University of California, Berkeley

Industrial Engineering & Operations Research

IEOR 242: Application in Data Analysis (Spring 2018)

Final Project: Prediction Kickstarter Project Success

Author:
Aishwarya Venketeswaran
Brenton Hsu
Sucheta Banerjee
Wiseley Wu

May 11th, 2018

1 Introduction

With social media, increase infrastructure, and other new technologies, entrepreneurs are able to pursue their ideas in numerous ways. In 2009, the Kickstarter platform was launched and allowed entrepreneurs across the world to easily pitch their ideas to millions of people to gain funding. In order to start a project to pitch an idea, entrepreneurs are required to follow the kickstart guidelines and only receive the crowdsourced funding if their funding goal is met. Otherwise, no money is charged to the consumers and the entrepreneurs will not receive a single dime - it's an all-or-nothing scheme. As a result, the knowledge of the attributes associated with success in a Kickstarter project, will allow entrepreneurs to gain additional insights and improve their chances of creating a successful project that receives funding. For the analysis of the success of Kickstarter projects, economic, temporal, and other attributes of the projects data is gathered to develop models to predict success.

2 Data Processing

The main data is obtained from https://webrobots.io/Kickstarter-datasets/, which contains monthly snapshot (in formats of CSV and JSON) of all projects available on Kickstarter. Additional resources were explored to collect data and merge them with this current dataset. Since Kickstarter projects are money driven, it is reasonable to guess that US unemployment rate would be a factor in determining the success of a project. The data for ***unemployment rate*** was collected from https://data.bls.gov/timeseries/LNS14000000 and formatted to a long format in R. Once that data was structured correctly, the unemployment rate and Kickstarter projects were mapped by doing an inner join between launch date month and the unemployment rate month. In addition to unemployment rate, ***daily stocks data for the S&P 500 Index*** was collected and normalized to be a daily % change to represent the US market economy. The S&P 500 was also inner join on the launch date to merge it to the main Kickstarter project dataset. Since *datetime* features were unusable in prediction model, numerical and binary temporal attributes for Kickstarter projects were derived from the original *datetime* data. For one, a ***holiday*** field was developed to indicate whether a project was launched on a holiday. Furthermore, ***project launch month*** was created to represent monthly seasonality. The ***number of days between the creation and launch*** of a project was also created to indicate how long entrepreneurs plan for their project before officially launching it publicly on Kickstarter. Similarly, the ***number of days between the launch and deadline*** was derived to indicate the amount of time a project has to secure funding.

In order to determine the popularity of Kickstarter as a service itself and whether it has any impact on the success or failure of the projects, the term *Kickstarter* was queried using google trends (https://trends.google.com/trends/explore) for the period of 2009 to 2018. The trends data provided number of times *Kickstarter* appeared as a search term per week. This data was added as a ***popularity*** feature to the final dataset.

Many of the features included in the original dataset are in JSON (Javascript Object Notation), a very common format seen in web API result and webscrapping. Therefore, JSON parsing was performed on those features using R library *jsonlite*. Categories information were first extracted since they were embedded inside the URLs. There were two levels of ***category*** info (ex: art/painting), but only the first level of category was used to identify a project in order to prevent creating overlying sparse category columns. Afterwards, ***location information*** was extracted from the location field that contained country, and state information. It was decided that only the state info would be extracted and stored in a new field due to the fact that the project scope was limited to projects in the United States.

The original dataset did not include many information regarding to the setup of the Kickstarter campaign itself, therefore a new webscrapping script was developed to further mine additional data from the Kickstarter website. Since the website itself has safeguard preventing user agents with similar IP address from accessing the site frequently in a short amount of time, the script did not employ any multithreading capability and instead rely on single thread process - which was extremely slow but less likely to be flagged by Kickstarter. Community page of each project was webscrapped to obtain a breakdown of current backers, which could be further divided into new backers (never backed a project before) and existing backers (backed at least one project before). This information was not used eventually as the scope of the project was focused on predicting a project success using information only available at launch. Rewards page of each project was webscrapped to obtain detailed information of the rewards tiers. Rewards is a key component of Kickstarter project, where entrepreneurs offer rewards to backers in exchange of funding. These rewards could ranged from a simple shout-out, the resulting product form the project, to a full-fledged international trips with the entrepreneurs themselves. Several new features were engineered with these information: ***median.tier.cost*** (median price of all reward tiers), pledge.tier.count (number of reward tiers available), ***tier.low/med/high*** (number of reward tiers below $100, between $100 to $1000, or above $1000), ***early.bird.frac*** (the fraction of reward tiers that offer "early bird", or tiers that offer great discount from original cost and were only available to early backers), ***limited.frac*** (fraction of reward tiers with limited availabilities), and ***ship.intl*** (whether any of the reward tiers offer international shipping). A last webscrapping was performed toward the end of this study on Kickstarter projects with the status "live": these indicated projects that were still ongoing at the time of collection, but were since completed as this study drew to a conclusion. Data were collected to simply determine whether the project met its goal (success), and the dataset containing live projects would be used in the end as an additional evaluation of model performance, other than the test data set, in the "real world" setting.

After performing analysis to find redundant data, some features were removed. The ***spotlight*** field represents projects that were featured on the spotlight page on Kickstarter, and it was removed since all successful projects were featured on the page. Additional, The ***staff***

picked field was also removed since this was not presented at the beginning of a project. Since the goal of this paper was to create a predictive model to classify a successful project before the campaign began, it made logical sense to remove features that were otherwise not available at the beginning of a project. [Table 3] listed all the features made it to the final data set. The final dataset were randomly divided into 80 % training, 10 % validation, and 10 % testing data.

3 Data Exploration

Initial data exploration on the Kickstarter dataset showed that 55 % of the projects were successfully launched while 39 % of the projects failed. Other states that were observed include "suspended", "cancelled" ,and "live" [Table 4]. As mentioned in the previous section, "live" data were filtered out separately as a "true test set" in additional to the test data partition. Since "suspended" and "cancelled" projects are also a form of failure, they were merged with other "failed" projects, and this became a binary classification problem of success versus failure.

