372 Final Report

Tyler Mainguy, Will Murray

April 2020

1 Abstract

In the age of online funding, decisions about product creation and investment are extremely difficult to intuit. Application of machine learning models on the data set of Kickstarter projects will provide insights into product development and investment that can be used by developers and investors alike. Prediction of success given metadata gives fairly good results, while generation of novel product descriptions is more difficult. Future applications of novel product generation or company names could aid in providing a more robust platform.

2 Introduction

The advent of the internet has provided prospective visionaries and innovators an entirely new way of acquiring capital for their businesses; online fundraisers. No longer is it required for companies to take the typical route of acquiring rounds of funding, as they can directly market their product to the customers.

With the plethora of new products being pitched on the popular fundraising site Kickstarter, those interested in investing must be selective in the products they choose to support. Not only is this a problem for those interested in purchasing products, but additionally those who are creating them. Investing both time and money into a creative venture that doesn't pay out can be equally damaging to both parties. While online funding seeks to remove barriers in the entrepreneurial process, there are some that inevitably remain due to the uncertainty related to innovation.

The largest hurdle from both the development and the purchasing side of this problem stem from the lack of tooling available to assess and understand new products that are being pitched on online funding platforms like Kickstarter. Those who have invested in the past and ended up disappointed doubt their own ability to pick promising products. Those who are taking their first step in the world of this new form of product development are rightfully skeptical as to how well their money and time will be spent. Inherent bias in opinions and beliefs cloud the ability to effectively judge which products may be successful, and which are destined for failure.

Deep learning (and machine learning, more generally) has shown itself to be at the forefront at detecting complex patterns in data. Whether in the context of believable text generation given a prompt [1], or the ability for agents to learn strategies in hide and seek [2], machine learning and it's more specific applications such as deep learning have shown that given enough data and the right model, we can gain incredibly deep insights.

At the time of writing, there have been 180,931 successfully funded Kickstarter projects, and 299,569 unsuccessful [3]. As has been the case in previous applications of machine learning models to large corpus' of data, it is expected that there are more nuanced patterns that exist within the structure of the data that indicate whether or not a product will be successful. We're looking to apply such models to the data set of metadata regarding Kickstarter data in hopes of finding reliable ways to measure the potential success of Kickstarter campaigns, and to aid in formulation of the Kickstarter pitch in order to optimize for success.

3 Problem Statement

It's too difficult to choose a Kickstarter that is likely to be successful, and it's difficult to know if it's worthwhile to create a product that may end up going nowhere. We have not taken the opportunity to view the insights available through all of the data collected throughout the years.

4 Proposed Methods

Our proposed solution consists of several analysis tools that will aid in the ability not only to invest, but also to create. These analysis tools aim to not only identify more promising companies, but also to aid in the generation of a pitch that is likely to earn the amount of money required for product development.

4.1 Data Collection

There were two primary data sets used in this work. The first captured metadata for roughly 300,000 Kickstarter projects, both successful and not (cite this). The second data set consisted of the 4,000 most successful Kickstarter projects (up until the point that data set was created)(cite this). The former data set will be used for preliminary analysis along with our classification and regression models, while the latter will be used for text-generation purposes.

4.2 Preliminary Analysis

Preliminary review of the data was paramount in determining the correct course of action when developing our models. By gaining a greater understanding of the data itself, we began to understand what attributes were important along with those that would simply obfuscate any patterns.

We began with simple histograms of various aspects of our data in order to gauge their distributions. Primarily, we viewed the distribution of the main categories of Kickstarter projects (i.e. Music, Publishing, etc.) to see if there were certain projects that would be more present throughout the data set. We also viewed histograms of the outcomes of the projects, to compare in our data set how many had been successful versus unsuccessful. This is important to quantify, as if most of the projects are unsuccessful our model will be ”swamped” with a certain class type.

4.3 Pitch Classifier

After analyzing our data, the most pressing matter was to build a classification model that could identify which companies were the most likely to succeed based on their metadata. There were several good options in terms of model selection; our initial selection that ended up being quite successful was that of the LightGBM model provided by Microsoft [4]. This model balances the accuracy of tree-based models, while also being fairly lightweight which aided in fast model iteration.

