Kickstarter Project Success Prediction and Classification using Multi-Layer Perceptron

Jahnavi Mehta
Information Technology
KJSCE
Mumbai, India
jahnavi.mehta@somaiya.edu

Hrishikesh Salunkhe
Information Technology
KJSCE
Mumbai, India
h.salunkhe@somaiya.edu

Hitansh Shah
Information Technology
KJSCE
Mumbai, India
hitansh.vs@somaiya.edu

Suchitra Patil
Information Technology
KJSCE
Mumbai, India
suchitrapatil@somaiya.edu

***Abstract*— Today, crowdfunding has become one of the most popular techniques to raise money for personal projects or group startups. Kickstarter, a leading platform for crowdfunding, provides vast opportunities for such projects. However, every two out of three projects fail to get their desired funding. Unfortunately, there is no such platform that provides in-depth analysis of projects uploaded on Kickstarter. Hence, the main objective of the project is to classify using deep learning algorithms the project into success or failure category along with statistics and analysis on various determining parameters of the project.**

Keywords- **Crowdfunding, Kickstarter, Classification, Deep Learning, Analysis**

I. INTRODUCTION

Crowdfunding is the practice where funds for a project or a venture is gathered by raising small amounts of money from a large number of people.Creators develop their business plan and idea and then upload it to the website with a financial goal in mind, and people can give money towards that goal if they believe in the business. It has become a leading alternative to Angel investors and VC fundings.

Kickstarter is one of the leading crowdfunding platforms that helps entrepreneurs get funding from the people [2].Project creators choose a deadline and a minimum funding goal. If the goal is not met by the deadline, no funds are collected. The kickstarter platform is open to backers from anywhere in the world and to creators from many countries.

However the problem faced by creators is that data shows, only one out of three projects on Kickstarter successfully receive their funding goal till they reach their deadline [5]. So with intention to help the project creators, we plan to create an interface which provides them with:

- Classification algorithm to determine whether the project is likely to succeed or fail.
- Statistics using graphs and data visualizations for each and every attribute of the project.

Using this interface entrepreneurs can fine tune each and every project parameter and improve their likelihood of a successful funding. We have collected datasets of Kickstarter projects [6][7][8][9] and implemented Deep Learning Model based on Multi Layer Perceptron Neural Network.

II. EXISTING WORK

In existing work, we see that reward dataset of Kickstarter projects have been analysed using various statistical methods[1] and it was concluded that projects with more rewards, with limited offerings and late-added rewards have a high chance of success.

Studies were done to analyse the project description pages [2], over a dataset of 26,000 Kickstarter projects to understand the factors that most affect the chances of having a successful funding. Random Forest was used on the model and 94.289 % accuracy was observed with precision of 94.5%

Apart from Kickstarter platform, another study was performed on popular chinese crowdfunding

platforms -JingDong and TaoBao[3].Logistic Regression model was used to understand how different platform designs and different mechanisms have an effect on the project success.
Another study [4] focused on the ratings of the crowdfunded products vs traditional products on Amazon. Machine Learning models were used to predict if the crowdfunded products will have higher ratings or not. It was concluded that crowdfunded products, on average, received lower ratings than traditional products.
In one study[5],the feature selection method, Support Vector Machines Recursive Feature Elimination, has been employed to find keywords that may affect the project's success. They also helped emerging entrepreneurs or anyone who needs to raise funds can have a higher chance of successful fundraising.
.

III. Proposed Methodology

We are making a web-based platform which will take input of project attributes from entrepreneurs and with the help of our deep learning algorithm we will provide them with detailed analysis that will tell them whether their project is going to succeed or fail, if it fails our system will provide statistics using graphs and data visualizations that they can use to improve their projects.

The classification and analysis of Kickstarter projects can be categorically divided into a series of steps which are as follows: Data Collection, Pre-processing, Natural Language Processing, Prediction, Classification & Statistical Graphs and Data Visualization.
The above-mentioned steps are depicted in the flowchart in Figure 1.

