KickStarter campaigns success predictor

Jose Garcia-Garcia

14/9/2020

Introduction and motivation

Even we could think that crowdfunding is a quite modern concept which started in the XXI century, that is not strictly true. Crowdfunding had indeed been present in the human history since the 1700's. In fact, some example of crowdfunding campaigns during 18th and 19th century are:

- 1700s, The Irish Loan Fund - Established to provide loans to poor but creditworthy people in Dublin, wealthy citizens donated money as a way of charity. This campaign was so successful that by 1834, there were 300 loan funds in Ireland ("A Brief History of Crowdfunding" 2018)
- 1850s, Auguste Comte - Philosopher and founder of the positivism, turned himself into an independent philosopher whose expenses were funded by the public people following his works ("Crowdfunding Comte" 2016)
- 1885, Statue of Liberty pedestal - Although the Statue of Liberty was a gift from the French government, additional $250000 were necessary to build the pedestal. The committee tasked with raising the money fell short by $100000 and infamous publisher Joseph Pulitzer launched a campaign that raised the required amount from more than 160000 donors ("The Statue of Liberty and America's Crowdfunding Pioneer" 2013)

However, it was not until 1997, in the early days of the internet, where the inception of the modern online crowdfunding ocurred. A British rock band called Marillon used a mailing list that had around thousand of fans emails in order to raise money to be able to came to the US in tour. The idea was that fans will put the money into the band bank account and would be returned to them if not enough money was raised ("Progressive-Rock Band Marillion Pioneered Crowdfunding" 2016). The modern online crowdfunding was born at that moment.

Marillon was succesful and were able to go on tour through the US thanks to that campaign and the brand new online crowdfunding platforms started to emerge in the following years:

- ArtistShare, 2001 - First dedicated crowdfunding platform for musicians that wanted to raise money for recording an album or going on tour
- Indiegogo, 2007 - One of the first, and the most succesful at that time, sites to offer crowdfunding for a large list of categories and not only related to the music business
- Kickstarter, 2009 - The platform that become the leader in the next following years with more than 2 billions raised between 2009 and 2017. In conjunction with Indiegogo, Kickstarter led to an explosion of the crowdfunding platforms

In the following years, lots of new platforms appeared in the market. Some of them being global for all kind of projects like the mentioned Kickstarter or Indiegogo, and others covering different market niches, e.g. Patreon for creative individuals (artists, podcasters, musicians,etc...) or Charitable for charity donations.

During the last decade, crowdfunding has become more and more popular As a consecuence of this popularity and growth in number of platforms, projects and users, North America generates $17.2 billion each year in crowdfunding, while Asia generates $10.5 billion each year and Europe $6.5 billion ("Crowdfunding Statistics" 2020.)

Despite this growth in number of platforms and money raised, Kickstarter remains as number one in all the popularity rankings. As per September 2020, Kickstarter had raised $5.2 billion. In Kickstarter, 188000 projects had been funded successfully, which is approximately a 37% of the projects.

This raises the question of which are the factors that determine if a fundraising campaign in Kickstarter is going to be successful or is going to fail. Take into account that Kickstarter is an all-or-nothing platform, only the projects that are able to raise at least the goal target are considered successful, as for those which aren't, money is returned to the pledgers.

As a result of this question, **our motivation for this project consists on trying to predict, before starting it, if our campaign is going to be succesful or is going to fail**, hence we will need to modify some features inherent to our campaign, in order to have more chances on accomplish our goal as fundraisers.

Project overview

In order to be able to predict if a Kickstarter campaign is going to be successful, we need some historic data on previous projects.

In Kaggle we can find a dataset with information about 321616 campaigns on Kickstarter from its very first days until January 2018 (https://www.kaggle.com/kemical/kickstarter-projects) This dataset is provided by Kaggle user Mickaël Mouillé and is presented in CSV format, hence is quite easy to load and manipulate.

Note: There are 2 dataset versions in there, one with data until 2016 and the one that we are going to analyze, with data until January 2018.

Main information contained in the CSV file, as different columns, that we are going to analyze and use for our prediction model will be:

- Name of the project
- Main category and category features (this last one can be really considered as a kind of sub-category)
- Currency and country of origin of the project
- Launched and deadline dates of the project
- Usd Goal Real, which is the goal for the project expressed in US dollars. External conversion was already included in the CSV and money conversions directly provided by Kickstarter are not reliable.
- State of the project: succesful, failed, canceled, live, suspended, undefined

There are other data/columns included in the dataset but as the purpose of our model is to predict if our project is going to be succesful **before** releasing it, so we will not introduce into our analysis and model:

- Number of backers, as that cannot be find out until our campaign has finished
- Usd Pledged Real, for the same reason, is unknown until our campaign has finished
- Additional data that can be deduced from date information but only after finalizing our campaign, e.g succesful rate for month/year of campaign release

Steps that we are going to follow for analysing the data and fitting a machine learning model that can help us to predict if a campaign is going to be succesful or not are:

1. Clean our dataset, removing all the columns/predictors that are not necessary for sure for our model and additionally, removing inputs that are not helpful
2. Split the dataset into validation dataset (only used for final validation of our choosen model), training and test/cross validation datasets (only used for comparing different possible models and choose the best one)
3. Analyze the rest of remaining predictors and find out if is worthy to include them into our model and/or those predictors need some kind of transformation
4. Train different models/algorithms and decide which is the best one that fits to our problem
5. Present the final results of our model and present, as well, future lines of work in order to improve the accuracy of the model

Data analysis

Cleaning the dataset and data partitions

As we mentioned in previous chapter, before starting the data analysis, we need to clean our dataset.

As a first step, we are going to remove the columns that are not useful for our prediction and those are:

Column	Reason
Goal	Information in USD already contained in another column
Pleged	Money raised, only known after project has finished
Usd Pledged	Money raised in USD, only known after project has finished
Backers	Number of donors, only known after project has finished

Additionally, we are going to filter the rows in our dataset depending on the final state of the campaign. The reason is that there are some other examples with states that we shouldn't include in our analysis:

- Live - Projects that are still fundraising money, hence we don't know the final state (if they are going to be succesful or not)
- Undefined - Projects where information regarding if they were succesful or not has not been provided
- Canceled - Projects that were canceled by the author for whatever reason. As we cannot figure out if they were going to reach the goal or not, we discard them from our analysis
- Suspended - Projects suspended by Kickstarted because there is some violation of their terms and rules

Once we have cleaned our working dataset, we are going to create the different partitions that we need for training our model. Before filtering by state we had 378661 rows, and after that filtering we have 331675.

That amount is not a huge quantity of examples (e.g. in previous projects, like doing a Movie Recommendation System, we had several millions of training examples).

Taking that into account, we want to keep as maximum examples for training but without compromising a proper validation of the results. So, we have decided to split the dataset into:

- 10% for final validation dataset for evaluating the choosen final model
- 90% of the remaining examples will be used as training examples for different models
- 10% of the remaining examples will be used as test examples for comparing accuracy performance of different models

