ISYE 6740 – Spring 2023
Project Final Report

Team Member Names: Kexin Cai (903141699), Cheng Wilson (903651530)

Project Title: Kickstarter: Predicting Successful Projects

Problem statement

Kickstarter, a popular crowdfunding platform, was founded in 2009 by Perry Chen, Yancey Strickler, and Charles Adler. The platform originated from Chen's idea to create a show during the 2002 Jazz Event but did not have enough money to do so. To date, Kickstarter has funded over 238,000 projects, raising 7.2 billion US dollars for creators across the globe. Kickstarter's success has made it a major player in the crowdfunding world, providing a platform for artists, entrepreneurs, and other creators to promote and highlight their projects and secure funding from a global community of backers.

While Kickstarter has many success stories, not all projects will succeed thus not all investments will have a return. Therefore, this project aims to analyze past Kickstarter campaigns to identify common trends and factors that contribute to project success. By exploring the characteristics of successful projects, we hope to gain insights that will help investors make decisions and improve the chances of future project success.

Data source

We used the Kickstarter Projects data set compiled by Mickaël Mouillé in 2018 and shared on Kaggle (source). Specifically, we worked with the ks-projects-201801.csv data set, which contains over 378,000 project entries and 15 columns of project information. This data set covers a wide range of project characteristics, including project name, category, currency, deadline, funding goals, project launching date, country, pledge amounts, number of backers, and most importantly, project success status. Notably, the author also provided pledged amount converted by both Kickstarter and Fixer.io API, and the Fixer.io API conversion was established based on the currency exchange rate at the time of the project. This information is relevant for projects initiated outside of the US. Because of the data set's size, we may need to randomly down-sample the data to achieve better performance and reduce the effort needed to clean the data set and train the models.

Methodology

Data preprocessing

An initial data exploration was conducted to gain a better understanding of the data set, including identifying the category of successful projects and examining the distribution of the funding raised among all projects. Next, we cleaned the data set to eliminate any missing values and ensure the models are not influenced by outliers in the data. In addition, we investigated redundancies in the data, where some columns, e.g., ‘goal’ and ‘usd_real_goal’, convey the same information. Removing these redundancies helps us reduce the complexity of the data, leading to more efficient and accurate models.

Some columns also did not provide much information on their own, but when combined, could provide useful information for the models to train on. For instance, the ‘launched’ and ‘deadline’ columns did not provide any useful information on their own as they were arbitrary to a project’s success. However, when combined to determine the targeted length of the project (‘project_length_days’), we were able to get much more useful information that was derived from those two columns.

Feature selection

Next, we analyzed feature correlations to determine the independence of features, resulting in a reduced number of features. For instance, looking at the initial feature correlation matrix (Figure 1), we can observe that there are many highly correlated features. For example, as mentioned above, ‘goal’ and ‘usd_real_goal’ are highly correlated and we can choose to keep one of them to simplify the data. After cleaning up some redundancies, we can see how the updated correlation matrix in Figure 2. This shows that `backers` and `usd_pledged_real` are still highly correlated, and we can choose to drop one or both (and even combine them in a way). In this case, we drop both because we are interested in determining what makes a successful Kickstarter project. It is obvious that if a project gets a lot of backers and funding, it will likely be successful, so we want to analyze the project independent of these 2 features. The final correlation matrix is shown in Figure 3.