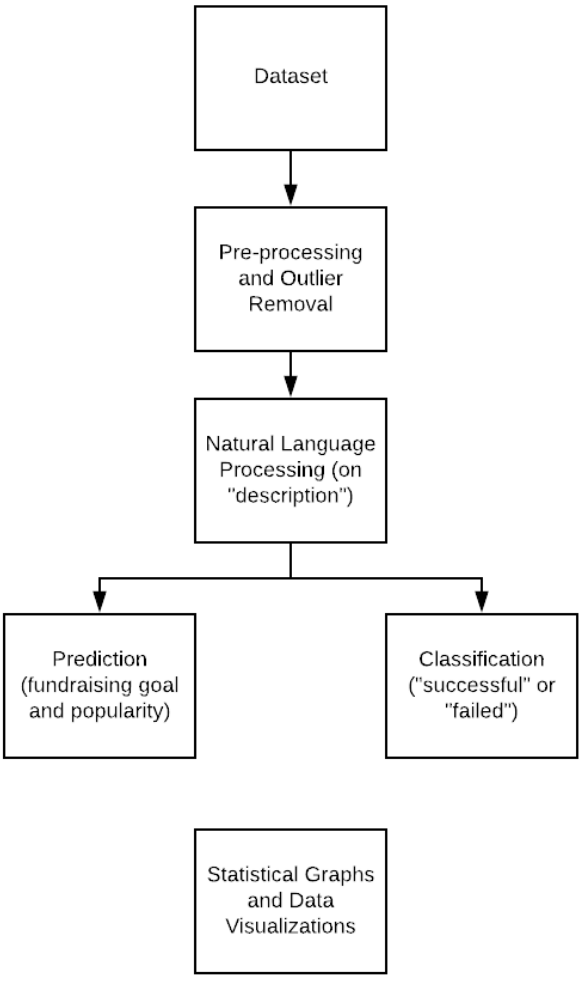

Figure-1

A. Dataset

We collected four different types of datasets of KickStarter Projects from Kaggle. There are 4 different datasets which have been used for various purposes like Prediction of certain parameters, Natural Language Processing on description of a kickstarter project and Classification of the state of a kickstarter project.The 4 datasets for our work are: Kickstarter Projects(General dataset) with size 300,000+ rows [6], Kickstarter-DataSet (Content dataset) with size 18,100+ rows [7], Kickstarter-2018- nlp(Description dataset) with size 379,000+ rows [8] and Kickstarter(Rewards dataset) with size 46,000+ rows [9]. We have used 3 folds for our Multi-Layer Perceptron implementation,on

all four datasets mentioned above. Hence each dataset is divided into three divisions.

- For our 1st fold, the first division is used for testing while the second and the third division is used for training.
- For our 2nd fold, the second division is used for testing while the first and the third division is used for training.
- For our 3rd fold, the third division is used for testing while the first and the second division is used for training.

Hence for one dataset the testing set constitutes as 33.33% and the training set constitutes 66.67%

B. Preprocessing And Outlier Detection/Removal

We started with pre-processing techniques which included removal of attributes that were not useful for analysis, replacing textual attributes like **Category, MainCategory, Country, State** to numerical values which was followed by replacing null and negative values in dataset with default values like 99999 and 0.
We searched for anomalies and outliers in our datasets by applying outlier removal techniques in removing them if they were present.

C. NLP On Description Set

Using the description of a kickstarter project in a textual format we calculated several NLP attributes such as number of syllables, number of words, number of unique words, number of characters , monosyllabic words , polysyllabic words etc. and also various readability scores on the dataset like:

- **flesch_reading_ease**

  measures how difficult the passage is to read,it uses the average length of sentences (measured by the number of words) and the average number of syllables per word in an equation to calculate the reading ease [10].
- **Smog Index**

  estimates the years of education a person needs to understand a piece of writing,involves finding count of polysyllables in a sentence[10].
- **gunning_fog index**

  readability test that takes a weighted average of the number of words per sentence, and the number of long words per word[10].
- **coleman_liau_index**

  readability metric used to indicate how difficult a text is to read based on the number of letters per word and words per sentence[10].

