	cast	crew
original language	revenue	original title
release date	revenue	

Table 1: Available features in the database

2.1.1 Analysis of the four quantitative features

It reveals two main observations. First, there are many inconsistencies in the data values; they were not reported very carefully or harmonized. Second, there exists some important collection bias in this data set. Therefore, these quantitative variables are not a panacea: other variables are very relevant in the models. Overall, the analysis of the different quantitative variables show the main flaws and defaults in the data base quality, fact that will be verified on other qualitative variables.

```
     budget          popularity         runtime         revenue
 Min.   :        0  Min.   :  0.000   Min.   :  0.0   Min.   :1.000e+00
 1st Qu.:        0  1st Qu.:  4.018   1st Qu.: 94.0   1st Qu.:2.380e+06
 Median :  8000000  Median :  7.375   Median :104.0   Median :1.681e+07
 Mean   : 22531334  Mean   :  8.463   Mean   :107.9   Mean   :6.673e+07
 3rd Qu.: 29000000  3rd Qu.: 10.891   3rd Qu.:118.0   3rd Qu.:6.892e+07
 Max.   :380000000  Max.   :294.337   Max.   :338.0   Max.   :1.520e+09
```

Figure 1: Summary of the four quantitative values we have at our disposal

To run this analysis, we plotted the statistics summary of those four variables (Figure 1). Budget and revenue have quite a similar behavior: the median value is much lower than the average value, max and min values seems unrealistic and the interquartile range is rather spread. The first fact means that a lot of values are very close to 0 regarding the average, that is to say very low budget and low revenue. One could argue that this is due to the presence of short film in the data base. However, the first quartile of run time is 94 minutes which corresponds to a regular film. So, our main hypothesis

is that the values were in expressed in USD or in the local currency, and were not harmonized. This should be true for revenue and budget. The first quartile of the budget is 0 meaning that a lot of films were supposedly produced with 0 $, which is not very coherent with the first quartile of revenue. This is probably due to missing values that were reported in the data base.

Popularity and run time features reveal more reasonable and reassuring facts about our data base. Popularity is a score given by professionals of the movie industry **?**. This score is in between 0 and 300 and mainly concentrated between 4 and 10. Therefore, most of the film have their popularity score in a very small range and this popularity score will probably not have a linear effect on the revenue. Also the run time column is made of regular movies with some very short movies. We suspect that time was not expressed in minutes. This leads to very small values of 1.2 or 1.4.

2.1.2 Analysis of the data collection bias

The second important aspect in the data, we carefully looked at was the distribution of features regarding the timeline. Reported films were released between the 20's until the late 2010's, which makes quite a large time scale. However, as shown in Figure 2, most of the reported film in the data base have their release date between 1990 and 2020. First, this is not very surprising given the actual trend of Hollywood, in 2000 only 371 films were produced compared to 878 in 2019. This justifies somehow the growing trend observed in the data. However, this does not take into account the other cinema industries that were flourishing in the 80's and 70's, European movie industry mainly. This is a first hint to claim that the data base is probably focusing on American movie industry mainly. Also, the average revenue and budget are decreasing over the time. This may come from three facts:

- The production costs increased by a factor 5 in 20 years
- These sums are not always expressed in USD or equivalent USD
- USD values were not actualized regarding the inflation. For instance, 1$ in 1990 is equivalent to 2 $ in 2020. Same, 1$ in 1970 is equivalent to 6.7 $ in 2020

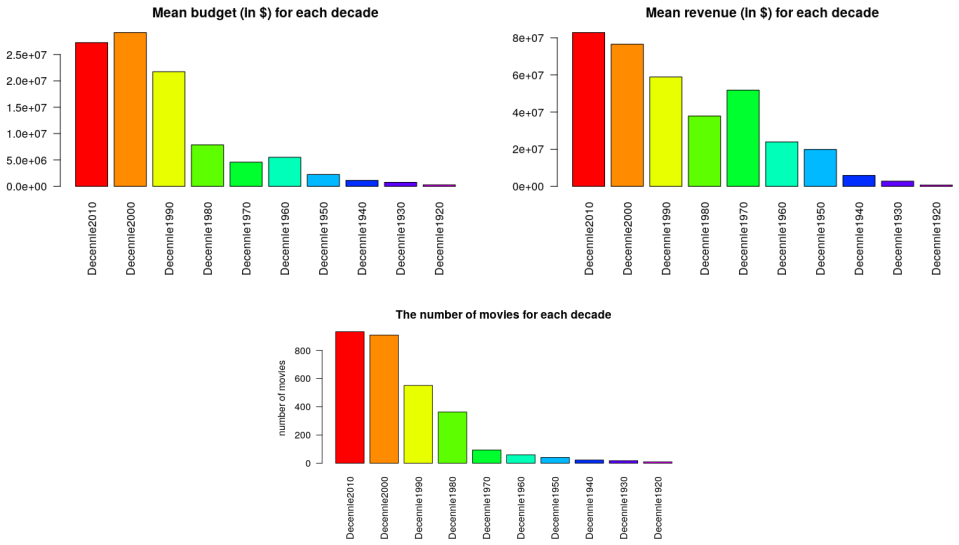

Figure 2: Caption

This clear temporal bias leads us to look up at other different biases that could have appeared in the data base. As the organization TMDB that collected the data is American, our first intuition is that the collection of data would be biased in favor of American movies and English movies. As we can see in the figure 3, a large majority of the films reported in the data base were produced in the USA. However, America was only the 5th producer of film (660) whereas India produced in 2018 more than 1,8000 films **?**. One could argue that this data set is historic and that Indian film industry is pretty recent compared to the traditional powerful Hollywood industry and that the American movies have a global impact rather than a very local and regional one for Indian movies. More than a bias in the data collection, this bias come from the industry structure in itself. However, we still encounter this major issues data collection bias that the majority of the films were produced in America. We have the same issue for the movies that have other languages than the English language. Therefore, predicting the revenue of films that are not American and not in English will likely to be a failure or at least will probably have some errors in the prediction as models often interpolate and not extrapolate. This is our main motivation to look at models on smaller data sets in Part REF which may outperform our global models.

