Predicting Kickstarter Success: An Analysis of Linguistic Factors and Model Selection

Albert Kuo
TTIC 31190: Natural Language Processing
University of Chicago
Chicago, IL, USA
albertkuo@uchicago.edu

Abstract

Crowdfunding sites like Kickstarter have allowed the online community to fund a growing number of entrepreneurial and artistic projects. In this paper, we will examine what factors contribute to the success of Kickstarter projects. Using a variety of models, we analyzed a corpus of 130k crowdfunded projects and found that the language used in project pitches can discriminate between projects that succeed and projects that fail up to an accuracy rate of 61%. However, with the addition of other project attributes, we can achieve an accuracy rate of 76%. This suggests that the influence of project pitches on a project's success may be limited in comparison to other factors.

1 Introduction

Kickstarter is an online crowdfunding platform where entrepreneurs look for funding for their creative projects. Users can pledge a contribution to projects that they are interested in and contribute their pledged amount if the project succeeds in reaching its funding goal. However, little research has been done on what factors, particularly linguistic factors, attract users to contribute to a project. Understanding these factors can shed light on the role of persuasive language in the context of crowdfunding and help project creators better craft their pitches.

To address this problem, we analyzed the pitches of Kickstarter projects and used these to predict a project's success while controlling for other factors known to have an impact on project success.

2 Related Work

Studies have shown that non-linguistic factors such as the inclusion of a video, higher goals, and a longer project duration lead to lower chances of success [4, 5]. Using such factors, past studies have achieved an accuracy rate of 67% in predicting whether a project would succeed or fail [2]. Mitra and Gilbert extended this to include linguistic content found in project pitches in their model [3]. By examining weights in their model, they extracted phrases most predictive of a project's success.

We extend past research by applying additional models, including a linear model and neural networks, and comparing the accuracy rates across the different models. Furthermore, we also examine possible phrases predictive of a project's success within the context of the different models.

3 Methods

3.1 Linear Classification Model

The linear classification model assigns a score to each output class based on binary features. Our input variable $x = \{$pitch, goal amount, staff pick, and project category$\}$ and our output variable $y = \{$success or failure$\}$. These variables will be discussed in greater depth in Section 4. We included unigram, bigram, and trigram features in our model. The score(x, y)= $\sum_{i=1}^{n} \theta_i f_i(x, y) + \sum_{j=1}^{m} \theta_j g_j(x, y)$, where $f_i(x, y)$ are indicator n-gram features and $g_j(x, y)$ are indicator features for our controlled variables: goal amount, staff pick, and project category. The perceptron loss function was used to learn the weights for the features.

3.2 Data Processing

The top 20 n-gram features with the highest weights in the linear model were selected as filters to process the variable-length pitches into fixed-length vectors for the logistic regression model and feed-forward neural network. Word vectors trained on Twitter data were used [1]. Each filter is convolved with every word (or n-gram) in the project pitch by calculating the cosine similarity of their word vectors to obtain a similarity vector for each filter. The highest similarity in each similarity vector was used to obtain a vector of length 20 (max-pooling). The vector for the pitch is then appended to the vector representing the controlled categorical variables for the final input vector. This vector is used as the input for the feed-forward network and the logistic regression model.

3.2.1 Feed-forward Neural Network

A zero-layer feed-forward neural network was used. A one-layer feed-forward neural network with 128 hidden nodes was also examined, but since there was no apparent gain in classification accuracy with the additional layer, the more interpretable zero-layer feed-forward neural network is discussed in our report. The well-known back-propagation algorithm learns the weights to obtain a score that is a nonlinear transformation of the input vector, using ReLU activation.

3.2.2 Logistic Regression

Logistic regression is a well-known probabilistic classification model. The model is

$$\log \frac{p(x)}{1-p(x)} = \beta_0 + \beta x$$

and we predict $y = 1$ when $p \geq 0.5$.

4 Experimental Setup

Kickstarter data was scraped by Web Robots[1] on January 28, 2016. The dataset contains 147,429 projects and of these, 132,853 projects have reached their end date. We ran our models on the subset of projects that have reached their end date. Within this dataset, 63,542 projects were successful, having reached their funding goal, and 69,311 projects failed.

The main explanatory variable is the project pitch: a short description of the project. The pitch is visible not only on the project page, but also whenever the project is featured on other pages. Thus, it plays an important and highly visible role in attracting potential contributors.

[1]https://webrobots.io/kickstarter-datasets/

The baseline variables consist of the following variables. We controlled for these variables by adding them to our final models.

