Crowdfunding Support Tools: Predicting Success & Failure

Michael D. Greenberg
2133 Sheridan Rd.
Evanston, IL 60208 USA
mdgreenb@u.northwestern.edu

Bryan Pardo
2133 Sheridan Rd.
Evanston, IL 60208 USA
pardo@northwestern.edu

Karthic Hariharan
2133 Sheridan Rd.
Evanston, IL 60208 USA
karthichariharan2012@u.northwestern.edu

Elizabeth Gerber
2133 Sheridan Rd.
Evanston, IL 60208 USA
egerber@northwestern.edu

CHI'13, April 27 – May 2, 2013, Paris, France.
ACM 978-1-XXXX-XXXX-X/XX/XX.

Abstract

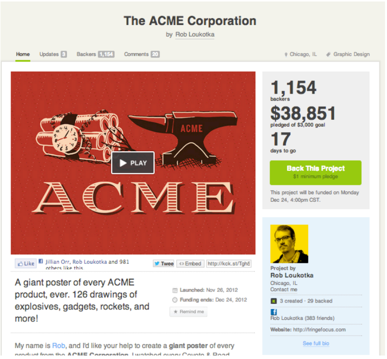
Creative individuals increasingly rely on online crowdfunding platforms to crowdsource funding for new ventures. For novice crowdfunding project creators, however, there are few resources to turn to for assistance in the planning of crowdfunding projects. We are building a tool for novice project creators to get feedback on their project designs. One component of this tool is a comparison to existing projects. As such, we have applied a variety of machine learning classifiers to learn the concept of a successful online crowdfunding project at the time of project launch. Currently our classifier can predict with roughly 68% accuracy, whether a project will be successful or not. The classification results will eventually power a prediction segment of the proposed feedback tool. Future work involves turning the results of the machine learning algorithms into human-readable content and integrating this content into the feedback tool.

Author Keywords

Machine learning, crowdfunding, crowdsourcing, sentiment analysis, Kickstarter, AdaBoost.

ACM Classification Keywords

I.2.6 [**Learning**]: Concept Learning – *decision tree, support vector machine, boosting.*



The ACME Corporation

by Rob Loukotka

This project will be funded on Monday Dec 24, 4:00pm CST.

A giant poster of every ACME product, ever. 126 drawings of explosives, gadgets, rockets, and more!

Launched: Nov 26, 2012

Funding ends: Dec 24, 2012

Website: http://fringefocus.com

My name is Rob, and I'd like your help to create a **giant poster** of every product from the **ACME Corporation**. I watched every Coyote & Road

Figure 1: An example page on Kickstarter.com

General Terms

Measurement, Performance, Design, Economics, Experimentation, Human Factors.

Introduction

Crowdfunding is the process of soliciting financial contributions from a large group of individuals to raise funds. Since 2007, online crowdfunding has emerged as a new means for creative types to receive funding for new ventures. Increasingly, though, novices are using online crowdfunding to raise funds for the first time [6, 11]. Yet, few tools exist to support novices.

As a broad goal, we are looking to develop tools to enable novice creators to successfully use crowdfunding, where success in crowdfunding is defined as reaching or exceeding a fundraising goal. For example, a project with a goal of $5000 that raises $4999 would be considered "failed," while one which raises $5001 would be considered "successful." A year long study of the crowdfunding community [5], revealed an urgent need for a tool for project creators to get feedback as to whether their projects were likely to be successful, so as to make revisions before launching. One component of this feedback tool would be the comparison of the traits of an individual's project to the traits of other, successful projects, in a manner similar to the research through design approach pioneered in HCI [13]. The next step would be identifying where the individual's project could be improved. Therefore, this research is motivated by the following research question:

Can we train a machine learner to identify the traits of a successful crowdfunding project before launch?

Since there is a huge amount of data online from the thousands of crowdfunding projects that have been posted we wish to explore the efficacy of using machine learning classifiers to determine whether projects will be successful before they launch. To this end, a novice crowdfunder could use a tool based on these algorithms to determine whether his or her project is likely to succeed and possibly correct errors in the pre-launch phase.

Dataset

We use a pre-scraped dataset of project pages from kickstarter.com provided by the owners of thekickbackmachine.com, a Web site that scrapes kickstarter.com and shows aggregated statistics on projects [10]. The dataset provides information on over 13,000 project pages on Kickstarter.com, the most popular US-based crowdfunding website [1]. While the KickBackMachine is open access, the data we used is not publicly available. We used data on all projects that finished between: 6/18/2012 and 11/9/2012.