It was also observed that the top three categories in which Kickstarter projects were created are music, art, and film [Figure 4]. This was also being reflected by the word cloud based on the term frequency of the stemmed words resulted from project blurbs [Figure 5]. Since incorporating stemmed words into a prediction model will result in significant increase of features dimension due to many one-hot-encoded column for each vocabulary, it's important to determine whether words featured in blurb would have any impact on the success of a project. This was achieved by splitting the dataset to "success" and "failure" group and created a word cloud for each group. The resulting word clouds shared many top frequency words and looked visually similar [Figure 6], thus it was concluded that a success of a project most likely does not depend on its blurb.

A brief time series analysis was also conducted by observing the number of projects launched per year, in addition to the number of successful and failed projects per year. It was evident that the number of projects peaked around 2014 and 2015 and had since been decreasing. Looking at the success vs failure rates per year, it's interesting to see that the success rate of projects has dropped substantially since 2013. The year with the highest success rate was in 2011 while 2015 had been the lowest. [Figure 7 and Figure 8]

4 Modeling

The goal of the paper was to make predictions on whether current/future Kickstarter projects would be successfully funded. An additional goal was to make additional inferences on factors that were known at the beginning of the project (such as project characteristics, state of economy, popularity of project types, and more) which might affect a project's probability

of success. Since this was a classification problem (or broadly speaking, a supervised analysis), the following methods were applied to the problem: Logistic Regression, LDA, CART, Random Forest, Boosting and Gradient Boosting. The different models were evaluated and compared using various metrics and the one with best performance in predicting a project's success was chosen.

Baseline Model

The Baseline Model serves as a benchmark for comparison with other models evaluated in this paper. In this case, the baseline model predicted all projects were successful (the most common outcome). Metrics for this model were: Accuracy was 0.564 which was a low threshold for other models to match. The model cannot be reliably used to make predictions as there was about 50 % chance of it being correct. TPR was 1 and FPR was 1 as all the projects were predicted to be successes. Overall, this set a minimum benchmark that could be compared with the subsequent models.

Logistic Regression and LDA

First, Logistic Regression was performed on the model to make predictions on the eventual success of projects. Because this was a classification problem, this was one of the highly interpretable models that could be used. Four different models of logistic regression were applied: Binomial Logistic Regression (Model 1) was a natural choice as this problem could be modeled as a binomial distribution with parameters n experiments with each one being success (probability p) and failure (probability 1-p). Then, the most insignificant feature *sp500_close_percent_change_launched_at* was removed and the same model was again re-run (Model 2). Note that while removing features, certain categories (like fashion) had low significance. They were not removed because there were other categories that were significant and it was not logical to remove all categories. Due to the slight dispersion of data, QuasiPoisson Logistic Regression was applied to the data as well (Model 3). Then, the most insignificant feature *category*was removed and the same model was again re-run (Model 4). In this case, almost all the categories were not significant. Hence, it made sense to remove it. However, as noticed through the metrics, applying QuasiPoisson did not yield good results in both cases. [Table 5] showed a summary metrics for all logistics regression model evaluated against the validation set, while [Figure 1] showed the ROC curves for the same model.

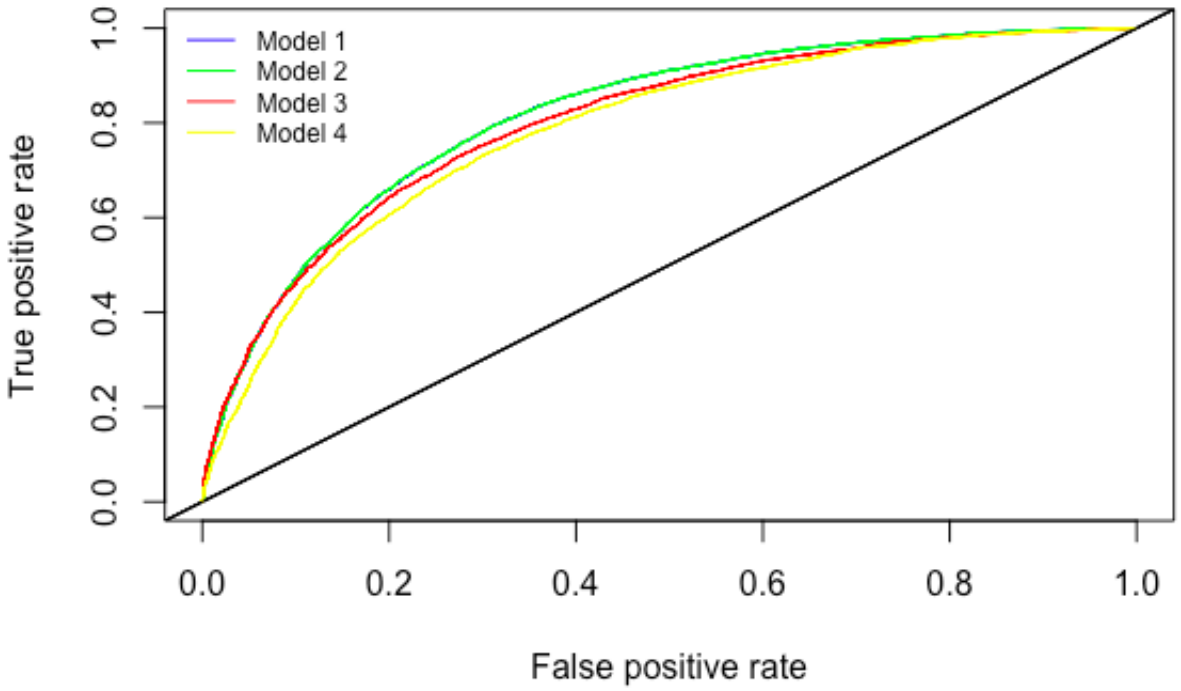


Figure 1: ROC curves for various logistic regression models

On the other hand, LDA was performed to make predictions because this was a problem following the Gaussian distribution. And, while applying LDA, it was assumed that the variables follow the Gaussian curve and that they had similar variances. By applying LDA, it was expected that a linear combination of features would be found that could be used to find two classes of 0s and 1s. The metrics that were observed were: Accuracy was 0.73 with AUC of 0.798 [Figure 9] - this model could make decent predictions and the performance was very similar to logistic regression. Nevertheless, logistic regression scored better in TPR and FPR with a higher AUC at 0.80.

CART and Random Forest

A Classification Tree is created in attempt to develop a model that has high prediction, while offering some inference capabilities. A 10 fold cross validation is used to fine tune the cp parameter to maximize the accuracy. The range of cp values that are tested are the values of .00001 to .0002 and are incremented by .00001. The model that offered the highest accuracy from cross validation is to have a cp of .0001, as seen in [Figure 10].