Project Goal Amount: The amount of money (in USD) the project aims to raise by the conclusion of the project duration. The project goal amount was highly right-skewed and the goal amount was therefore converted into a categorical variable. Projects were categorized into high ($\geq$\$10,000), middle (between \$3,000 and \$10,000), and low ($<$\$3,000) goal amounts.
Project Categories: There are 143 unique categories of projects in our dataset (e.g. “Digital Art,” “Indie Rock,” “Poetry,” etc.).
Staff pick: Kickstarter projects may be featured as a staff pick on the website.

The dataset was randomly split into a training, a devtest, and a test set. The training set consists of 90% of the data, and the devtest and test set each contains 5% of the data. Model tuning and selection was conducted on the devtest set and final accuracy rates were calculated on the test set.

To evaluate our results, the classification accuracy rate was used. This is the number of projects predicted correctly out of the total number of projects in the test set.

4.1 Linear Classification Model

For our n-gram features, we used a feature count cut-off of 1 in the training set. This resulted in 130k unary features, 935k binary features, and 1392k ternary features. The model was tuned using early stopping on the devtest set. The model was run for 10 epochs.

4.2 Feed-forward Neural Network

The neural network was run with a learning rate of 0.01 and 300 epochs. To tune the model, regularization, dropout and early stopping on the devtest was used. The neural network was run using TensorFlow.

4.3 Logistic Regression

A L1 penalty was used to tune the model, such that the likelihood of the weights is

$$l(\theta) = \ln(\prod_{i}^{N} P(y_i|x_i,\theta) - \lambda|\theta|)$$

where λ is the penalty parameter. The penalty parameter was selected using the highest devtest accuracy. Logistic regression was run using the sklearn toolkit in Python.

5 Results and Analysis

Across the different models, classification accuracy is 10% to 20% higher with baseline features than with text. Accuracy is similar when using baseline features or when using baseline features with the addition of text. This suggests that text does not add much to predictive power beyond the baseline features. In other words, the signal that is present in the text can already be captured by the baseline features.

5.1 Linear Classification Model

Bigrams and trigrams did not improve classification accuracy, as simply using unigrams alone achieved a classification accuracy of 0.6118, in comparison to 0.6113 when bigrams and trigrams are added. However, to examine potentially predictive n-grams of longer length, we included bigrams and trigrams in the model.

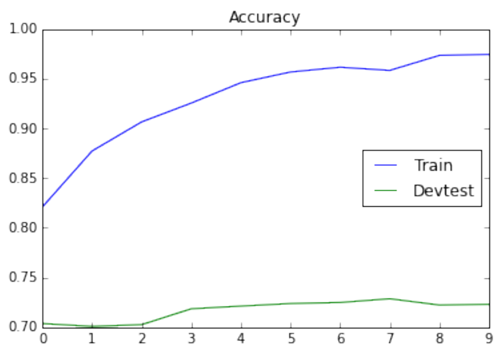

Figure 1: Accuracy Rate for Linear Model

When both text and baseline features are used, classification accuracy only improves for the training set and has little impact on the devtest set (Figure 1). This suggests that the predictive textual phrases we learn may not generalize well to other projects.

The weights of the features were used to determine the predictive power of n-grams (unigrams, bigrams, and trigrams) in the model. Table 2 lists the top n-grams for successful and failed projects, when controlled for baseline features. For the “failed” features, a theme of uncertainty or hesitancy seems to be present (“trying to start,” “trying to bring,” “releasing their first”).

Successful	**Failed**
pens	arias
stretch goal	trying to start
really need	trying to bring
zine	sure to
bronx	hobby
residency	neon
visual novel	a higher
tesla	releasing their first
layered	needing
help to raise	live in the

Table 2: Top 10 Weighted N-Grams

As expected, other factors also played an important role in predicting success and were highly weighted. In particular, the categories of “Documentary,” “Tabletop Games,” and “Shorts” were predictive of success, while the categories of “Young Adult,” “Hip-Hop,” and “Mobile Games” were predictive of failure. Staff picked projects and projects with a low goal amount were also highly predictive of success.

Table 1: Summary of Accuracy Rates For Each Model

Model	Data	Classification Accuracy
Linear Model	Text	0.6113
	Baseline	0.7090
	Text+Baseline	0.7194
Logistic Regression	Text	0.5790
	Baseline	0.7561
	Text+Baseline	0.7585
Feed-forward Neural Network	Text	0.5687
	Baseline	0.7557
	Text+Baseline	0.7561

5.2 Feed-forward Neural Network

The difference in the weights for a successful outcome and a failed outcome for each n-gram features are plotted in Figure 2.