Since project pages on Kickstarter are all similarly structured, scraping data from Kickstarter is straightforward. The structure of crowdfunding pages includes a video (optional), a goal, a project description, reward structure, and links to social media platforms (Figure 1). From each project page, we scraped and calculated a variety of attributes, which can be seen in Table 1.

The attributes *sent, fkgl, and sent_count* were calculated from the text of the project description (the main body of text on the project page), and were not scraped directly. For the sentiment attribute, we used the Mashape Text-Processing API, a public, and pre-

Attribute	*Description*	*Type*
goal	Goal in dollars of the project	Integer
parent_category_ string	Project category (eg. Music, or Dance, or Video Game)	String
reward_count	Number of rewards available	Integer
duration	Length of project in Days	Double
twitter_url	Connected to twitter	Boolean
has_video	Video present	Boolean
facebook_connected	Connected to Facebook	Boolean
facebook_friends	Number of facebook friends	Integer
twitter_followers	Number of twitter followers	Integer
sent	Sentiment (pos, neg, or neutral)	String
fkgl	Grade level	Double
sent_count	Number of sentences in project description	Integer
project_success	Outcome variable	Boolean

Table 1: Scraped and Calculated Attributes

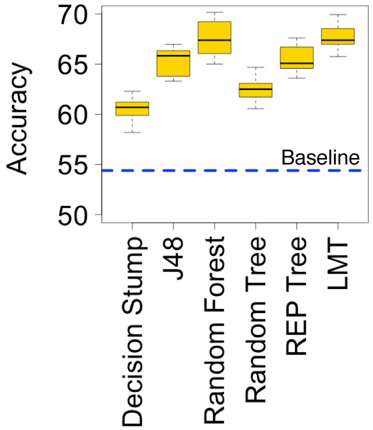


Figure 2:Performance of Decision Tree Algorithms

trained implementation of the NLTK natural language processing library to classify the sentiment of text [3].

The Text-Processing API is a useful implementation of a sentiment classifier as it is pre-trained and allows 35,000 free classifications per month. The attributes, *fkgl* and *sent_count,* were calculated from a Python script [3].

Learning Methods & Software:

We ran the dataset (with the attributes described in Table 1) through a variety of different classification algorithms. Our baseline was the a priori probability of successful Kickstarter projects, which in this dataset was 54.35%. Since we were only interested in classification given the initial conditions, we did not consider attributes of a project that are obtained after or during the funding process (These attributes included the number of comments posted on a project as well as the number of resulting backers).

We were interested in evaluating the performance between various decision tree algorithms and support vector machines with different kernel functions. We evaluated the performance of radial basis, polynomial and sigmoid kernel functions with varying costs for support vector machines. For decision trees, we used J48 Trees, Logistic Model Trees, Random Forests, Random Trees and REPTree. Next, we decided to choose the highest performing set of algorithms and boost them using the AdaBoost algorithm to see if accuracy improved [12].

To run the learning methods on the data set described in section 2, above we have used WEKA (v.3.7.7), a machine learning package from the University of Waikato [7]. Weka comes pre-installed with a variety of machine learning algorithms. To use a SVM learning method however, requires an additional package: LibSVM, which was installed separately [4]. Each method was run with through 10-fold cross validation to gather a distribution of the resulting accuracy.

Results:

The results we achieved through the basic set of variables described in Table 1 are encouraging, we are able to predict the success of a crowdfunding project with 68% accuracy, for an improvement of roughly 14% over the baseline. Figures 2, 3, and 4 graphically represent the results. Figure 2, compares the performance of various decision trees to a priori classification rate, while Figure 3 compares the a priori results to the SVM classifiers. Figure 4 compares the best performing algorithms to AdaBoost-ed counterparts.