Naturally, a Random Forest model was then attempted to maximize accuracy. First, the number of trees (*ntree*) to build for each random forest model was set to 100. The minimum bucket was initialize to be 5. Lastly, the number of variables (*mtry*) to have for each tree was fine tuned between the values 1 and 20 and sequenced by 1. The best random forest model was chosen based on the m that provided the highest accuracy. As one can see in [Figure 11], $mtry = 5$ provided the highest accuracy, therefore, the random forest model was trained on the whole training dataset and was used as the final random forest model. The final out of sample performance of the random forest model was .7916 accuracy, .2810 false positive rate, and .8546 true positive rate.

Gradient Boosted Tree

In this paper, XGBoost, a variant of Gradient Boosted Trees was used to train one of the predictive models. It is highly memory efficient, which enables it to scale to large data set. It is also well known in many data contests and frequently used by top scorers in Kaggle competitions.

Initials sweeps were first performed to identify the optimal number of boosting iterations (number of boosted trees) with other default parameters, where 1000 was identified to be the number with the highest cross-validated accuracy [Figure 12 and Figure 13]. With the number of boosting iterations set, another parameter was setup to identify optimal values for learning rate (shrinkage / eta), maximum tree depth, subsampled fraction, and gamma. [Figure 14] showed that the optimal settings are 0.1, 3, and 1 respectively. The features importance shown in [Figure 15] was generated using the optimal model in the previous step, and it was apparent that features *launched_at_month*, *launched_at_holiday*, and *deadline_at_holiday* had almost no impact to the gradient boosted model. This suggested it did not matter what month the project was launched, and whether the project was launched or ended during a holiday.

With those features removed, a new parameter sweep was conducted to further fine-tune the learning rate, gamma, and fraction of subsampled columns. [Figure 16] showed the result of such fine-tuning, with optimal value 0.1, 0, and 0.4 respectively. It was also interesting to see that all these parameter sweeps could not push the accuracy beyond 80 %, a potential sign that this might be the limit for all the features that were used in the predictive model. [Figure 2] showed the features importance of the final optimized model. Several important features include goal (required fundraised amount), tier.low (number of reward tiers equal or under $100, tier.med (number of reward tiers above $100 and equal or under $1000), and *launched_to_deadline_days* (length of Kickstarter campaign). The most important feature, however, was the *unemployment_rate*.

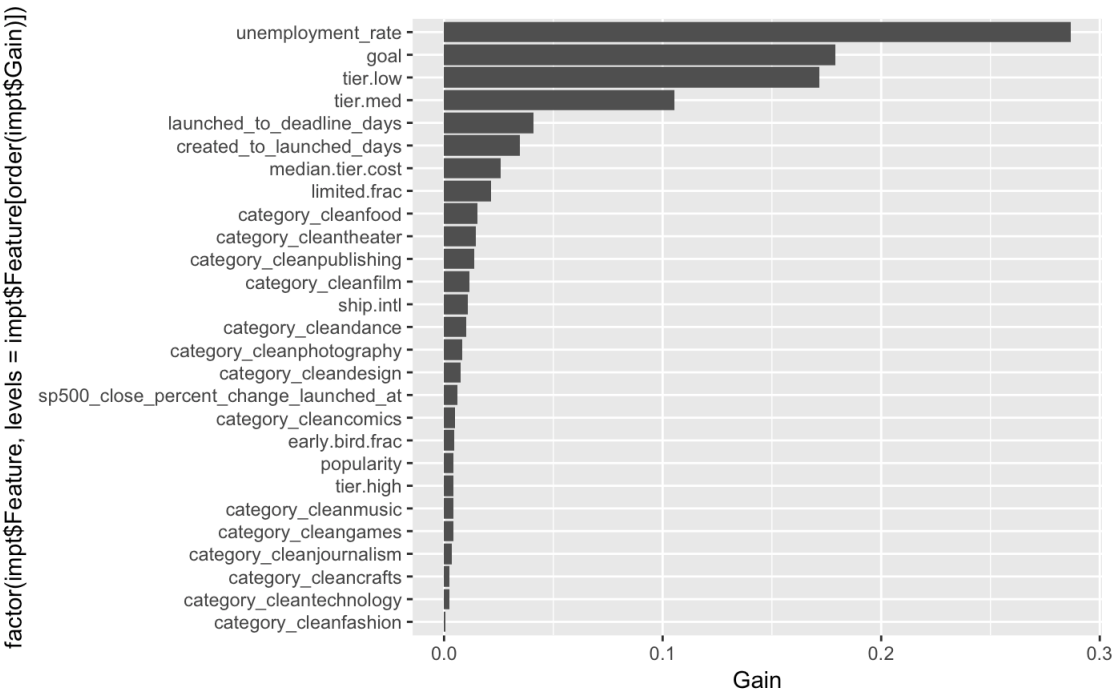
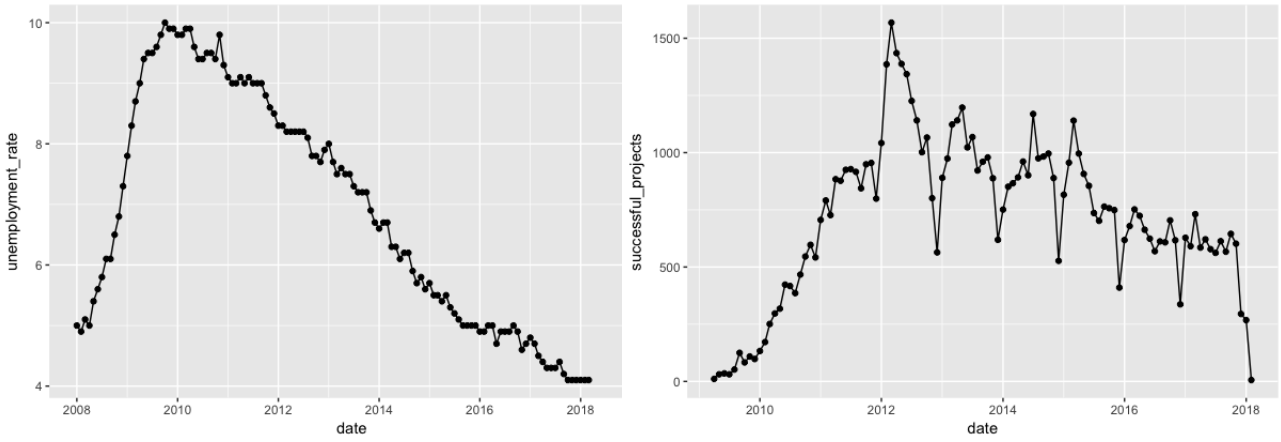