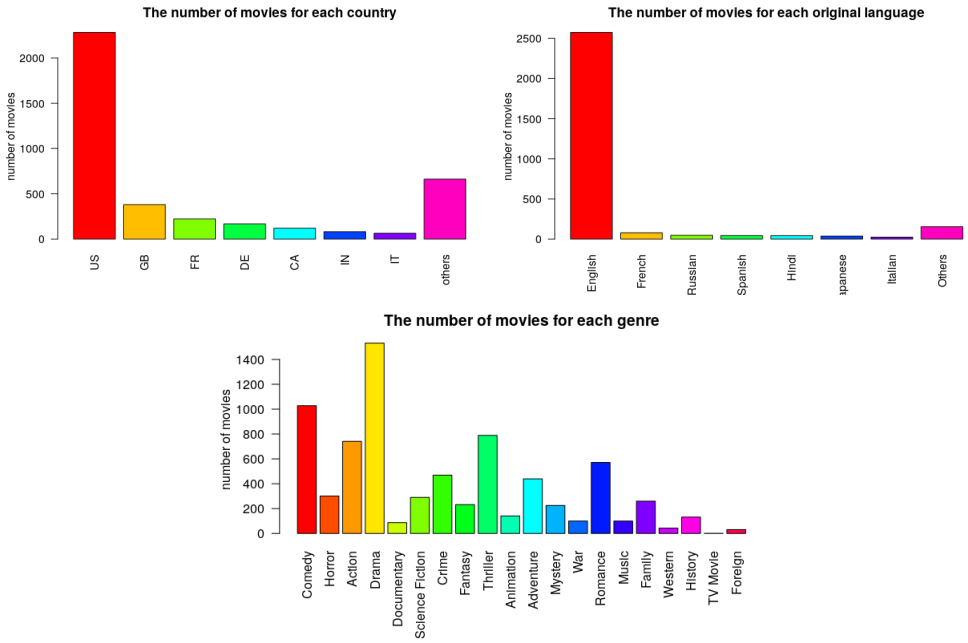

Figure 3: Illustration of bias in the collection process of the data

2.2 Pre-processing Work

After having well studied the data set and therefore getting aware of all the different flaws of our data set, some pre-processing work was made on this table in order to be able to deploy different ML methods and models. Two main lines of work can be identified:

- Treatment of verbose features colomns
- Dummy matrix encoding

The first step was one of the most tedious work we had in this project. Our first idea and feeling was that a film directed by Steven Spielberg has more chances to be a success than a film directed by an anonymous Spanish director. We had the same reasoning with the other verbose columns such as cast or keywords. But the very tricky part was that for many films, crew and cast listed the entire people involved in the film and combined id numbers and names. So our method was to create a new variable for every name in these cells and as soon as the person contributed to the film a 1 was put in this column, and 0 otherwise. This work required from us a lot of energy and effort to analyze efficiently this horribly encoded features. In this column crew, there was 40,000 different people and a similar number of actors in the cast feature. For the easy columns, for instance genre, as before we created a new column for every genre and a 1 was put if the film was of that genre or not. After repeating the same steps for suited features (keywords, production companies, production countries, original language, belongs to collection), we obtained a dummy matrix whose entries were zeros or ones and that was very sparse. This is what we called the dummy matrix encoding.

This pre-processing was really challenging as the crew and cast columns were very messy and difficult to handle in order to obtain our clean table. After this pre-processing step, we obtain our final data set to be a table of 3000 films with $\sim$ 90000 variables. We are now serenely able to deploy ML methods and models on a train data set of 2500 films and a validation set of 500 films.

A quick remark about our inactivity on supposed outliers: it is very difficult to determine if a film was an economical catastrophe or this film was badly reported. A certain amount of films were small productions from eastern Europe or from the Middle-East and the revenues and budgets are probably small or lower than blockbsuters production in the US. As our model should also predict those films, we decided not to remove any points at first.

3 Global Strategy Description

3.1 A very first attempt

Having 89375 variables for 3000 movies is challenging. Any models would be overparametrized and therefore would need a much higher number of observation to train the model. Therefore, we performed a dimensionality reduction algorithm to obtain more reasonable data in low dimension. We will then be able to fit a machine learning model to make our prediction on the selected variables.

To reduce the dimension in the data, we performed a classic Lasso algorithm on it. We used the `glmnet` package. To select the right parameter λ, we use a cross validation technique as shown in Figure 4. With this particular value of parameter λ, we select 19 variables, which is much more flexible than 90000 variables.

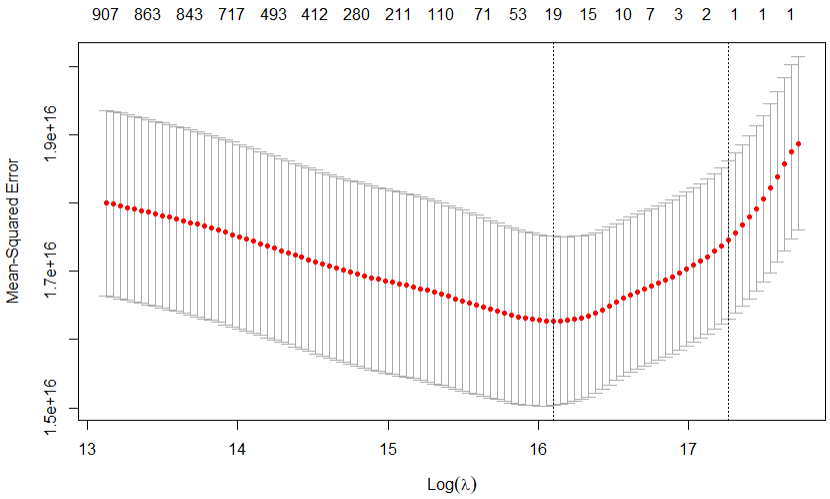

Figure 4: Cross validation in Lasso helps set the best value of λ

We tried a random forest on those selected variables, and obtained an indicator R^2 of 0.27 on the test sample, which is very low and not satisfying at all. This weak result is not very surprising though. We did not put any hierarchy in the variable in the Lasso procedure. As a result, the budget variable and any random actor have the same weight in the variable selection method. One could argue in return that we selected variables that should explain most of the variance. Thus, budget should be selected and not this random actor in practice. Obviously, this method is a failure and we need to think about a new strategy that should take into account the differences in terms of importance between all these variables. To assess a certain hierarchy on the different variables and features, we plotted the importance of every variables in the Random Forest. Very easily the three first (budget, popularity and run time) are far more important than the others, that are binary variables (see Figure 5)

3.2 Division into three categories

That very first model, which was a failure, leads us to think about and elaborate our strategy. 90000 variables seemed to us a huge number of variables. Also, a major part of those variables were factors