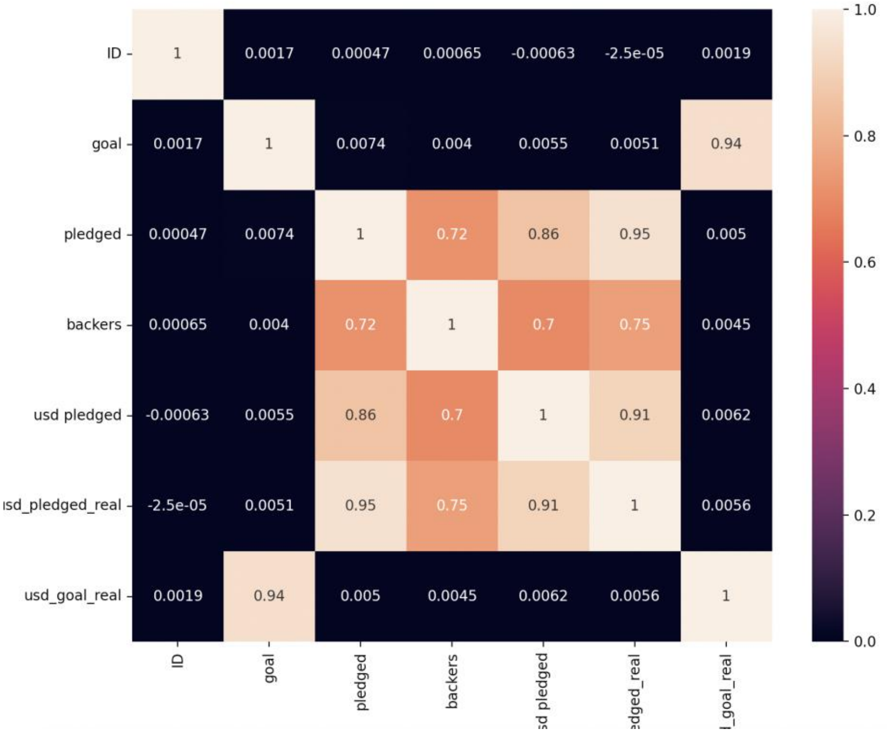

Figure 1: Correlation matrix before pre-processing

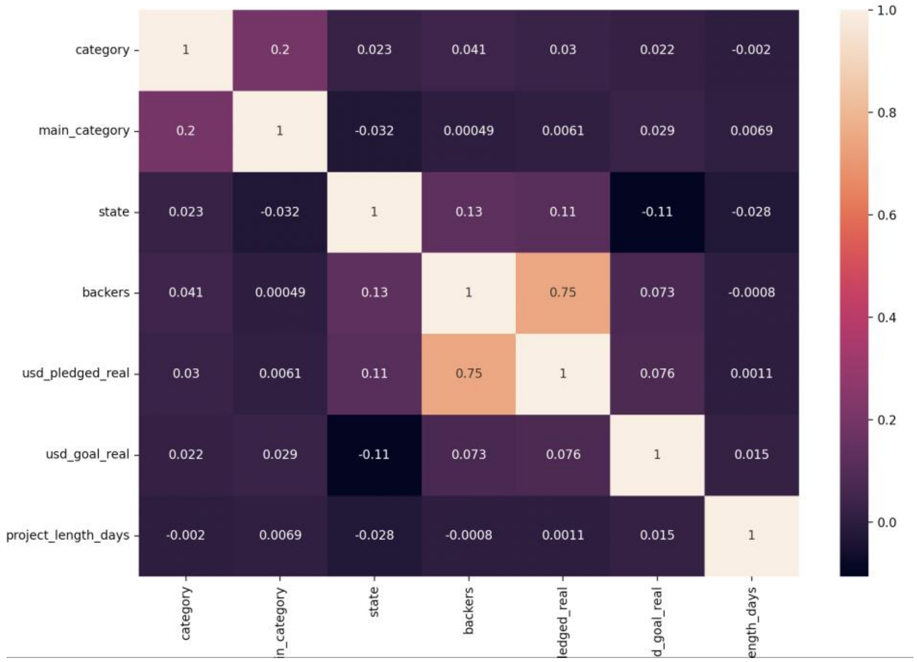

Figure 2: Correlation matrix after some initial pre-processing

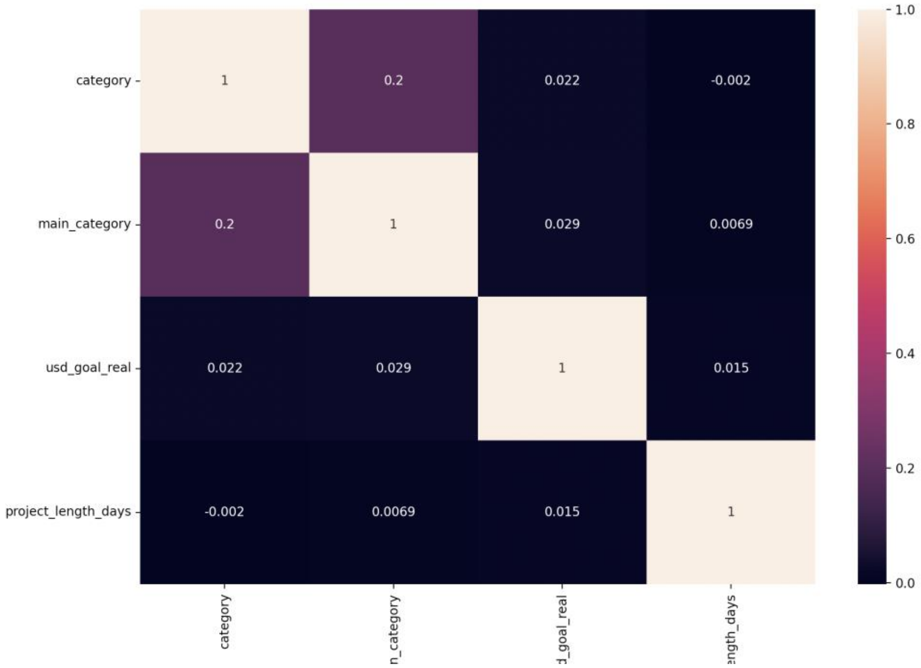

Figure 3: Final correlation matrix

Then, we performed feature selection via mutual information to identify the most significant project characteristics that contribute to a successful Kickstarter campaign. Figure 4 shows that ‘backers’ and ‘usd_pledged_real’ were the strongest indicators of a successful project. As discussed previously, we