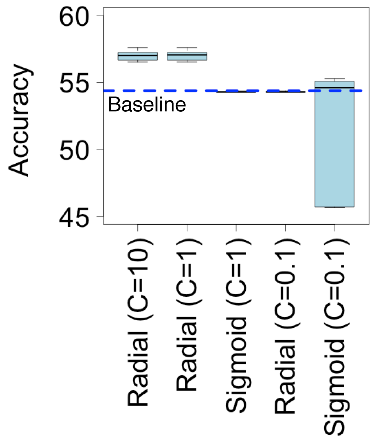


Figure 3: Performance of SVM's with varying cost functions and kernel functions

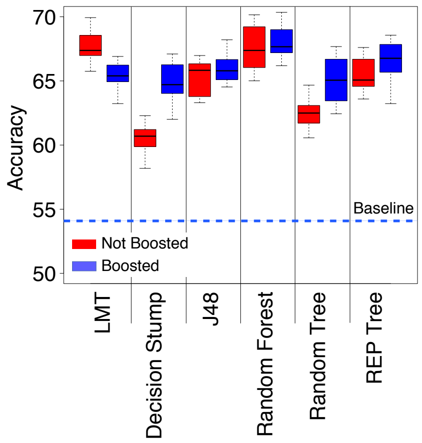


Figure 4: Comparison of boosted to un-boosted algorithms. Blue represents un-boosted, while the corresponding red value represents the boosted result

On the whole, simple classification algorithms, such as decision trees performed the best (Figure 2), while more complex algorithms such as SVMs performed at or around the baseline level (Figure 3). Furthermore, we found that boosting simple algorithms using the AdaBoost algorithm further improved the results with simple algorithms (Figure 4). Simple algorithms will work best with the feedback tool we are currently developing as it will allow for near instantaneous feedback for the end-user.

In all cases, decision tree algorithms perform around 10% better than a baseline guess for all projects to fail. The range of accuracy for the six decision tree algorithms ranged from just below 60% to just above 70% in one case. In practice it appears that random forest and logistic model trees perform the best.

We see that SVMs provide an average accuracy around 54.43%, which is almost the same as the baseline accuracy. Running an SVM with a radial basis function returned results marginally better than the baseline value, while a SVM with a Polynomial Kernel function performed slightly worse than the baseline. In reality the SVM mostly returned classifications for all projects to succeed, which explains why they mostly hover around the baseline value.

Since decision trees were a clear winner in this contest, we wanted to investigate if using AdaBoost would improve our classification percentage. For this experiment, we ran each of the six decision tree algorithms from before with AdaBoost [12]. Again, each learning method was run with 10-fold cross validation. Our results are illustrated in figure 4.

Boosting provides mixed results, in the case of simple algorithms such as decision stumps and random tree, boosting provides a bit of an improvement, around 3% accuracy. However, for more complex decision trees, boosting provides little improvement. Additionally in the case of logistic model trees, boosting actually decreased accuracy.

Discussion

Overall we believe that the performance of our classifier were satisfactory. But our accuracy seems to hit an upper bound of 67% irrespective of how we break down the dataset. This suggests that there is a possibility of the existence of a hidden variable that would help us classify better. Possible additional variables could be the audio quality of the video posted on the Kickstarter page, past experience with crowdfunding, age, gender, location and network connectedness, as well as the actual content of the text and video.

Another interesting phenomenon that we noticed was an analysis of the running times of some of these algorithms on our data compared to their accuracy. So we picked our six best performing algorithms and have illustrated the results in Table 2.

In the case of a user-facing tool, building a model with minimum resources (like time and memory) is of high importance. It is very encouraging in this case, that a simpler model like a Random Forest or a Boosted Decision Stump performs almost as well as a complex model like Logistic Model Trees. In order to get a boost of a few percentage points, we have to run a model significantly more complex and computationally intensive that the simple model. This seems to subscribe well with the theory of diminishing returns

Algorithm	Run Time (s)	Accuracy (%)
LMT	215.8	67.68
Random Forest	1.52	67.53
JRIP	3.32	67.17
REPTree	0.57	65.56
Boosted Decision Stump	0.60	65.10
Logistic Regression	0.71	65.09

Table 2: Timed Results of Model Runs

presented by Hand, and would allow an end-user of a tool powered by these algorithms to receive results rapidly [8].

Furthermore, if we include the number of backers in the model, our accuracy jumps to around 90%, while if we run the model with number of backers as the only attribute, accuracy hovers around 77%. This would be useful for the tool, as we could tell users that if they can motivate a certain number of people to contribute to the project, we can predict their success with a greater degree of confidence. This would give users goals to strive for, and could improve the usefulness of the support tool.

