Jinlun Zhang 16jz61@queensu.ca

APPLICATION OF MACHINE LEARNING ON PREDICTING SUCCESSFUL KICKSTARTER CROWDFUNDING PROJECTS

ABSTRACT

The aim of this paper is to construct a machine learning model that can be moderately reliable to predict whether a potential project crowdfunding on Kickstarter will be successful or not before it is launched in Kickstarter, which can be considered as solving a binary classification problem. To solve the problem, 4 different model architectures are selected to perform this binary classification collectively by analyzing some basic features related to the potential project such as the project name and the goal of funding amount. A remarkable characteristic of one of the 4 chosen model architectures is that it can predict the outcome of a project crowdfunding simply based on analyzing the name of the project, which yields reasonably good prediction accuracy by capturing the certain relationships between the names of the projects and the success of the project crowdfunding. The proposed solution of this paper has shown that the moderately reliable model for predicting whether a potential project crowdfunding on Kickstarter will be successful or not before it is launched in Kickstarter can be soundly constructed, while a sufficiently accurate model for this problem is unlikely to be obtainable due to the nature of the problem such as the unlimited information and non-quantifiable factors affecting the outcome of a potential project crowdfunding on Kickstarter.

INTRODUCTION

Unknowing whether being capable of gathering sufficient fund for your own project has always been a prohibitively difficult challenge to bring innovative ideas into reality for young generations or people who are risk-averse or have very small amount of savings for their own pursuits or dreams. If you are a computer science student who is just graduated from University and is eager to create your own game or software, you may start your plan by yourself or with some of your friends who are on the same wavelength with you. However, in order to create a meaningful or at least attractive game or software, a certain amount of sunk investment must be made upfront on your side, because your team is unlikely to be able to address any requirement or specification you come up with. For example, a game might need a voice actor, and a software might need a good design of interface, which requires people with expertise on those specific fields of works to help you out. Thus, as a student who just graduated from University and do not have much savings, you may need to gather some funds from others to keep the plan rolling. However, the outcome of the fund raising is likely to determine if you can continue to work on your plan or not, because the limited financial investment on your game or software is likely to be translated into the unsatisfactory quality of the game or the software, meaning that your plan or impulsion is likely to end up as a failed undertaking as the game or the software failed to generate adequate amount of returns to compensate your financial investments and personal dedications. Therefore, before you decide to take the risks to develop your own game or software after graduation from University rather than to acquire a formal paid job, you may be urgently wondering whether you can gather sufficient funding for your ideas or plans, and where can you gather the sufficient funding from.

Jinlun Zhang 16jz61@queensu.ca

As the science and technology progress rapidly in the past 20 years, the approaches and options of obtaining funding have expanded dramatically, and one of the most emerging and popular funding options that is widely accepted by individuals and small business groups in the last 10 years is called the 'online crowdfunding'. The advantage of the 'online crowdfunding' is that the project initiators or the start-ups do not need to pay much upfront fees and spend much time and efforts to gain the required investments from the world-wise potential investors, which leads the 'online crowdfunding' to be a very suitable funding option for the individuals or small groups with little backgrounds or assets comparing to the other traditional funding options like bank loans.

To formally organize those 'online crowdfunding' campaigns, a website named 'Kickstarter' is founded in 2009 and is well-known for its reliable 'global crowdfunding platform' focused on creativity until now. Therefore, the question about where the sufficient funding can be gathered from is resolved as you can post your ideas or plans as a project on Kickstarter directly and start to raise fund. However, there is a policy in Kickstarter crowdfunding and that is a project is only funded when the backers' pledge meets its funding goal amount, otherwise, no money is given. Hence, to achieve the success of an online crowdfunding on Kickstarter and answer the urgent wondering about whether you can gather sufficient funding for your ideas or plans, a variety of factors is needed to be considered, which brings the machine learning technology into the picture as it can construct a model for predicting whether a project crowdfunding will be successful or not.

Therefore, the motivation of this paper is to construct a moderately reliable model to help those potential investees to assess what are the likelihoods of their project crowdfunding on Kickstarter being successful before their projects are displayed on Kickstarter. As a result, the innovators who are risk-averse or have very small amount of savings for their own pursuits or dreams can be encouraged to bring their innovations into reality by checking their probabilities of gaining sufficient development funding at the start.

