Statistical Learning project: "A successful Kickstarter campaign"

Riccardo Sturla 13457A

September 2023

Contents

1 Abstract

This project requires answering a research question through the use of supervised and unsupervised learning techniques. My question is "is it possible to predict the success of a Kickstarter campaign using statistical learning methods?". For the supervised part, I will use different models in order to find the best one considering both accuracy and interpretability, while for the unsupervised method, I will use k-means clustering in order to see if the algorithm is able to "catch" the relations between the most important variables.

2 Introduction - about Kickstarter

Kickstarter is an American public benefit corporation with a crowdfunding platforms for gathering money from the public, which circumvents traditional avenues of investment. The company's stated mission is to "help bring creative projects to life". People can put their idea on the platform and collect money to fund it: they choose a deadline and a minimum funding goal. If the goal is not met by the deadline, no funds are collected.

3 Data

3.1 Dataset

The dataset used for the analysis can be downloaded here. It is composed of 378661 observations and 15 columns; the variables are:

1. **ID**: ID code of the project;
2. **Name**: title of the project;
3. **category**: category of the project (ex. Technology, Art...)
4. **main category**: more general category;
5. **currency**;
6. **deadline**;
7. **goal**: minimum amount of money to be reached to consider the project successful;
8. **launched**: date in which the campaign is launched;
9. **pledged**: amount of money actually pledged;
10. **state**: categorical variable that identifies if the project is successful, failed or cancelled;
11. **backers**: number of backers;
12. **country**
13. **usd pledged**: conversion in US dollars of the pledged column (conversion done by kickstarter).
14. **usd pledged real**: conversion in US dollars of the pledged column (conversion from Fixer.io API).
15. **usd goal real**: conversion in US dollars of the goal column (conversion from Fixer.io API).

3.2 Data cleaning and preprocessing

First of all, I checked for NA values and I deleted them together with some clear errors in the data (projects launched in 1970), then I removed redundant columns, useless for our analysis, keeping only the amount of money in US dollar converted with the API (usd goal real, usd pledged real). I also removed projects whose "state" was not "successful" or "failed", because they don't give us useful information for our analysis. Then I split the launched date in launch month and launch year, in order to have two more categorical variables to study, and I also computed the duration of the campaign as the difference between "launched" and "deadline". "Currency", "ID", "name" and "deadline" are now dropped. Finally I plotted the data to drop some more outliers and encoded "state" into a dummy variable.

3.3 The "backers" variable

An important question for the analysis is if the number of backers should be included in our classification problem. In fact "backers" is a variable known only after the campaign is over, so the answer to our question changes based on our goal:

- Yes, it should be included when we want to build a model to classify projects with the same data structure, but no "state" variable.
- No, it should be not included if we want to make a prediction about the success of our project, *before* launching the campaign.

We want to obtain both the results, so we are going to carried two analysis, one including the variable and one without.

4 Data visualization

I decided to plot some variables to better understand their relations and see if they could have a significative impact on the success of a projects. In order to do so, I computed the average goal for each category and the success rate.