Success rate per category analysis

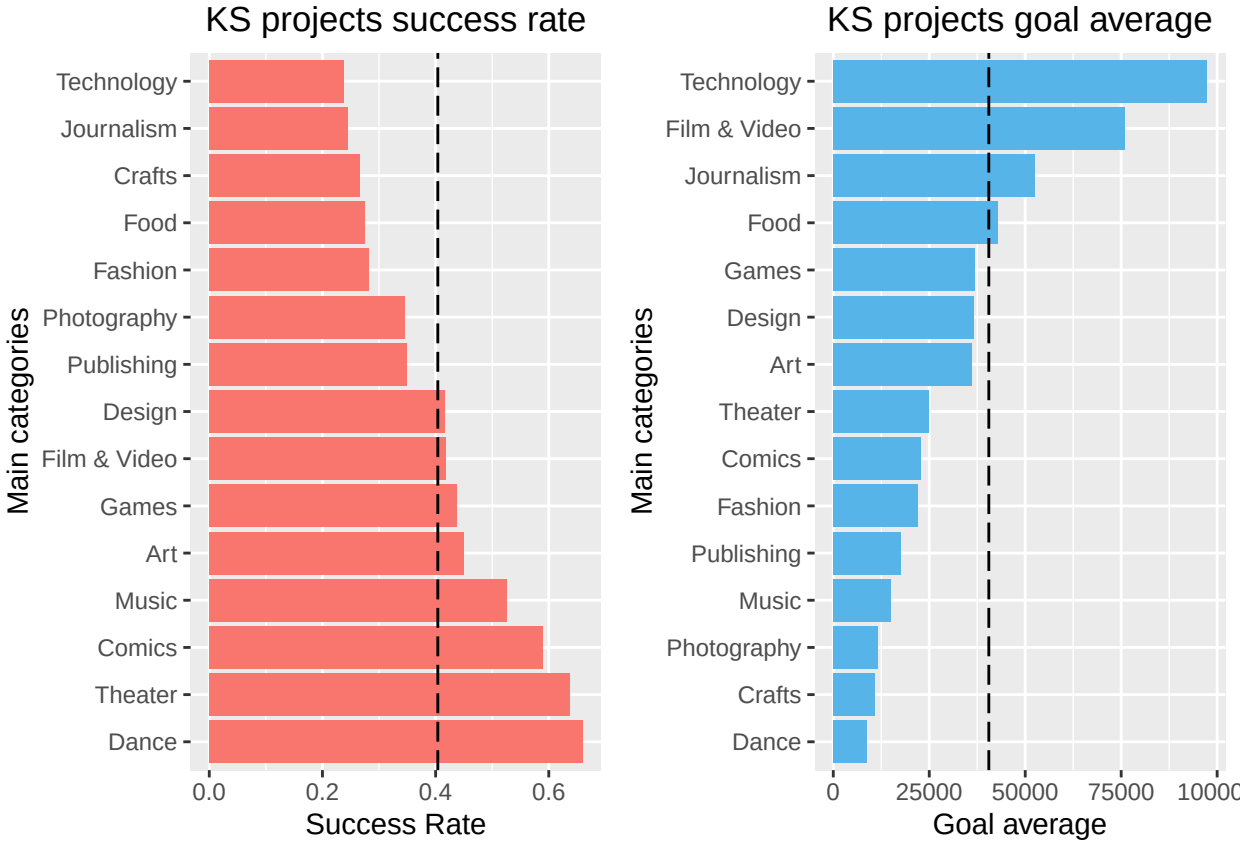
As a first possible factor to take into account in our prediction model, we are going to study if the category election is making any difference in the probabilities for a campaign being succesful.

Our hypothesis consists on considering that there are categories which the public could consider more interesting, hence would have a higher probability of being crowdfunded successfully than others.

In our analysis we will take into account as well the average goal amount required for each category. The purpose is to discard that differences of the sucess rate between categories are due to some projects in some categories (e.g. filming movies) requiring a huge amount of money compared to other "cheaper" projects in other categories (e.g. publishing books).

So, first step consists on calculate the average success rate and average money goal for all the campaigns and use it as a base for compare which categories are more or less successful than the average. The average success rate for our training dataset is 0.404 (40.4%), while the average goal is $40519.

Now we are going summarize by main category and analyze the success rate vs goal average per each main category.



If we take a look to the graphs, we can observe that:

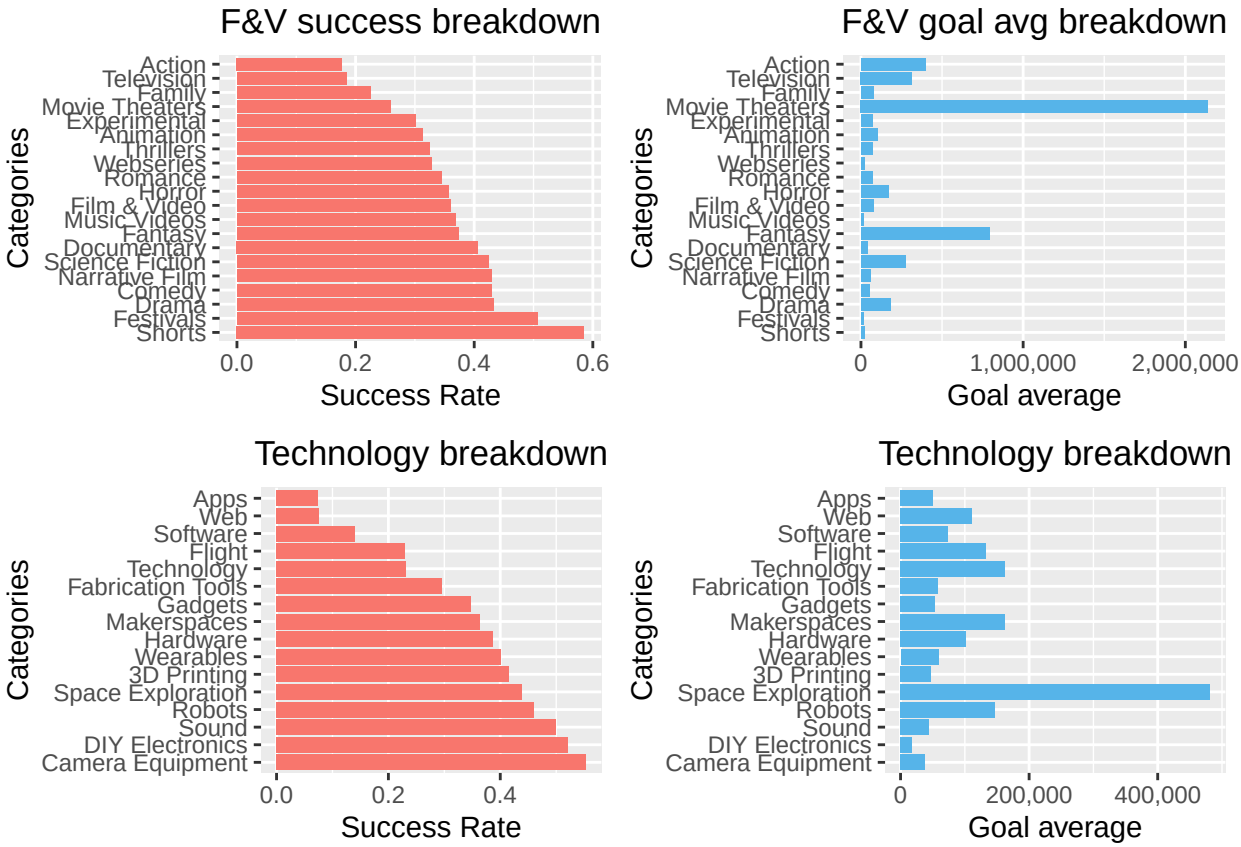
- There are important differences in the success rate between different categories. For example, the least succesful category, which is "Technology", has a success rate of 0.237 (23.7%), while the most succesful one, which is "Dance", has a success rate of 0.66 (66%)
- Although we can see some correlation between goal average, main categories and success rate, only for the ones that are at the ends (Technology, Journalism, Dance), for most of them there is no correlation. For example, "Games" is the 6th main category with higher success rate even if it is also one the most "expensive" ones, being 5th in that ranking.

So, from these observations, we can deduce that the main categories information is going to be a good candidate to be a predictor/feature in our model.

On the other hand, is also possible that goal for each project could be also a good feature to be included in our model. However, we need a further analysis in order to be able to confirm that assumption (will do it in next sections).

Additionally, we are going to analyze if there are big differences for the categories inside each main category. The reason is that if there are important differences, instead of just taking into account for our model the main category, we should include the category breakdown as well.

We are going to select 2 examples of main categories for that analysis, the one with worst success rate (Technology) and one with success rate very close to the average (Film & Video). We are going to summarize success rate per category and also the goal average as well, in order to discard that differences are due to goal objective instead of category itself (as we did with main categories).



In the graphs we can observe that there are big differences between categories inside the same main category and that are not related to the goal objective average for each category. Hence, we can conclude that is worthy to add the categories into our model in conjunction to the main category.

Finally, if we check the number of different categories we will find out that there are 159 different categories but the number raises to 170 if we take into account the main category plus category combination. That means that there are categories which are repeated at least in 2 different main categories.

So, in order to simplify our model and have one only predictor related to the category of the project, we are going to create a new column called *unique_cat*. In that new feature, we are going to join the main category and category for a campaign.

Table 2: Examples of main cat and category features joined into unique cat feature

ID	category	main_category	unique_cat
1000002330	Poetry	Publishing	Publishing|Poetry
1000003930	Narrative Film	Film & Video	Film & Video|Narrative Film
1000004038	Narrative Film	Film & Video	Film & Video|Narrative Film
1000007540	Music	Music	Music|Music
1000014025	Restaurants	Food	Food|Restaurants
1000030581	Drinks	Food	Food|Drinks

Funding goal and campaign duration analysis

We are now going to focus on studying if there is some kind of correlation between the funding goal that a campaign need to raise and the probability of success for that campaign.