On the other hand, the challenges of this paper include but not limited to that building a sufficiently accurate model to predict the outcome of a crowdfunding activity can be technically difficult to achieve. The reason for this is that the success of a crowdfunding activity depends on both of the quantifiable attributes such as the goal amount needed to raise to start the project and the non-quantifiable factors such as the local current trends of technology, arts, or music when the project is posted on Kickstarter. Though those quantifiable attributes about a project crowdfunding can be collected by crawler search and be utilized to train the model, those non-quantifiable features cannot be easily recognized and gained due to the unlimited amount of noise information existing in the real world, leading those non-quantifiable features to be excluded from the training feature space of the model, otherwise, the training feature space will be tremendous in scale (while we only have limited amount of computational resources to do the model training). Moreover, even if those non-quantifiable features are included in the training space, the true underlying patterns within those features cannot be easily learnt by the model as the quantization of those non-quantifiable features can be highly subjective and biased (which can result in overfitting and low generalizability). Thus, the evaluation requirement for the accuracy of the model should be accommodated, meaning that a model achieving an evaluation metric score within the range of 70% to 80% should be considered as a moderately reliable model (as a model with 90% accuracy rate or higher for this problem is technically impossible to be acquired due to the substantial number of non-quantifiable features and unforeseeable factors affecting the outcomes of the project crowdfunding on Kickstarter).

Another challenge of this paper is that the number of attributes (15) given in the dataset is relatively small comparing with the number of samples (378661) the dataset contains, while some of the attributes are considered as redundant under the assumptions of the problem and they are also overly correlated with the target attribute (the results of the project crowdfunding). For example, the number of backers is highly correlated with the success of a project crowdfunding, and is not supposed to be known before the posting of the project on Kickstarter. Therefore, those redundant attributes need to be dropped out, meaning that the training feature space will be further trimmed down, leading to a smaller number of valid features to train the model, which causes more difficulties to the learning of the model due to the dense distributions of the samples in the low-dimensionality space.

Contribution of This Paper:

- Encourage innovators who are risk-averse or have very limited amount of savings for their own pursuits or dreams to bring their innovations into reality by checking whether they can gain sufficient development funding at the start
- Constructed a moderately reliable model to quickly assess what is the likelihood of a project crowdfunding on Kickstarter being successful before the project is posted on Kickstarter.
- Used ensemble method to combine the predictions generated by multiple machine learning models to make the final/output prediction more robust and convincing
- A Bidirectional RNN model is implemented to predict the outcome of a project crowdfunding simply based on analyzing the name of the project, which yielded a testing accuracy around 65%. This strategy is introduced for the first time on Kaggle by me, and the model provides empirical evidence of the solution on this Kaggle task/dataset

PROBLEM STATEMENT

The problem this paper attempts to resolve is that if a trained machine learning model can be moderately reliable to predict whether a potential project crowdfunding on Kickstarter will be successful or not before it is actually launched or posted in the Kickstarter website.

PROPOSED METHOD (OVERVIEW)

To begin with, certain built-in Python methods/functions are applied on the given dataset to produce some statistics and visualizations of these statistics about the values under the features of the given dataset to gain ideas about the distributions of the feature values and possible future interpretations of certain training results. Then, several customized pre-processing steps and techniques are applied on the given dataset for the data cleaning purpose. Once a clean dataset is acquired, samples in the dataset will be randomly partitioned into 3 groups called the ‘training set’, the ‘validation set’, and the 'testing set' according to the proportions 50%, 20%, and 30% respectively. Afterwards, a pipeline for training classification models is established by connecting additional pre-processing steps with the configured models needed to fit on the training set. 4 model architectures are selected for fitting on the training set, they are SVM, Gradient Boosting, XGBoosting, and Bidirectional RNN respectively. After applying the pipeline to each of the 4 specified models, the 'random search CV' method is employed for the hyper-parameter tuning process of the corresponding model to optimize the model performance. Once the specified model finishes its training by showing no clear sign of underfitting, it will be tested on the validation set to check its generalizability. If a signal of overfitting is detected, then the search space of the hyper-parameters will be adjusted and the model will be trained again on the training set until both of the underfitting and the overfitting are avoided. At the end of the solution, an Ensemble strategy (bagging) is implemented to combine the predictions generated by the 4 models to improve the accuracy and the robustness of the final prediction or the output of the solution.

In summary, the general direction of the solution for solving this binary classification problem is to train models such as XGB classifier and Bidirectional RNN to produce moderately accurate predictions on the outcomes of the known project crowdfunding based on analyzing the projects' attributes such as the name, country, target amount of funding, and etc (and continuously improve the models' performances according to the feedbacks gained from testing the models on the validation set). After completing the trainings of the models, the Ensemble strategy will be implemented to combine the classification models together to improve the accuracy and the robustness of the final prediction (as a single classification model's performance may not be adequately satisfactory).