We would expect features predictive of success to have a positive difference and features predictive of failure to have a negative difference. The bars are colored according to whether the feature was predictive of success or failure in our linear classification model; the top "successful" n-gram features are white and the top "failed" n-gram features are gray. Thus, we would expect white bars to be positive and gray bars to be negative. However, this is not what we observe. Though 9 out of 13 of the "succesful" features have positive differences, only 2 out of 7 of the "failed" features have negative differences. This suggests that the features learned in the linear model are used differently in the feed-forward neural network. Though not shown in the plot, the goal category has a negative difference and staff pick has a high positive difference, which is what we expect, suggesting that these features outweigh the top n-gram features and are more consistently predictive.

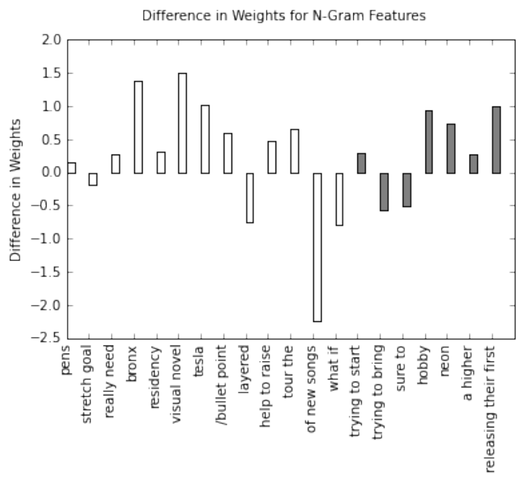

Figure 2: Weights of N-gram Features

5.3 Logistic Regression

The highest odds ratio for the n-gram features are plotted in Figure 3. An odds ratio greater than 1 indicates that the feature increases the probability that a project will fail. Two features have high odds ratio ("sure to" and "hobby"). Comparing these weights to Table 2, we see that in general, features predictive of success, which have white bars, have lower odds ratio than features predictive of failure, which have gray bars. However, there are several exceptions, suggesting that there are again limits to the predictive power of these n-grams. Furthermore, though not displayed on the plot, the goal category has an odds ratio close to 2 and staff pick has an odds ratio of 0.1, which is what we would expect, suggesting again that these features are more consistently predictive than linguistic ones.

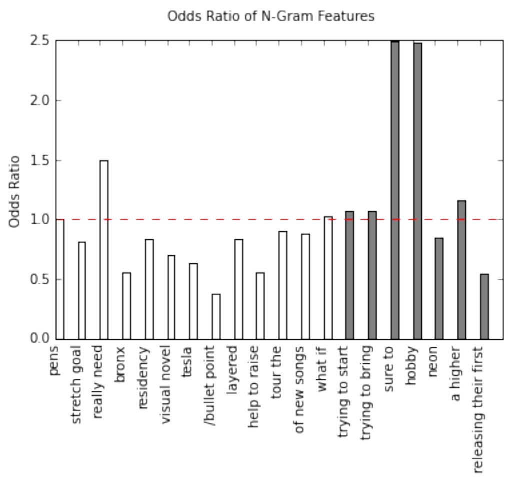

Figure 3: Odds Ratio of N-gram Features

6 Conclusion

Our results indicate that the language in project pitches do have some predictive power, as linguistic factors alone achieve an accuracy rate between 56% to 61%. However, when baseline features are added, they contribute little to the model. As a result, the predictive information contained in the project pitches may simply reflect these other factors, such as the category of the project.

The low accuracy rates also suggest that there might not be substantial differences among the project pitches. Thus, extracting predictive power from the linguistic content in project pitches alone may have an upper bound. Possible areas for further exploration would be to add text from other areas in the project page to the model. While project pitches are likely to be the most carefully honed textual content, other areas may not be and may therefore be more indicative of whether a project can attract potential backers and ultimately succeed.

Furthermore, the scope of this paper focused on n-grams learned using a linear model. Other models such as convolutional neural networks may find different predictive n-grams and is another area for future work.

References

[1] Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., and Smith, N. A. (2011). Part-of-speech tagging for Twitter: annotation, features, and experiments. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 42-47, Portland, Oregon, USA. Association for Computational Linguistics.

[2] Greenberg, M. D., Pardo, B., Hariharan, K., and Gerber, E. Crowdfunding support tools: predicting success & failure. CHI EA 13 (2013).

[3] Mitra, T., Gilbert, E. The Language that Gets People to Give: Phrases that Predict Success on Kickstarter. CSCW'14, ACM (2014).

[4] Mollick, E. The dynamics of crowdfunding: Determinants of success and failure. SSRN scholarly paper, Social Science Research Network, Rochester, NY, July 2012.

[5] Muller, M., Geyer, W., Soule, T., Daniels, S., and Cheng, L.-T. Crowdfunding inside the enterprise: employee-initiatives for innovation and collaboration. Proc. CSCW'13, ACM (2013), 503512.