In order to study that funding goal effect, we are going to plot the goal vs the succes rate through an histogram. However, we need first to do a preliminar analysis in order to find the best way to visualize this data.

So, as a first step, we need to find the median, maximum and minimum funding goals in our training dataset:

- median goal is $5000
- maximum goal is $151395870
- minimum goal is $0.49

We can observe that there is a huge difference between the maximum and the minimum, while the mean was around $40000 and the median is $5000. This is going to represent a problem when creating a proper histogram that can be easily interpreted.

In fact, 97% of our training samples have a goal between 0 and $100000, so we will use that limit in order to plot our histogram, and will be a valid approximation of the whole population for our purpose.

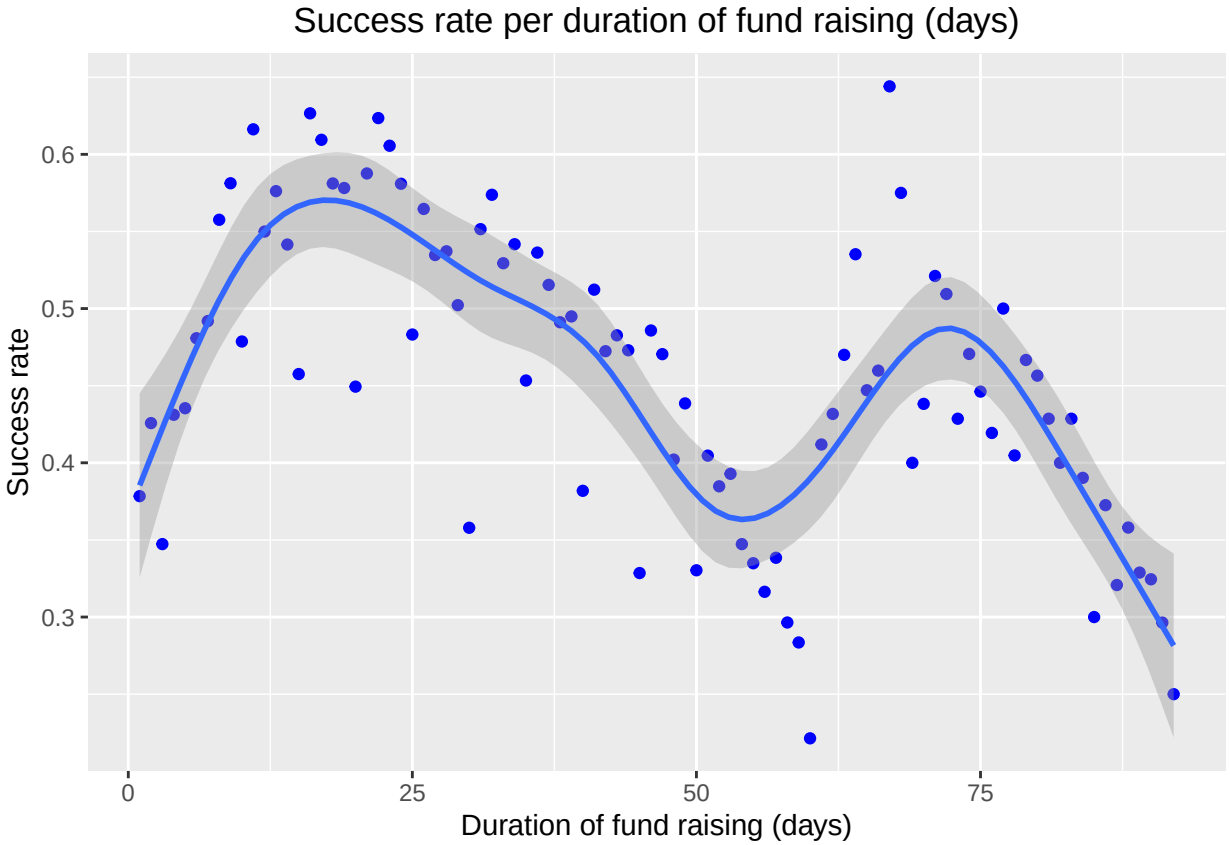
For lower goals, up to $4000, the success rate is quite high compared to the average and even the number of succesful campaigns is higher than the number of failed projects for goals up to $2000. Additionally, we can also observe that there is an important percentage of the total number of projects with a funding goal under $6000.

Now we are going to calculate the number of days that the campaign was open for donations and check if this feature has some effect in the success rate.

As a first step for this analysis, we are going to create a new "days period" column in our dataset. That new feature will represent the days that the campaign was open. We calculate that new feature substracting the deadline from the launched date.

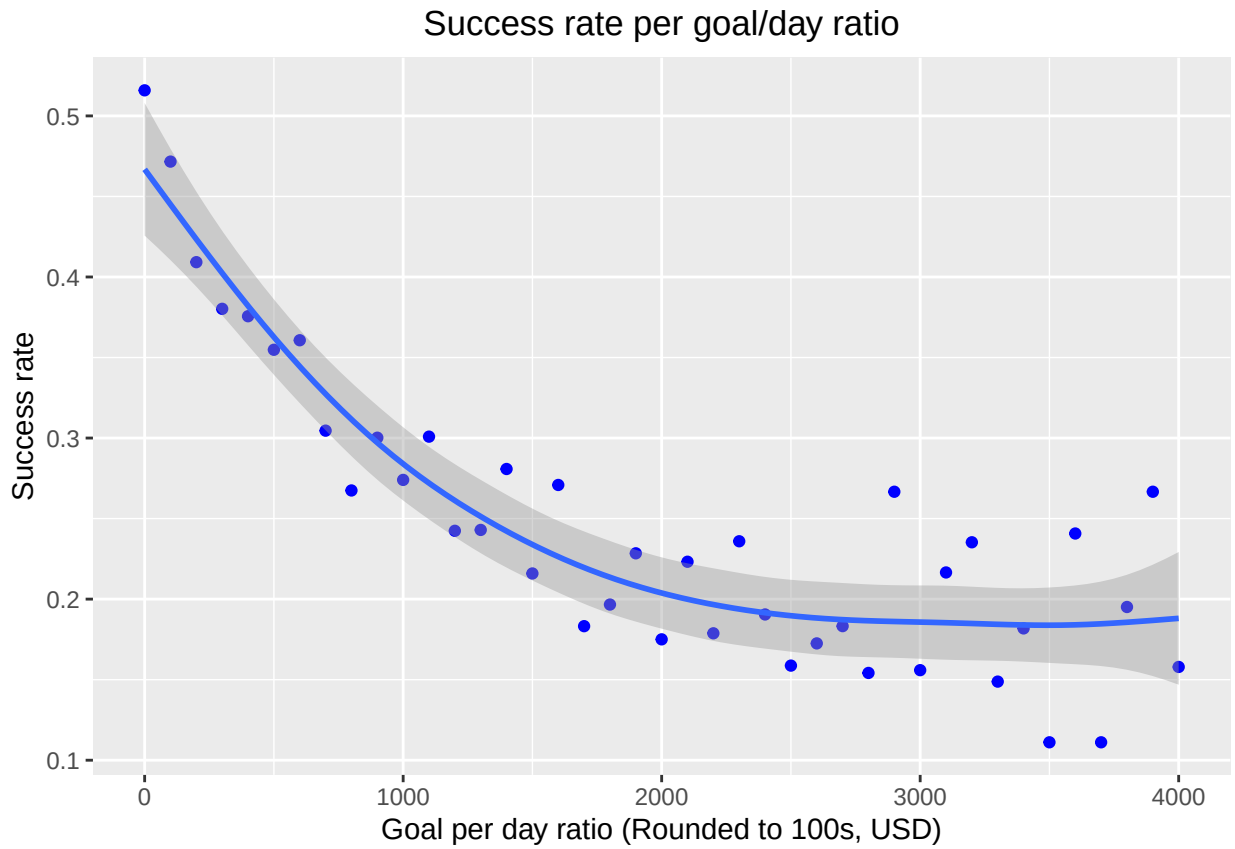
ID	launched	deadline	days_period
1000002330	2015-08-11	2015-10-09	59 days
1000003930	2017-09-02	2017-11-01	60 days
1000004038	2013-01-12	2013-02-26	45 days
1000007540	2012-03-17	2012-04-16	30 days
1000014025	2016-02-26	2016-04-01	35 days
1000030581	2016-02-01	2016-03-17	45 days

Now that we have a feature with the duration of the fundraising campaign, expressed in number of days, we are going to plot the success rate vs duration of the campaign and analyze if there is some kind of correlation between both of them.