The first step in this model was selecting the parameters to be used. The most promising attributes were; category, main category, country, USD pledged, USD goal, and the number of backers to the project. Other time related data was excluded from this model. Once the attributes were selected, transformations and standardization were performed in order to remove as much bias as possible in each attribute. Numeric attributes had missing values replaced by its median and scaled to a uniform distribution, whilst missing values in categorical data was removed, and each unique attribute value one-hot-encoded.

The target class for this problem is the project status; whether or not the project has succeeded. In the case of our data set, there were several classes; successful, failed, cancelled, undefined, live, and suspended. Undefined projects are few and due to the ambiguity in naming, were removed from the data set. Live projects were also removed from the data set, as it is not yet possible to determine their success. The remaining negative states (failed, suspended, cancelled) were grouped into the more general class "unsuccessful".

Our problem is now that of binary classification. The remaining transformed data was separated into train/test splits, with a testing size of approximately 30%. Grid search for hyper-parameter optimization was used to determine the best combination of hyper-parameters based on the training set. Accuracy of the model was then determined using the testing set.

An issue we detected when building this model came from the use of both the "USD pledged" and the "USD goal" attributes in a single model. Our model was learning the correlation between the two variables as a trend; if USD pledged was greater than USD goal, it was classifying it as a successful product. Given the dataset we had, this made sense, but ultimately was not the pattern we were trying to detect. We removed the attribute for "USD pledged" to avoid this complication.

4.4 Kickstarter Funding Estimator

Our next goal was to provide a reasonably good estimator of how much funding a given project would receive. To achieve this we first sought to determine the relevant features relating to the final amount pledged. This feature selection was done through a correlation test and manual regression analysis on the data. Basic linear regression models serve our purpose in this instance, being a low cost, low complexity way to determine a relationship between each feature and the final amount pledged.

This data has a rampant problem with outliers, as such, a standardization and scaling technique that does not depend so heavily on the mean and unit variance was needed. In this case, RobustScaler was used. This technique scales the numerical values by the using the median as opposed to the mean and the quantile range of the data instead of the unit variance. The median was also used to fill empty values. Categorical features were ran through a OneHotencoding. This encoding simply creates a numeric array with each string a unique value. Missing categorical values were simply removed.

With the relevant features understood and the data cleaned, a model needed to be created that would predict the final amount that would be pledged to a project, KNeighborsRegressor was used in this instance. Intuitively it makes sense for similar projects to garner a similar amount of interest and fundraising. This model takes advantage of this logic by establishing a prediction based on results of the projects most similar to the project being estimated.

This now becomes a solvable regression problem, though required to be more robust and generalizable than the simple regressions performed earlier. The data was separated into training **and testing groups and a grid search done to optimize the model parameters. The final model was tested on the test data and the r2 score was calculated to determine the accuracy of the model.