D. Prediction

We developed a **linear regression** model to predict parameters(while taking all the other project parameters as input) such as:

- **amount** that could be pledged
- **popularity** (number of backers expected)

E. Classification

In this step we will classify the kickstarter project as either success or failed. For classification, we will be using a Multi-Layer Perceptron on the Kickstarter Projects dataset consisting of 3,00,000 rows [6].
Multi-Layer Perceptron is a neural network that is inspired by the way biological nervous systems work in the human body, just like the way our brain processes information. The main element of Multi-Layer Perceptron is the way the information processing system is structured and designed. It is made up of a large number of highly interconnected processing elements (neurons) working together to solve a specific problem.

1. **Normalizing the dataset:**

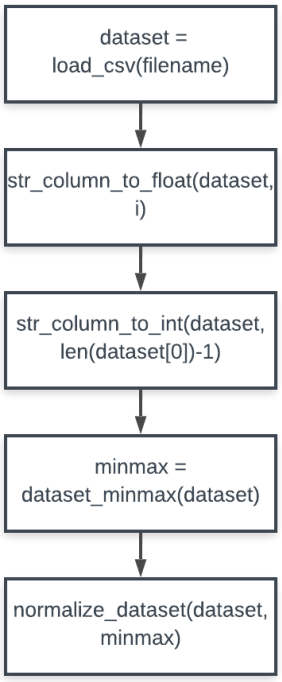

Figure-2

Before implementing the Multilayer Perceptron (MLP) model for classification we normalize the dataset that we use for providing the input parameters and the class variable values for training the MLP model. We start with the "load_csv" function where we load the dataset from a .csv file to a python list. After that we convert all the input parameter values into float type and the class variable values to integer type by using the "str_column_to_float" and "str_column_to_int" functions respectively. Having done that we calculate the minimum values and maximum values of each input parameter using the "dataset_minmax" function and store these values in a python list named "minmax". Finally we pass this list "minmax" along with the python list "dataset" into "normalize_dataset" function which normalises the python list "dataset" using min-max normalization.

2. Implementing MLP:

The classifier used is a Multilayer Perceptron (MLP) model using Back-propagation algorithm.

The hyperparameters of the classifier are as follows:

- The number of folds in which the dataset is divided into training and testing sets is 3.
- The learning rate is equal to 0.05.
- The number of epochs is 1000.

These specifications apply to all the MLP classifiers used for every dataset.

Pseudo Code for the algorithm:

```
evaluate_algorithm(dataset):
        folds = cross_validation_split(dataset)
        for each fold in folds:
    Generating the train_set and test_set from the rows in fold.
    predicted = back_propagation(train_set, test_set):
        Generating n_inputs and n_outputs using the rows in train_set.
        network = initialize_network(n_inputs, n_outputs)
        network.append(train_network(network, train_set, n_outputs):
            for each epoch in n_epoch:
                for each row in train_set:
                    outputs = forward_propagate(network, row)
                    backward_propagate_error(network, expected):
                        Calculating a list of errors for all the layers in the network
                        For each neuron in the layer:
                            neuron['delta'] = errors * transfer_derivative(neuron['output'])
                    update_weights(network, row)
                    return network
        )
        for each row in test_set:
            prediction = predict(network, row):
                outputs = forward_propagate(network, row):
                    inputs = row
                    for each layer in network:
                        new_inputs = []
                        for each neuron in layer:
                            activation = activate(neuron['weights'], inputs)
                            neuron['output'] = transfer(activation)
                            new_inputs.append(neuron['output'])
                        inputs = new_inputs
                    return inputs
                return outputs.index(max(outputs))
            predictions.append(prediction)
        return(predictions)
```

```
        Taking the actual values from the rows in fold.
        accuracy = accuracy_metric(actual, predicted)
        scores.append(accuracy)
    return scores
```

Above mentioned is the pseudo code for the "evaluate_algorithm" function of class MLP which is called in order to train the MLP model and calculate accuracy of the same. In this function the dataset is first divided into the given number of folds using the "cross_validation_split" function. Now for each fold we generate the training set and testing set as pass those to the "back_propagation" function to get the values predicted by the classifier.