As we can see in the plot, the slope for the success rate is raising from 0 to 15 days, then decreasing from 15 to 55 days, increasing again from 55 to 72 days and decreasing from then again. So, there is some kind of correlation, but is clear that is not linear and not easy to quantify.

On the other hand, it could be interesting to study as well the relationship between duration, funding goal and success probabilities for a campaign. The reasonsing behind the decision to study that relationship is that, for example, should be easier to raise $5000 in 10 days, than $3000 in just 2 days, even is a bigger amount of money the ratio usd per day is lower ($500 per day vs $1500 per day in the second case).



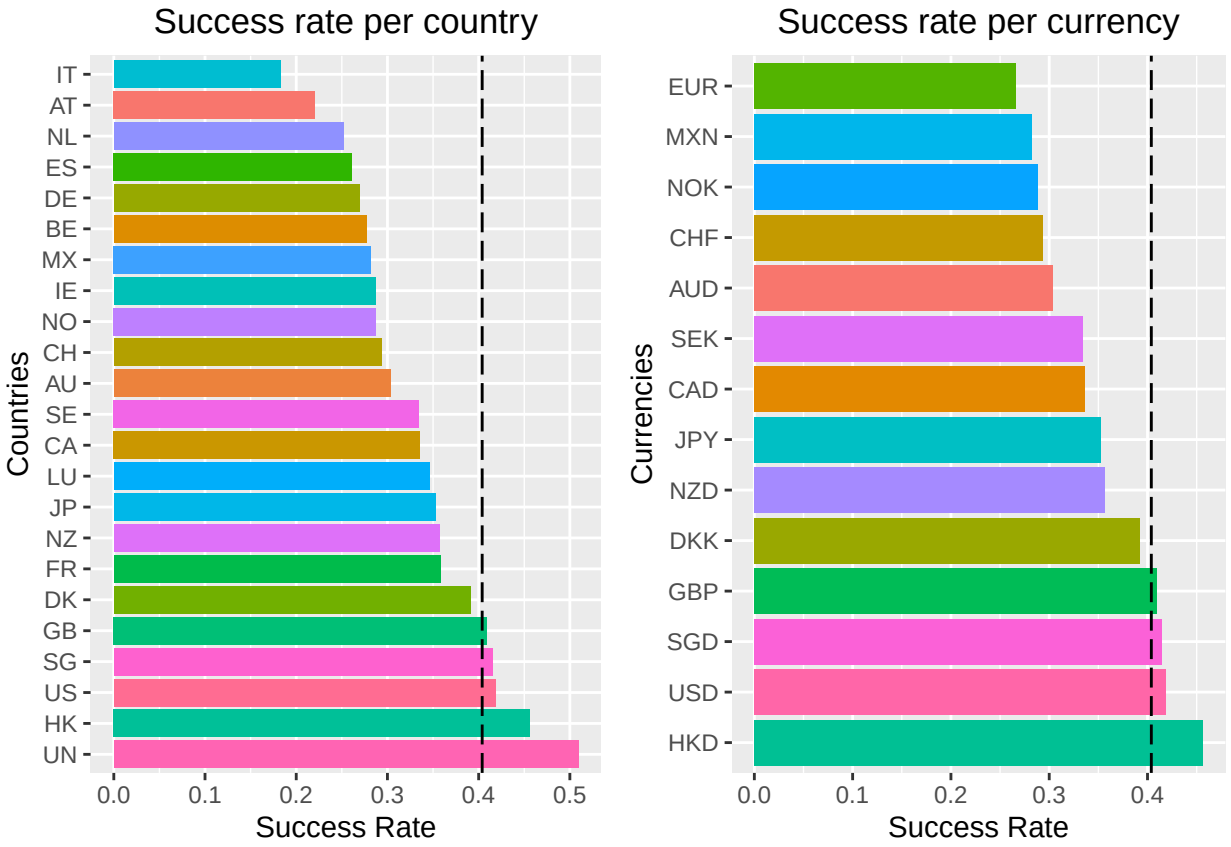
Note: In order to improve visualization, we have filtered those campaigns with a fundraising goal bigger than $100000 and those with a goal per day higher than $4000. In any case, we are still covering 97% of our training dataset population in our graph.

After analyzing this goal per day ratio, we can observe a clear tendency in the graph: the probability of success of our campaign will descend logarithmically a long as our goal per day ratio increases. As it seems it could be an importante feature for our prediction model, we are going to include it as a new column in our dataset.

Currency and country analysis

In Kickstarter, project creation is available to individuals in the US, UK, Canada, Australia, New Zealand, the Netherlands, Denmark, Ireland, Norway, Sweden, Germany, France, Spain, Italy, Austria, Belgium, Switzerland, Luxembourg, Hong Kong, Singapore, Mexico, and Japan.

So, we are going to study if the project location could have some relationship with the probabilites for the campaign to be succesful. In order to do that, we need to summarize the success rate per country and currency and check if there are differences between countries and/or different currencies.



Note: There is a country code (N,0) which seems to be some kind of error in the original data. There are 165 entries in our training dataset for that one, that we are going to change to Unknown (UN).

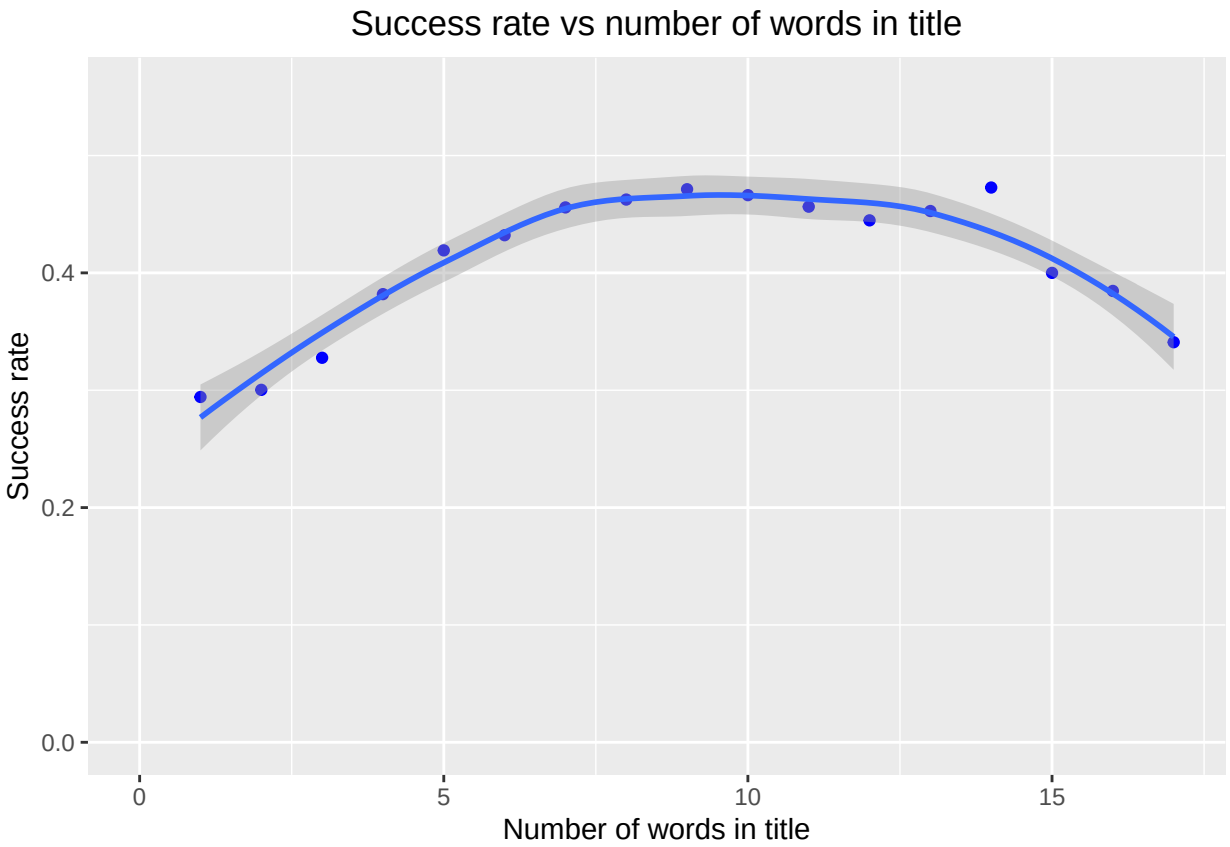
As we can observe, success rate is almost the same for a country and its currency. The only remarkable differences are for countries in the EURO area, which share a common currency. So we can discard currency as a feature if we have selected the country as well.