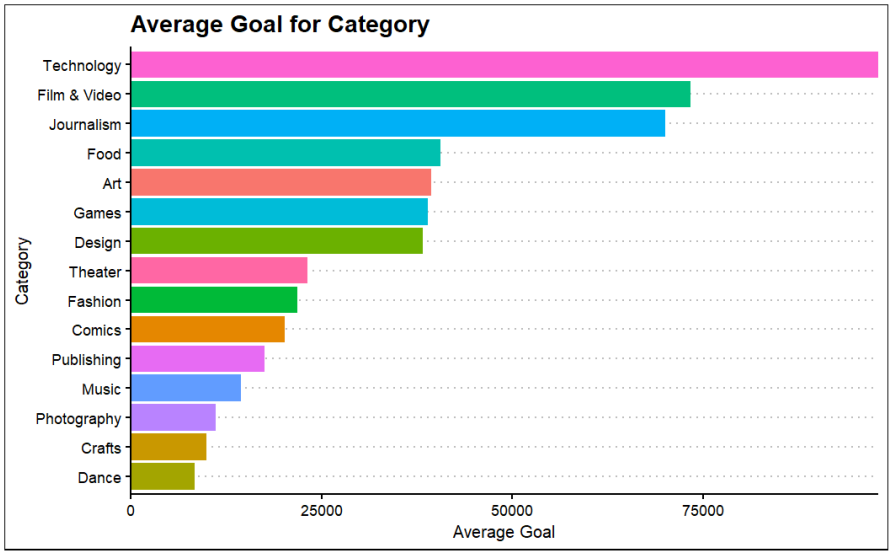

Figure 1: Average goal for category

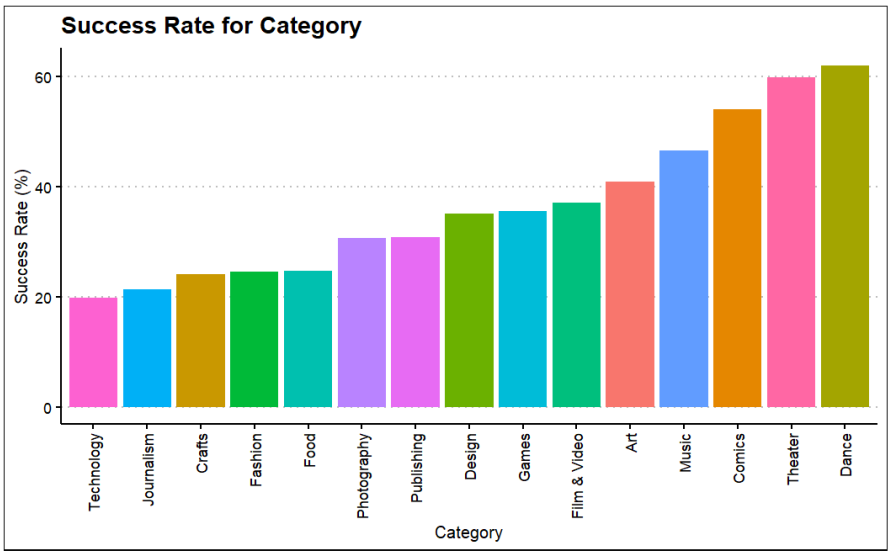

Figure 2: Success rate for category

In Figure 1, we can see that there are some categories with much higher goals and in Figure 2 some of them have the lowest success rates: this could be a relevant correlation.

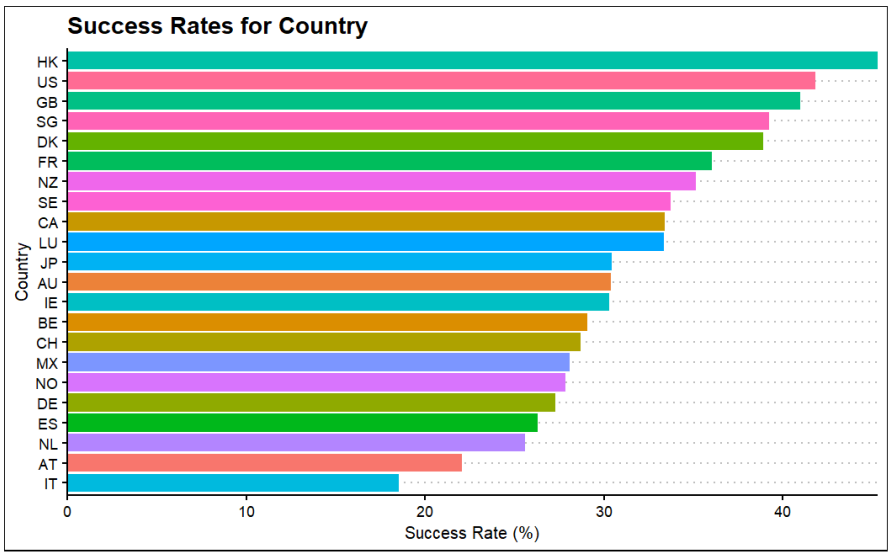

Figure 3: Success rates for country

In Figure 3 we see how some countries have higher success rates than others, showing a potential pattern. These were just a few insights to see if some variables could be actually relevant for the analysis carried on later.

5 Supervised learning

5.1 Goals and Objectives

As said in the abstract, my goal was to see if the success of a crowdfunding campaign could be predicted using statistical learning methods. In order to achieve that, I've carried an analysis for both the cases listed in 3.3, to see if the algorithms can correctly classified the projects knowing or not how many backers there are.