Inside the "back_propagation" function we first initialize the network by using the "initialize_network" function and the number of inputs and the number of outputs required to initialize the network are taken from the rows in the training set. After the network has been initialized it is trained using the "train_network" function wherein for each epoch, of the given number of epochs, output for each row of the training set is calculated. This output is calculated by using the "forward_propagate" function where the values of neurons are computed are propagated in the forward direction to generate a single output at the output layer. The expected values for taken in the range of number of outputs and are passed along with the network to the "backward_propagate_error" function. The "backward_propagate_error" function calculates a list of error values for all the layers in the network and these are multiplied to the transfer derivative of each neuron's output to calculate the 'delta' values of each neuron. These 'delta' values are used to update the weights of the network using the "update_weights" function. A python list named 'network' is provided as an output to the "train_network" function which is appended to the python list 'network' of the class MLP.

Now for each row of the testing set we calculate the predicted values using the "predict" function. Inside the "predict" function the "forward_propagate" function is called to generate the outputs. Then for each neuron in each layer of the network we calculate the activation of each neuron using the "activate" function and transfer this activation to that neuron's output using the "transfer" function. These outputs of all neurons are appended to a list and this list is returned by the "forward_propagate" function. The index of the maximum of all outputs is returned and is the value of prediction. In this way, predicted values are appended to a python list named "predictions" and these values are compared with the actual values in the dataset by using the "accuracy_metric" function.

Finally in this way accuracy for each fold is calculated and appended to a python list named 'scores' which is returned by the "evaluate_algorithm" function.

Pseudo Code for the "predict" function:

```
predict(network, row):
    outputs = forward_propagate(network, row):
        inputs = row
        for each layer in network:
            new_inputs = []
            for each neuron in layer:
                activation = activate(neuron['weights'], inputs)
                neuron['output'] = transfer(activation)
                new_inputs.append(neuron['output'])
            inputs = new_inputs
        return inputs
    return outputs.index(max(outputs))
```

Above mentioned is the elaborated pseudo code of the "predict" function. After the classifier has been trained and it's accuracy has been calculated we can classify Kickstarter projects based on their input parameters as "successful" or "failed" project using the "predict" function.

We have used the Backpropagation algorithm which is a supervised learning method for multilayer feed-forward networks from the field of Artificial Neural Networks. The functions used in the classifier are:

- Initialize Network
- Forward Propagate:
    - Neuron Activation
    - Neuron Transfer
    - Forward Propagation
- Back Propagate Error
    - Transfer Derivative
    - Error Backpropagation
- Train Network
    - Update Weights
    - Train Network
- Predict

F. Statistical Graphs and Data Visualizations

This is the last step to be performed. This step will give necessary insights to our entrepreneurs via statistics using graphs and data visualizations. With help of analysis on each visualization they can gain valuable insights to improve their project.

- From the General dataset[6], we display probabilities of success and failure, mean goal, mean amount pledged, mean backers and mean duration of all the projects. From the Rewards dataset[9], we display mean reward levels of all the projects.
- From the Content dataset[7], we display mean facebook friends,mean facebook shares,mean number of projects created that were successfully backed, mean number of images, mean number of words in description.
- From the Description dataset[8], we display mean number of words in the description,mean number of characters in the description and mean number of sentences in the description.

Kickstarter offers various categories in which projects can be uploaded on its platform. So we have also designed our product to provide category-wise data visualizations for various attributes of the three datasets as mentioned above.