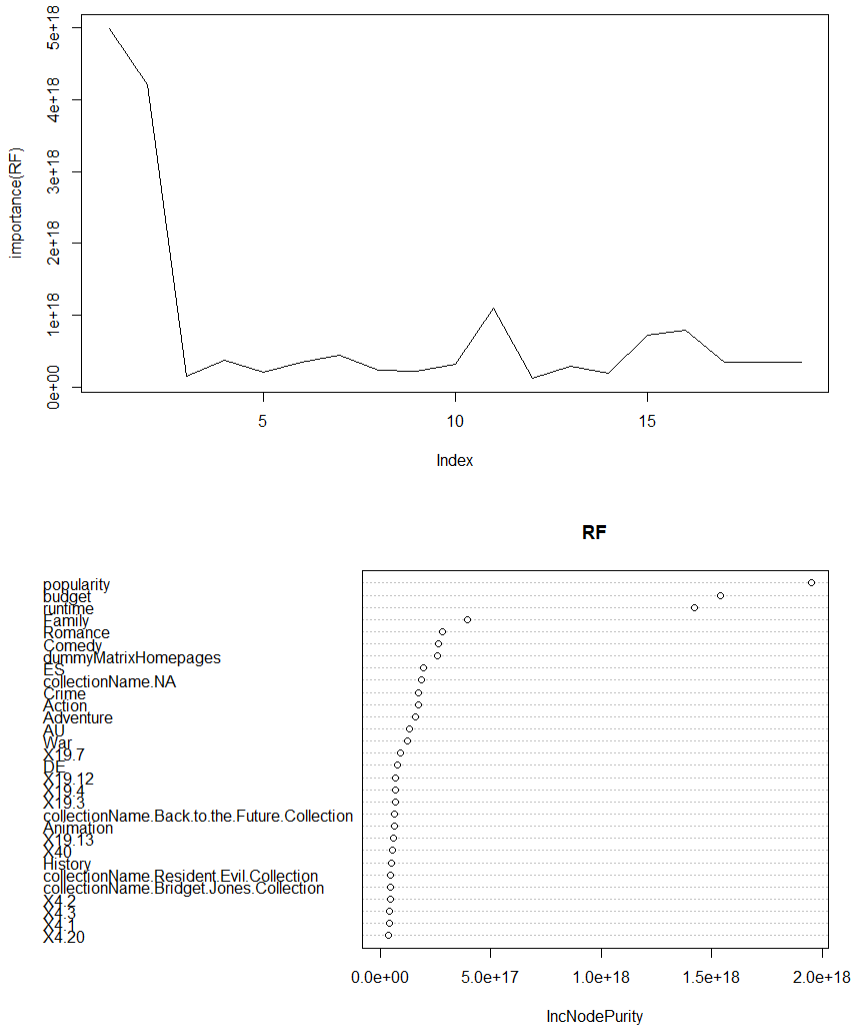

Figure 5: Importance of our 19 variables for the Random Forest

and therefore this data table is very sparse. This leads us to think that grouping the variables into three sets and applying different models to first predict then try to capture better performance by modeling the residuals at each iteration of the categories would make an efficient strategy for our study.

We chose the different categories of variables by their, supposed, importance in the prediction and their correlations between themselves. Figure 6 indicates to us that the two more correlated variables to the revenue feature are the budget and the popularity. Our first category should contain them. We also added the run time as there is no correlation between the revenue and it (even though that does not imply that these variables are independent).

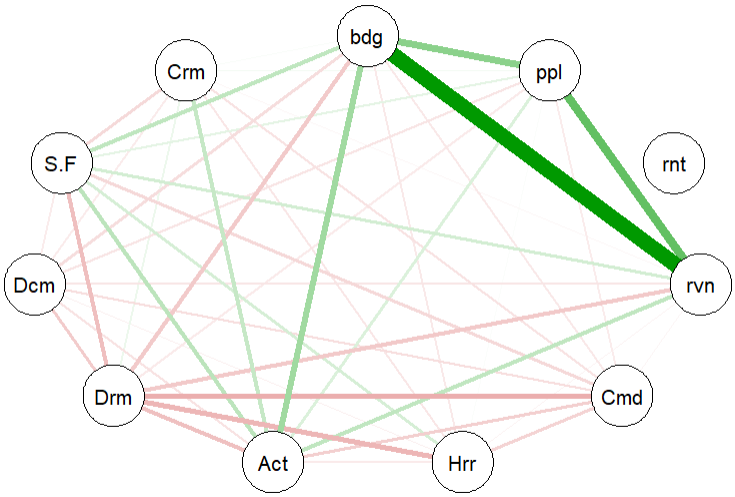

Figure 6: Correlations between several important variables selected by the lasso procedure. Green links denote a positive correlation and red a negative one. The thickness of a link denotes its importance

Then, the choices made for the determination of the two other categories were more of an accounting nature. The second category should not contain too much variables and the third one a lot of variables. The second category is then set to be made of all the categories except crew, cast and keywords as these three categories represents most of the variables. Mainly, category # 2 is made of language, production companies and genre of the films and that represents $\sim$ 5000 variables. Finally, the category # 3 is made these three categories that are very sparse for each film: crew, cast and keywords. This represents almost 85000 variables. We summarize our grouping into the following table.

CAT I	CAT II	CAT III
Budget	Genre	Crew
Run time	Language	Cast
Popularity	Production Company	Keywords
	Country	
	Collection	

Table 2: Summary of the three categories

3.3 Our final strategy

In this section, we aim at formalizing a bit more our strategy for it to become very clear to our reader. Denote CAT_i each category. In the following we described almost as an algorithm our final action plan for this project

1. Fit and deploy a ML model to obtain a relation $R \approx f_1(CAT_1)$.

2. Define the residuals of that model $Err_1 = R - f_1$. For each movie, Err_1 is defined as the difference between its revenue, and what our first model f_1 would have predicted.
3. Fit and deploy a ML model to obtain a relation $Err_1 \approx f_2(CAT_2)$.
4. Iterate this procedure over the third category: Fit and deploy a ML model to obtain a relation $Err_2 \approx f_3(CAT_3)$.

Finally, our final estimator function f will be defined as the sum of these defined models

$$f = f_1 + f_2 + f_3$$

4 A meaninful baseline model built on category # 1

As described before, variables were divided into three sets, with decreasing degree of importance. The first category is made of the most important variables according to us. These variables were *budget, run time* and *popularity*. We first tried a regression tree to have a glimpse on the importance of each variable.

4.1 A Regression Tree

As a first attempt to model our data and predict the revenue of a film, we tried to implement a regression tree using the package `rpart`. This helped us understand the hierarchy between these three variables. The parameters we used were

- `minsplit` value of 300, that is to say that the algorithm should split at least 300 films at each node.
- complexity parameter of 0.001, which enforces R^2 to increase at each splitting. This small value was chosen in order to have the parameter run time appear.
- `minbucket` value of 100.