decided to analyze without those features, and we can see the results of removing those 2 features in figure 5. It appears that ‘category’ and ‘usd_goal_real’ are the strongest indicators of a successful project.

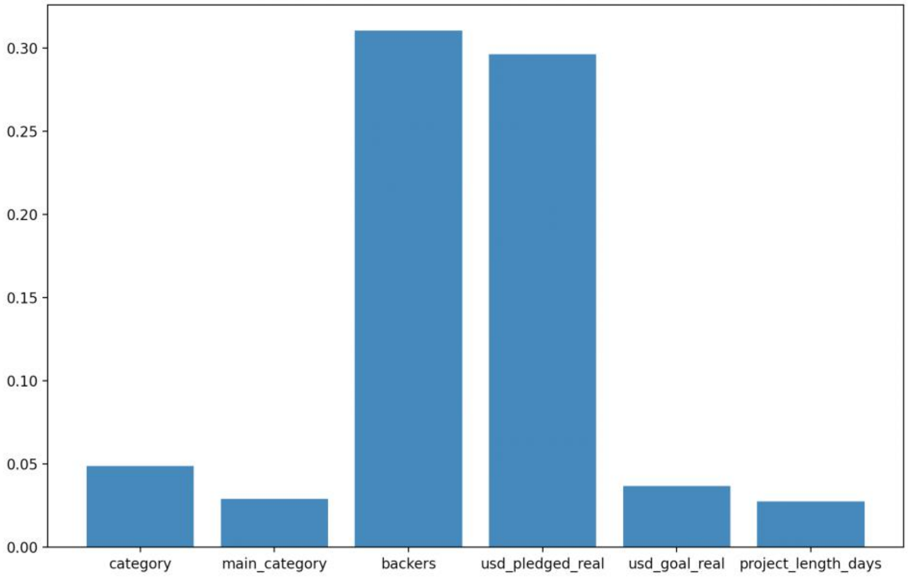

Figure 4: Feature selection scores after initial data pre-processing

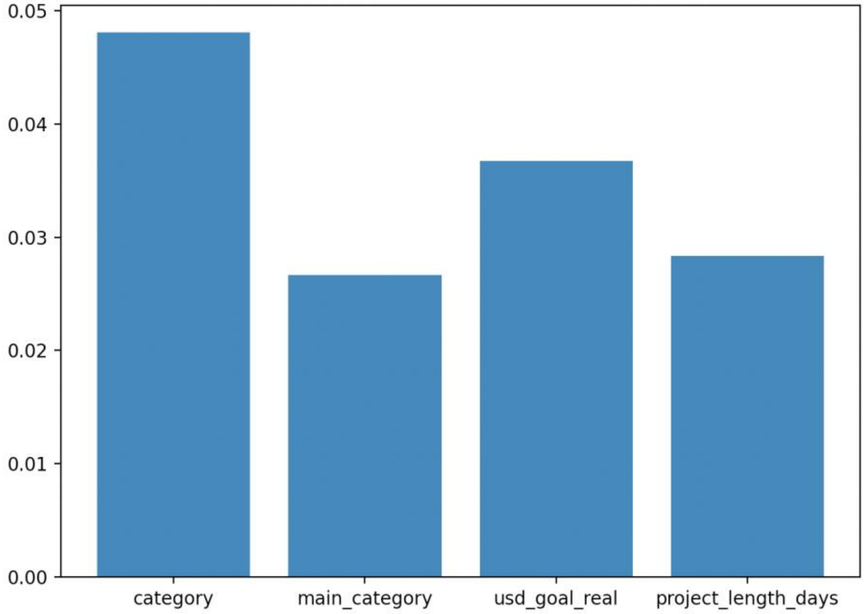

Figure 5: Feature selection scores after final data pre-processing

Sampling

Due to the size of the data set, model training time was a concern. Thus, we downsized the data set into 5000 entries through random sampling.

