University of Cologne

Business Analytics and Econometrics

Machine Learning in Applied Settings

Team Report

Predicting Success of Kickstarter Projects using Facial Detection

Group Members:

Carolin Freude (7406279)

Angelina Skryabina (7412542)

Vu Duc Anh Nguyen (7354039)

Rolf Schnaufer (7409069)

Submission Date: May 29, 2023

Instructor: Christopher Coors

Table of Content

1 Introduction

Financing new entrepreneurial projects has changed significantly in the last few years. Previously, bank loans or venture capital were conventional methods of raising funds that often limited the implementation of projects. With the appearance of innovative crowdfunding platforms like Kickstarter, that has changed. Kickstarter gained popularity quickly for its usability and community-building features. However, thousands of projects are competing for funding nowadays, with most failing to reach their financial goal.

Previous research has already investigated different variables, models, and features that influence the success of Kickstarter projects. In this context, the purpose of funding and duration are identified as important factors. Other studies related to this have also already looked at the characteristics of the creator and the effects of social networks. Nevertheless, there are research gaps in understanding the impact of face detection in predicting the success of a project (Canhoto et al., 2020; Leggett et al., 2013; Luo et al., 2020).

Human faces elicit emotional responses and can create personal bonds between creators and sponsors. Advances in face detection technology in recent years facilitate face detection in this regard. Given these factors, this research project focuses on analyzing the impact of face detection on the success of Kickstarter projects to fill the research gap. This leads to the research question: How does face detection in a funding campaign improve success in Kickstarter and which prediction model performs the best?

Binary prediction models with and without face detection are examined and different machine learning models are compared to identify the best-performing prediction model.

Studying the impact of face detection and comparing different models is of great interest to project initiators. By understanding the impact of face detection, projects could potentially improve their chances of getting funded. But also for sponsors, predictive models can be indicators for funding decisions. Ultimately, it will become easier for Kickstarter platform operators to identify areas of improvement and increase the visibility of promising, but also less successful projects.

In the first step this collected data set is described in more detail. Afterward, the data wrangling and preprocessing steps including the addition of new variables for the analysis are explained. Based on these steps, exploratory data analysis is performed so that an initial understanding of data is developed, and patterns are identified. By combining data wrangling and pre-processing with exploratory data analysis, a solid foundation is laid to train and

evaluate the machine learning models using forward feature selection, nested cross-validation, and hyperparameter tuning.

2 Dataset

2.1 Data Collection

The study is based on data from the Kickstarter platform, acquired from the Web Robots website (Nicerobot, 2023). The data is available in CSV format and was obtained using a scraper robot that crawls all Kickstarter projects monthly starting in March 2016. To ensure data collection for a wide variety of projects, the scraper robot systematically crawls all subcategories on the Kickstarter platform. The current study was implemented on the dataset of April 18, 2023. This dataset is the newest of all available and contains recent snapshots of Kickstarter projects.

Additionally, it is important to note that starting in April 2015, Kickstarter introduced restrictions on the number of projects that can be viewed in one category, which affected the number of historical projects included by Web Robots in the dataset. Despite this limitation, the dataset covers the majority of the campaigns at the time of data collection.

2.2 Data Description

The data collected for this study is presented in a tabular format, where each row represents a Kickstarter project, and each column reflects specific attributes of the projects. The original dataset consists of 35 columns and 17,670 observations. Out of these 35 columns, we have already left out and removed the following 18 columns from the dataset during the description of the columns, as they will not contribute meaningfully to the goal of this analysis project: backers_count, converted_pledged_amount, country_displayable_name, currency_symbol, currency_trailing_code, current_currency, disable_communication, fx_rate, is_starrable, pledged, slug, source_url, spotlight, staff_pick, static_usd_rate, usd_exchange_rate, usd_pledged, usd_type.

It is important to note that in this original dataset, some columns contain JSON objects in string format and therefore must be transformed. A brief description of each of the 17 remaining columns is provided in Table 1. Column names marked with asterisks refer to JSON objects.

2.3 Data Wrangling and Pre-Processing

Based on the dataset described above, the missing values were examined in more detail. Ultimately, only 14 missing values (N/A) were found in the "Location" column and along duplicate rows removed from the dataset by deleting the rows, resulting in 15,523 unique projects.

Subsequently, features were extracted from the JSON objects stored as strings in the dataset. Specifically, the "parent_category" and "sub_category" features were extracted from the "category" column, the "country" and "city" features were extracted from the "location" column. Additionally, the "project_url" and "reward_url" features were extracted from the "urls" column, while the "photo_url" feature was extracted from the "photo" column.

The time information, stored in Unix format in milliseconds since 1st January 1970 00:00:00 UTC, was processed. The date of creation of each project was calculated, as well as the durations of different campaign phases, namely, from creation to the funding deadline and from launch to funding deadline.