As we can observe in Figure 18, the variable *budget* is of first importance compared to the two others as it is used in the two first splitting and in many more afterwards. Then, popularity is used several times and seems a powerful criterion to discriminate between two set of films. Finally, the run time seems less important regarding the two others, however still able to discriminate between two populations. Thanks to this regression tree, we obtained a relative importance order in the variables. We should find that in the following models the budget explains most of the results, then the popularity less and finally the run time even less.

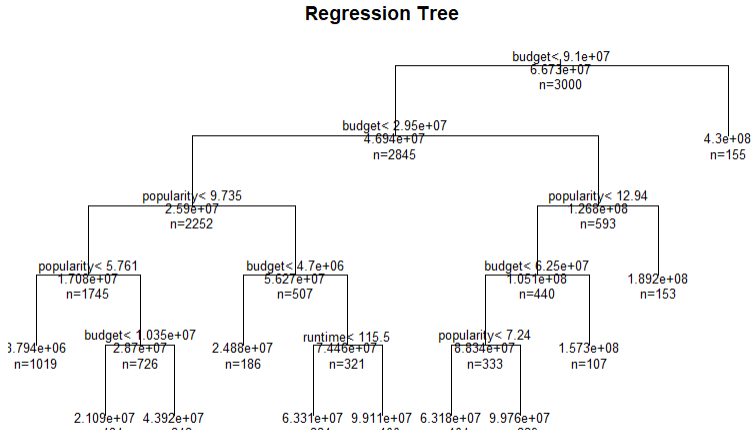

Figure 7: Regression tree for Prediction using Category # 1

4.2 Linear model

After this regression tree which led us to understand the importance order between variables, we decided to run a basic linear model between our variables. In this model, if we note R_i the revenue, B_i the budget, P_i the popularity and T_i the run time of the film i, we have the following relation between those quantities.

$$R = \beta_1 B + \beta_2 P + \beta_3 T + \varepsilon$$

with ε being a Gaussian noise.

β_1	β_2	β_3
2.4	2.5×10^6	1.7×10^5

Table 3: Estimated coefficient values for the linear model

We would like to estimate the different coefficients in this equation to obtain the final residuals as small as possible. However, as summarized in the table 3, the estimated coefficients reach non significant values because the indicator R^2 was of 0.24 which is not satisfying enough.. Looking at the residuals plot (Figure 8) shows major flaws in the prediction, meaning that the model is not suited for the prediction and should not be taken as baseline.

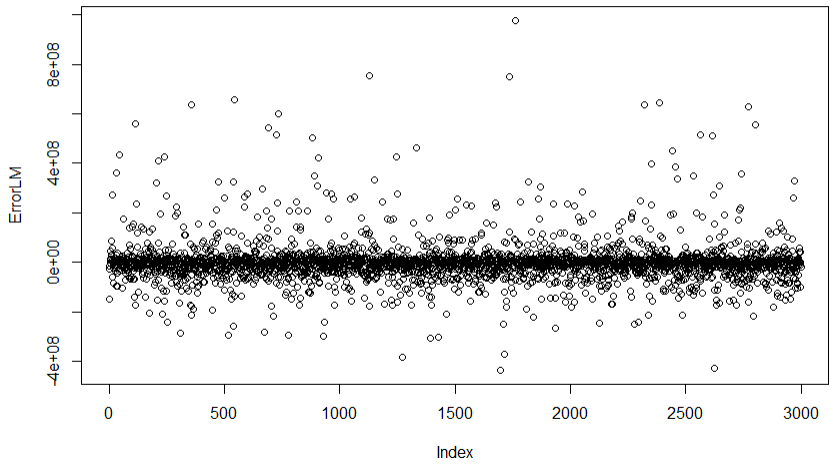

Figure 8: Residuals for the linear model

The validity of the linear model is quite flawed: why on earth a linear relationship between movie length and its revenue ? However, this linearity hypothesis seems relevant for the budget or the popularity features. Therefore, we decided to move towards a generalization of this model : a Generalized Additive Model.

4.3 Generalized Additive Model

In a GAM model, we assume taht the data is generated according to the following non linear equation

$$R = f_1(B) + f_2(P) + f_3(T) + f_4(B,P) + f_5(P,T) + f_6(B,T) + f_7(B,P,T) + \varepsilon$$

with all the f_i being smooth functions. We added the terms of interactions as we think that the behavior of those variables are correlated (see infra). These functions were estimated on a spline base functions (always thin plate spline basis) with a certain dimension k that was determined with generalized cross validation. All of this was performed with the package `mgcv`.

We obtain an indicator R^2 of 70 % on the train data and 60 % on the test data, which is decent even though that means that there is a slight over-fitting phenomena. As a sanity check, we plotted the residuals and response vs fitted values. Compared to the linear model, the GAM residuals are much closer to zero and the response vs fitted plot is coherent.

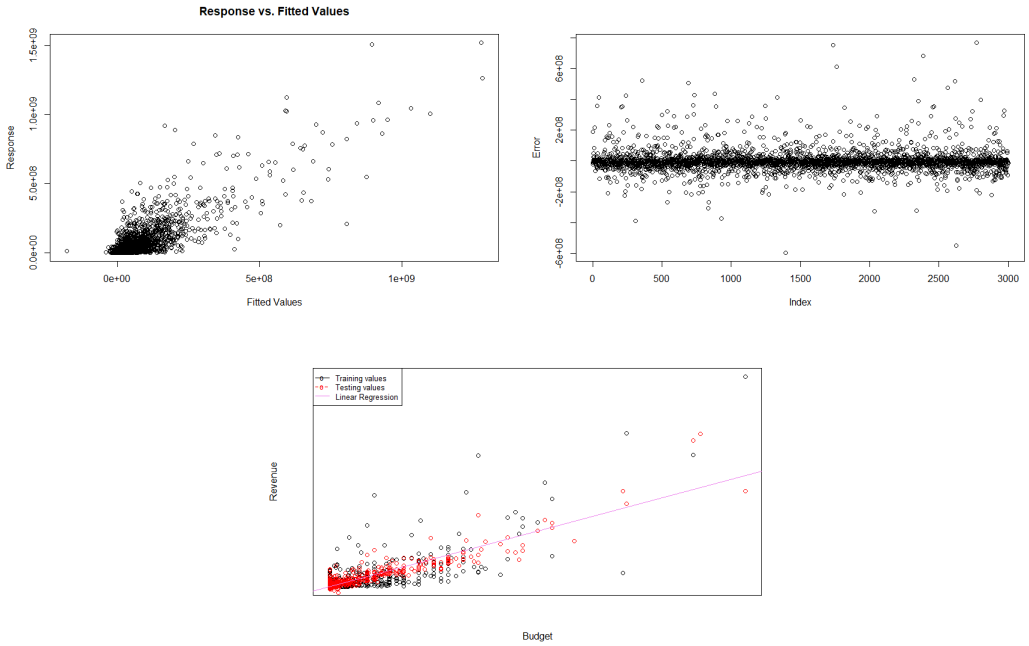

Figure 9: (Top): response vs fitted and residuals plot for the GAM (Bottom): Revenue with respect to budget which shows a linear dependency between those two features