PROPOSED METHOD (DETAILS)

Initial Exploration & Visualization

The dataset used for solving the problem is a CSV file that contains 378661 unique projects hosted on Kickstarter between May 2009 and March 2018, where each project falls into one of the 15 main categories and one of the 159 subcategories specified by Kickstarter. We begin the solution by loading the CSV file into a Pandas dataframe and use some Python built-in functions/methods like '.info()' and '.value_counts()' to gain some initial ideas about the attribute values and their corresponding distributions. Then, through applying certain data visualization methods on the categorical attribute 'state', which is the intended target attribute, the Figure 1 is generated as shown on the left. Through observing the Figure 1, we notice that there are 6 possible outcomes for a project crowdfunding on Kickstarter, which are 'failed', 'successful', 'canceled', 'undefined', 'live', and 'suspended', while only around 2% of the total projects in Kickstarter are having state 'undefined', 'live', or 'suspended'. According to our objective, which is training models to predict whether a potential project crowdfunding on Kickstarter will be successful or not, the crowdfunding activities that end up as

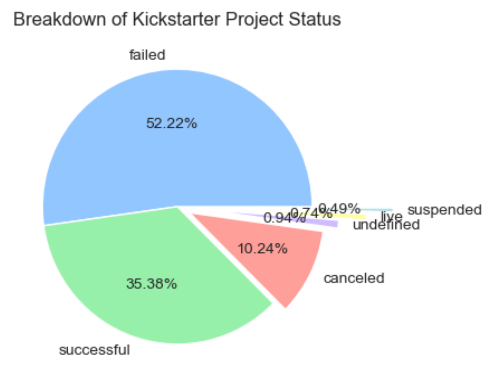

Figure 1: Distribution of the attribute 'state' values

being other than 'successful' or 'failed' should not be in the scope of the considerations (as 'canceled', 'undefined', 'live', and 'suspended' cannot be counted as 'failed crowdfunding' in a strict sense), meaning that the samples with 'state' attribute values being other than 'successful' or 'failed' will be dropped out. As a result, the problem can be transformed into a typical binary classification task, where the 'successful' and the 'failed' are the 2 class labels for the problem (target attribute). Furthermore, we notice that the percentage of the failed project crowdfunding on Kickstarter (52.22%) is moderately greater than that of the successful project crowdfunding (35.38%), meaning that the 'accuracy' might not be regarded as the optimal evaluation metric due to the moderately imbalanced instances of the 2 class labels. Thus, the chosen evaluation metric to the models' performances are both of the 'AUROC' and the 'F1 score'.

After visualizing the 'state' attribute values, we applied the data visualization methods on the categorical attribute 'main_category' and the 'category', generating 2 figures as shown below. Through comparing the Figure 2 with the Figure 3, we find that there are some overlaps between the attribute 'main_category' and the 'category' such as the value 'Music', 'Film & video', 'Food', etc. This may

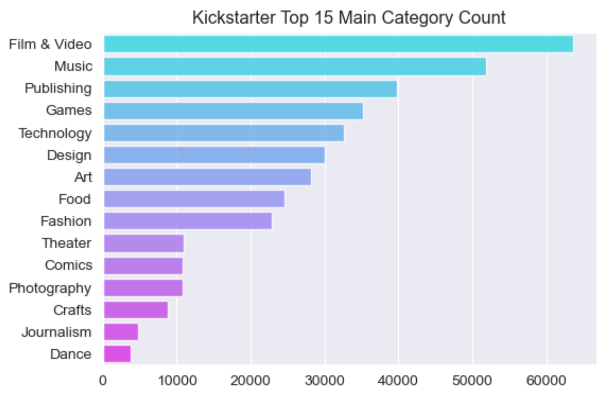

Figure 2: Rank the 'Main Category' attribute values by their corresponding instances

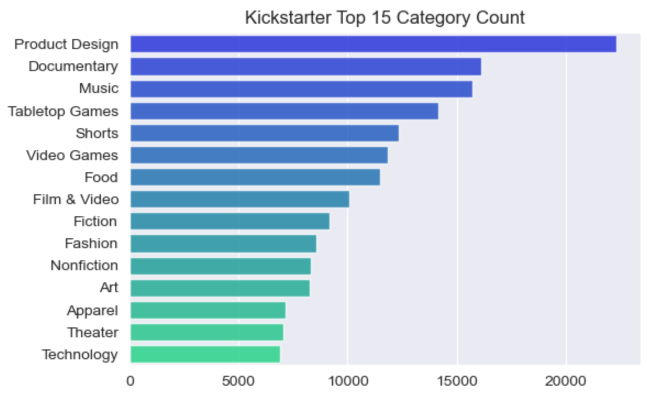

Figure 3: Rank the 'Category' attribute values by their corresponding instances

imply that the information provided by the 2 features for the model's learning can be quite similar, which might cause the classification of the samples to be more difficult than expected.