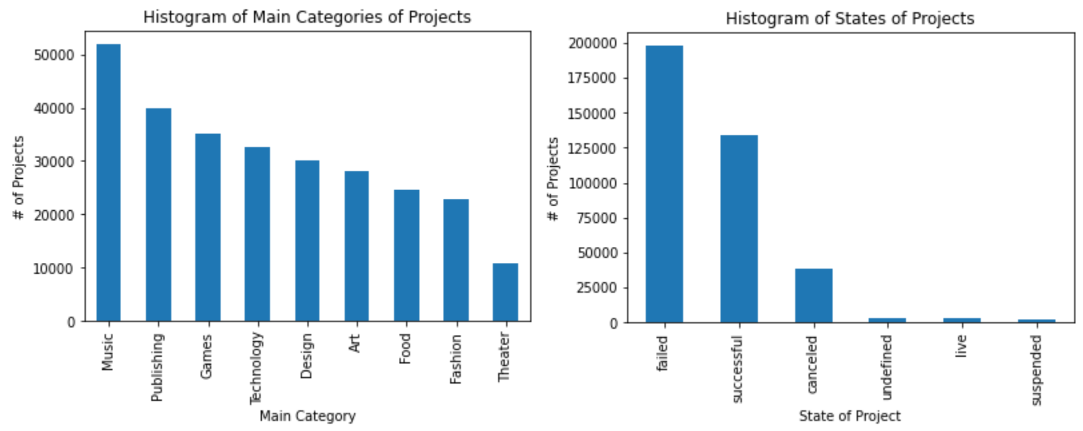
4.5 Kickstarter Blurb Generator

Our final feature was to build a text-generation model that could aid in the pitch generation of a project. The concept is as follows: given the company name and the main category of the product, the text-generation model will produce a pitch for said product using the language of the most successful Kickstarter ventures. The inherent belief in this approach is that there's something unique about the language used in the most successful pitches, and that accessing the underlying patterns in that language will aid in your ability to persuade potential customers.

Cleaning of the text data was undertaken first. Removal of punctuation (along with other strange ASCII characters) allows for the model to only train on the text portions of the data, without needing to worry about patterns in punctuation. Afterwards, the remaining text was converted into integer form using a mapping from character to integer value (as RNNs can only take in numeric input). All of the existing blurbs were then combined into one sequence of characters, such that our model could then train on it. The model for this text-generation system was based using multiple LSTM layers, with a fully-connected dense layer at the end of the network for character prediction.

The model we're aiming to generate needs to create a novel pitch, and avoid plagiarizing previous works. In an attempt to achieve this, the temperature of the softmax layer was increased. Temperature is an approach available for activation functions that helps change the "exploration" of the model; when the temperature is low, the model will stick to patterns it's seen before, maintaining low grammatical errors along with coherent text. By increasing the temperature, we're allowing the model to explore new avenues of character generation. This in turn will allow for more creative pieces of text to be written.

Another method used in order to avoid overfitting was to include dropout layers within the neural network. Dropout layers are a probabalistic method in which you



(a) (b)

Figure 1: Histograms of important aspects of our data. (a) gives the histogram of the main categories of the Kickstarter projects in the data set, whilst (b) shows the histogram of the status of the Kickstarter projects in our data set.

”remove” certain neurons from a forward/backward pass through a neural network. This is especially useful in the case of large networks with relatively small data sets, where it is likely that the network will overfit the data. By adding in several dropout layers between the LSTM layers of the network, we were effectively able to regularize the network and avoid overfitting.

5 Results

5.1 Preliminary Analysis

Histograms generated from main category and project status columns can be seen in Figure 1.

The correlation analysis on the numerical data can be seen in Figure 2.

5.2 Pitch Classifier

After training the model for several iterations, the final result for accuracy was 93.43%. The distributions of classifications can be seen in the confusion matrix in Table 1.

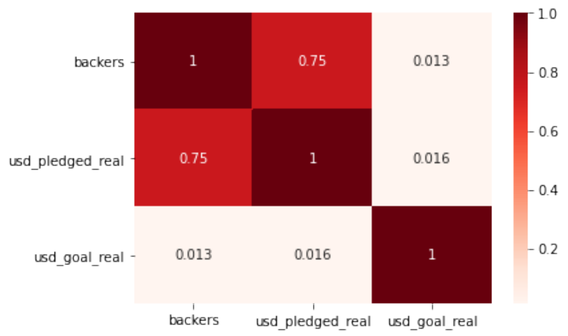


Figure 2: Correlation analysis of numerical data.

		True diagnosis		
		Positive	Negative	Total
Prediction	Positive	67,537	4028	71,565
	Negative	3307	36,818	40,125
	Total	70,844	40,846	111,690

(1)

Table 1: Confusion matrix for the testing set classification of the pitch classifier model.

5.3 Kickstarter Funding Estimator

After training the model for several iterations, the final regressor accuracy was 58%.

5.4 Kickstarter Blurb Generator

Results from various seeds are shown below; it is clear that the model remained underfit, an unfortunate after effect of lacking computing resources for sufficient training. The model was trained on approximately 150 epochs, with dropout layers containing 30% dropout likelihood. Temperature was set to $T = 2$, and applied before the final softmax function.