Table 4 shows the estimated degree of freedom (edf) of each non linear contribution. The edf of the non linear effect of the budget is 1, so the effect is actually linear which is confirmed by the Figure 9. The effects of both popularity and run time have the same non linear behavior and the same edf equals to 9. They have a very non linear effect which is reassuring regarding the logic. Also, the interaction between budget and popularity may have no non linear effect on the response. However, the associated p-value to this fact is high (0.56) which means that this remark is not statistically significant. Also, the `ref.edf` is 27.

Finally, we plotted the predictor response with respect to the three variables (Figure 10) to analyze how it performs regarding the three variables. A first sanity check is to remark that the bigger the

function	edf
$f_1(B)$	1
$f_2(P)$	9
$f_3(T)$	9
$f_4(B, P)$	0
$f_5(P, T)$	27
$f_6(B, T)$	24
$f_7(B, P, T)$	4

Table 4: Estimated degree of freedom for every non linear terms in the GAM

budget and the higher the popularity are, the bigger the prediction is. However, these plots are not uniform and reveals several interesting facts about our GAM and its behavior.

- First, there is a symmetric valley in the prediction for middle run time in the plane (P, T). That is probably due to the fact that most of our films are either very short or relatively long. Therefore as the models could not really learn in this area.
- Second, we observe another valley in the plane (B, T). However, this is not symmetric and the run time regarding the budget seems to have the behavior of a 0-1 function, meaning that a sufficient run time is necessary to high revenue but not sufficient. Also the link between budget and prediction is linear.
- Finally, the prediction attains a local maximum in the plane (B, P), which means that this point may be of interest in terms of profitability for film makers. Also the predictor seems to have an exponential growth regarding the popularity, except around this local maximum.

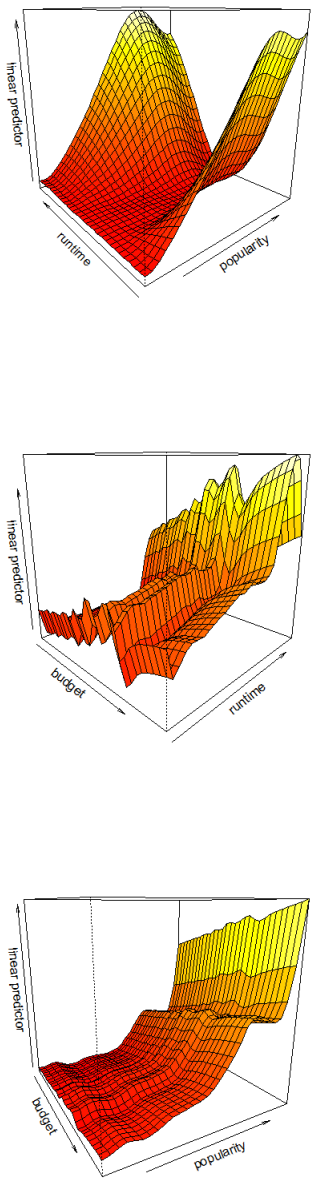

Figure 10: Linear predictor with respect to the three variables of category # 1. Yellow corresponds to high values and red to low values.

5 Models on the residuals: able to break the baseline model ?

In this section, we present how we refined our first GAM model built on variables present in category # 1. The main goal of this is to increase the global prediction performance and our target will be the R^2 score. We strongly believed that the Category #2 is a mine of information and should be treated in depth. Therefore, we spent quite some time on trying to optimize all the approaches we started off. On the other hand, the category # 3 due to its very sparse but memory heavy nature was more tricky and we used a very straightforward method quite similar to the first attempt we did (reduce dimension then apply ML method).

5.1 Category # 2

First, we plotted the residuals of the GAM obtained in the previous part (Figure 11). Residuals are located close to the 0 line. However, a large fraction of the points are non zero. This does not look like noise to us: we can apply our ML algorithms directly. We chose two models for our prediction task: a Random Forest and a Gradient Boosting algorithm (XGBoost). This choice was motivated by the following observation: nowadays, in most of the *Kaggle* competition, those two methods are the most competitive ones (excluding neural networks).

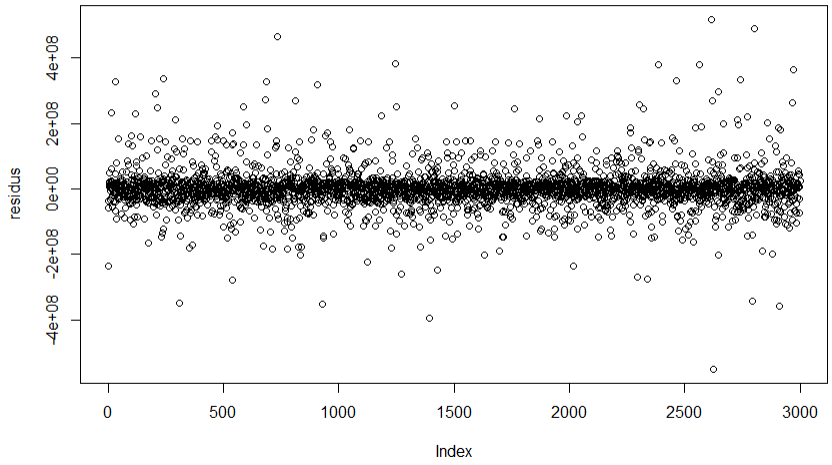

Figure 11: Err_1 with respect to the different films

The first method we tried was Random Forest. It has at first glance a very high R^2 indicator of 0.11 on the train data set. However, it decreases to the value 0.05252634 on the test data set whihch means that the RF overfits a lot. we tried several different parameter for this model and none of them did better than these results. Figure 12 shows how erratic Fandom forest rsiduals and predicton are compared to the XGBoost ones. THerefore, we focused on the latter model rather than Random Forest.