Afterwards, the data visualization methods are applied on the categorical attribute 'country', generating 1 figure as shown on the right. Through observing the Figure 4, we see that the projects from US dominate the Kickstarter crowdfunding website. Even if we aggregate all the projects from countries other than US, the number of projects from US is still larger than the number of projects from the aggregation. Moreover, the English-speaking countries like US, Great Britain, Canada, and Australia are forming the top 4 of the project origins in Kickstarter, which may imply that this feature may not be very helpful for training the model to predict if a potential crowdfunding post will be successful or not, because most of the samples are from the English-speaking countries that tend to share similar cultures and interests, meaning that this feature might not be able to provide abundant information for the model's learning.

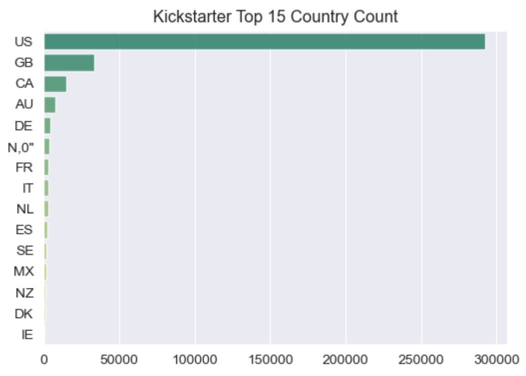

Figure 4: Rank the 'Country' attribute values by their corresponding instances

Also, we can use the visualization methods to generate the figure about the last categorical attribute, 'currency', in the given dataset as shown on the right. Through observing the Figure 5, we realize that the 'currency' attribute might be highly correlated with the 'country' attribute as we can observe nearly identical numeric patterns/trends within the 2 features. For example, the projects requiring US dollar as funding currency still dominate the Kickstarter crowdfunding website. Even if we aggregate all the projects requiring funding currencies other than US dollar, the number of projects requiring US dollar as funding currency is still larger than the number of projects from the aggregation. Moreover, the currencies circulated in English speaking countries like US dollar (USD) in US, Great Britain Pound (GBP) in UK, Canadian Dollar (CAD) in Canada, and Australia Dollar (AUD) in Australia are in the top 5 of the projects required funding currency in Kickstarter. The only difference between this observed pattern with the pattern observed in 'country' feature is that the currency circulated in European Union (EUR) is ranked the third in the top 5 of the projects required funding currency. A possible explanation for this observation is that for European countries like Germany (DE), France (FR), and Italy (IT), they ranked as the fifth, seventh, and eighth respectively in the top 15 of the project origins in Kickstarter (as seen from Figure 4), and those European countries all consider EUR as the sole currency in their countries to be used, so the number of projects requiring EUR as funding currency should be the summation of the projects with origin in European countries such as Germany (DE), France(FR), and Netherlands(NL). Thus, it is reasonable to see that EUR is ranked the third in the top 5 of the projects required funding currency as Euro can be used in multiple countries, while currencies like USD and GBP can only be used in their 'home countries'.