Figure 2: Features importance of the second gradient boosted model trained with insignificant features removed



(a) Unemployment rate over the years (b) Number of successful projects over the years

Figure 3: Comparison of the distribution between unemployment rate and number of successful projects showed some similarities

[Figure 3a] and [Figure 3b] showed the distribution of unemployment rate and successful Kickstarter projects binned by month. Both features had very similar distribution shapes, which would have explained the high importance value of unemployment rate. Nevertheless, this was unexpected as the feature was included under the assumption that a low unemploy-

ment rate would engender a rise of successful Kickstarter project as more consumers having jobs could afford spending money on things other than daily necessities. An alternative interpretation to this correlation would be that as unemployment rate rose, employees who had lost their jobs ended up becoming entrepreneurs to secure funding for their projects on Kickstarter.

Model	Accuracy	TPR	FPR	AUC
Baseline	56.4 %	100.0%	100.0 %	0.500
Logistic Regression	74.8 %	81.0 %	33.2 %	0.815
LDA	73.0 %	78.2 %	33.9 %	0.815
CART	76.5 %	83.3 %	33.0 %	0.827
Random Forest	79.2 %	85.5 %	29.9 %	0.783
Gradient Boosted Tree	79.9 %	85.9 %	27.8 %	0.879

Table 1: Metrics for various models evaluated using validation data set

Model Summary

A summary metrics of all the models trained could be found in [Table 1]. Since gradient boosted trees had the highest accuracy and AUC while offering acceptable inference via the features importance, it was selected as the final model to be tested against the test and live data. Even though logistics regression offered superior interpretation and inference, the accuracy difference did not justify its selection over gradient boosted trees. [Table 2] displayed the metrics of the XGBoost model on various fractions of data set. It was curious to see that almost no drop in accuracy was observed between the validation and test data set, a sign that the model was well generalized. More importantly, the model performed relatively well on the "real life" live data set, scoring 75.3 % in accuracy. This suggested the model could make an accurate prediction of a Kickstarter project status, 3 out of 4 times. Furthermore, only information available at the start of a project was used in order to predict its status in the end. This was a significant improvement over random guessing, which would only be correct 2 out of 4 times by chance.

Data Set	Gradient Boosted Tree Accuracy
Train Data Set	80.6 %
Validation Data Set	79.9 %
Test Data Set	79.4 %
Live Data Set	75.3 %

Table 2: Accuracy of the gradient boosted tree model evaluated using different fractions of data set

5 Impact

This paper has demonstrated the ability to train an effective prediction model of a Kickstarter project success using machine learning method and limited amount of information. Existing features were carefully chosen, and new features were engineered meticulously in order to reflect a project's potential to success as close as possible. Furthermore, a plethora of machine learning methods were evaluated and tested to identify one that worked best with the provided set of data. Linear method such as logistics regression and linear discriminant analysis were examined due to its simplicity and interpretability; their linear nature gave these models the ability to identify feature significances along with their signed correlation with the predictor. On the other hand, non-linear method such as CART, random forest, and gradient boosted trees were explored due to their abilities to model complex behavior; the ensemble nature (for the latter two methods) also compensated their weaknesses as model with high variance by either averaging model behavior (random forest) or learning on mistake generated by previous model (boosted trees). The result speak for themselves: these nonlinear models performed significantly better than the linear counterpart, obtaining high accuracy in exchange for interpretability. Unfortunately, the only interpretation available in these nonlinear methods were features importance, with no ability to interpret the directions of these importances. Nevertheless, the high accuracy of model trained by XGBoost gives the ability to predict Kickstarter project success with high fidelity. With this model, Kickstarter could predict a project's success when entrepreneurs submitted their campaign on their website with high accuracy. The website could provide additional mentoring, suggestion, and advertisement in a form of paid service to these projects that are in danger of failure, as Kickstarter could only earn a profit if a project get successfully kickstarted. Unfortunately, this model does not provide as much benefit to entrepreneurs due to the difficulty interpreting the model. Nevertheless, the model still provide several important takeaways: a project success does not strongly depend on daily S&P 500, its offering of early bird/limited reward and international shipping, nor does it depend on when a project starts or ends.