After preprocessing the dataset, variables with low information value were eliminated. Some of these variables served only to gather supplementary features for our classification models, such as the project-, reward-, and photo-URLs for Kickstarter projects. The project ID and name were also deemed irrelevant due to their uniqueness to each project, rendering them unsuitable for classification purposes. The blurb variable was also removed, as it was mainly intended for sentiment analysis. The creation date was also removed, as its primary purpose was to calculate campaign phase duration. However, we retained the month variable to test our assumption that it would not significantly impact project success.

Thus, the dataset was left with important columns that potentially contribute to predicting success, namely: 'state', 'goal', 'parent_category', 'sub_category', 'country', 'city', 'currency', month, 'create_to_launch', 'create_to_deadline', 'launch_to_deadline'.

2.4 Face Detection Variable

In this project face detection was performed using two libraries: OpenCV and RetinaFace. OpenCV is an open-source computer vision library designed for image analysis, classification, and processing.

OpenCV includes many different functions and efficient algorithms, resulting in its versatility. Tradeoff for its efficiency is that it lacks accuracy in face detection under demanding conditions like rear face angles.

Nevertheless, compared to RetinaFace, the time required for face detection is much shorter (OpenCV, 2023a; OpenCV, 2023b).

RetinaFace is a powerful facial recognition algorithm that uses deep learning techniques to achieve highly accurate and reliable facial recognition. It is designed to recognize faces in images even in complex scenarios with variations in pose, scale, occlusion, and lighting conditions. However, the algorithm is very sensitive to unusual facial features and takes a very long time to evaluate the images. Nonetheless, RetinaFace is currently considered state-of-the-art due to this precision and the ability to reliably localize faces under various conditions. Deng's article highlights that RetinaFace has set a new standard for face recognition by combining deep learning techniques (Deng et al., 2019).

Comparing the two facial recognition tools will help us test the assumption that enabling face detection as a feature improves the accuracy of our predictions. Since RetinaFace theoretically recognizes faces better than OpenCV, an increase in accuracy is expected when using RetinaFace. By comparing the results of both tools, we can determine which one is more effective at face detection and how much it affects the accuracy of our predictions.

To simplify the face detection process, the DeepFace library was used, which contains OpenCV and RetinaFace as internal options for face detection in images. The URL of the project photo was passed to the library along with the selected backend, and the library returned an array containing the coordinates of any detected faces along with the confidence value. When the confidence value is 0, no faces are detected in the image. This made it possible to identify project images with and without visible faces.

It is important to note that the execution time of the face detection process varied depending on the selected backend. When using OpenCV, the face detection process took approximately 1 hour and 15 minutes. However, due to RetinaFace's more sophisticated deep learning methods, it took much longer to analyze a single image, with the average processing time of each image being about 10 seconds. Such a sharp contrast in execution time led to the total time of the face detection process taking up approximately 49 hours.

Given the long processing time of RetinaFace, the face detection process was carried out sequentially, while progress was maintained after analyzing each image. This made it possible to ensure the stable execution of the analysis, as well as to export the current results at regular intervals.

2.5 Exploratory Data Analysis

This subcategory provides initial findings from exploratory data analysis, looking more closely at patterns, distributions, and correlations.

Based on the initial analysis of the data, 56% of the projects were successfully funded, while 44% of the projects were unsuccessful. This indicates that just a slight majority of projects are successful (Figure 1)

When analyzing the percentage distribution within the individual subcategories of the projects, it is noticeable that "Classical Music" and "Web" are most strongly represented with a share of around 16% (Figure 2). The distribution of the subcategories also indicates the parent categories. Slightly less than half of the projects can be assigned to the "Technology" category with44%, reflecting progress in technological innovations. The other categories "Theater" with 31% and "Music" with 25% follow. This detailed breakdown highlights the fact that Kickstarter offers a platform for visibility to both technology projects and art projects. The focus is thus on these two subject areas (Figure 3). Looking more closely at the funding success in the various project categories, it is evident that the distribution of success rates differs significantly from the distribution of project frequencies. In particular, the category "Music" shows a probability of success of 86% and is consequently the most successful in terms of funding on average. In contrast, the success rate in the "Technology" category is only 34%.

When looking at the distribution of projects in the various countries, the USA records the highest share with around two-thirds of all projects. The UK is in second place with about 17%. Canada, Germany, Australia, Italy, France, and Mexico each have a share of only less than 5%. Remarkably, Africa and Asia have only a few projects. This suggests that Kickstarter has a distinctly Western bias in project proposals. (Figure 4)

By analyzing project goals, the average value is around $70,000, with the largest part of the distribution in the $1 to $15,000 range. The average time span from launch to deadline of a project is 35 days, with most projects aiming to reach their project goal within 25 to 35 days. At the same time, the majority of projects first require between 0 and 40 days to create the project. (Figure 5 and Figure 6)

Regarding the objective of this paper, the success rates of RetinaFace and OpenCV for successful projects where faces were recognized as well as where no faces were recognized respectively were analyzed in more detail. Retinaface's final success rate for projects with a recognized face is around 68%. Projects without a recognized face only show a value of 48%. The situation is similar for OpenCV, where the success rates of both variants are 2-4% higher. This suggests that the success rates for projects with a recognized face are higher than for projects without a recognized face. This observation is confirmed by a comparison of the success rates in the individual parent categories (Table 2).