Regarding the country effect, is clear that there are big differences in the project success rate between different countries, so it's a good candidate to be included as a feature in our prediction model.

Title length analysis

As a final feature to be analyzed, we are going to study if the lenght of the title for a project has some kind of relationship with the probability of success for that project.

Our hypothesis is that if the title of the project has some kind of very brief explanation related to the purpose of that project, backers are going to be more interested on it and there is a bigger probability that they will fund it.



Note: For better visualization purposes, only shown until 17 words. That corresponds to 99.9% of population in the training dataset.

As we can observe in our graph, when the title length is between 5 and 15 words, the success rate is higher than the average, with a maximum when title length is 9 words. However, when the length is lower than 5 or higher than 15, the probability of the project being succesful decreases quite quickly.

Hence we could think in including the title length as well as a feature into our model. We are going to include a new column into our dataset with the number of words that are in the title length.

Model selection

Logistic regression

One of the most simple, easy to understand and probably most used supervised machine learning algorithms is the linear regression algorithm. In fact, in any introduction course to machine learning algorithms is the first one to be taught.

Linear regression is a suitable algorithm when the expecte output is a continuos variable. However, our problem is a classification one, where there are 2 possible output classes: 1 (succesful campaign) or 0 (failed campaign).

For those cases, we should use logistic regression instead of linear regression. Logistic regression is an extension of the linear regression algorithm using logistic transformation:

$$g(p) = log\frac{p}{1-p}$$

where conditional probability has been modeled as:

$$g\{Pr(Y = 1|X = x)\} = \beta_0 + \beta_1 x_1 + ... + \beta n x_n$$

Logistic regression algorithm will always return a value between 0 and 1. That will be the probability for a project for being succesful, and we will transform it in the following way:

- if probability p $>$ 0.5, then we will predict the project as succesful (1)
- if probability p $\leq$ 0.5, then we will predict the project as failed (0)

Before start training our logistic regression model, we need to do some additional transformations in our dataset:

- transform the *success* column into a numeric, 1 for successful project and 0 for failed
- transform the days column into a numeric
- do the same transformations that we did in previos sections to the cross validation dataset as well

We are going to execute different iterations of the logistic regression algorithm with different features in order to find out which one fits best.

In our first iteration, the most simple one, we are going to use as features only the ones that are not factors: goal per day ratio, days of duration of the campaign and words in the title. After using those features, we get an accuracy of 61.64% in our predictions.

For the second iteration, we are going to include the 2 features expressed as factors as well, those are the unique category key that we created and the country. The accuracy for our predictions has improved to 67.39%.

However, the execution time has increased exponentially as the R implementation of the algorithm is transforming each different factor in a feature into a new feature with possible value 1 or 0:

- unique category feature is transformed into 170 new features (1 for each different possible unique category)
- country feature is transformed into 23 new features (1 for each different country)

Table 4: Six first coefficients for the logistic regression model. Note that is creating a different one for each category

	x
(Intercept)	-0.3562861
days_period	-0.0221279
goal_day_ratio	-0.0004758
title_words	0.0775112
unique_catArt|Ceramics	0.1247731
unique_catArt|Conceptual Art	-0.3292919

So, as a third iteration, in order to improve the performance of the algorithm and be able to interpret the coefficients easily, we are going to try to represent the category and country effect in a different way. We will try to represent them through their average success rate, so instead of factors they become a numeric feature.

Table 5: Examples of our new features: success rate per unique category and country

ID	unique_cat	ucat_success_rate	country	cty_success_rate
1000002330	Publishing|Poetry	0.3902681	GB	0.4091879
1000003930	Film & Video|Narrative Film	0.4286098	US	0.4182172
1000004038	Film & Video|Narrative Film	0.4286098	US	0.4182172
1000007540	Music|Music	0.5089207	US	0.4182172
1000014025	Food|Restaurants	0.1759259	US	0.4182172
1000030581	Food|Drinks	0.2821553	US	0.4182172

Now the algorithm is quite faster compared to the second iteration of our model (where we were using the factors directly), coefficients easier to interpret and the accuracy is almost the same: 67.01%

As we can see, using linear equations to model the effect of our features has some limitations. If we observe the relationships between features and success rate presented in the previous data analysis, we can see that this relationship is not linear in most of the cases.

As a fourth iteration, we are going to include higher order terms into our model trying to achieve a better accuracy, even if that means that more computational time is required. We are going to introduce second order terms into our model.

However, as we have examples where features present huge values (e.g. $11.6 million for the goal per day ratio) and same or other features with tiny values (e.g. $0.012 for same goal per day ratio feature), the algorithm is failing to converge. Hence, it is not returning any valid value, as we can see in the following warning thrown by R studio when executing the model:

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

That is happening even if we increase the number of max iterations for the underlying optimization algorithm using the *maxiter* parameter.

In order to avoid this problem, we are going to scale our features. So, for example, all of them will have a value between 0 and 1:

- unique category success rate and country success rate already between 0 and 1, no resize required
- for duration of the campaign in days, we just need to divide by the maximum number of days, which is 92

- for number of words in the title, divide by the maximum number of words, which is 33
- for the goal per day ratio, we need to complicate it a little bit, as the maximum of $11.6 million is to high to use because most of the population will be 0 after changing the scale. In this case, we are going to:
  - consider $5000 as the maximum possible value (covering 98.3% of population) and assigning that value to higher goal per day ratios (only 1.7%)
  - now divide all the examples by that new goal ratio maximum

Table 6: Examples of scaled features: duration of campaign, words in title and goal per day ratio

ID	days_period	period_scaled	title_words	words_scaled	goal_day_ratio	ratio_scaled
1000002330	59	0.6413043	6	0.1818182	25.99915	0.0051998
1000003930	60	0.6521739	8	0.2424242	500.00000	0.1000000
1000004038	45	0.4891304	3	0.0909091	1000.00000	0.2000000
1000007540	30	0.3260870	7	0.2121212	166.66667	0.0333333
1000014025	35	0.3804348	3	0.0909091	1428.57143	0.2857143
1000030581	45	0.4891304	8	0.2424242	555.55556	0.1111111

Now, if we try to use higher order polynoms, also including inference between different features, the algorithm not only is converging but also doing it quite quickly. The accuracy results are now:

- 0.6748, using 2nd order polynoms
- 0.6757, using 3rd order polynoms

As we can observe, we have reached the accuracy limit that we can achieve using logistic regression and higher order polynoms. We are going to experiment now with other algorithms in order to find one with better accuracy.

Table 7: Accuracy for logistic regression using different features and orders

Features	Accuracy
Only non-factor	0.6164618
Including factors	0.6739138
Factor as numeric	0.6701618
Scaled and 2nd order	0.6748183
Scaled and 3rd order	0.6756558

K-Nearest Neighbours

There are a big number of machine learning algorithms that are based on measuring distances between different observations. If we think in Euclidean distance as the ordinary distance between 2 points, we can describe this distance with the formula:

$$dist(P1, P2) = \sqrt{\sum_{j=1}^{n}(x_{1,j} - x_{2,j})^2}$$

where:

- $P1$ and $P2$ are point/example 1 and point/example 2
- $x_{1,j}$ and $x_{2,j}$ are values of feature j for example 1 and example 2

Those algorithms are based on the principle that as closer are those observations, closer (or very similar) is supposed to be the output value for both of them.

For example, in our case, if we have 2 observations with same country, category, title length and a small difference in the goal per day ratio, the distance will be quite small. Hence, we should expect that both of them share same result (both are succesful or failed).

In particular, we are going to use the Knn algorithm (K nearest neighbours) which makes an estimate/prediction for a given sample looking for the k nearest neighbours and computing the average of the output for those neighbours.

Hence, for example, if we want to predict if a given project is going to be succesful or not, and we have set k equals to 100, the algorithm:

- will look for the 100 nearest examples in our training set
- will calculate the average of the outputs for those 100 examples. I.e.if 80 of them are succesful projects and 20 are failed project, the algorith will predict that our project will be succesful with a probability of the 80%

Before starting to train our algorithm, we should notice that we should use the scaled values for this kind of algorithms. That way we will keep the value of our predictors between 0 and 1.