6 Appendix

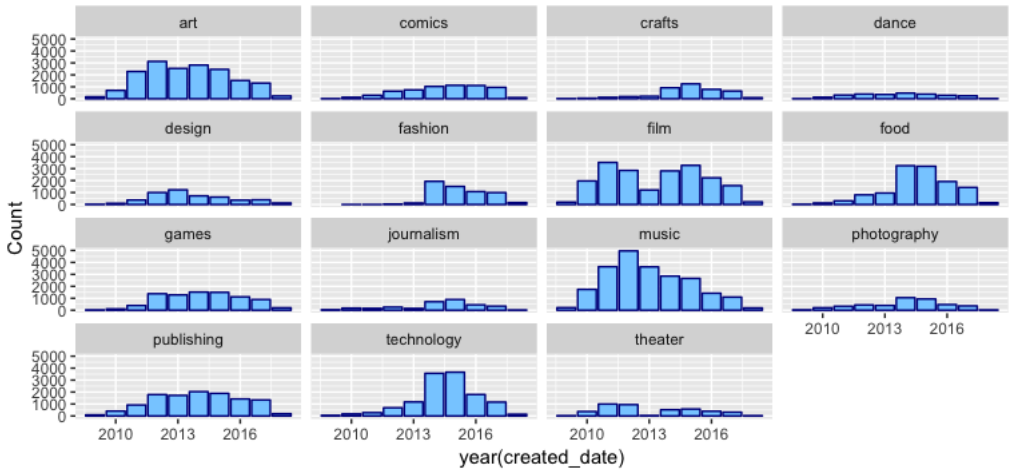


Figure 4: Number of projects launched by categories

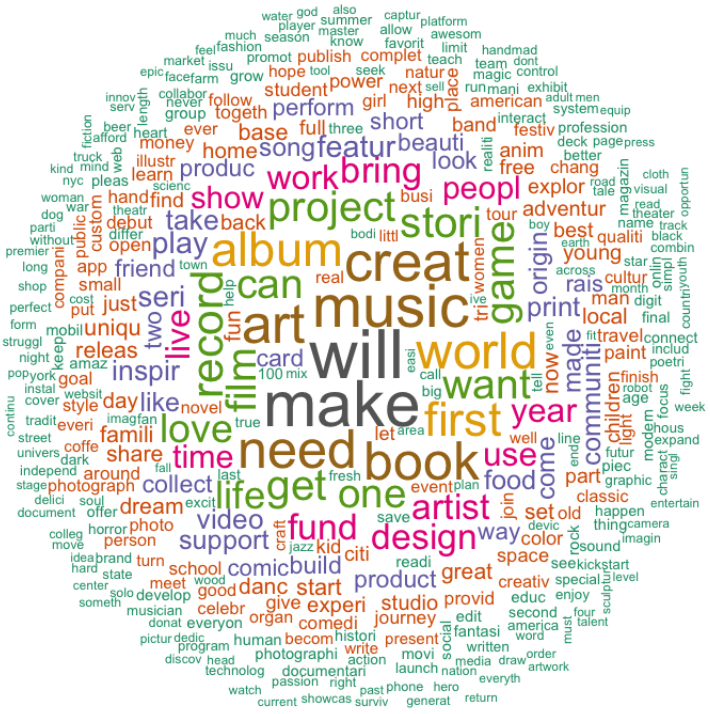


Figure 5: Word cloud generated from blurbs of all projects

Successful	Failed
	

Figure 6: Word cloud comparison between successful and failed projects

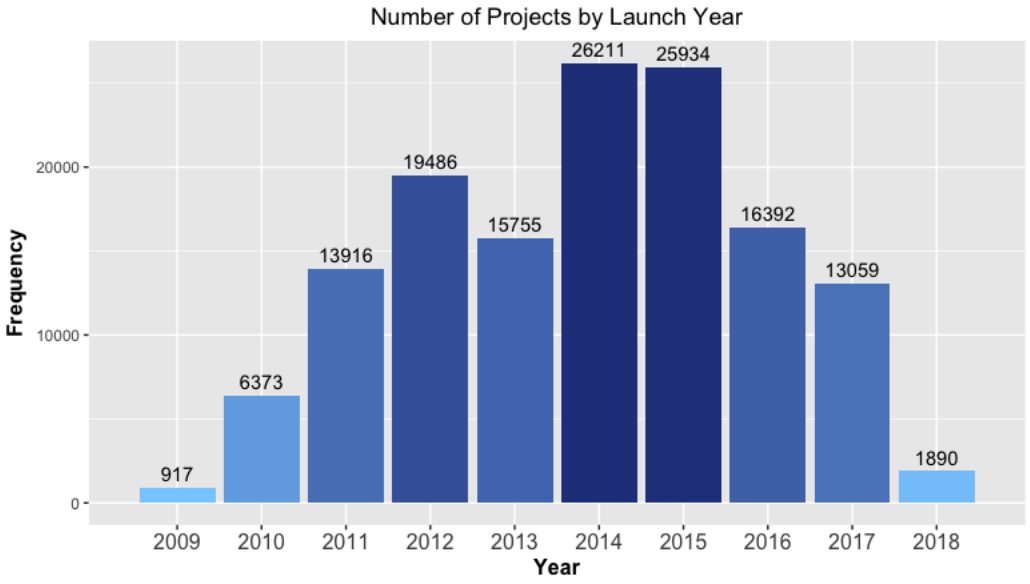


Figure 7: Number of projects launched by year

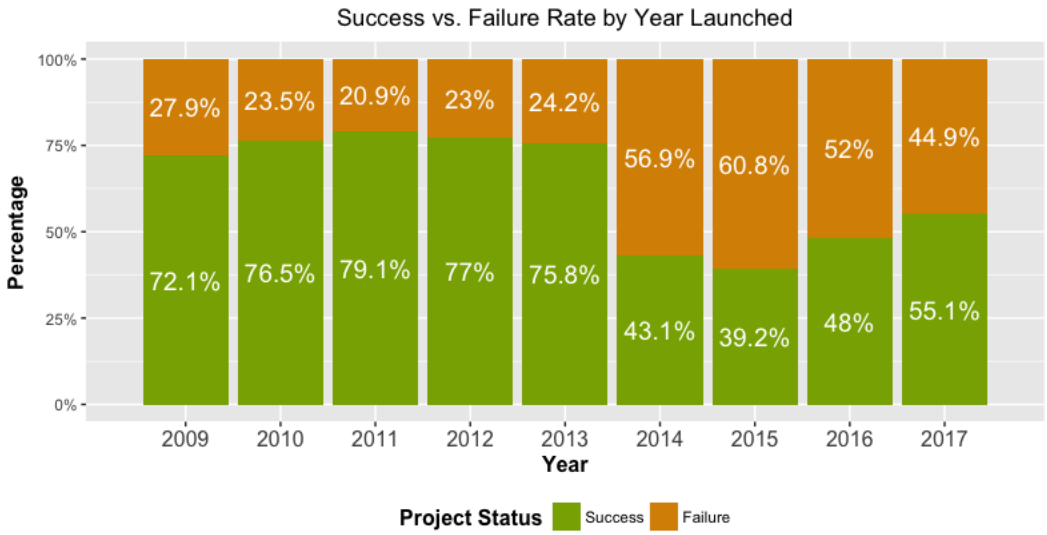