When examining correlations between selected variables and the success variable "state", the variables "is_face_retinaface" (0.19) and "is_face_opencv" (0.14) are particularly noticeable. There is a slightly positive correlation between each of the variables and the success variable, with RetinaFace registering a slightly higher correlation. This suggests that face detection tends to be associated with a higher probability of project success. Additionally, there is a slightly negative correlation (-0.16) between the duration of the project launch to deadline and the success variable. Thus, a longer duration of this time period tends to be associated with lower probabilities of success. (Figure 7)

3 Methods

3.1 Feature Selection and Engineering

To address the right-skewness of values in the "goal" variable, a logarithmic transformation was applied to achieve a normal distribution (Figure 8). This transformation helps mitigate the influence of outliers and equalizes the impact of extremely large values, making the variable more manageable for analysis. Standardization of other numeric variables was not performed due to the absence of distance-based algorithms such as SVM or k-NN, which are known to be sensitive to scale. For the Neural Network the numeric variables were standardized to help the model converge faster. Based on the available data, the values that are identified as outliers appear to be reasonable and not the result of errors or anomalies. Therefore, any modification of these values may distort the relationships or patterns within the data.

To enable the use of categorical variables in machine learning algorithms, a label encoding technique was employed. Label encoding converts categorical variables into numeric labels, where each unique category is assigned a unique integer value. In the code, the categorical variables 'parent_category', 'sub_category', 'country', 'city', and 'currency' are being label encoded using the *LabelEncoder* from scikit-learn. This encoding allows the classification models to effectively learn and capture the relationships between the categorical variables and the target variable, enhancing the accuracy of predictions. The target variable, indicating project success, was encoded in binary form. The value 0 was assigned to unsuccessful projects, whereas successful projects were represented by 1.

3.2 Machine Learning Models

Several classification models were used to predict the success of Kickstarter projects. The models were selected based on their suitability for handling both numerical and categorical features commonly found in Kickstarter datasets. The following models were evaluated:

- Logistic Regression: Logistic Regression is a widely used model of linear classification that provides a basic framework for comparison with more complex models.
- XGBoost: As an ensemble learning technique, XGBoost is known for its robustness and performance. The model was implemented to leverage the advantages of gradient boosting and improve classification accuracy.
- Random Forest: Given the success of ensemble methods, random forest was further explored as a model that combines multiple decision trees. Random forest models have the capability to handle high-dimensional data and provide feature importance rankings.
- Neural Network: Neural Network model was experimented to capture complex patterns and interactions in the data. Neural Networks are well-suited for handling large-scale datasets and nonlinear relationships.

Using this diverse set of models, the research was aimed at determining the most effective approach for predicting the success of Kickstarter projects and investigating whether face detection improved the performance.

3.2.1 Nested Cross Validation with Forward Feature Selection

To accurately assess the performance of the models and mitigate potential bias, nested cross-validation was employed. This technique involves an outer loop for model evaluation and an inner loop for feature selection via forward selection and hyperparameter tuning. By iteratively splitting the data into training and validation sets, the aim was to obtain reliable estimates of the models' generalization performance.

The forward feature selection using cross-validation was performed only on the training set to prevent any information leakage. If feature selection were performed on both training and validation sets, the model would optimize on the selected features using the validation set data, which would lead to an upward bias in the performance evaluation. Moreover, relying only on an approach based on the validation sets is not enough to provide a robust model performance evaluation, since exploratory analysis showed very volatile performance depending on the train test split. Therefore, to obtain reliable estimates, nested cross-

validation was performed with an inner cross-validation consisting of 5 folds to determine the best set of features for each iteration.

In each iteration of the forward feature selection, the feature that resulted in the highest accuracy gain was added to the best feature subset from the previous iteration, until no further accuracy improvement was achieved or a maximum of 5 features, to reduce computational time, was reached. Due to the high collinearity between the variables "create_to_deadline" and "create_to_launch," only one of these variables can be included in the feature subset. Additionally, to facilitate a meaningful comparison between the two face detection libraries, only one of them was included in the feature subset, as including both would be redundant.

Subsequently, outer cross-validation evaluated the performance of the model, and the mean of these evaluations served as an overall model performance estimate. This approach helped to reduce the likelihood that the model would work exceptionally well on a validation set due to chance.

3.2.2 Hyperparameter Tuning

A process of hyperparameter tuning was conducted to optimize the performance of the Random Forest model. Within this process, different combinations of hyperparameters were systematically investigated, such as the number of trees, maximum depth, and minimum samples per leaf. The tuning process involved a randomized cross-validation search approach, evaluating multiple parameter combinations to identify the optimal configuration that maximized a model's accuracy. Additional tuning of hyperparameters for the XGBoost and Neural Network models was also applied.