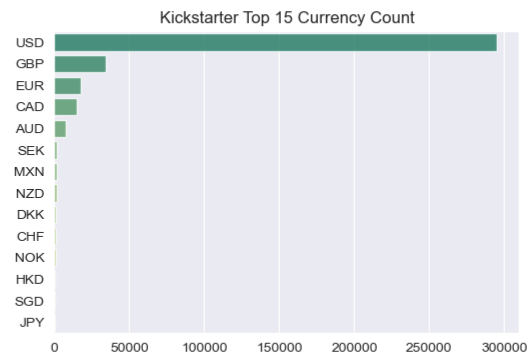

Figure 5: Rank the 'Currency' attribute values by their corresponding instances

Pre-processing/Preparation of the Dataset

After the visualization and the initial exploration of the given dataset, we move to the pre-processing of the dataset. Firstly, we drop out the samples with 'state' attribute values being other than 'successful' or 'failed' from the dataframe, then we notice that the 'deadline' time feature and the 'launched' time feature can be combined together to produce the 'number of active days' of a project crowdfunding on Kickstarter as a new feature. Hence, we check the number of samples with missing values under the feature 'deadline' and the feature 'launched' before we create the new numeric feature named 'days_active' by subtracting the 'launched' date from the 'deadline' date. The reason why we check the number of the missing values instead of directly dropping out the samples with missing values is that if there is a large number of samples having missing values under the feature 'launched' or 'deadline', it may not be a wise approach to simply drop out the samples with missing values as we can lose considerable amount of information and data. Fortunately, it turns out that there is no missing value under the feature 'deadline' or the feature 'launched', so we can directly combine the two time features without worrying about how to handle the samples with missing values. Once the feature 'days_active' is gained, we can visualize the correlations between the feature 'days_active' and the other numeric features such as 'backers', 'usd_pledged_real', and 'usd_goal_real', generating a figure as shown on the right. Through observing the Figure 6, we notice that the only one strong correlation (0.75) within the 4 selected numeric features lies between the feature 'backers' and the feature 'usd_pledged_real', which is intuitive as the core idea of the crowdfunding is to gather as many individuals as possible and encourage them to invest as much as they can, meaning that for the successful crowdfunding projects, there should be a positive linear relationship between the number of backers and the amount of money pledged, because a person must invest or contribute some money to the crowdfunding project to be counted as a backer, and usually the contribution or investment from a single investor is not considerable comparing to the total fund raised, so the number of backers must be large when the amount of money pledged is also large (provided that the crowdfunding is successful). Unfortunately, the number of days a project lasted on the Kickstarter website tends to be uncorrelated with either of the number of backers or the amount of money pledged, while the correlation between the number of active days and the goal amount of funding is also weak, which might imply that the 'days_active' feature may not be able to provide much useful information for the model to predict if a project crowdfunding will be successful or not as the number of backers and the amount of money pledged should be highly correlated with the outcome of a project crowdfunding (while the 'days_active' is uncorrelated with them), leading the classification of the samples to be more difficult. It is noteworthy that the feature 'backer' and the feature 'usd_pledged_real' cannot be used for the training of the model, because the objective is to predict whether a potential project crowdfunding on Kickstarter will be successful or not before it is actually launched or posted in the Kickstarter website, meaning that the number of backers and the amount of money pledged should be unknown when the project is not even posted yet. Additionally, if the feature 'usd_pledged_real' is not removed, the model performance will be perfect, because the model can simply compare the 'usd_goal_real' feature with the 'usd_pledged_real' feature and know if the project is funded successfully or not according to the policy in Kickstarter.

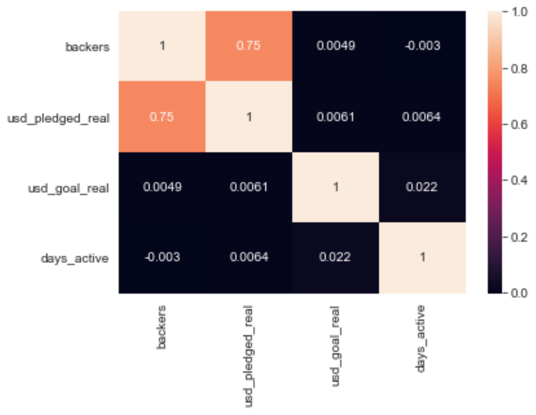

Figure 6: Correlations between 4 numeric features

Furthermore, from the initial exploration of the dataset, we notice that the minimal value for the 'usd_goal_real' feature can be as low as '1.0e-02', which is equal to 0.01$ and looks like a prank or mischief. Also, since we consider the crowdfunding of a project to be successful as long as its pledged amount is greater than its goal amount, some successful samples might be inherently biased as their goal amounts are set to be unreasonably low and so can be reached easily with just a few backers. Hence, some data analysts on Kaggle may just choose to set a threshold value (ex: 1000) on the 'usd_goal_real' feature to filter out those 'disturbing outliers'. However, from my perspective, my think that for some meaningful projects that have low cost in nature, it is possible that the project initiator cannot gather a small amount of funding to proceed on his/her project. For example, for people with low level of income, if they want to invent or sell some cheap but meaningful or useful widgets, it is reasonable for them to set their goal amount to be low, and the number of backers should be relatively large to prove that their projects are intrinsically attractive or useful to others. Therefore, instead of simply filtering out the projects that are not ‘earnest’ by setting a threshold on the goal amount of funding, we can perform 2 filtering operations. The first one is to filter out the ‘real’ prank/mischief projects by setting a threshold value 100$ on the goal amount of funding as it is not convincing that anything useful, meaningful, or original can be created with less than 100$. Then, the second operation is to filter out the projects that have both low goal amount (less than 1000$) and the number of backers (less than 10) to ensure that potentially meaningful projects are not filtered out by simply setting a threshold on the goal amount of funding. (There are in fact around 19000 projects that have a low goal amount (less than 1000$) but tend to be meaningful as there are more than 10 backers supporting each of the projects)

In addition, through reading the specifications about the features in the dataset, we realize that some of the features might be redundant and may not be used during the model training process. For example, the ‘pledged’ and the ‘goal’ data columns include the pledged and the goal amounts in the project required currency, while the ‘usd_pledged_real’ and the ‘usd_goal_real’ data columns includes the USD conversion of the pledged and the goal amounts made by the fixer.io api, which means that these corresponding features are highly correlated and the ‘usd_pledged_real’ and the ‘usd_goal_real’ should be preferred to ‘pledged’ and ‘goal’ as the numeric patterns and trends within the ‘usd_pledged_real’ and the ‘usd_goal_real’ can be better captured due to the uniform measurement unit. Moreover, the time feature ‘deadline’ and ‘launched’ are also redundant as we have combined them into a numeric feature ‘days_active’ already. Hence, we can drop out all the redundant features from the dataframe.