Figure 8: Number of success vs. failed project by year

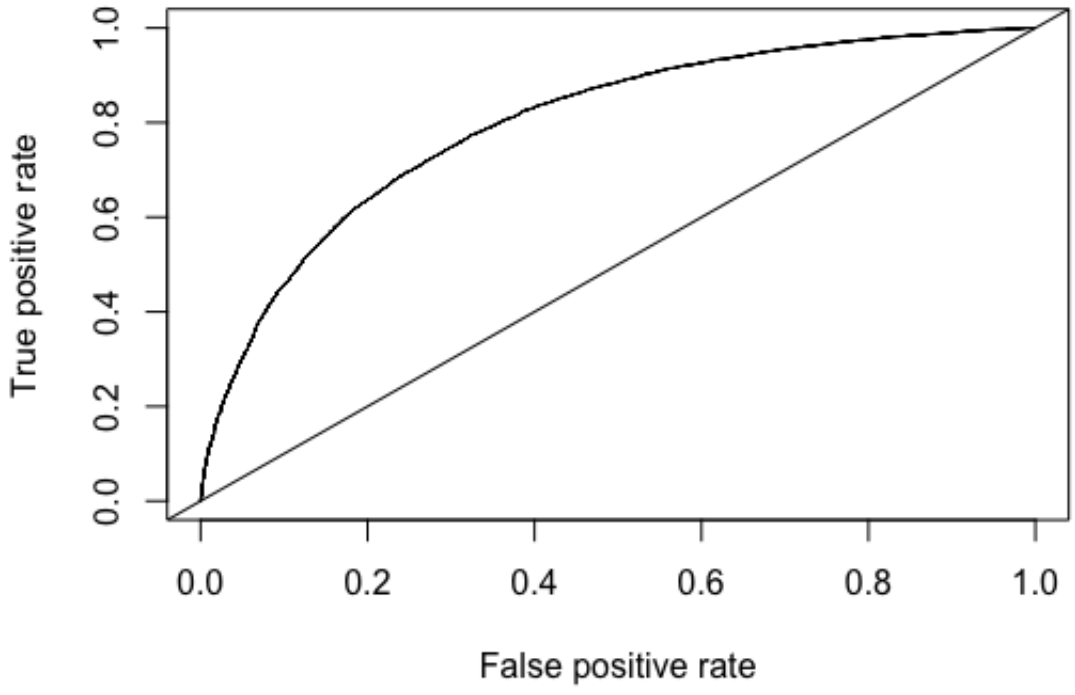


Figure 9: ROC curve for LDA model

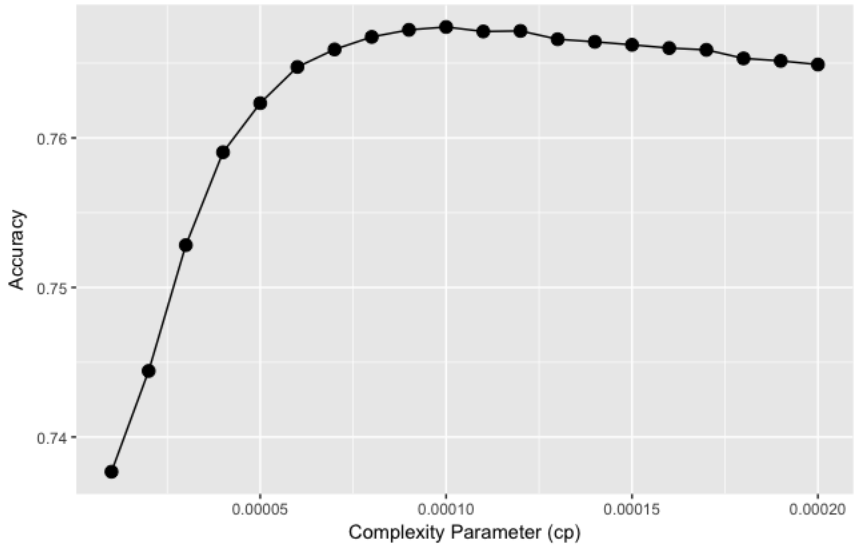


Figure 10: Accuracy vs CP from Cross Validation

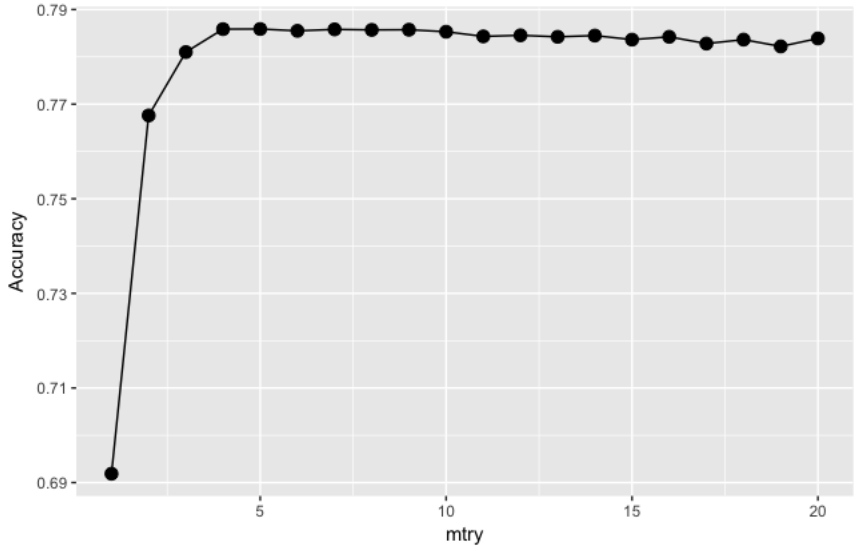


Figure 11: Accuracy vs mtry (number of variables for each model)

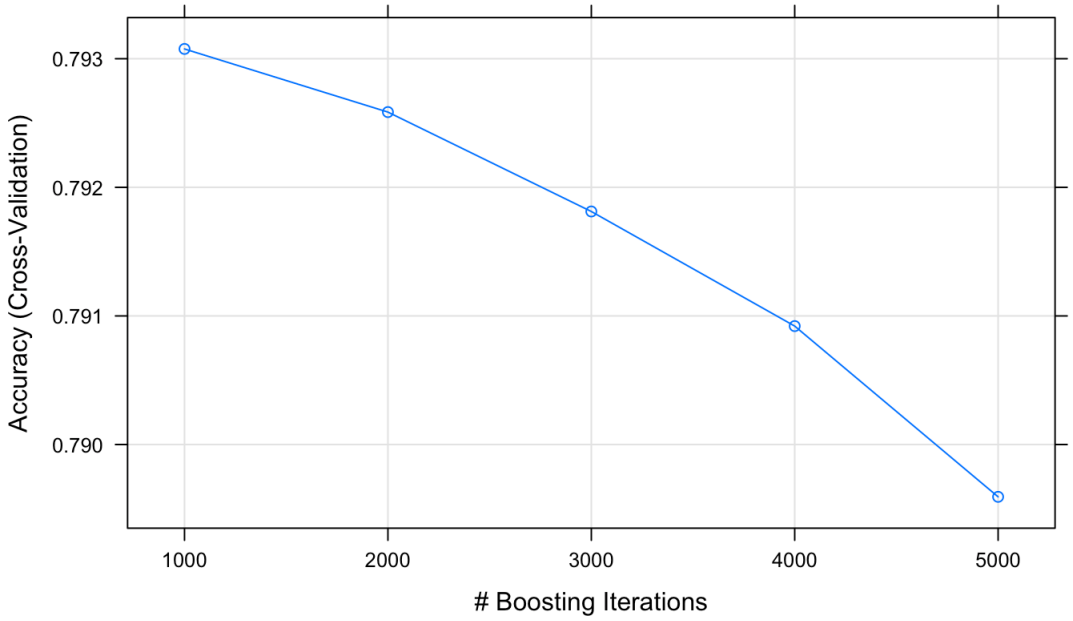


Figure 12: Cross-validation of accuracy versus number of boosting iterations (1000 - 5000)