3.3 Performance Evaluation

To evaluate the performance of each model, evaluation metrics such as accuracy, precision, sensitivity, and specificity were used. However, to ultimately gain a comprehensive understanding of the contribution to the success rate of the OpenCV and RetinaFace face detection variables, an additional systematic approach was adopted. At first, whenever the best feature set of a given iteration contained either the OpenCV or the RetinaFace face detection variable, a separate model was trained without the respective variable. Using this approach, the average increase in prediction accuracy with the inclusion of the specific face detection variable was observed. In addition, further experiments were performed, replacing the RetinaFace variable with the OpenCV variable that was not included in the best feature set. The purpose of this comparative analysis was thus to determine the extent to which the

alternative face detection variable did or did not perform better in terms of predictive accuracy. Finally, the final model's best feature selection is obtained and its stability is evaluated by retraining it on 90% of the data. The remaining 10% validation data is then used to assess the model's ability to predict the success of certain subcategories.

4 Results

Table 3:

Performance Metrics of the Models

Models:	**Log Reg**	**RF**	**XGB**	**NN**
Accuracy	71.1%	68%	71.7%	68.5%
Precision	73.1%	68.2%	71.4%	71.8%
Sensitivity	76.4%	79.7%	82%	71.8%
Specificity	64.5%	53.2%	58.7%	64.4%
Retinaface improvement	0.67%	1.93%	1.68%	1.51%
OpenCV improvement	0%	0%	0%	-0.87%
Retinaface in feature selection	20%	100%	90%	20%
OpenCV in feature selection	0%	0%	0%	20%
Retinaface over OpenCV improvement	0.42%	1.55%	1.35%	1.27%

After analyzing the performance of various machine learning algorithms on the Kickstarter dataset, we found that XGBoost is the best-performing model with an accuracy of 71.7%. Random Forest and XGBoost almost always include ‘RetinaFace’ and 'goal_log' as variables in their best feature subset. While XGBoost also often includes some of 'currency', 'launch_to_deadline', 'create_to_deadline', and 'city', Logistic Regression mostly chooses 'goal_log' and 'sub_category' for their best feature subset. We also observed that the inclusion of ' RetinaFace' in XGBoost provides an improvement of 1.68% in accuracy and performs better than OpenCV, which never appears in the feature selection.

However, training the model and determining hyperparameters and the best feature subset using a validation set approach shows very unstable best feature sets and performances for each of the machine learning algorithms. The accuracy varies strongly, but also RetinaFace is barely included in the best feature subset. Interestingly, for the Neural Network, the best

features do not include RetinaFace when evaluating the general performance of the model, but when training with a different split or more data, RetinaFace is included.

In most of the music sub-categories, the models have a very high accuracy of 80-100% in determining success or failure, while wearables perform worse with around 55-67% accuracy. Software is consistently predicted quite well around 72-78%.

Finally, in terms of processing time, XGBoost finishes the process in 2:10 hours, Random Forest in 52 minutes, Logistic Regression in 12 minutes, and NN in 3:30 hours. Therefore, we conclude that the XGBoost algorithm with RetinaFace and goal_log features is the best-performing model for predicting the success or failure of Kickstarter projects, with an accuracy of 71.7%.

5 Conclusion

The main objective of the study was to compare different machine learning models to best predict the success rate of projects using nested cross-validation, forward feature selection, and hyperparameter tuning. The evaluation of the models involved analyzing various performance indicators and examining the effectiveness of face detection techniques such as OpenCV and RetinaFace in predicting outcomes within the implemented models.

The main findings of this research reveal important information about performance and the impact of various models and variables influencing the prediction of Kickstarter projects' success. Among the evaluated models, XGBoost has become the top-performing model with Logistic Regression following closely behind in terms of accuracy and even outperforming XGBoost in precision and specificity. RetinaFace's face detection variable displays a slightly higher accuracy rate than OpenCV when making predictions. However, it should be evaluated with caution as the best feature subset is not consistently stable. The accuracy of single folds also varies heavily between 49-81%, which is why it is strongly recommended to use cross-validation to mitigate the randomness of a validation set approach when evaluating models. Additionally, nested cross-validation should be used to prevent information leaks when hyperparameter tuning and selecting the best feature subset (Figure 9).

It is important to consider certain limitations. The current research involves a sequential process of selecting the best subset of features using default hyperparameters, followed by hyperparameters tuning. The choice of features and hyperparameters could be further optimized by exploring various combinations. Secondly, the current research used a randomized cross-validation search, which may limit the exploration of hyperparameters.