After converting the 2 categorical values under the target attribute ‘state’ into numeric class labels (1 for ‘successful’, 0 for ‘failed’), we move into the last stage of the pre-processing. It is noteworthy that the categorical features ‘category’, ‘main_category’, ‘country’, and ‘currency’ are considered to be great fit to the OneHotEncoder pre-processor (as these 4 features fit the definition of ‘category’ precisely such as no built-in hierarchies within the categorical values), and so their pre-processing is left to the following pipeline training structure. Also, the pre-processing of the numeric features is left to the pipeline training structure as well. Thus, the only one left feature without pre-processing is ‘name’, and we decide to do 2 things with the feature ‘name’. One is that we will train a Bidirectional RNN model to predict if the crowdfunding of a project will be successful or not simply based on its ‘raw’ project name (no data cleaning for the attribute ‘name’), and the other is that we will replace the ‘name’ feature by generating relevant features from the ‘name’ feature to train the following models in the pipeline structure (ex: the length of the names, if certain punctuation is used in names, etc). In order to proceed with the 2 operations, we firstly need to filter out the samples whose ‘name’ attribute values are unknown or missing as we cannot train the bidirectional RNN on ‘invalid’ text data. Then, we generate the first new feature ‘name_len’ by finding out the string length of each project name, and by using ‘.describe()’, we see that the minimal value of ‘name_len’ is 1, which means that some of the project names are as simple as only one character, which might be a prank or mistake. Afterwards, we generate 4 additional Boolean features in their integer representations (1 for ‘True’, 0 for ‘False’). They are ‘name_contain_exclam’ (checking if a project name contains an exclamation mark), 'name_contain_question' (checking if a project name contains a question mark), 'name_is_tile' (checking if all the words in a project name start with a upper case letter), and 'name_is_upper' (checking if all the words in a project name are in upper case). The reason of producing the 4 Boolean features is that we believe certain ways of formulating the representations of the project names can catch more attentions regardless of the meanings of the names, which can subtly motivate more potential backers to look into the details of those projects with eye-catching names, and contribute some money if they are really interested in the projects. Thus, in other words, the distinctive representations of the project names may lead to broader public exposure/attention of the projects at a certain degree, which may be correlated with the success of the project crowdfunding. Finally, we generate the last new feature 'name_word' by finding out the number of words each project name has, and by using '.describe()', we see that only 25% of the projects names are formed with 5 words to 8 words, which may imply that this feature can provide useful information for the model's learning as the concise project names formulated by an appropriate number of words should convey a sign of adequate professional ability to the potential investors, leading the projects whose wordings of the names are concise and comprehensive to have a higher chance of being clicked (broader public exposure).

Model Training

After the pre-processing of the given dataset, we randomly partition the samples in the dataset into 3 disjoint groups called the 'training set', the 'validation set', and the 'testing set' according to the proportions 50%, 20%, and 30% respectively. Then, we can build the pipeline for the trainings of the classification models by selecting the needed features from the dataset and connecting them to the configured corresponding transformers ('IterativeImputer' is used to fill in the missing values in the numeric features, while 'SimpleImputer' is used to fill in the missing values in the categorical features, and the 'OneHotEncoder' is applied on the categorical features after the fillings of the categorical features). Before we establish the hyper-parameter search space for the first chosen model, we need to drop out the target attribute from the training, validation, and testing sets. Afterwards, we can connect the built pipeline to the chosen model architectures and perform the model trainings on the training set only with using the 'RandomizedSearchCV' to search for the possibly optimal hyper-parameter settings for the models.

The first model architecture we selected is SVM. Since the number of features in the original dataset is not large (though we have generated/added additional features from 'name'), we may need to use the Kernel functions to increase the dimensionality of the feature space and make the dataset sparser, so that the decision boundaries with low biases can be acquired. Thus, SVM might be a wise choice to try first as it implements Kernel functions conveniently and co-operates with them effectively. After training the SVM on the training set with using the built pipeline and the 'RandomizedSearchCV', we evaluate the SVM model on the validation set. If the evaluation metric score gained from testing the model on the validation set is not satisfactory, we will adjust the search space of the hyper-parameters of the model and train the model again on the training set. This hyper-parameter tuning loop terminates when no sign of improvement on the model performance is likely to be achieved, then we will apply the

SVM model on the testing set to acquire the predications of the SVM model on the class labels of the testing samples for the later implementation of the Ensemble strategy.