The main reason is that if we use a feature which an absolute value much higher than others, that feature will become much more important when calculating distances than the others.

As a first iteration, we are going to train our algorithm with a fixed value of k equals to 100 and we will try to use scaled and non-scaled features in order to compare the difference in the accuracy.

So, if we try to train our knn algorithm with value k = 100, we will get an accuracy of:

- 0.6466 (64.66%) using as features duration of the campaign in days, goal per day ratio, words in title, unique category success rate and country rate
- 0.6768 (67.68%) using scaled features for duration, goal and words in title plus unique category and country rate which were already between 0 and 1

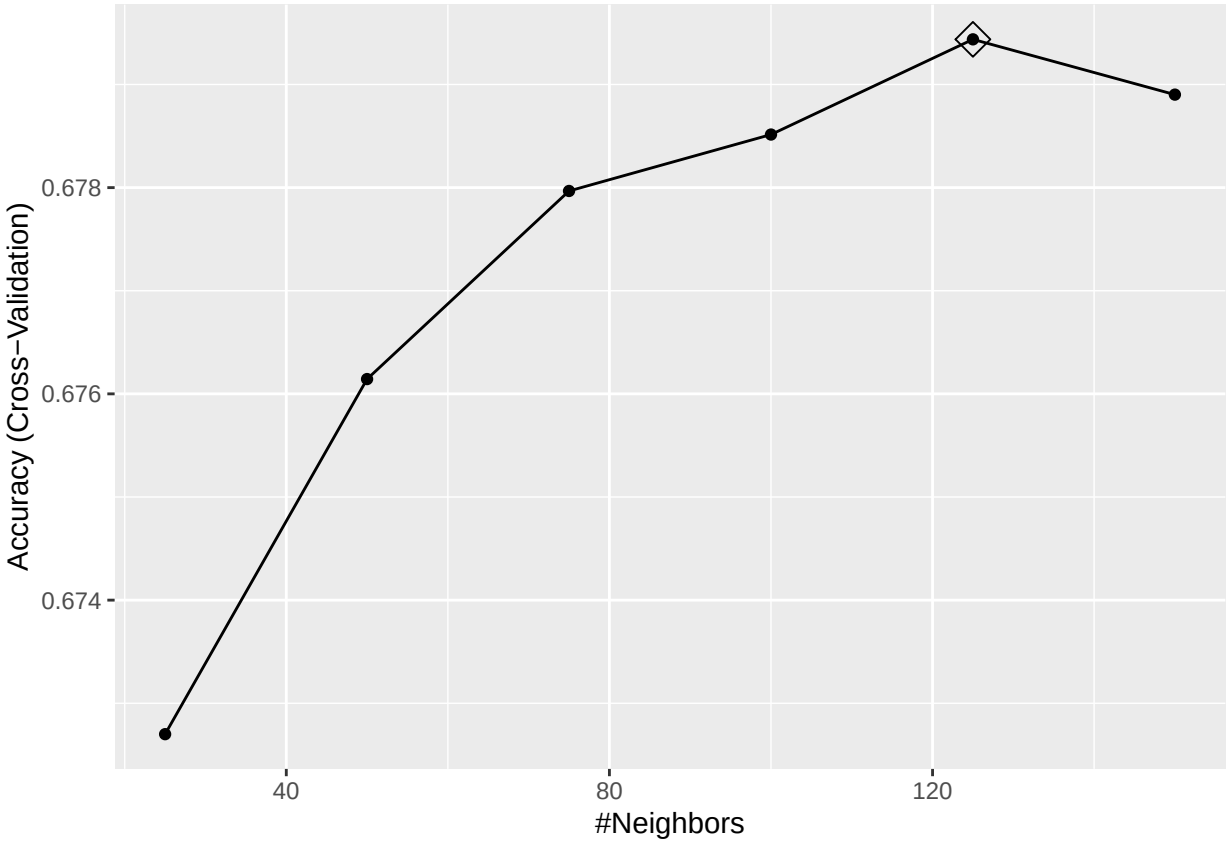
As expected, the accuracy is higher when we use the scaled version of some features.

As a second iteration, we are now going to look for the optimum parameters we can tune our algorithm in order to get the most accurate model for our problem.

If we take a look to the model, we can see that the tunning parameter for our model will be the k one (number of neighbours to take into account). We will make use of the *caret* package in R, that will help us to find this optimum k.

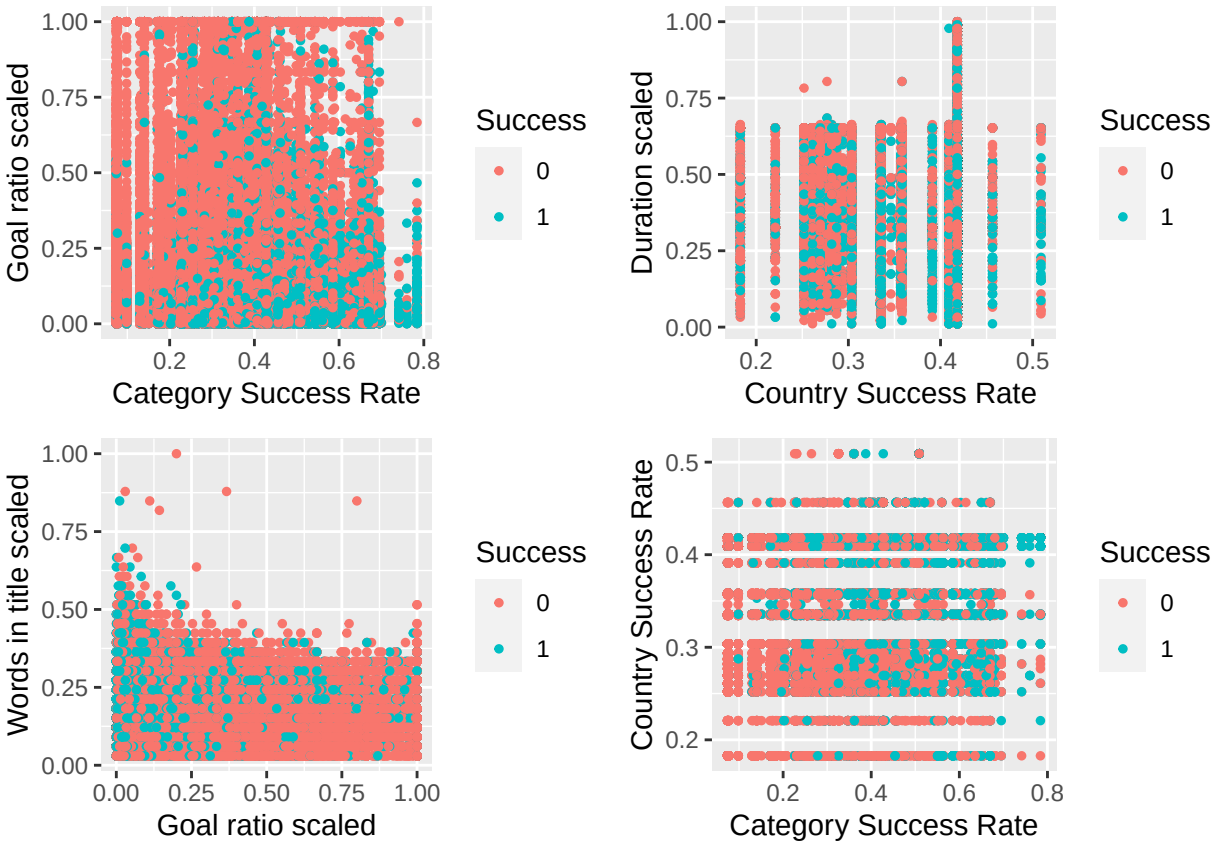
In the *train* function of the *caret* package, we will use k-fold crossvalidation, setting the control parameter to execute 5 iterations for each value of k using a 80% of the dataset as training dataset and 20% as testing one in each of those executions.

Additionally, we will set the *tuneGrid* parameter with a possible value of k between 25 and 150 in steps of 25 (25,50,75,100,125,150).