5.2 Logistic regression

Dealing with a binary classification problem, the first supervised model that comes to mind in terms of simplicity and efficiency is probably the logistic model. So, after splitting the data in training and test sets, I implemented it for both the cases (backers and without backers).

```
Call:
glm(formula = state_bi ~ main_category + usd_goal_real + duration +
    launch_year + launch_month + country + backers, family = "binomial",
    data = ks_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-8.4904  -0.5422  -0.0284   0.2971   8.4904

Coefficients:
                           Estimate Std. Error  z value Pr(>|z|)
(Intercept)              -1.330e-02  2.312e-01   -0.058 0.954131
main_categoryComics      -5.219e-01  4.317e-02  -12.088  < 2e-16 ***
main_categoryCrafts      -6.333e-01  4.343e-02  -14.583  < 2e-16 ***
main_categoryDance        6.973e-01  5.650e-02   12.341  < 2e-16 ***
main_categoryDesign      -6.171e-01  3.475e-02  -17.757  < 2e-16 ***
main_categoryFashion     -5.720e-01  3.363e-02  -17.007  < 2e-16 ***
main_categoryFilm & Video 2.032e-01  2.465e-02    8.241  < 2e-16 ***
main_categoryFood        -3.992e-01  3.484e-02  -11.460  < 2e-16 ***
main_categoryGames       -1.505e+00  3.684e-02  -40.862  < 2e-16 ***
main_categoryJournalism  -5.966e-01  6.274e-02   -9.510  < 2e-16 ***
main_categoryMusic        1.234e-01  2.508e-02    4.923 8.54e-07 ***
main_categoryPhotography -3.924e-01  3.995e-02   -9.822  < 2e-16 ***
main_categoryPublishing  -5.104e-01  2.771e-02  -18.415  < 2e-16 ***
main_categoryTechnology  -5.660e-01  3.803e-02  -14.882  < 2e-16 ***
main_categoryTheater      7.117e-01  3.791e-02   18.771  < 2e-16 ***
usd_goal_real            -2.253e-04  1.514e-06 -148.809  < 2e-16 ***
duration                 -1.419e-02  5.335e-04  -26.594  < 2e-16 ***
launch_year2010          -3.596e-01  9.721e-02   -3.699 0.000217 ***
launch_year2011          -3.604e-01  9.454e-02   -3.813 0.000138 ***
```

Figure 4: Some of the Logistic Regression coefficients

Due to the presence of a lot of categorical variables, the list of coefficients is extremely long (in Figure 4 only a few are reported) and this lead to a difficult reading and interpretation of the model.

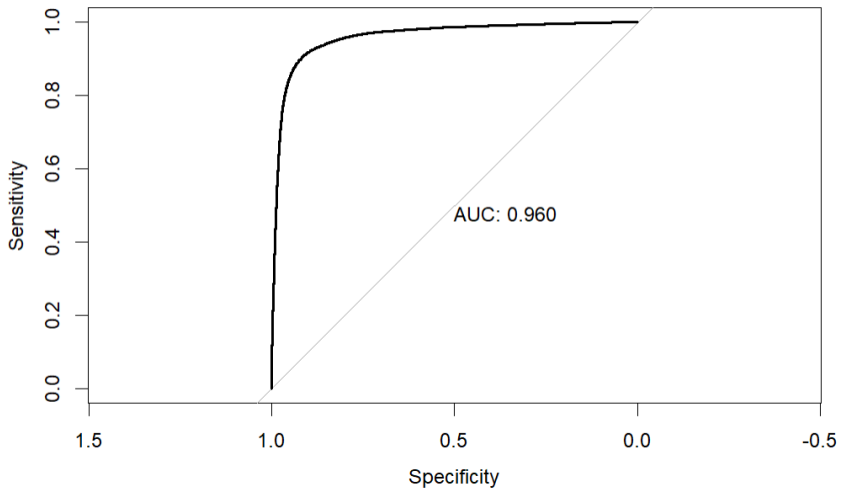

Figure 5: ROC curve and AUC

```
                Predicted_Value
Actual_Value  FALSE   TRUE
           0 131893   6252
           1  15407  78262
[1] 0.9065673
```

Figure 6: Confusion matrix and accuracy

The accuracy of the logistic model is 0.907: it is a very good method but, in this particular case, not so easy to interpret.
Running the algorithm without "backers" leads to a significative lower accuracy of 0.65, showing how important the missing variable is to predict the output.

```
                Predicted_Value
Actual_Value  FALSE   TRUE
           0 111157  26988
           1  54355  39314
[1] 0.6491023
```

Figure 7: Confusion matrix and accuracy (without backers)

5.3 The Lasso

From now on, for computational reasons, a smaller random sample of the dataset is going to be used.
Due to the complexity of the logistic model output, I've tried the lasso, a shrinkage method, hoping to put into effect a feature selection and exclude some of the less relevant variables. I've proceeded with the tuning of the lambda parameter for both the cases.