Further exploration of the exhaustive grid search could provide a more accurate understanding. Moreover, the computational requirements for training and evaluation are significant and the execution time is up to 4 hours. Both highlighted limitations are a tradeoff for better computational efficiency. Additionally, the research mainly relies on the sklearn library, which may have limitations in terms of its complexity and fine-tuning capabilities, especially for Neural Networks. Considering alternative libraries such as torch could offer more advanced tuning and improve model performance.

Furthermore, the Neural Network in the study uses label-encoded features, which can affect the training, as it introduces ordinality between categories. This approach was chosen to avoid an excessive number of features that could arise as a result of a single one-hot encoding. However, it is important to consider the potential impact of ordinality on the training process and the ability of the model to reflect the true relationships between categories. Although dimensionality reduction methods such as principal component analysis (PCA) could be effective for reducing the number of features, their implementation would eliminate the possibility of selecting features.

Limited time prevented the implementation of other methods for advancing the project. These methods are mentioned in order to give an overview of further possibilities. First, additional machine learning models like SVM or k-NN, which are based on the calculation of distances, could be compared to the current models to gain new insights. Another approach would be to use clustering for identifying patterns among the detected faces. Additionally, a multi-class prediction with the third class being projects that are 80% close to achieving its goal could provide value. Such close-to-success projects could then be identified and supported more strongly. To expand the dataset in a further step, performing sentiment analysis could be considered, especially for video transcripts. By implementing these ideas, prediction could possibly gain precision and more comprehensive support for projects could potentially be provided.

6 References

Canhoto, A. I., Clark, M., & Powell, P. (2020). Good Creators, Good Crowdfunding Campaigns? The Role of Creator Characteristics in Online Reward-Based Crowdfunding Campaigns. In Frontiers in Service Conference (Vol. 2020, No. 1, pp. 15).

Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., & Zafeiriou, S. (2019). RetinaFace: Single-stage Dense Face Localisation in the Wild. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1905.00641

Leggett, A., Prescott, J. E., & Ely, K. (2013). Factors influencing the success of Kickstarter campaigns. In Proceedings of the 11th International Symposium on Open Collaboration (pp. 8). ACM. Short form: (Leggett et al., 2013)

Luo, J. H., Wu, D., Li, Y., & Fang, Y. (2020). A Novel Approach for Kickstarter Success Prediction using Machine Learning with Limited Data. arXiv preprint arXiv:2010.04790. Short form: (Luo et al., 2020)

Nicerobot. (2023). Kickstarter Datasets. Web Scraping Service. Retrieved May 23, 2023, from https://webrobots.io/kickstarter-datasets/

OpenCV. (2023a). OpenCV documentation. Retrieved May 23, 2023, from https://docs.opencv.org/

OpenCV. (2023b). OpenCV Website. Retrieved May 23, 2023, from https://opencv.org/

7 Appendix

Table 1: *Description of column names of the dataset*

Column name	Format	Description
blurb	chr	a short description / pitch of the project
category*	chr (JSON objects)	the specific category to which the project belongs (providing details such as the name of the sub-category (e.g., hip hop for music projects), the parent-category (e.g., music), and other related data, including URLs)
country	chr	the country where the project was launched
created_at	num	the date when the project was created on the website
creator*	chr (JSON objects)	the information about the project creator, including their id, name, email, and other relevant details
currency	chr	the currency used for the funding goal
deadline	num	the date representing the deadline for funding the Kickstarter project
goal	num	the funding goal set by the project creator
id	int	a unique identifier for each Kickstarter project
launched_at	num	the date when the project was launched
location*	chr (JSON objects)	geographical location of the Kickstarter project including their id, name, short name and displayable name
name	chr	the name / title of the project
photo*	chr (JSON objects)	the URL of the project's main photo
profile*	chr (JSON objects)	information about the project creator's profile on Kickstarter
state	chr	the current state of the project, indicating submission, success, fail, or is in another state
state_changed_at	num	the date when the project's state was last changed
urls*	chr (JSON objects)	the links to the project and rewards

Table 2: *Table summarizing the success rates of RetinaFace and OpenCV in general and for the different parent categories where faces were detected and also where no face was detected*

	RetinaFace				**OpenCV**			
	Overall	Technology	Music	Theater	Overall	Technology	Music	Theater
successrate for project where **face detected**	68.27%	38.13%	87.57%	65.28%	70.22%	36.95%	89.48%	66.97%
successrate for project where face **not detected**	48.44%	33.20%	83.95%	59.50%	52.32%	33.89%	84.60%	60.38%