Modeling

We used multiple packages from scikit learn for various aspects of modeling in this project. The modeling is comprised of two parts:

- Use classification methods to find out the best model for determining which class (success or failure) a project will fall in;
- Use regression method to predict the amount of funding a project will potentially collect given its features.

Before model training, the data set is split into 75% training and 25% testing sets.

Classification

For classification purposes, we chose Naïve Bayes and Logistic Regression because they are simple, easy, and fast to implement, and do not require much parameter tuning. We also adopted Random Forest because of its potential to reduce overfitting through combining multiple decision trees and its capability in handling a mix of categorical and continuous data. Neural Network requires some tuning of parameters but can deal with complex nonlinear relationships, which makes it a potential candidate for our project.

We trained the models with default parameters for Naïve Bayes Classifier and Logistic Regression and used grid search or random search to find the best parameters for Random Forest and Neural Network. The approach to determining the parameters is discussed in the Evaluation section.

Prediction

Linear Regression is used to predict the funding raised (column 'usd_pledged_real') based on other features within the cleaned data set (e.g., 'category', 'project_length_day'). The model is evaluated by calculating the Mean Squared Error (MSE) and coefficient of determination. The results are discussed in the Evaluation section.

Evaluation and final results

After building and training the models, we evaluated the various models based on several criteria:

- Compared various classification models based on accuracy scores (or error rates) to determine the most suitable classification model;

- Looked at metrics such as Mean Squared Error (MSE) and coefficient of determination, which is reported as R-squared (R2);

Neural Network model fitting

Parameters of the neural network classifier (scikit learn MLPClassifier) are selected via plotting the training and test accuracy score curves as a function of the number of training epochs for different values of the L2 regularization term (alpha) and the hidden layer size.

Selection of alpha

The tested number of epochs is set within a range of 0 to 100 and the alpha values tested are 0.001, 0.01, 0.1, 1, and 10. Based on the plots, although all alpha values tested produce relatively low error rates and converge after about 20 epochs, the model performs the best when alpha is 1. Sample output plots for alpha selection are shown in Figure 6 below

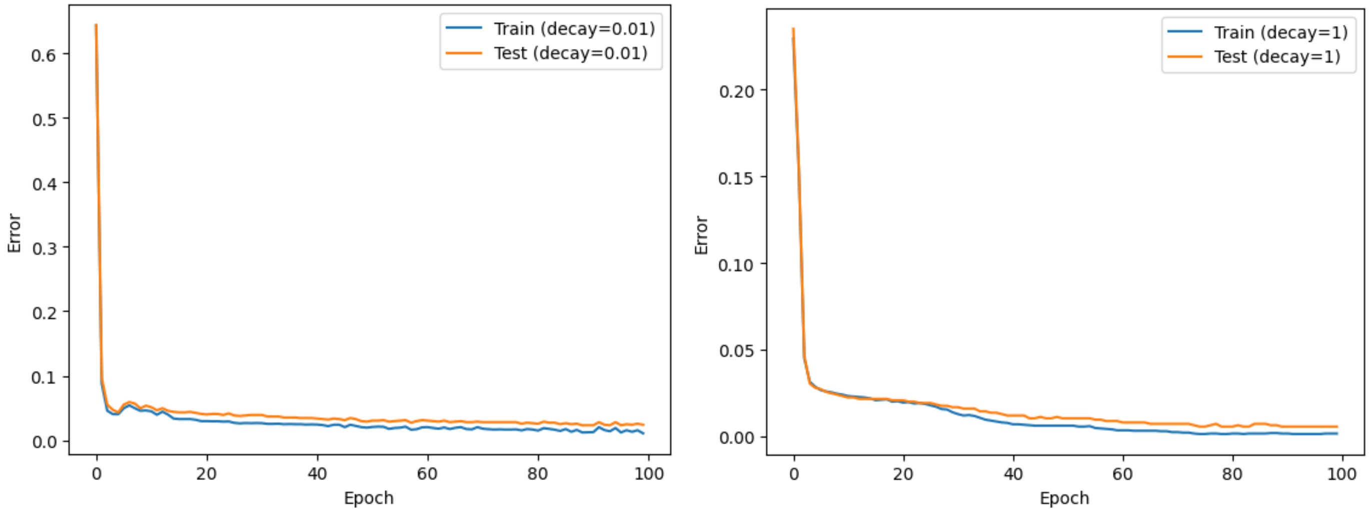

Figure 6: Alpha value selection in MLPClassifier (left: alpha = 0.01, right: alpha = 1)