As we can observe, we get the maximum accuracy with a k value of 125, being that accuracy for predicting our test dataset of 0.6769 (67.69%)

If we compare with the previous algorithm (logistic regression), we can see that there is no so much improvement in the accuracy of our predictions. We are going to try to explain it analyzing some graphs showing success output versus some pairs of features. As we cannot visualize a 5D graph (including all of the features together) we are going to show the success output for some pairs of features and extrapolate our analysis for the 5 of them together.



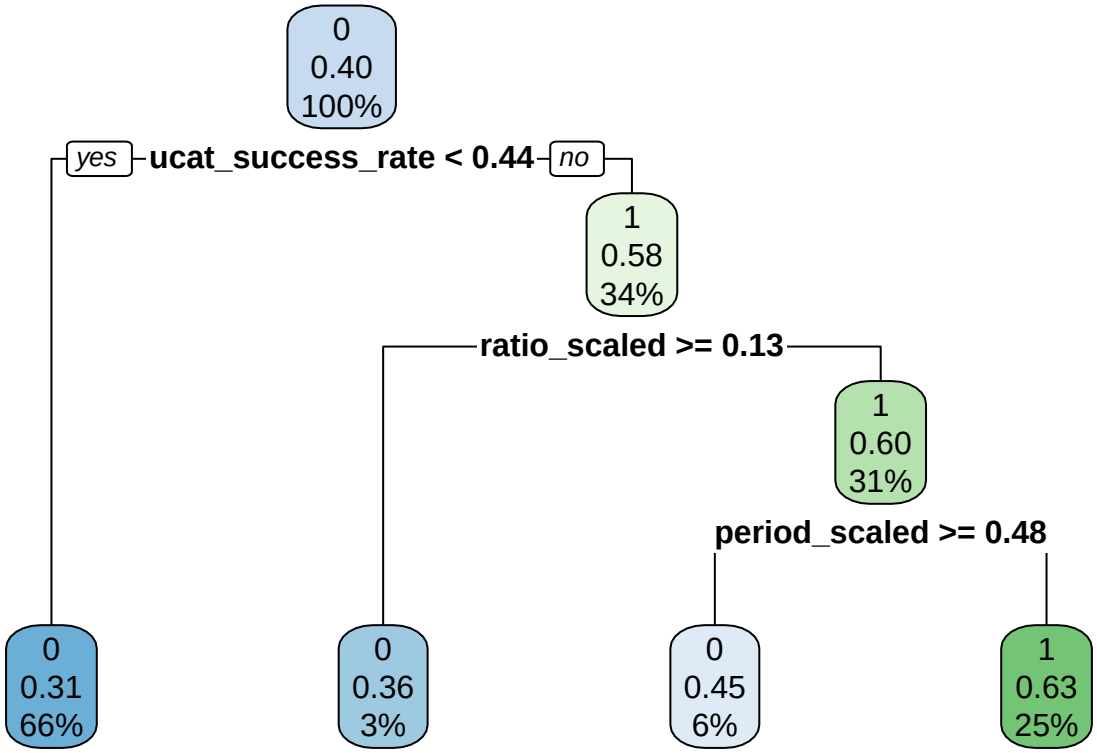
So, as we can see, it's quite difficult to discern clear clusters of successful and non-succesful projects with these predictors. There is a lot of randomness and variability. Hence it's quite difficult to predict if a project is going to be succesful or not just observing the output status of its nearest neighbours.

Classification trees and random forests

Classification and regression trees are algorithms based on creating a flow chart of yes or no questions. When the outcome is continous is called regression tree and, as in our case, when the outcome is categorical is called classification/decision tree.

In classification trees, we predict the output variable by partitioning the predictors. Predictions are done by calculating most common class in the training dataset within the partition.

As a first iteration for this model, we are going to create our classification tree with default parameters. As we can observe in representation of the tree shown below, classification trees are quite easy to visualize and easily interpretable.



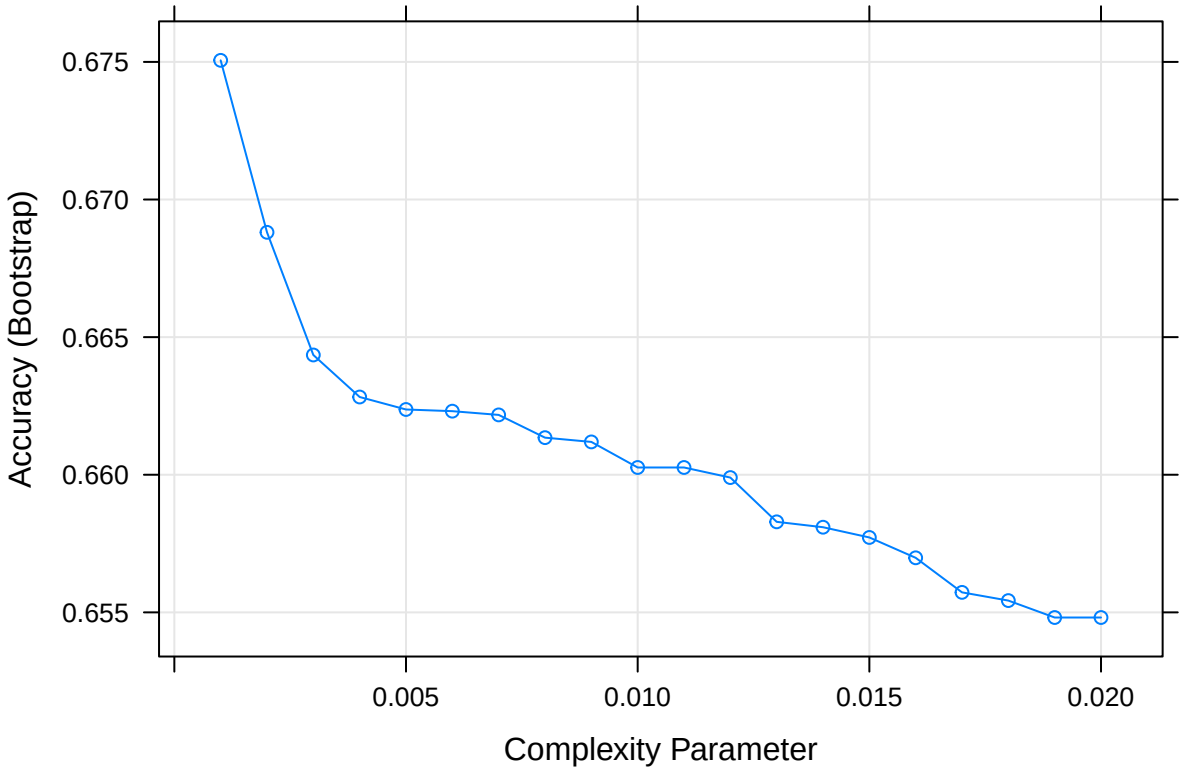
In fact, from that tree we can easily take decissions and predict if our campaign is going to be successful or not:

- if we select a category for a project with success rate less than 0.44, then is not going to be successful
- if our category success rate is higher than 0.44 but goal per day ratio scaled is equal or higher to 0.13, then is going to fail as well
- if category success rate is higher than 0.44, goal per day ratio scaled less than 0.13 but duration of campaign in day scaled is going to be bigger or equal to 0.48, then is going to fail as well
- in any other case, we are going to predict that our project is going to be succesful