The second and the third model architectures we selected are Gradient Boosting and XGBoosting. The incentives of choosing these 2 models are that the 2 boosting models empirically perform well for the ordinary binary classification tasks, while these 2 model architectures are also widely chosen by the data analysts on Kaggle as they outperform the other types of models such as Random Forest or Logistic Regression. Hence, these 2 boosting models are selected to be combined with the SVM model to improve the accuracy and the robustness of the final output/prediction through the implementation of the Ensemble method at the end of the solution. Thus, after finishing the hyper-parameter tunings of the 2 boosting models, we will apply the 2 models on the testing set to acquire the predications of the 2 models on the class labels of the testing samples for the later implementation of the Ensemble strategy.

In addition, the reason why choosing/training both of the Gradient Boosting and the XGBoosting that belong to the same model family (boosting algorithm) is that the XGBoosting is a specific implementation of the Gradient Boosting method which uses more accurate approximations to find the best tree model and a more regularized model formalization to control the overfitting. Hence, the XGBoosting generally can yield a better model testing performance than the Gradient Boosting, meaning that if the two models end up giving predictions of similar but unsatisfactory accuracy after several times of tunings, it may indicate that the patterns within the data cannot be captured by the models easily due to either of the large amount of noises in the data or the insufficient number of features provided that causes the separation of the two groups of samples to be computationally difficult. Therefore, by comparing the performances of the 2 models following the same principle of gradient boosting, we can not only determine whether the two models have already been finely tuned up or not (reaching the limitation of the model if the prediction accuracies are similar), but also we can conclude that the gained unsatisfactory prediction accuracy is likely to be resulted from the 'deficient' training dataset. Hence, if we want to further improve the prediction accuracy, we can either switch to another different model family such as Neural Network to falsify our conclusion or to collect more data to acquire larger feature space to make the dataset sparser and the classification easier.

The last model architecture we selected is Bidirectional RNN. This decision is inspired by a data analyst on Kaggle who attempted to use the natural language processing (BoW model) on the names of the projects to predict if the project crowdfunding will be successful or not. Since the natural language processing is applicable to perform the prediction, then a Bidirectional RNN model should be considered as a better fit than BoW model to perform such prediction. Furthermore, from the logical perspective, all the backers should be attracted by the names of the projects initially, and once they are hooked by the names of the projects, they will have the motivation to look more into the postings of the projects and invest some money if they are really interested in. Thus, there should be certain relationships between the names of the projects and the success of the project crowdfunding, and using the Bidirectional RNN to capture such relationships should be novel and outstanding.

The training process of the Bidirectional RNN model is different with the trainings of the previous 3 models. A tokenizer must be trained on the values under the 'name' attribute of the training set at the start to generate the vocabulary dictionary (embedding matrix). Afterwards, all the ‘name’ attribute values in the training, validation, and testing sets are pre-processed through applying the trained tokenizer on the ‘name’ attribute values and using the ‘pad_sequences()’ method to transform the ‘name’ attribute values into text sequences with uniform length (transform the unstructured data into structured data). It is noteworthy that the uniform text sequence length is set to be 35, which is approximately equal to the median value of the lengths of the project names according to the distribution of the ‘name_len’ attribute values. Then, after finishing the configuration of the Bidirectional RNN structure (3 bidirectional LSTM layers with accumulated memory vector size being 100), we can train the Bidirectional RNN model on the training ‘name’ attribute values, and perform the hyper-parameter tuning loop as before. Once the loop is terminated, we apply the Bidirectional RNN model on the testing set to acquire the predications of the model on the class labels of the testing samples for the following implementation of the Ensemble strategy.

At the end of the solution, we have acquired 4 lists of predictions on the class labels of the testing samples generated by SVM, Gradient Boosting, XGBoosting, and Bidirectional RNN, and we will use a weighted average function as the aggregation method to combine these 4 lists of predictions into 1 list as the final predictions on the class labels of the testing samples. The weights assigned to the predictions generated by SVM, Gradient Boosting, XGBoosting, and Bidirectional RNN are 20%, 30%, 30%, and 20% respectively according to their corresponding evaluation metrics scores gained as shown in the Result section of this paper (the weights also indicate the models’ relevant importance to the final predictions on the class labels). The combined prediction is used as the final/output prediction on the outcome of an input project crowdfunding activity.