For the Gradient Boosting method, we carefully picked the best parameters maximizing our RMSE. Actually, we did not choose the R^2 criterion because the particular XGBoost model had worse performance than the XGBoost model taken with the RMSE criterion. We tried 40 different combinations

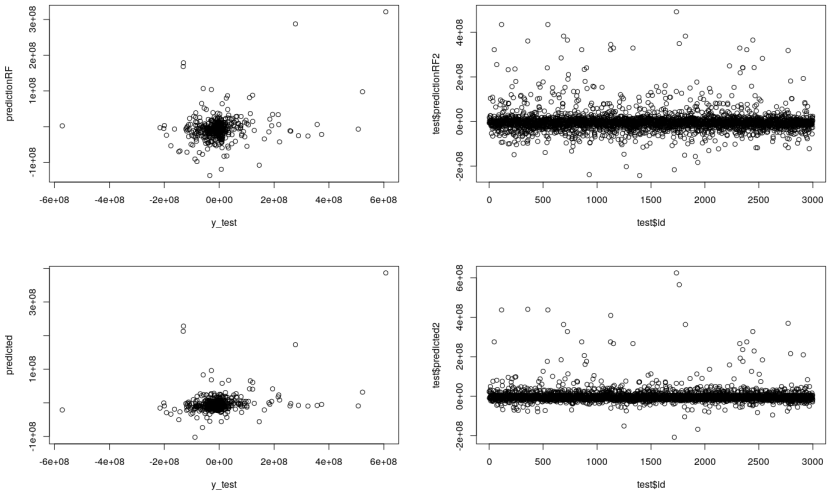

Figure 12: Predicted values and residuals with respect to the films Top : Random Forest Bottom : XGBoost

of parameters and then pick the best ones in order to maximize our final performance indicator (See table 5). The final parameters were nrounds = 100, max depth = 5, $\eta = 0.1$, $\gamma = 0$, colsample bytree = 0.5, minchildweight = 1 and subsample = 1. The final R^2 was 0.099 on the test data set.

We also looked at the difference in the variable selection induced by these methods. In both of the two methods, collections have a very important impact on the results, then some languages (English for instance) and some genres (comedy). This is not quite a surprising fact that these two methods have similar importance plot. However, having the collection as the most important variable is quite non-explainable as language or genre seemed to be more important to our eyes.

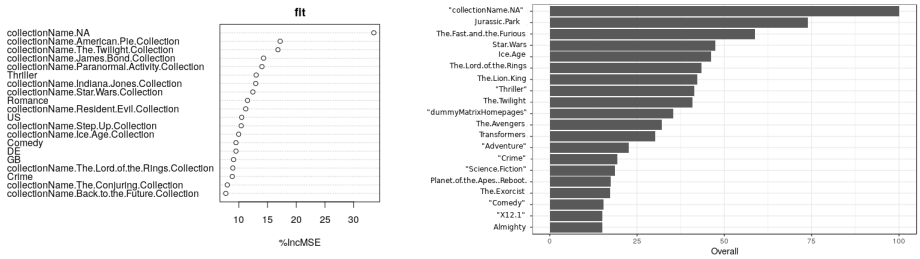

Figure 13: Importance of variables in Top: RF Bottom: XGBoost

Finally, we plotted the predicted values in comparison to the true values. It is clear that these values are very close to each other. Indeed, nothing is very surprising in that fact: the two predictors on Category 2 are all heavily relying on the predictor on category 1, the difference is made of a slight refinement. However, the performance of the gradient boosting refined predictor are better than the others ($R^2 = 0.68$ against $R^2 = 0.62$ and $R^2 = 0.64$ for respectively GAM and GAM+RF models). Hence, we select the XGBoost model to predict residuals of the GAM.

max depth	colsample by tree	nrounds	RMSE	R^2	MAE
3	0.5	100	75081411	0.07850649	41344242
3	0.5	200	75106880	0.08575261	41543800
3	0.6	100	74925010	0.08210483	41238245
3	0.6	200	74954089	0.08856462	41414487
3	0.7	100	74830287	0.08448410	41184992
3	0.7	200	75018109	0.08836912	41429770
3	0.8	100	74874892	0.08372020	41197468
3	0.8	200	74986201	0.08940739	41387728
3	0.9	100	74815554	0.08535706	41149674
3	0.9	200	75017963	0.08985098	41378105
5	**0.5**	**100**	**74562806**	**0.09614171**	**41291633**
5	0.5	200	74968610	0.09721221	41737751
5	0.6	100	74923723	0.09251095	41481111
5	0.6	200	75454629	0.09259327	41987674
5	0.7	100	74899286	0.09160295	41433248
5	0.7	200	75434186	0.09195593	41891138
5	0.8	100	75204729	0.08994527	41654823
5	0.8	200	75747457	0.09020351	42121346
5	0.9	100	74890340	0.09427961	41547726
5	0.9	200	75423258	0.09470117	41960755
10	0.5	100	75217601	0.10145411	42079318
10	0.5	200	76013745	0.09936467	42826428
10	0.6	100	75776222	0.10014593	42604473
10	0.6	200	76431228	0.09853525	43221212
10	0.7	100	75810474	0.09992112	42740492
10	0.7	200	76506137	0.09863502	43354981
10	0.8	100	76313676	0.09512347	42869369
10	0.8	200	77016945	0.09298457	43447591
10	0.9	100	76891189	0.09282396	43177001
10	0.9	200	77559743	0.09226333	43740308
15	0.5	100	76143453	0.10584606	43456850
15	0.5	200	76845544	0.10369244	44168629
15	0.6	100	76558469	0.10042819	43585337
15	0.6	200	77235207	0.09905981	44308735
15	0.7	100	77244789	0.09563022	44075652
15	0.7	200	77871813	0.09391503	44724674
15	0.8	100	78367858	0.09216501	44483246
15	0.8	200	79114492	0.09010209	45178833
15	0.9	100	79414766	0.08568495	45097916
15	0.9	200	80037370	0.08550412	45803821
20	0.5	100	77024372	0.10040836	44435972
20	0.5	200	77965966	0.09797956	45338992
20	0.6	100	77567436	0.10164243	44839312
20	0.6	200	78594856	0.09864081	45915545
20	0.7	100	78599985	0.09359129	45609782
20	0.7	200	79441045	0.09166335	46467701
20	0.8	100	79504973	0.09258396	46138587
20	0.8	200	80608447	0.08896890	47111949
20	0.9	100	81931066	0.07903982	47232302
20	0.9	200	82680682	0.07792140	47928800

