Predicting The Success of Crowdfunding

Yan Chen, Yiwen Guo, Chenchen Pan

Department of Statistics, ICME, MS&E, Stanford University

Abstract

Crowdfunding platform like Kickstarter, where entrepreneurs and artists seek for support from a large number of contributors, has become prominent over the past decade. Better understanding and more accurate prediction of the success of a project can help both contributors and creators make better use of their resources. Previous works applied traditional machine learning methods, such as SVM and Random Forest, with only categorical and numerical features, such as goal and duration, or only textual features, such as project description and keywords. To our knowledge, we are the first work to apply deep neural networks with all three types of features. Using a dataset of 100K+ crowdfunding projects, our model achieves 72.78% accuracy on test set, which is significantly better than the baselines and the previous works. We also show that the trained model can help us understand the success and failure of crowdfunding projects better.

About The Dataset

- Use a dataset from Kaggle [1], which has 108,129 examples in total. 34561 of them are successful, and the rest are failed.
- Illustrate with a positive example:
  - desc: I like drawing pictures. and then i color them too. so i thought i would suggest something for me to draw and then if someone wants...
  - goal: 20
  - keywords: drawing-for-dollars
  - disable_communication: FALSE
  - country: US
  - currency: USD
  - deadline: 1241333999
  - launched_at: 1240602723
  - final_status: 1
- Select input features, which are description, goal, keywords, disable_communication, country, currency, and the time difference between when the project is launched and the deadline

Data Preprocessing

- Turn country, currency, and disable_communication columns into categorical variables having values from 0 to the number of class minus one.
- Add a new column called duration, which is computed by subtracting the date when the project is launched from the deadline date.
- Split into 90% training set, 5% dev set, and 5% test set

Baseline Models

- Random forest classifier with 100 trees
- Shallow neural network
  - One-hot vector word embedding with 10K vocabularies
  - Adam optimization
  - L2 regularization with the penalizing parameter $\lambda = 0.01$
  - binary cross-entropy cost

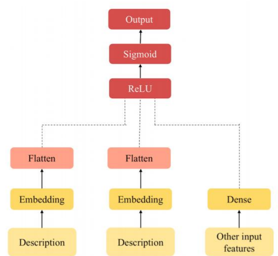

Figure 1: Shallow neural network model

RNN Model - Hyperparameter Tuning

Hyperparameters	Range
Dropout rate	0-0.5
Number of LSTM layers	2-7
# of hidden units in one LSTM layer	64,128,256
Learning rate	0.0001-0.01

Table 1: Setting of hyperparameters tuning

RNN Model

- Pre-trained word embedding with 300-dimensional GloVe vectors
- Loss function:
$$J = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(\hat{y}^{(i)}) + (1-y^{(i)})\log(1-\hat{y}^{(i)})]$$
- Adam optimization
- Regularization: early stopping + dropout
- Our best hyperparameters are as follows:
  - LSTM state size is 64.
  - Dropout rate is 0.485.
  - FC layer output sizes is 64.
  - Learning rate is 0.000145
  - Epoch is 50.

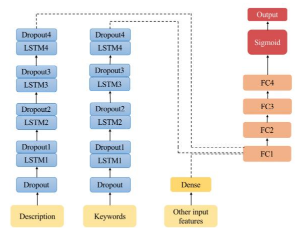

Figure 2: RNN model

Result

Our result is summarized in the following table. We can see that the RNN model gives the best result.

Models	Dev Accuracy
Random Forest Classification	70.00
Shallow Neural Network	69.35
RNN Model	**72.42**

Table 2: Experiments Results

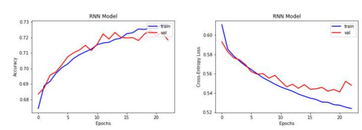

Figure 3: Accuracy and cost for RNN model

Discussion

- About 26% of errors are due to dataset issues, including label error (6%) and incomplete information (20%, there are some missing content with quotation marks in project's description)
- 29% of the errors are due to the fact that the model sometimes only uses partial information for predictions ('game', 'documentary' and 'film' for success and 'app', 'food' and 'mobile' for failure).
- The rest 45% of errors are model errors. This is a quite challenging task for human as well because a human can only get around 50% correct (no better than random).

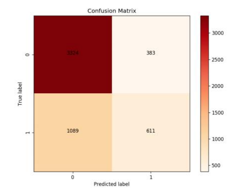

Figure 4: Confusion matrix

Future Work

- Address the mismatched data issue. Include more features of the projects (images, videos).
- Try bi-directional LSTM. Apply batch normalization. Adjust the loss function to give different weights to the losses regarding different misclassifications.

Reference

[1] https://www.kaggle.com/iamsajanbhagat/kickstarter/data
[2] Kevin Chen, Brock Jones, Isaac Kim, and Brooklyn-Schlamp.Kickpredict: Predicting kickstarter success, Technical report, California Institute of Technology, 2013
[3] Sawhney, Kartik, Caelin Tran, and Ramon Tuason. 'Using Language to Predict Kickstarter Success.'