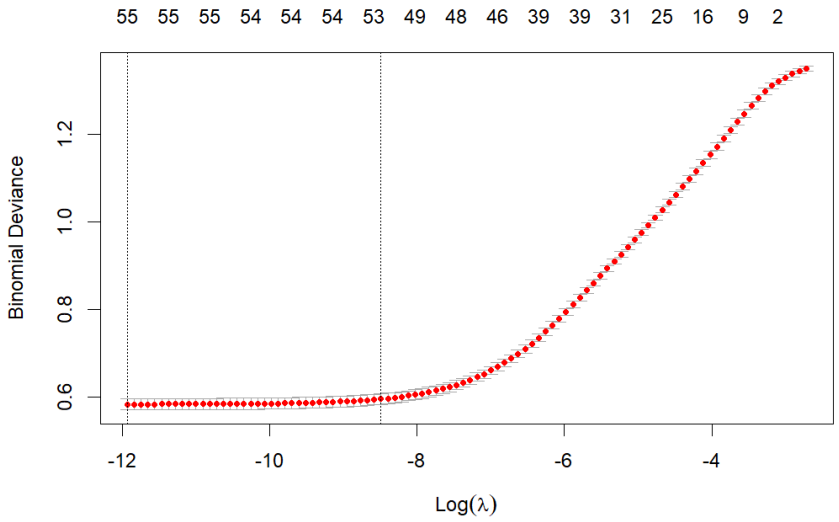

Figure 8: Tuning Lambda

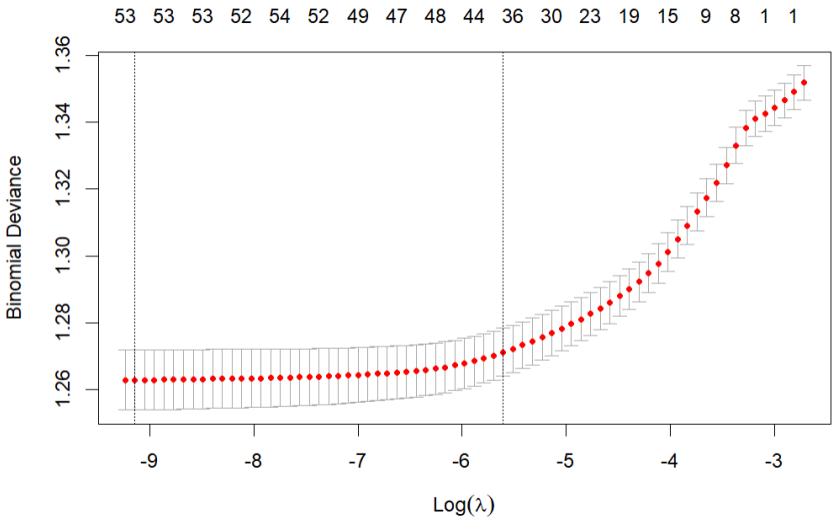

Figure 9: Tuning Lambda (no backers)

Then, I've run the models with the best lambda and looked at their performances.

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1674  185
         1   80 1025

               Accuracy : 0.9106
                 95% CI : (0.8997, 0.9206)
    No Information Rate : 0.5918
    P-Value [Acc > NIR] : < 2.2e-16
```

Figure 10: Confusion matrix and accuracy

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1414  679
         1  340  531

               Accuracy : 0.6562
                 95% CI : (0.6388, 0.6733)
    No Information Rate : 0.5918
    P-Value [Acc > NIR] : 3.295e-13
```

Figure 11: Confusion matrix and accuracy (no backers)

There was a little enhancement in the accuracy in both cases, but, contrary to what I hoped, there was not a decent feature selection and only a few coefficients were shrunk to 0. For these

reasons, I decided to try a model that is usually easy to interpret and to visualize: the decision tree classifier.

5.4 Decision Tree

Decision tree classifiers are simple to understand and to interpret. They can also be visualized. To build a tree, I started creating a very large one then I pruned it until a certain threshold of Cost of Pruning *CP* was reached. The model obtained does not overfit and the error can be controlled.
I implemented a tree for both cases and tested their performances.