Table 5: Determination of the best parameters for the XGBoost model

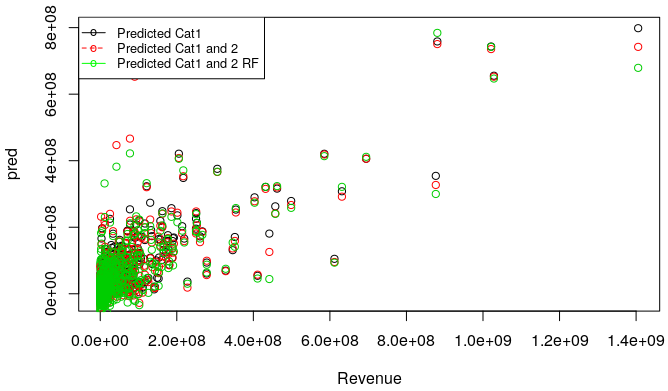

Figure 14: Comparison of the prediction made by (i) predictor based on Category 1 (black) (ii) predictor based on category and residuals with XGBoost (red) (iii) predictor based on category and residuals with Random Forest (green)

5.2 Category # 3

Having predicted the residuals of the first ML method, we are now willing to fit the residuals of these residuals that we noted Err_2. The last variables we have at our disposal and that were note used in any of the previous ML models, are the third and last category of variables, containing $crew$, $actors$ and $keywords$. All of types of variables are are 0-1 variables and form a very sparse and heavy dummy matrix (3000 films times 85000 variables). Hence, our methods will tend to reduce this really high dimensionality and in meantime, predict values of residuals.

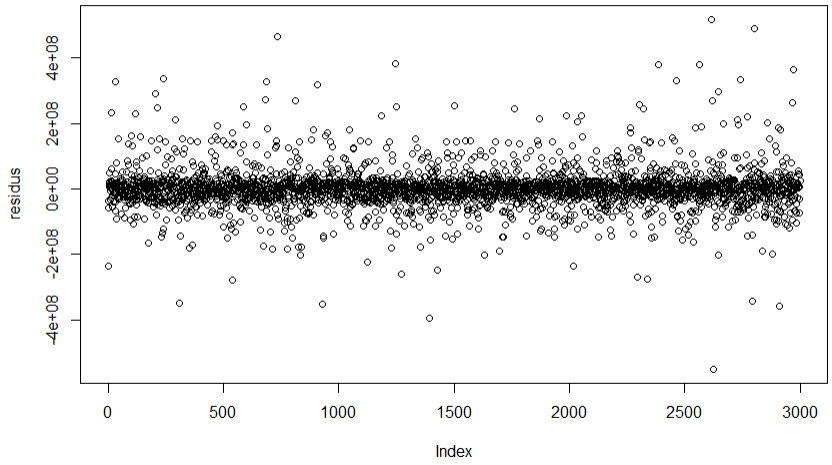

Figure 15: Regression tree for Prediction using Category # 1

We plotted the Err_2 vector in Figure 15. This is obviously very noisy and that fact is very reassuring. As we previously bootstrapped two previous models in order to predict this Vector of residuals, our first hypothesis was that this residuals were only the noise left. As the noise in our model is a white noise, these residuals should follow a Gaussian distribution. To check that statement, we used a `shapiro.test` to statistically test if the distribution of the residuals was Gaussian. We rejected the null hypothesis that was 'the residuals were sampled from a Gaussian distribution' and obtained a very low p-value of $2.2 \times 10^{-}16$. Hopefully, these residuals carry information. This is conditioned to the fact that these residuals were not sampled from another noise distribution.

5.2.1 Dimensionality reduction

In this setting, we are now facing the curse of dimensionality as the number of variables we are dealing with exceeds 85000 variables. Therefore, a first solution considered was to reduce the dimension of our data and then to apply any ML model that might be efficient. The stregy we adopted is similar to the very first attempt. We wanted to directly run a Random Forest. However, it appears that if the input matrix exceeds 30 Gigabytes, running a Random Forest is not possible. Our data matrix size is $\sim$ 500 GB, any direct application of any ML model is doomed. Accordingly to that remark, a Lasso method was used to reduce and select variables. We use the `glmnet` package to cross-validate a fitted lambda for our lasso (Figure 16). The form of the obtained curve does not have the ordinary and expected shape. The optimal λ is almost zero, and any lambda would select the same 3 variables. This is probably due to the very noisy behavior of the residuals Err_2.

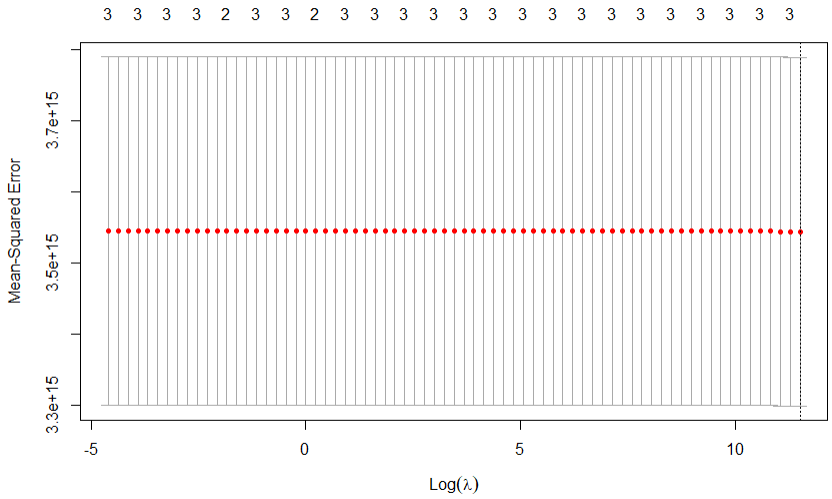

Figure 16: Cross validation for our Lasso # 1

5.2.2 Random Forest

We now apply a Random Forest model on the first 1000 variables of our lasso output to confirm whether or not 3 variables are sufficient to explain all the features. We plotted in Figure 17 the importance of variables regarding the random forest, to see whether or not the selection of these 3 variables is truly significant. Only three variables have importance in the output. Therefore, working on only those three seems legit and justified.