Figure 1: *Distribution of successful and failed project proposals*

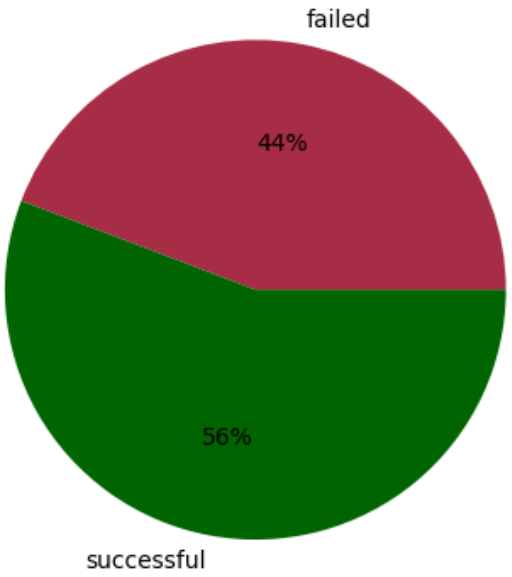

Figure 2: *Distribution of Sub-Categories*

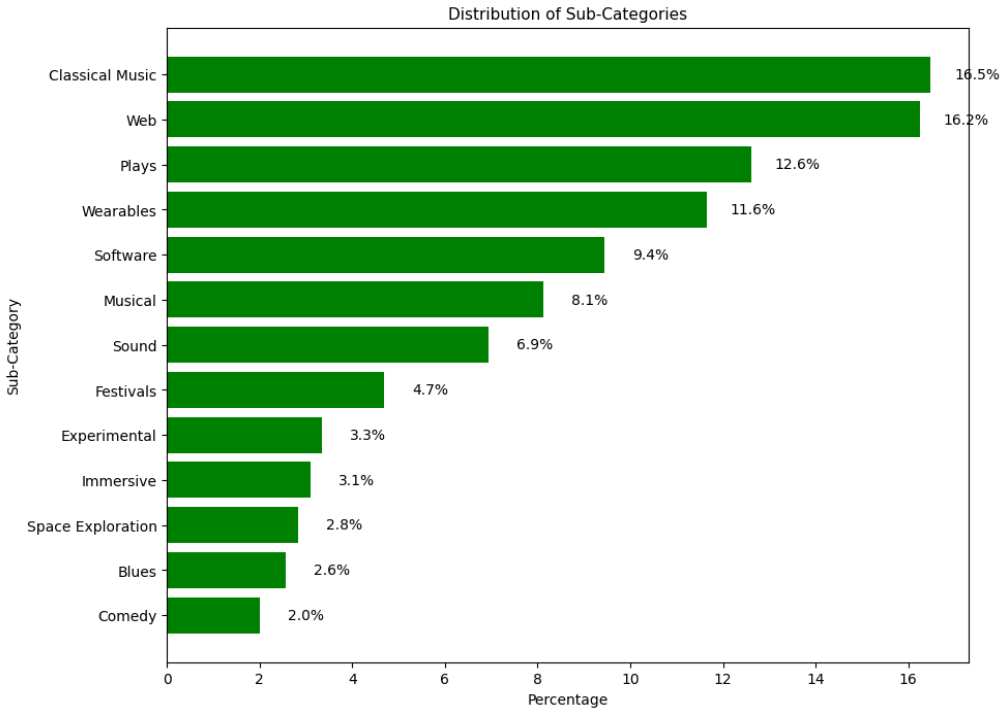

Figure 3: *Distribution of Parent-Categories*

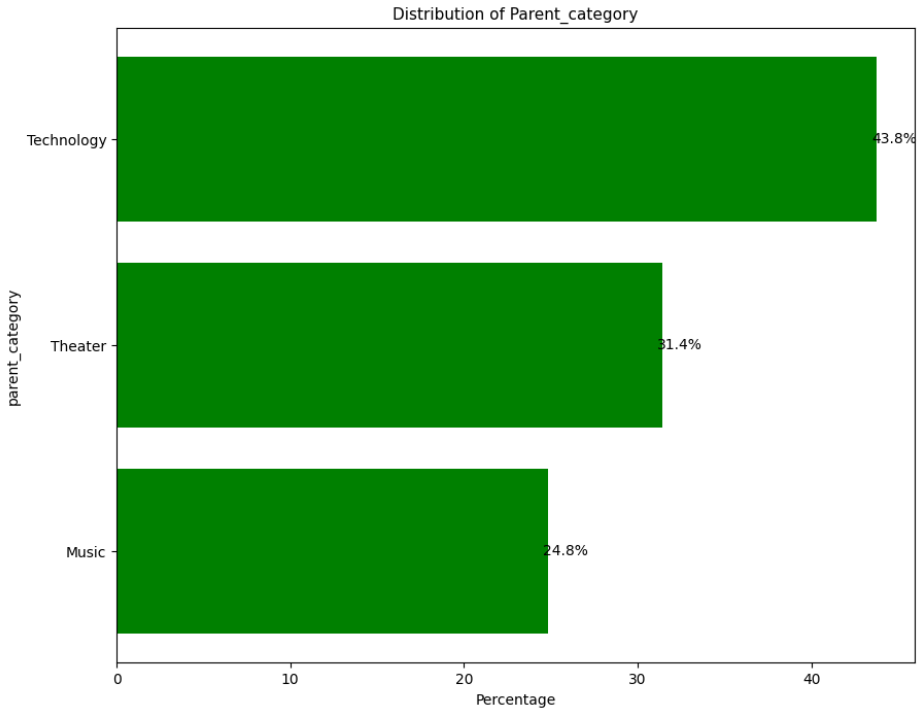

Figure 4: *Overview of the distribution of project plans by country*

Note: The darker a country is colored green, the more projects from this country can be found on Kickstarter. However, if a country is colored blue, this indicates that there are no projects in that country.