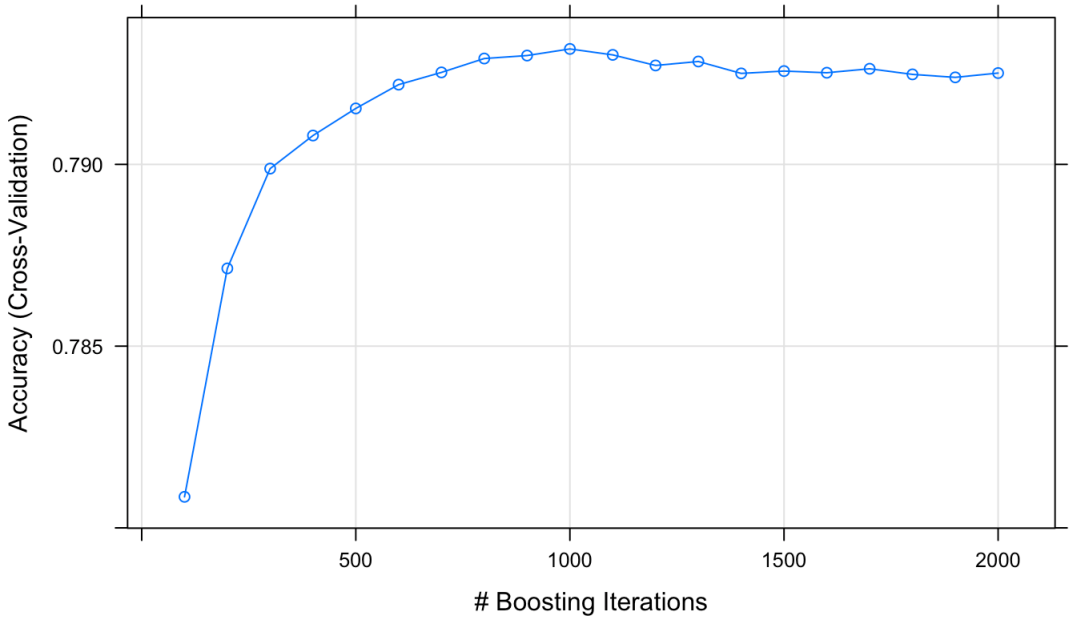


Figure 13: Cross-validation of accuracy versus number of boosting iterations (100 - 2000)

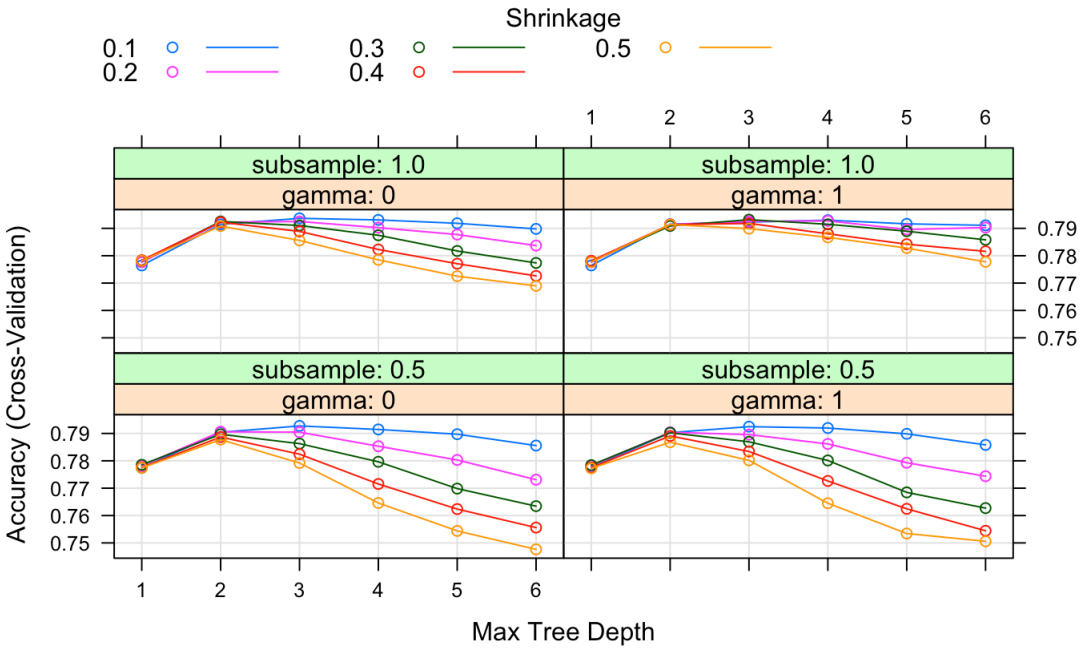


Figure 14: Grid search of optimal learning rate (shrinkage / eta), maximum tree depth, subsampled fraction, and gamma