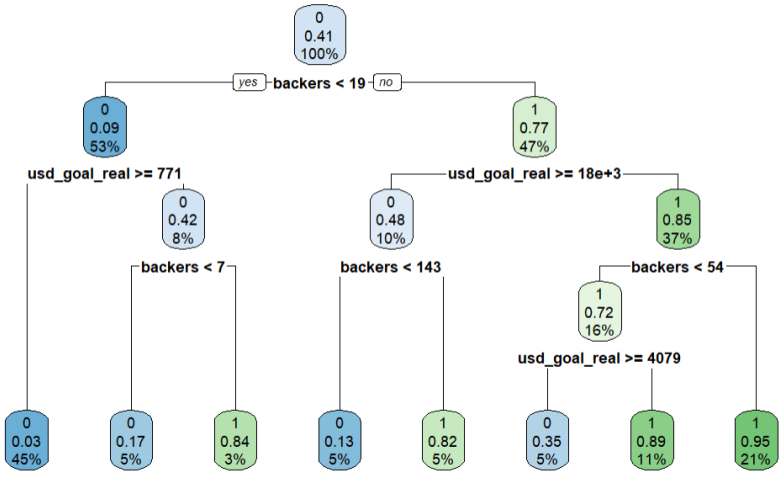

Figure 12: Tree with backers

The tree in Figure 12 looks too simple, it considers only the two variables "backers" and "goal" in its partitions, showing how much important they are, but at the same time reducing the real complexity of the problem. For these reasons, it's a good model to visualize and interpret, but I would prefer a more "complex" one, even if the accuracy is good (0.91).

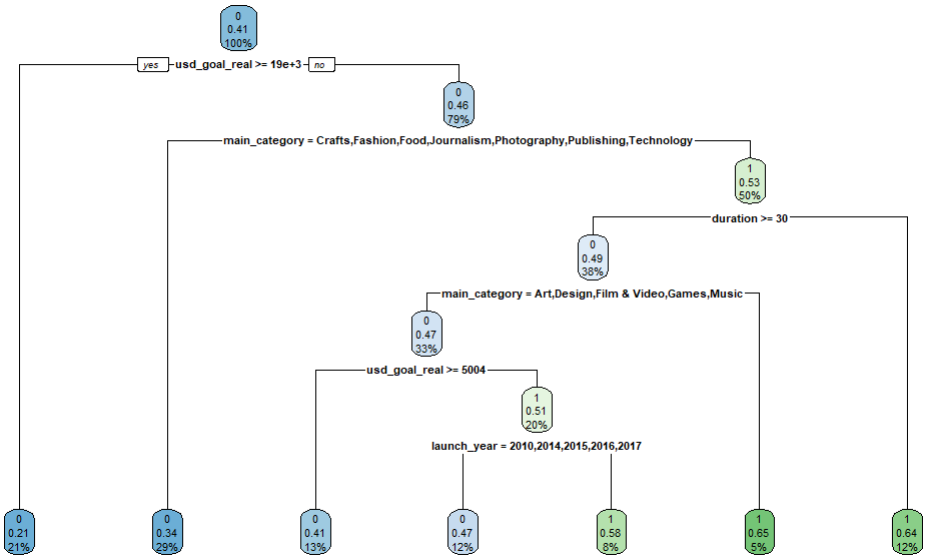

Figure 13: Tree without backers

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1507  735
         1  247  475

               Accuracy : 0.6687
                 95% CI : (0.6514, 0.6856)
    No Information Rate : 0.5918
    P-Value [Acc > NIR] : < 2.2e-16
```

Figure 14: Confusion matrix and accuracy without backers

I was instead very impressed by the tree in Figure 13, because of its performance (accuracy of 0.67, higher than any other model), but also for how good it looks. It considers many of the variables and it is very easy to interpret, even with the presence of so many categorical features.

5.5 Random Forest

Random forest is a powerful tool and it usually performs better than the tree classifier, due to the fact that it combines more trees together to improve the precision.
Fist of all, I tuned the *mtry* hyperparameter which identifies the number of variables randomly chosen every time to split a node, and then I tuned the best number of trees, plotting the *OOB* error and looking when it was minimized.

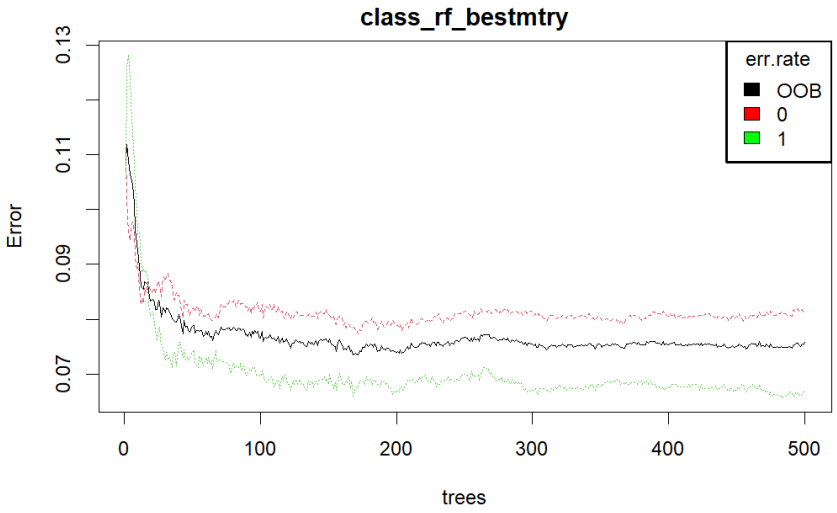

Figure 15: OOB error and number of trees

I've built the model with the tuned hyperparameters and I've evaluated the performances with the confusion matrix.