Selection of hidden layer size

Next, we used alpha value of 1 and grid searched through a number of hidden layers between 0 and 100 to create an error rate plot (Figure 7). We can see that the error rate reduces and converges after about 5 units.

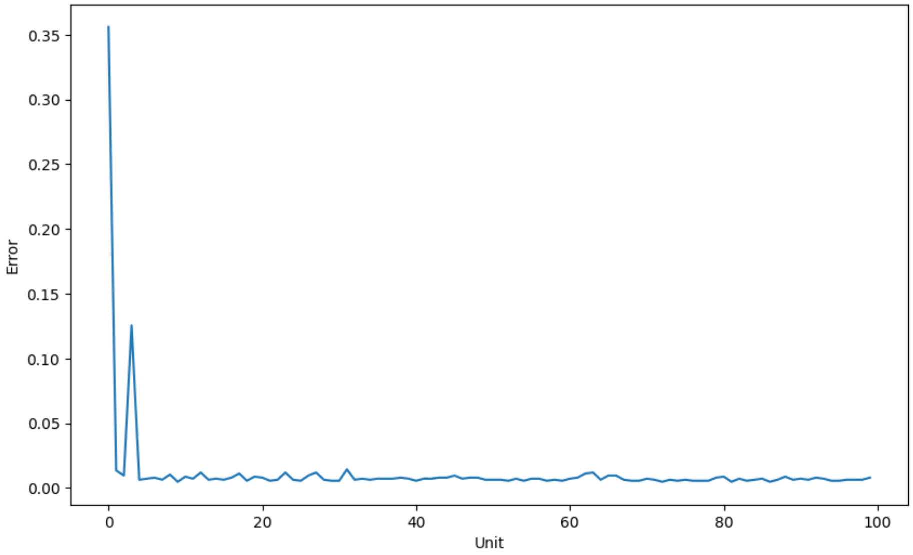

Figure 7: Hidden layer parameter selection in MLPClassifier

Thus, we will use alpha of 1 and hidden layer of 5 in the Neural Network MLPClassifier.

Random Forest model fitting

Randomized searches are used to determine the parameters of Random Forest. The parameters tested are, number of trees, number of features to consider at each split, minimum number of samples at each split, maximum depth, and method of selecting samples for each tree. The best parameters are used to re-train the Random Forest model. The re-trained Random Forest model is then compared with the base model (with default parameters) via measuring the model accuracy.

However, we found that the model trained with ‘best parameters’ performed the same as the base model, both having an accuracy of 98%. Thus, we will use the default parameters for the Random Forest model.

Classification models comparison

Next, we compared the model performance via measuring the model accuracy on the training and testing data sets for the four classification models with default parameters for Naïve Bayes, Logistic Regression and Random Forest classifiers, and with selected parameters for Neural Network classifier. The results are reported in Figure 8.

The Naïve Bayes model has an overall accuracy of between 73% and 78%, and a trend of decreasing accuracy as the number of training samples increases.

The Logistic Regression model has a close to 99% accuracy among smaller and larger numbers of training samples.

The Random Forest model has an over-90% accuracy among all training sample sizes. An improvement in accuracy is observed with a larger training sample size.

The Neural Network model has the lowest accuracy when training sample size is small. But the accuracy improves as the number of training samples increases, achieving over 90%.

Based on the observations stated above, it could be concluded that given both the model simplicity and performance, the Logistic Regression model is the preferred model among the four classification models.