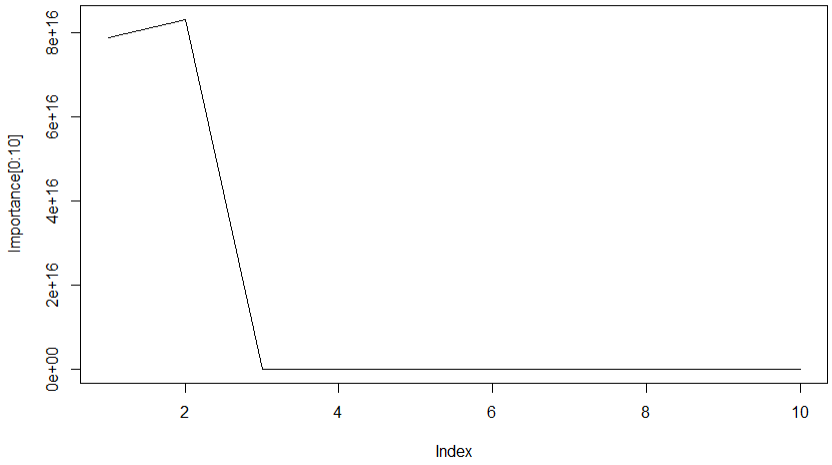

Figure 17: Relative importance of variables in Random Forest

We decided to run a Random Forest on those three variables as they are very sparse and 0-1 vectors. Indeed, the tree structure will capture the behaviors of our residuals and will fit well the different parameters. RF were preferred to regression trees as they introduced randomness and robustness in the prediction. Obtained R^2 on the test set is 0.2. This value is quite low. However, we are fitting residuals of residuals that are very erratic and noisy. Nonetheless, this will still help us improve and our global model as we are now able to predict residuals of residuals. As a sanity check, we plotted the error obtained for different number of trees. It is a relief to observe that the error decreases with the number of trees.

5.3 Global performance of our final model

To process these three categories, we separated data into two samples of movies: the training set Train (of size 2500), to fit the models, and the testing set Test (size 500), to evaluate how well our models behave on new data.

Likewise the beginning of this section, denote f_1, f_2, f_3 these three models. We now need to compute the R^2 of these models applied on our testing sets, thanks to the following formula:

$$R^2 = 1 - \frac{\text{sum((pred-actual))}^2)}{\text{sum((actual-mean(pred))}^2)}$$

where pred is the predicted value for the revenue, and actual the real revenue. Depending on the category $i = 1, 2, 3$, the predicted value will be $\sum_{1 \leq k \leq i} f_i(\text{Test})$. Thus, the R^2 after each step in iterative strategy we described in part II is easily calculable (see table 6).

This quantity R^2 is increasing, meaning that each category made a none trivial contribution to the result. Each step managed to refine our revenue prediction in a better way. Indeed, as stated by the *Kaggle* community in this competition, many models could permit to achieve a R^2 of 0.6. However, more refinements were needed to improve this.

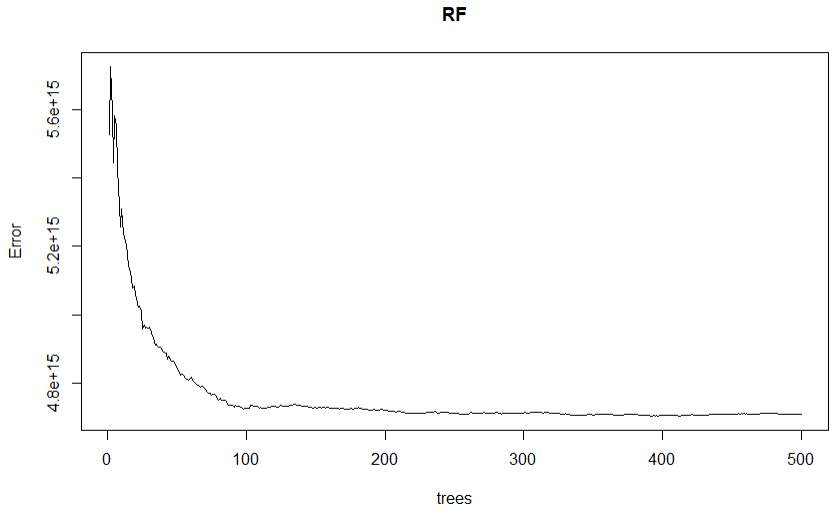

Figure 18: Random Forest errors 1

f_1(Test)	f_1(Test) + f_2(Test)	f_1(Test) + f_2(Test) + f_3(Test)
$R^2 = 0.63$	$R^2 = 0.68$	$R^2 = 0.71$

Table 6: R^2 depending on the considered categories

6 Enhancing model performance by limiting the impact of data bias

As explained and described in the introduction, we probably face bias issues in the data. Indeed, we can recall, for instance, that 2282 out of the 3000 movies in our data set were produced in the US. Most of them are recent or very recent (2000's and 2010's). Huge budget movies do not make the same profit the same way small ones do. Even though some of our models (regression trees in particular) are able to handle and capture very different behaviors, they are insufficient. The GAM regression on the first category could be of better accuracy if we eliminate or reduce bias at first in the data set.

In order to refine our models and to quantify the importance of these biases, we used subsets of the entire movie data base. These subsets take into account features that are susceptible of being biased. Subsets we created were: recent movies (2000's and 2010's), American movies, High budget movies (more than \$2 Millions). On every of these subsets done, we run the CAT1 procedure (a GAM model) and compute the R^2 indicator. We would like that the variance explained will significantly increase in the testing sets. However, that was not the case for every variables, as the following table shows (table 7).

Selected Movies	US	Budget>1M	Budget<2M	Short (<90min)	Recent(2010-2020)
Size	2282	2012	1016	413	952
R^2	0.54	0.57	0.55	0.57	0.65

Table 7: R^2 and sizes of our model on different subsets

Our R^2 obtained by considering all movies was of 0.63. We did not manage to obtain better indicators for most of the subsets. This is not of particular surprise as revenues and budgets, in dollars, do not have the same meanings through time. Only the subset made of the recent films outperforms our first model.

A way to alleviate these burdens could be to take into account the price of the dollar related to time. This will partially solve these issues. Furthermore, a study of the total amount of money poured in the movie industry each year (an exponentially increasing quantity) could have been of use here, for the same reason.

7 Conclusion

In this project, we wanted an original topic enabling us to apply several ML models. The Movie Data Base (TMDB) data set was our unique source of information. This data set was messy and needed a lot of pre-processing work before applying any ML model on it. After this first line of work we undertake a first baseline with a GAM. To refine it, we then modeled two times in a row the residuals of the previous model and residuals of the residuals. Our final is made up of a GAM plus a Gradient boosting algorithm (XGBoost) and finally a Random Forest (RF). The final R^2 is 0.71 which is way higher than the average 0.6 appearing in the *Kaggle* competition homepage.

Our main lines of work in this project was:

1. Pre-process the data set
2. Derive an efficient Strategy to tackle this prediction problem
3. Apply this strategy to obtain a model as fine as possible
4. Critically review the results and enhance performance even more.

We really want to thank Yannig Goude and the other groups of the M2 StatML for all the fruitful discussion we had.