```
          Reference
Prediction    0    1
         0 1608   92
         1  146 1118

               Accuracy : 0.9197
                 95% CI : (0.9093, 0.9292)
    No Information Rate : 0.5918
    P-Value [Acc > NIR] : < 2.2e-16
```

Figure 16: Confusion matrix and accuracy

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1427  653
         1  327  557

               Accuracy : 0.6694
                 95% CI : (0.6521, 0.6863)
    No Information Rate : 0.5918
    P-Value [Acc > NIR] : < 2.2e-16
```

Figure 17: Confusion matrix and accuracy without backers

In both cases, the accuracy is the highest among all the models (0.92 and 0.67), but for the one without the backers variable, the increase is so small that I would still prefer to use the single tree classifier, due to its interpretability. For the first model, there is no doubt that random forest is the

best method to correctly classify the projects.
An additional advantage of building a random forest model is the possibility to measure the importance of every variable and how much the error would increase without that feature.

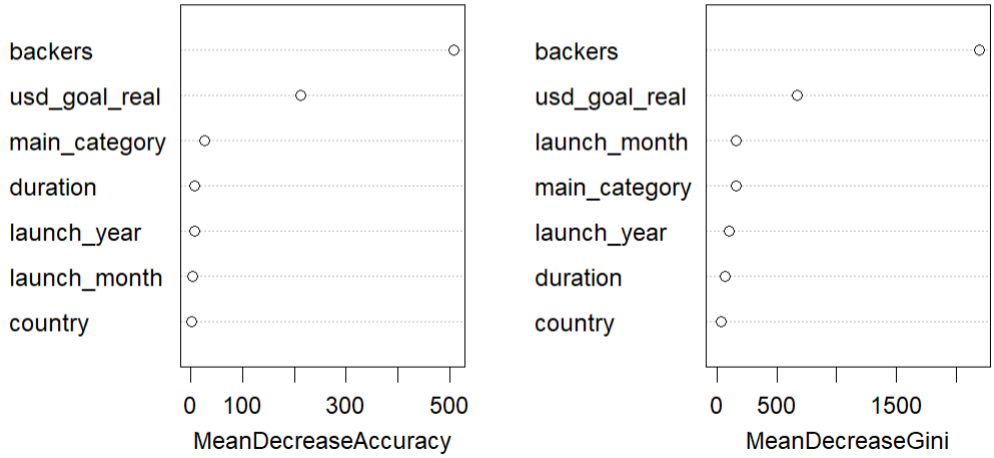

Figure 18: Importance of the variables in the model

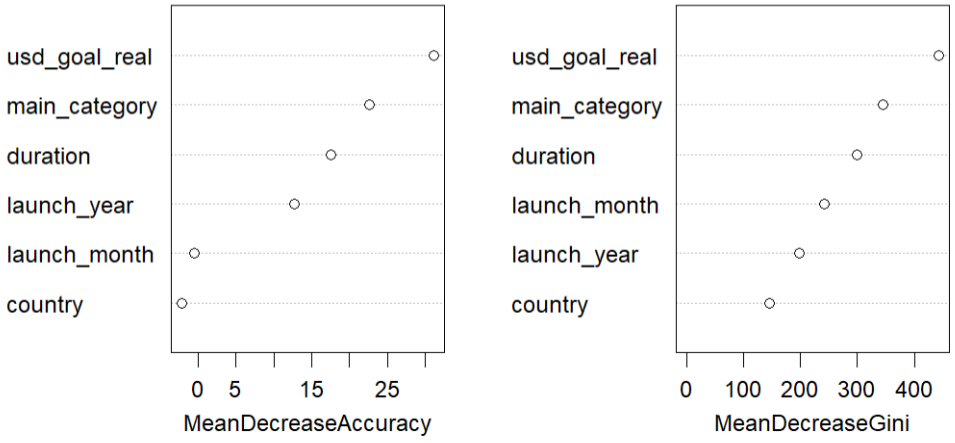

Figure 19: Importance of the variables in the model without backers

The *Mean Decrease Accuracy* plot expresses how much accuracy the model losses by excluding each variable. The *mean decrease in Gini coefficient* is a measure of how each variable contributes to the homogeneity of the nodes and leaves in the resulting random forest. The higher the value of mean decrease accuracy or mean decrease Gini score, the higher the importance of the variable in the model. As expected "backers" is the most important one in both the measures, followed by "usd goal".

5.6 Conclusions and Keypoints

After implementing and testing all the models, we can say that:

- Using the data to predict if a project is successful is possible and the best supervised model to use is Random Forest.