Given: " oolet the slimmest smart wallet for the modern man "

Response: "ing product design the worlds first smart project product design the worlds first smart project product design the worlds first smart project product design ..."

Given: "the newest dungeons and dragons tabletop game"

Response: ”the worlds first smart product design the worlds first smart project product design...”

6 Discussion

Analysis of the histogram of the main category data gave an intuitive idea as to what were the most relevant features in the data set. It was clear that many projects that were undertaken were in the music, publishing, and gaming industry; all creative avenues for design. Understanding this, we can compare the results of our classification model against these creative categories and see how well they perform. This can give us an idea as to whether or not the magnitude in which the projects are creative actually reflects the previous successes of such creative endeavours.

The classification model performed exceptionally well on the task of classifying whether or not a business would be successful. The model was able to perform with 93.43% accuracy on the testing set, with only the attributes mentioned above. While this still leaves some room for uncertainty regarding whether or not you should invest in a business, treating this tool as part of a preliminary analysis can give the user a much clearer idea as to whether or not this business would succeed. The distribution between false negatives and false positives was relatively equal, which in our application is satisfactory. If we wanted to be much more stringent in terms of weeding out poor products we could shift our model away from false positives, but we don’t believe the margin of error to be large enough such that this would be an issue.

The correlation analysis prompted a deeper dive on the backers and goal features. A regression analysis on the backers showed an R2 value of 0.55 and 0.05 for the goal. Regressions were then done on the categorical features, the one with most relevance being main category, however the R2 value was only 0.02 so it was not a huge contributor to the final amount pledged. An amount of data had to be dropped in order for the regressions to run without returning nonsensical results; outliers were dropped from the dataset manually. Anything with less than 10 backers was eliminated. For a project to fail to garner even 10 backers there is likely a major pitfall in advertising and thus wasn’t included here. Projects which exceeded 5 Million USD pledged were excluded as were projects with goals beyond 1 Million USD because this amount of success and high expectations are not frequent nor easily repeatable.

The funding estimator using the information gained from the deeper dive into the data returned somewhat satisfactory results, though not as accurate as initially hoped. The model had 58% accuracy on the testing set which serves as a useful ballpark estimate on the level of success a project will have. Paired with the classification model explored earlier this becomes a very useful second check on how well received and supported the project will be.

Unfortunately, the blurb generation model consistently underfit to the data set, resulting in sentences that always tended towards a certain sequence of characters; in this case, it was "the worlds first smart project product design". There could have been several reasons that this occurred; dropout probability was too high, data wasn't trained on enough epochs, and the presentation of the data was flawed in some way. The dropout layers involved in our network aimed to decrease the overfitting nature of LSTM character-generators, but instead may have limited the models ability to fit the data. An obvious issue with this trial was the lack of epochs that the model was trained on. With an expected epoch time of 30 minutes, running hundreds or thousands of epochs to get this model effectively trained was nearly impossible. Settling with approximately 150 epochs, it was evident that our model was beginning to learn effective patterns, but didn't have the time to expand on them. Finally, potential changes to the text cleaning and preprocessing may have aided in our work; stemming was not used, which could have made it easier for our model to fit to the data set (smaller vocabulary). The three techniques mentioned in this work have shown promising results in predicting successful businesses, along with aiding in the generation of a successful pitch. Future work of text-generation models that can be trained on the entire pitch itself (roughly 20 times larger than the blurb) could act as another method of assistance for pitch generation.

References

[1] Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language models are unsupervised multitask learners. OpenAI Blog. 2019;1(8):9.

[2] Baker B, Kanitscheider I, Markov T, Wu Y, Powell G, McGrew B, et al. Emergent tool use from multi-agent autocurricula. arXiv preprint arXiv:190907528. 2019;.

[3] Kickstarter Stats;. Available from: https://www.kickstarter.com/help/stats.

[4] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, et al. Lightgbm: A highly efficient gradient boosting decision tree. In: Advances in neural information processing systems; 2017. p. 3146–3154.