EXPERIMENTAL RESULTS

Table 1: Summary of the evaluation metrics scores, presented in unit % with two decimal places, gained by the 4 trained models evaluated on the testing dataset. The final 2 rows of the table represent the evaluation metrics scores gained by the ensemble/aggregation strategy applied on the predictions generated by the 4 trained models

	SVM	**Gradient Boosting**	**XGBoosting**	**Bidirectional RNN**
Testing AUROC	61.70%	68.65%	68.90%	62.29%
Testing F1_Score	61.54%	70.95%	71.04%	64.77%
Final/Output Testing AUROC	74.55%			
Final/Output Testing F1_score	71.32%			

As we can observe from Table 1, both of the SVM model and the Bidirectional RNN model have achieved testing evaluation metrics (F1 and AUROC) scores around 62%, which means that the Bidirectional RNN model that only utilizes the 'name' attribute values for training can yield predictions that have similar accuracies with the ones generated by the SVM model that is trained on all the other non-target attribute values such as the project origin country, and the goals of funding amount. This observation confirms our previous assumption/reasoning that there should be certain relationships between the names of the projects and the success of the project crowdfunding as the Bidirectional RNN prediction accuracy rejects the null-hypothesis (AUROC score is much larger than the 50% of random guessing), while the observation also indicates that SVM may not be a suitable model architecture for this classification problem as we expected (as it cannot even outperform the model that only analyzes the 'name' attribute values).

As for the 2 boosting models, we can see that both of the XGBoosting model and the Gradient Boosting model have achieved testing evaluation metrics scores around 70%, showing that the 2 boosting models outperform the SVM model and the Bidirectional RNN model. Moreover, though 70% may not be as satisfactory as 90% in value, these 2 boosting models should be considered as moderately reliable due to the nature of the problem as we discussed in the Introduction part of this paper (unlimited non-quantifiable factors that cannot be easily recognized and quantified without subjective supposition). For example, if a project named 'face mask improvement' was introduced in Kickstarter when the 'Covid-19' outbroke, its crowdfunding outcome should be predicted as 'successful' by the model due to the high demand for quality masks at the time. However, the real-time information is not included in the model's training space, and it cannot be included as well, because there is unlimited information in the real world (while we only have limited computational resources) and translating these information into features that can be processed by computers without any subjective biases tends to be unrealistic. Thus, as the unlimited additional information outside the scope of the model should affect the outcomes of crowdfunding activities substantially, a very reliable model for this problem should be technically impossible to be acquired.

Finally, we can see that both of the AUROC score and the F1 score have improved slightly due to the implementation of the Ensemble method, leading the final accuracy of the solution to be around 71% or 72%.

CONCLUSION

Constructing a machine learning model that can be moderately reliable to predict whether a potential project crowdfunding on Kickstarter will be successful or not before it is launched in Kickstarter is computationally and practically achievable by analyzing some basic features related to the project such as the project name and the goal of funding amount. However, a sufficiently accurate model for this purpose is unlikely to be obtainable due to the nature of the problem, the limited computational resources we have, and the unlimited information and non-quantifiable factors affecting the outcome of the potential project crowdfunding on Kickstarter.

Jinlun Zhang 16jz61@queensu.ca

REFERENCE

Crockett, J. (2021, January 22). Kickstarter success prediction [WIP]. Retrieved April 15, 2021, from https://www.kaggle.com/jennifercrockett/kickstarter-success-prediction-wip#Natural-language-processing

Elsinghorst, S. (2018, November 29). Machine learning basics - gradient boosting & xgboost. Retrieved April 16, 2021, from https://www.shirin-glander.de/2018/11/ml_basics_gbm/

Icyblade. (2017, February 12). GBM vs Xgboost? Key differences? Retrieved April 17, 2021, from https://datascience.stackexchange.com/questions/16904/gbm-vs-xgboost-key-differences

Kosovan, O. (2019, September 24). Kickstarter: LGBMClassifier [0.681]. Retrieved April 14, 2021, from https://www.kaggle.com/kosovanolexandr/kickstarter-lgbmclassifier-0-681#LGBMClassifier

Leonardo, F. (2019, September 30). Kickstarter Projects EDA + Stat-tests & Pipeline. Retrieved April 16, 2021, from https://www.kaggle.com/kabure/kickstarter-projects-eda-stat-tests-pipeline#Dealing-with-date-features

Nilan. (2018, September 08). EDA of Kickstarter projects. Retrieved April 15, 2021, from https://www.kaggle.com/nilanml/eda-of-kickstarter-projects

Samuel, D. (2018, November 12). Kickstarter Success Classifier [0.685]. Retrieved April 15, 2021, from https://www.kaggle.com/majickdave/kickstarter-success-classifier-0-685#Try-Linear-Regression-model-to-predict-Log-USD-Pledged

The potential dataset can be viewed and retrieved at: https://www.kaggle.com/kemical/kickstarter-projects