- The prediction of the success of a campaign, before knowing the number of backers is difficult. However, in my opinion, the best choice is the Decision Tree Classifier, because it is easy to interpret and it has one of the best accuracy.
- One way to improve the model and the prediction could be to include a sentiment analysis on the projects names.
- It is relevant to remember that data cannot "catch" everything. Even after choosing the best set of variables to improve the probability of success, the most important thing is the intangible idea behind the project.

6 Unsupervised learning

6.1 Goals and Objectives

- See if clustering observations using the most important variables leads to the right classification for successful and failed projects
- We are not making a prediction, but we want to see if the relations that the algorithm finds are in some way significant, knowing the real labels

6.2 K-means with two variables

For the unsupervised learning methods, I have used a small random sample of the data (about 1000 observations), due to visualization and computational reasons.
I have chosen k-means clustering because I find it an easy but powerful method to implement, moreover I have used only scaled numerical variables, so there was no need for more "complex" algorithms. For the first analysis, I have tried to cluster the data including only "backers" and "usd goal real", which, as we have seen in the supervised learning part, are the most relevant variables to classify the projects. After scaling the values and removing some outliers, I have plotted the graphic (Figure 20) showing the theoretical best number of clusters, i.e. the k. However our goal is to see if clustering could lead to a correct classification, in order to do so I used $k = 2$, even if it wasn't the optimal value.

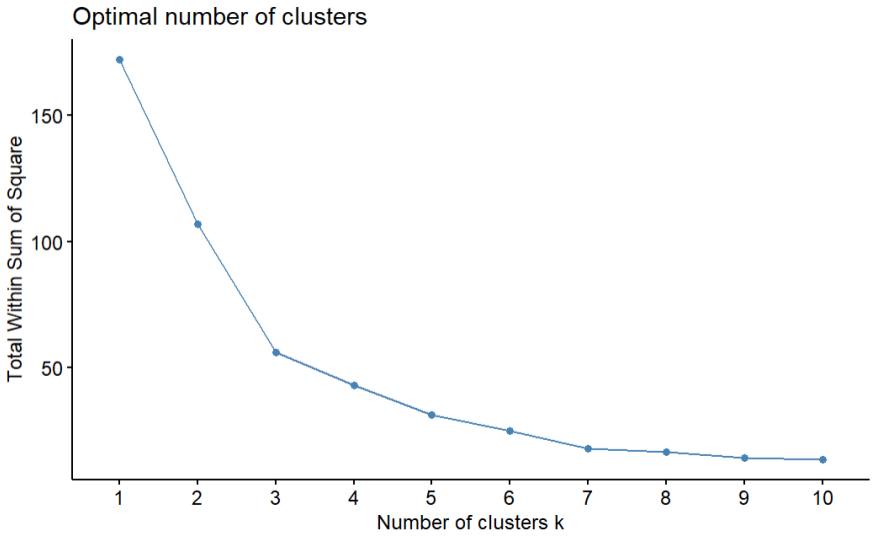

Figure 20: Best number of clusters

After that I performed the k-means and plotted the results. Moreover, I compared the output with the correct label of each observation, creating a sort of confusion matrix.