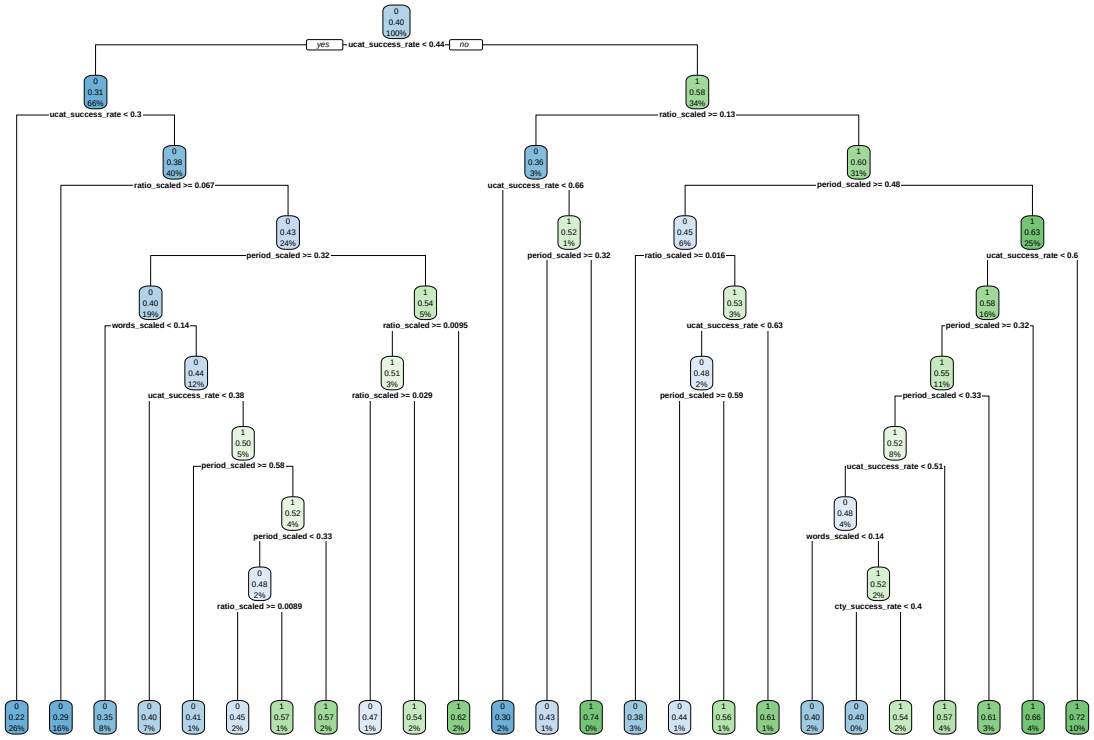
However, although is quite simple and nice to visualize and interprete, the accuracy is quite low compared to other algorithms. For example, if we use our test dataset to calculate the accuracy, we get 0.6595 (around 66%), which is lower than we obtained with logistic regression or k-neareast neighbours model.

Now are going to use the *caret* package again in order to train our classification tree with optimized parameters thus we can achieve a better accuracy.

If we take a look to the model, we can see that the tunning parameter for our model will be the *cp* (complexity parameter). The complexity parameter is the minimum improvement in the model needed at each node. The cp value is a stopping parameter. It helps speed up the search for splits because it can identify splits that don't meet this criteria and prune them before going too far.



In this case, the best result that we can get is when the *cp* value is 0.001, and we are getting an accuracy of 0.6778(67.78%) for our test dataset when applying the optimized model. However, on the other hand, the tree becomes more complex thus is more complicated to interpret. In our case, the number of nodes has increased considerably and is a little bit more complicated to predict if our campaign is going to be succesful or not than it was in the previous tree.



Finally, as a more complex variant of classification trees, we are going to try to train a random forest algorithm. Random forests improves prediction performance by averaging multiple classification trees. In order to ensure the randomness that the algorithm needs, those multiple classification trees are generated using: random samples (bootstrap), a randomly selected set of predictors between all our available predictors and/or different node sizes.

The main disadvantage of these algorithms are: we lose interpretability compared to simple classification tress and that computational requirements to train the algorithm are quite higher compared to other simple methods like classification trees, knn or logistic regression.

We are using *caret* package for parameter optimization combined with one of the particular implementations of the random forest algorithm called *Rborist*. We are setting the tunning parameters *predFixed* (number of predictors to use for each invididual random tree) to 4 or 5 and *minNode* (minimal node size) between 30 and 80.

As a final result, we get an accuracy for our test dataset of 0.6640 (66.4%).

Results

After analyzing the accuracy of the predictions for our test dataset applying different models (logistic regression, k-nearest neighbours, classification trees) we can now compare best accuracy results for predicting if the projects in the test dataset are going to be successful or not.

We are going to represent the best accuracy obtained for each method:

Table 8: Prediction accuracy for different methods and test dataset

Algorithm	Accuracy
Logictic regression	0.6164618
K-Nearest Neighbours	0.6756558
Classification Tree	0.6778667

We have found that the one with better accuracy is the classification tree algorithm with *cp* (complexity parameter) equals to 0.001.

Once we have selected the best algorithm for our problem, we are going to calculate the final performance of the model using our validation dataset. The validation dataset was created at the beginning and has not been included in any part of our analysis or algorithm training, hence is completely independent of our model.

First step will consist on applying on validation dataset the same transformations that we did for training and cross-validation datasets: create unique category and success rate for category, calculate goal per day ratio and scale it, etc...

After doing the corresponding transformations and applying the trained classification tree model to our validation dataset, we get a final accuracy of 0.6740 (67.4%).

As we can observe, the accuracy performance, 67.4%, is quite good (we are going to predict correctly for 2 of each 3 campaigns) but not as high as the performance we can get for other simplier problems.

This is due to the nature of our problem, where:

- there is a high dependency of human preferences in liking or disliking a project (and the human being is quite unpredictable)
- there is a lack of additional features that could be important for predicting the future success of a project like author (previously succesful authors have more probabilities of success for new projects), rewards (authors give rewards to sponsors if they commit to donate certain amounts of money), etc...

In fact, as we calculated at the beginning the global average of succesful projects (40.4%), we may be tempted to predict the success of a campaing just randomly guessing (with probabilites of 40.4-59.6) and think that the performance is going to be quite similar to our trained model.

We are going to compare the accuracy for both of them and additionally, as we know, overall accuracy is not the only way of measuring different algorithms. So We will take into account as well:

- sensitivity/recall → proportion of actual positives (successful campaigns) correctly identified (true positive / true positive + false negative)
- specifity → proportion of actual negatives (failed campaigns) correctly identified (true negative / true negative + false positive)
- precision → proportion of positives (succesful campaigns) that were correctly identified as succesful (true positive / true positive + false positive)

- F1-score → harmonic average of precission and recall. It is used as a simple number summary to evaluate the algorithm. The formula to calculate it is quite simple:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Table 9: Comparative metrics between just guessing and our trained algorithm

Metric	Guessing with 40-60 prob	Trained Classification Tree
Accuracy	0.5220996	0.6740835
Sensitivity	0.6013049	0.8078090
Specificity	0.4051956	0.4767095
F1 Score	0.6000151	0.7471582

NOTE: For calculating sensitivity, specificity and f1-score, the failed campaign factor is used as positive value. Caret package automatically uses it as positive value because is the most numerous. In any case, it really doesn't matter as it doesn't change the validity of our metrics.

As we can observe in the table, is not only the accuracy which is much better in our predictions done with the trained algorithm, compared to guessing with a 60-40 probability, but also sensitivity, specificity and f1-score.

Conclusion

In this report, we have studied the probability of success for any Kickstarter campaign based on historical data provided since its inception in 2008 until January 2018.

We have demonstrated how different features like the money/goal we want to raise or the duration of the crowdfunding campaign can increase or decrease the probabilities of success for our project.

We have been able to provide a model that may help us to predict if our Kickstarter campaign will be successful or not with an accuracy of 67.4% (2 of each 3 will be predicted correctly).

It's important to remark that in this analysis, our purpose was always to predict the probability of success **before** launching our campaing. That way, only features that we can choose as authors and we have historical success rate information availabe for them have been included in our model: goal to raise, title and duration of the campaign and country of origin.

We could probably achieve a higher accuracy including other features that are available (or can be deduced) from the dataset, i.e., success rate for projects on same month/year.

However, we are not considering them for our model because that information wouldn't be available when we launch our project. I.e. if we launch our project on September 2020, we are not going to have that information available until all the projects launched at same time have finished.

Additionally, as part of a possible future work, we can study if this accuracy performance would be improved collecting and using additional data that is not represented in the original Kaggle dataset and we think could have some impact:

- author and previous campaigns success rate → it is probable that authors that had previously promoted succesful campaigns have more chances of being succesful again
- rewards for the campaign and minimum contribution in order to get it → it is probable that campaigns where rewards require a lower minimum donation have more chances on being succesful
- marketing campaigns (money spent on marketing) → the chances for our campaign to be successful are going to be higher if we have hired some marketing as social media influencers, Facebook & Instagram ads, etc...

References

"A Brief History of Crowdfunding." 2018. https://www.startups.com/library/expert-advice/history-of-crowdfunding.

"Crowdfunding Comte." 2016. http://positivists.org/blog/archives/5959.

"Crowdfunding Statistics." 2020. https://blog.fundly.com/crowdfunding-statistics/.

"Progressive-Rock Band Marillion Pioneered Crowdfunding." 2016. https://www.westword.com/music/progressive-rock-band-marillion-pioneered-crowdfunding-8425775.

"The Statue of Liberty and America's Crowdfunding Pioneer." 2013. https://www.bbc.com/news/magazine-21932675.