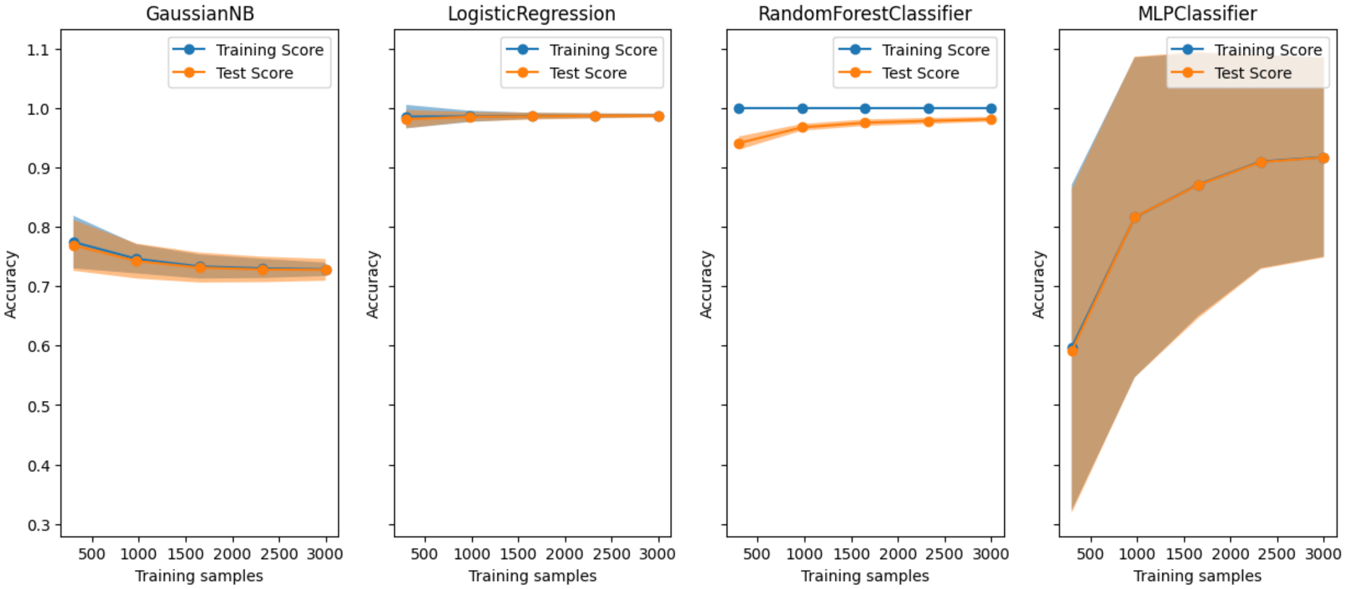

Figure 8: Effects of number of training samples on model accuracy for Naïve Bayes, Logistic Regression, Random Forest, and Neural Network (from left to right)

Linear Regression

The Linear Regression model has a resulting coefficient of determination (reported as the R2 score) of 0.65 (1 being the perfect prediction) and MSE of 2.13 billion. The ranking of the coefficients suggests that the pledged funding in USD is likely to be mostly driven by the main category of the project and the number of backers.

Conclusion and future work

We achieved our goal to guide investment decisions on Kickstarter projects by identifying key features to a successful project, suggesting the best model to predict whether a project would succeed, and by building a linear regression model to estimate the potential funding raised for a project. We found that for this data set, the ‘category’ and ‘usd_goal_real’ are the strongest indicators of a successful project. Logistic Regression might be the best model for project success/failure classification. We believe our project provided useful information for both Kickstarter enthusiasts and creators.

However, throughout the process, we also identified areas of improvement:

- Given the large amount of data involved, computational efficiency played a key role in model selection and training. For the purposes of this project, we had to downsize the data set to 5000 samples for our analysis. In the future, one could explore other methods such as changing the optimization function (solver).

- We converted columns comprised of textual data such as ‘category’ into numerical/categorical values and ignored the column ‘name’ in our analysis. One could also consider conducting text analysis on the column ‘name’ as that could provide valuable information to Kickstarter project creators on how to name their projects to make them more attractive to potential investors.

Team contribution

Proposal: Cheng & Kexin

Data cleaning and feature selection: Cheng

Modeling and evaluation: Kexin

Final report: Cheng & Kexin

References

A Beginner’s Guide to Scikit-Learn’s MLPClassifier: https://analyticsindiamag.com/a-beginners-guide-to-scikit-learns-mlpclassifier/

About Kickstarter: https://www.kickstarter.com/about?ref=global-footer#the-full-story

Hyperparameter Tuning the Random Forest in Python:https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

How to Speed up Scikit-Learn Model Training:https://www.anyscale.com/blog/how-to-speed-up-scikit-learn-model-training

Plotting Learning Curves and Checking Models’ Scalability:https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html#sphx-glr-auto-examples-model-selection-plot-learning-curve-py