Figure 5: *Distribution of project goals, as well as drawing of average goal, minimum goal and maximum goal*

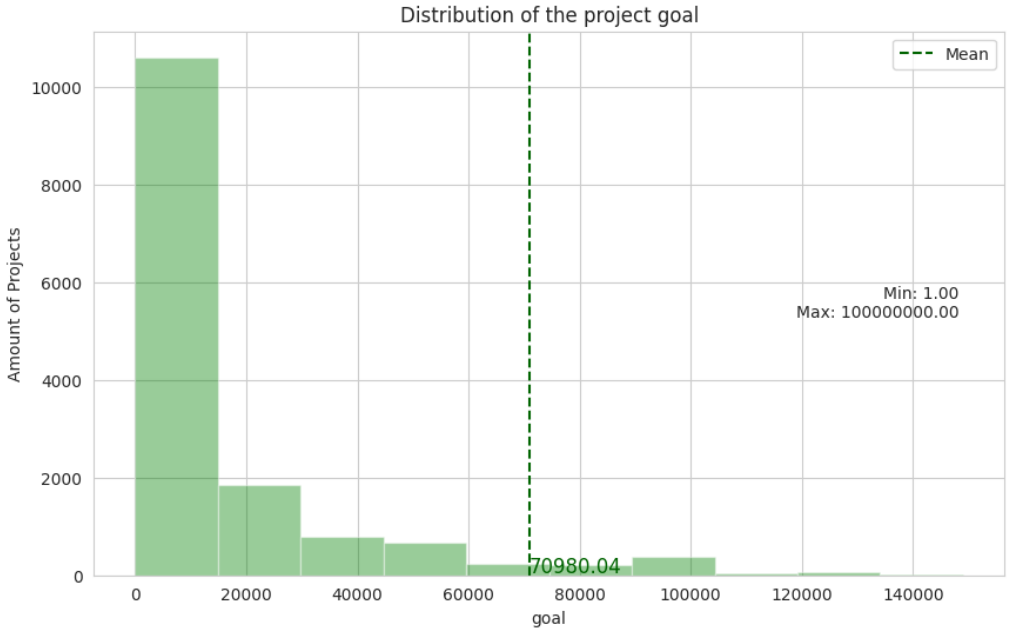

Figure 6: *Distribution of the time between the creation and the launch of a project, as well as recording of average time, minimum time and maximum time*

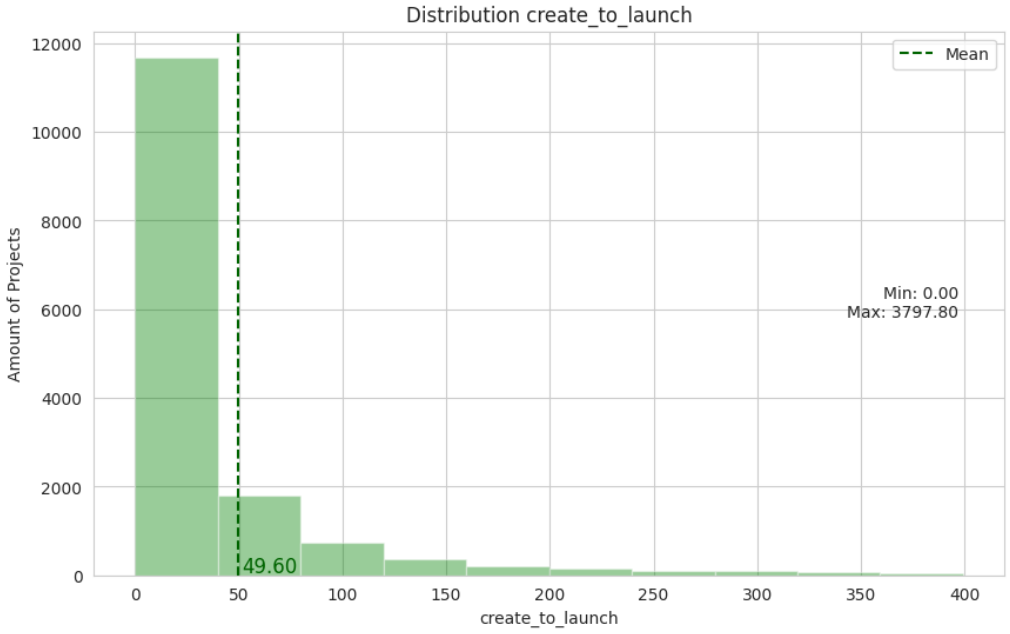

Figure 7: *Heat map showing the correlations between the variables state, goal, create_to_launch, create_to_deadline, launch_to_deadline, is_face_retinaface, is_face_opencv and the individual variable state.*