Future Work:

In the future, we are going to build these machine learning algorithms into a larger scale, user-facing feedback tool, which could give guided feedback, such as: "We noticed your project doesn't have a video. Projects with videos are 10% more likely to be funded." While the current idea is to assist users in the pre-launch stage of online crowdfunding projects, the methods we describe here could be adapted to a broader-scale creativity support tool.

The machine learning algorithms we describe are powered by a scraped dataset. Further processing on the scraped data might improve the prediction accuracy of the algorithms. In the future, we will run more analysis on the text content of the project page. An approach using a Naïve Bayes classifier on project text would be an interesting approach, and would begin to get at the actual content of project's pitch, but it would require scraping the text of each project as it launched. We will investigate this type of approach in the future.

In addition, we would like to investigate how the impact of a crowdfunding project creators' social network (both online and offline) influence rates of success.

However, every approach we have considered up and until this point relies strictly on scraped data. We are certainly aware that crowdfunding success is not directly related to scrape-able attributes, and should be affected by abstract concepts such as the effectiveness of the pitch or the professionalism of the associated video. To this end also wish to consider the possibility of using Amazon Mechanical Turk workers to evaluate the abstract strengths and weaknesses of each project.

The design process requires iteration [13]. Another way we could construct this system would be to build an application that would predict the success after the completion of each campaign day. Such an approach would encourage end-users to iterate their project design during the length of the campaign, as their success score varies. This approach would require us to have training data of a set of Kickstarter projects over their duration and do an analysis over that which we do not currently possess.

Currently there exists very few tailor-made tools for crowdfunders to assist in the planning of crowdfunding projects [9]. Concurrently, the population of crowdfunders, and the amount of money being raised by crowdfunding is growing at a tremendous rate [2]. The tool we are currently working on has the opportunity to have enormous impact within this new and growing community. The results presented above, indicate that machine learning techniques could be used to help crowdfunders in project planning.

Conclusions

Prospective crowdfunders need tools to help predict their campaign's success before they launch. We used Machine Learning algorithms to help them do so.

In this project we applied machine learning techniques to a dataset of Kickstarter projects to determine whether we could classify projects as successful or failures at the time of launch. Our work in this area is in support of a user-facing tool to assist novices with project planning. The idealized end result of this tool is a prediction engine that can be used to advise users in project creation, and to open access to crowdfunding to those who haven not previously completed creative ventures.

To support this prediction engine, we ran a variety of classification algorithms, ranging from decision trees, to SVMs. The decision trees provided the best results, and ran the fastest, hovering around 67% accuracy, 14% above a baseline value. We are encouraged by this result, but we look to explore future improvements with additional attributes in the coming months. As a broader-scale goal, we look forward to using these findings to power a user-facing tool for generating feedback novice crowdfunders. As a growing community, a tool like this could have a lasting and meaningful impact which we hope to provide.

References

[1] Alexa: Kickstarter: *http://www.alexa.com/siteinfo/kickstarter.com*.

[2] Best of Kickstarter 2012: *http://www.kickstarter.com/year/2012*. Accessed: 2013-01-09.

[3] Bird, S. 2006. NLTK: the natural language toolkit. *Proceedings of the COLING/ACL on Interactive presentation sessions* (2006), 69–72.

[4] Chang, C.C. and Lin, C.J. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*. 2, 3 (2011), 27.

[5] Gerber, E.M. et al. Crowdfunding: Why People are Motivated to Participate.

[6] Greenberg, M.D. and Gerber, E. 2012. Crowdfunding: A Survey and Taxonomy. *Segal Technical Report: 12-03*. (2012).

[7] Hall, M. et al. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*. 11, 1 (2009), 10–18.

[8] Hand, D.J. 2006. Classifier technology and the illusion of progress. *Statistical Science*. 21, 1 (2006), 1–14.

[9] Hui, J.S. and Gerber, E. 2012. Easy Money? The Demands of Crowdfunding Work. *Segal Technical Report: 12-04*. (2012).

[10] Kickstarter: *http://www.kickstarter.com*. Accessed: 2012-09-11.

[11] Lambert, T. and Schwienbacher, A. 2010. An empirical analysis of crowdfunding. *Social Science Research Network* (2010).

[12] Schapire, R.E. 1999. A brief introduction to boosting. *International Joint Conference on Artificial Intelligence* (1999), 1401–1406.

[13] Zimmerman, J. et al. 2007. Research through design as a method for interaction design research in HCI. *Proceedings of the SIGCHI conference on Human factors in computing systems* (2007), 493–502.