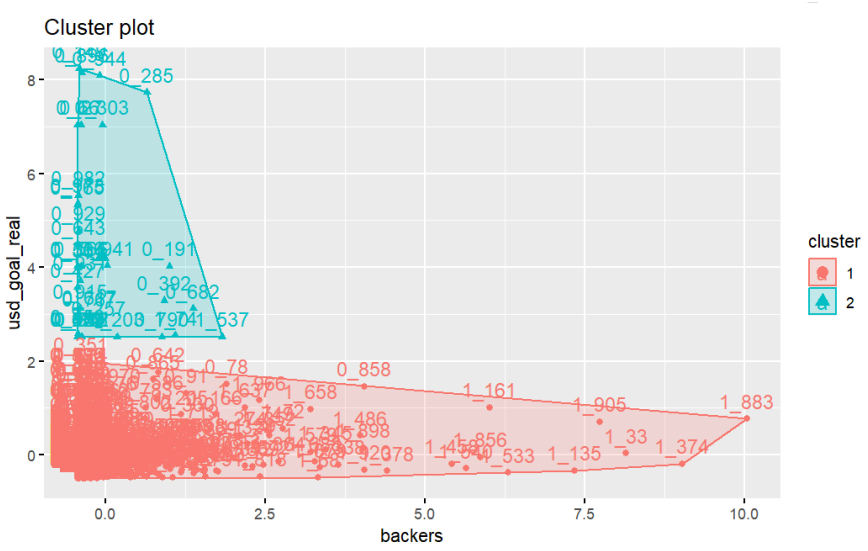

Figure 21: K-means clustering: backers and goal

```
km.clusters   0   1
          1 590 356
          2  34   2
```

Figure 22: Confusion matrix

The clusters result to be basically a division "*low goal - high goal*", but obviously that does not correspond to successful and failed projects, so I've tried to add some complexity.

6.3 K-means with three variables

In this case, I did the same thing as before, but I added the variable "duration" (scaled) in the analysis. So I plotted the clusters and computed the confusion matrix.

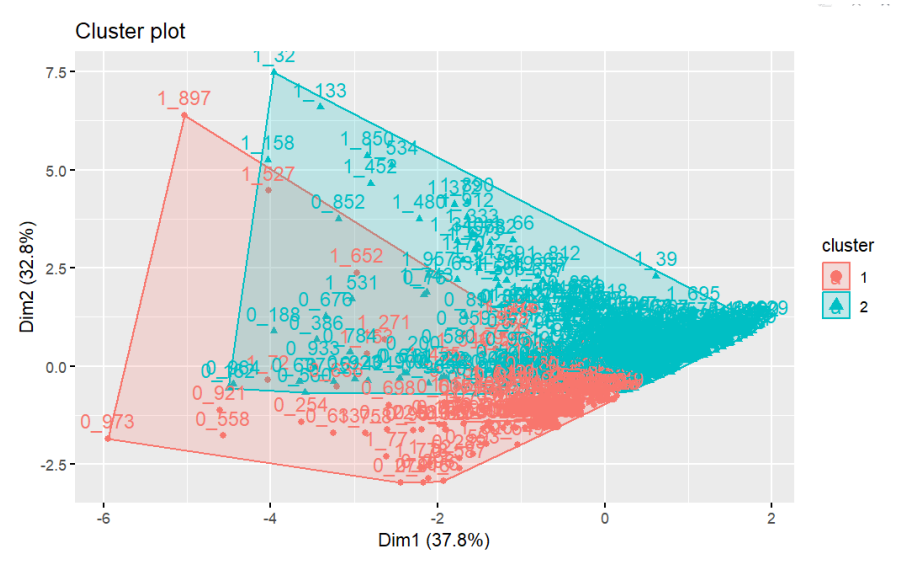

Figure 23: K-means clustering: backers, goal and duration

```
kmd.clusters   0   1
           1 134  55
           2 483 301
```

Figure 24: Confusion matrix

This time the clusters are more precise and can split some of the data according to their "state". Obviously, that is not enough to say that k-means clustering is able to catch the relations between the variables and classify the data correctly. This is probably due to the fact that there are some "anomalous" observations, for example campaigns with a lot of backers, but a very small goal and

vice versa, but surely also due to the fact that the original problem is more complex and includes many more variables.

6.4 Conclusions and Keypoints

- Only "duration", "backers" and "usd goal real" are used in the analysis
- K-means clustering is not able to group the projects according to their "state"
- There are "anomalous" data which could influence the correct clustering
- Anyway, clustering could be used for other purposes, such as find patterns and relations in the data

7 Appendix

This project was realized using R and all the code can be found on my GitHub: RiccardoSturla.