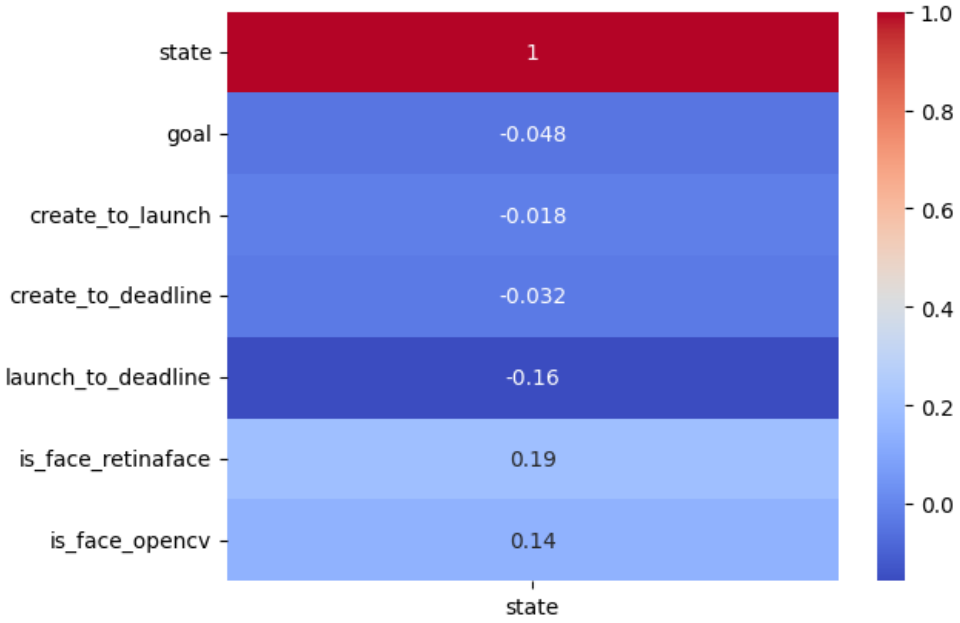

Figure 8: *Distribution of the project goal for the log transformation of the variable "goal"*

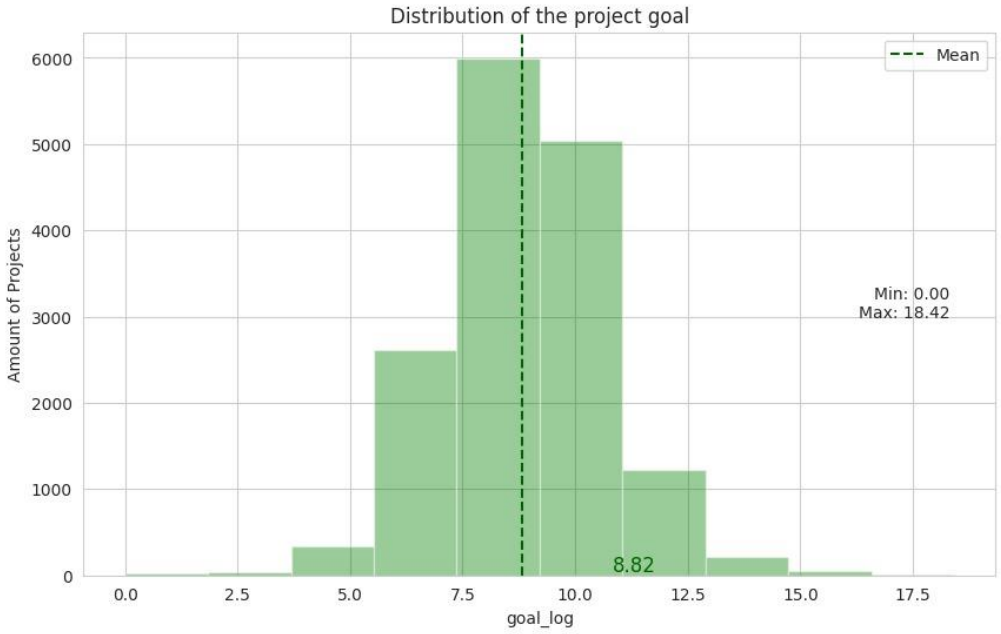

Figure 9: ***Image snippet of varying accuracy in golds***

```
X_explore = features[['goal_log', 'is_face_retinaface']]

clf = RandomForestClassifier()
scores = cross_validate(clf, X_explore, y, cv=10)['test_score']
accuracy = np.mean(scores)
print(accuracy)
print(scores)
```

```
0.6616588445376756
[0.5492595  0.69993561 0.80553767 0.69136598 0.60889175 0.48904639
 0.65141753 0.76353093 0.73453608 0.62306701]
```