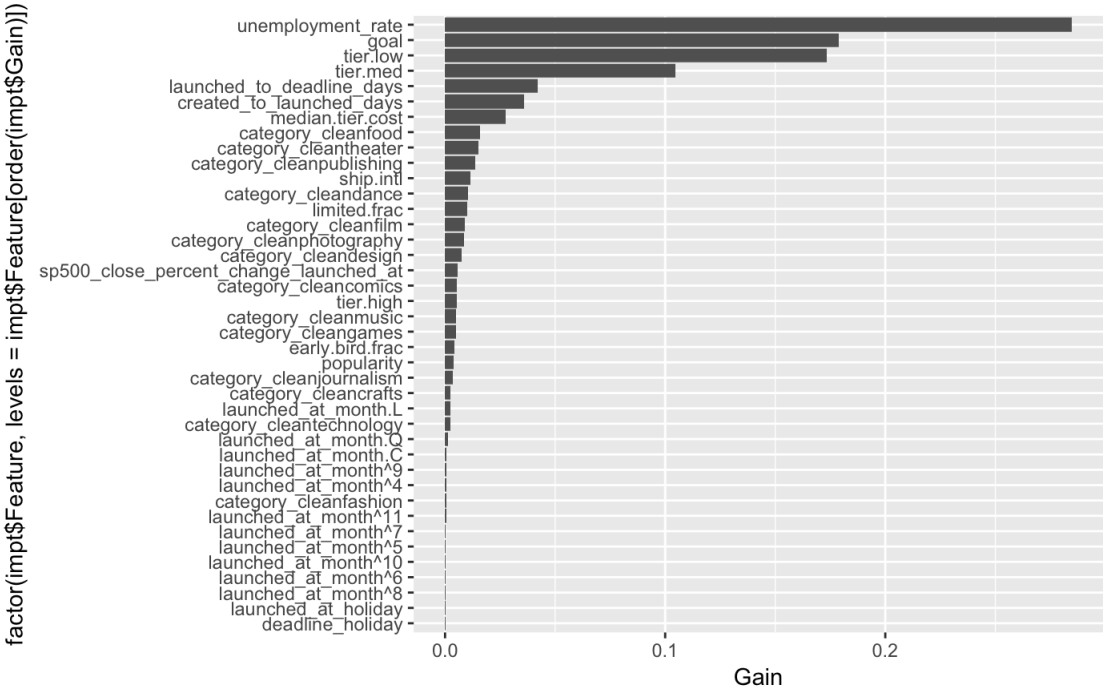


Figure 15: Features importance of the first gradient boosted model trained with all features

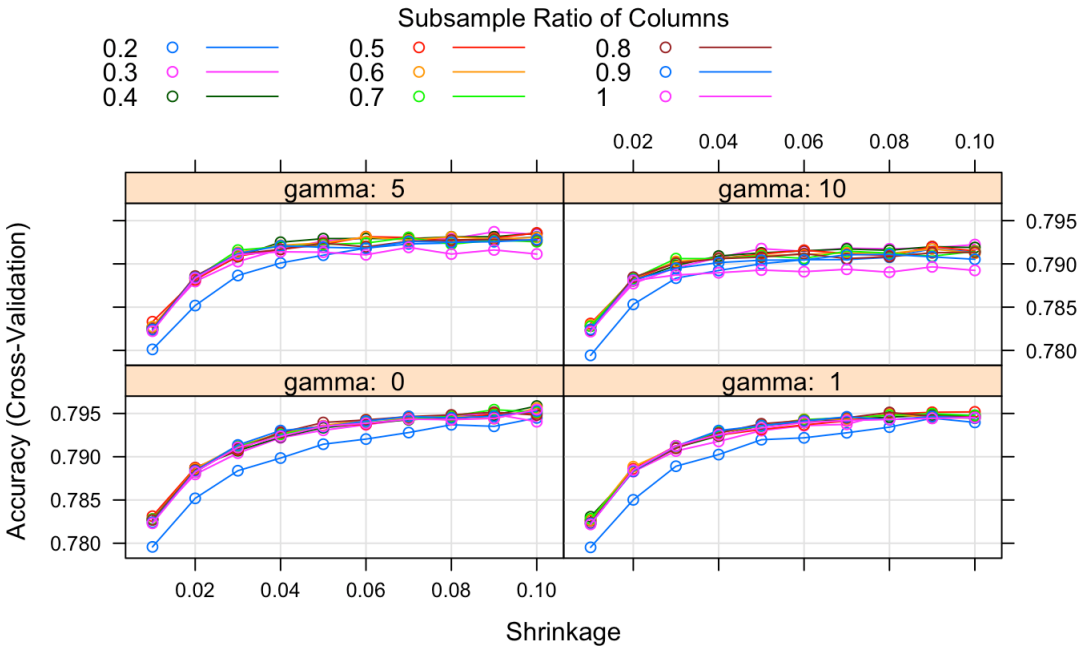


Figure 16: Grid search of optimal learning rate (shrinkage / eta), gamma, and fraction of subsampled columns

Column Name	Data Type	Description
goal	Double	Project fundraise target (in USD)
category_clean	Text	Project category
launched_at_month	Text	Month when the project was launched
created_to_launched_days	Double	Days between creation and launch of a project
launched_to_deadline_days	Double	Days between launch and deadline of a project
launched_at_holiday	Logical	Whether a project was launched on a public holiday
deadline_holiday	Logical	Whether a project was concluded on a public holiday
sp500_close_percent_change_launched_at	Double	Daily S&P 500 Index normalized to daily % change
unemployment_rate	Double	Monthly Unemployment rate when the project was launched
early.bird.frac	Double	Fraction of reward tiers that offer early bird option
limited.frac	Double	Fraction of reward tiers that offer limited quantities
ship.init	Logical	Whether a project offered international shipping for any of its reward tiers
popularity	Integer	Normalized query count of the term "Kickstarter" on Google at the month the project was launched
median.tier.cost	Double	median cost of all reward tiers in the project
tier.low	Integer	Number of reward tiers having a cost $\leq$ \$100
tier.med	Integer	Number of reward tiers having a cost $>$ \$100 and $\leq$ \$1000
tier.high	Integer	Number of reward tiers having a cost $>$ \$1000
state	Logical	Whether a project is successful (1) or not (0)

Table 3: Features selected for various model trainings

Project State	% of All Projects
Successful	55.0
Failed	39.0
Live	1.8
Suspended	0.2
Cancelled	4.4

Table 4: State distribution across all projects

Model	Name	TPR	FPR	Accuracy	AUC
1	Binomial Logistic Regression	0.810	0.332	0.748	0.815
2	Binomial Logistic Regression (insignificant featured removed)	0.809	0.332	0.748	0.815
3	QuasiPoisson Logistic Regression	0.749	0.296	0.729	0.801
4	QuasiPoisson Logistic Regression (insignificant featured removed)	0.746	0.319	0.718	0.784

Table 5: Metrics for various logistic regression models evaluated with validation set. Model 1 had the best performance