IV. Results and Discussions

1. Multi-Level Perceptron using Backpropagation Algorithm

The general model used for the Multi-Layer Perceptron is depicted in the figure below:

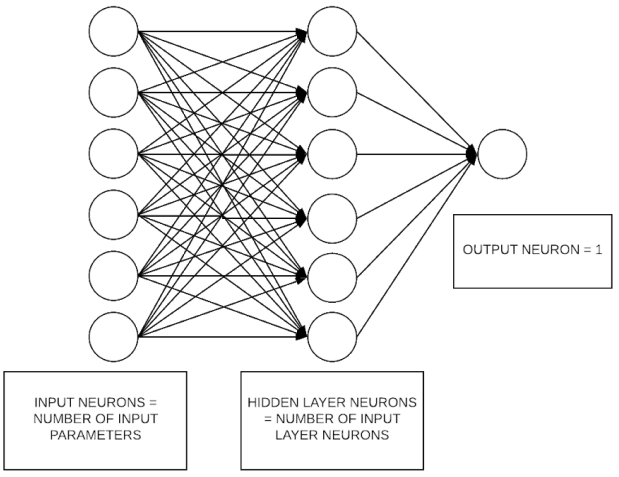

Figure-3

The model consists of three layers:

- Input Layer : This takes the input to our model which is always equal to the number of input parameters.
  The following are the number of input neurons for the four datasets:
  - General dataset[6]: 6
  - Content dataset[7]: 18
  - Description dataset[8]: 22
  - Rewards dataset[9]: 8
- Hidden Layer: The number of neurons in this layer is equal to the number of input layer neurons. The input to the hidden layer neurons comes from the input neurons as seen in the figure above.
  The following are the number of hidden layer neurons for the four datasets:
  - General dataset[6]: 6
  - Content dataset[7]: 18
  - Description dataset[8]: 22
  - Rewards dataset[9]: 8
- Output Layer: This is the final layer. It consists of only 1 neuron (for all datasets). The output is our final result, that is, project is classified as ”successful” or ”failed”.

For the Multi-Layer Perceptron implementation we have used the following attributes of datasets as input parameters:
Inputs:

- **General[6]**: Goal, Category, Main Category, Country, Currency, Duration as shown in Figure-4.
- **Content[7]**: Number of rewards, Goal, Duration, FB-friends, FB-shares, Number of Projects Created, Number Of Projects Backed, Number Of Videos Uploaded, Images Uploaded, Number of words in description, Number of words in risk and challenges, Number of FAQs, Currency, Category, Main Category, FB-Connected, Has Video, has Website, as shown in Figure-5 & Figure-6
- **Description[8]**: Description of Project, Goal, Category, Main Category, Country, Currency, Duration as shown in Figure-7.
- **Rewards[9]:** Goal, reward levels, Duration, Main Category, Category, Min Reward, Max Reward ,Mean Reward as shown in Figure-8.

The Target Output: Success/Failure Of Project .

Following are the screen images of the web-interface for MLP:

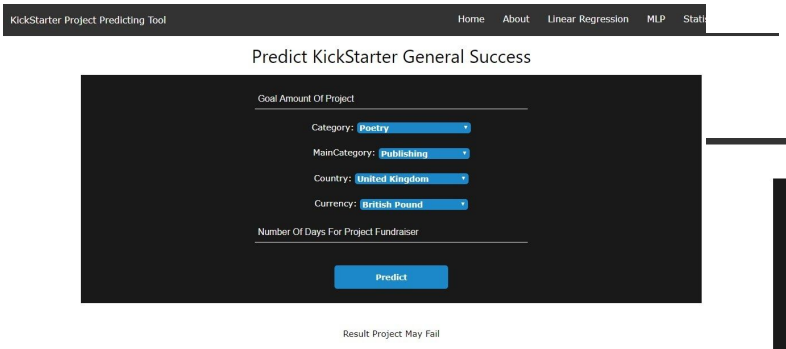

Figure-4

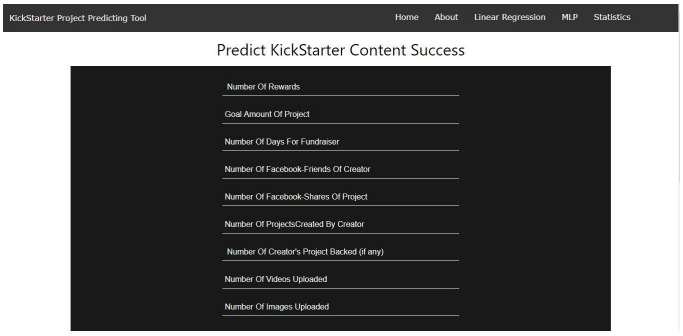

Figure-5

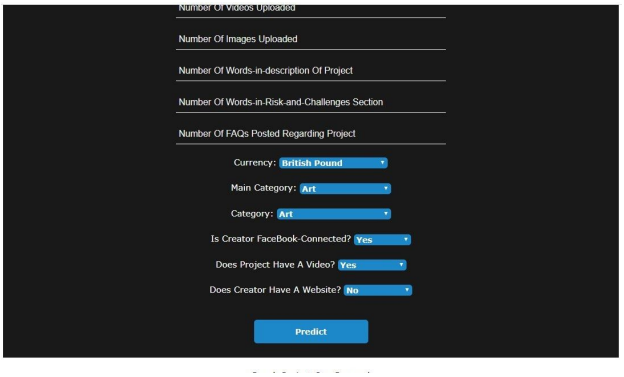

Figure-6

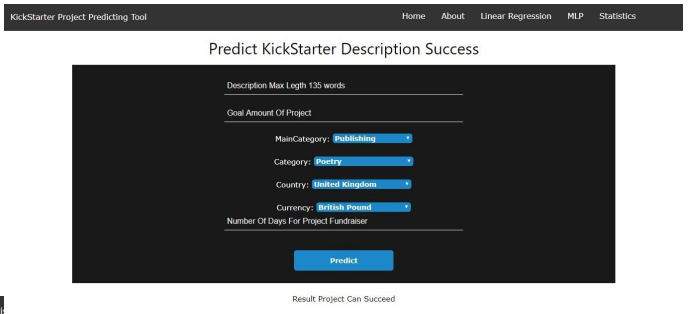

Figure-7

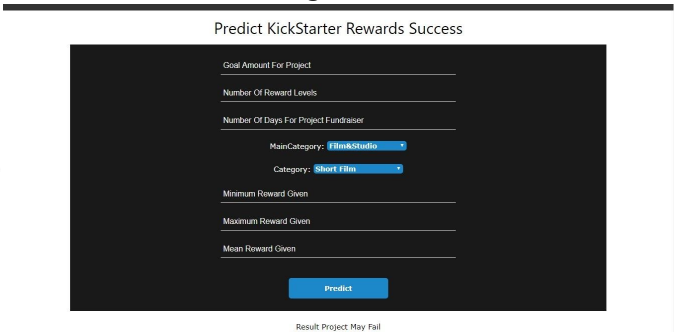

Figure-8

We use a function defined as accuracy metric which takes in two input parameters that is actual class value from the training dataset and the predicted class value from the predict function of the classifier. Then we simply compare whether these two values are equal or not and maintain a count to calculate the total percentage of correct predictions that is accuracy.

2. Linear Regression

We use the inbuilt 'score' function of scikitlearn's Linear Regression classifier to calculate accuracy for the same. The table below provides all the accuracies of Linear Regression models developed in this project against the

4 datasets used with respect to two output parameters namely amount pledged and number of backers. The performance of the Linear Regression models is evaluated in terms of their accuracies.

Output Parameter	General [6]	Description [8]	Content [7]	Rewards [9]
Amount Pledged	89.5	89.6	93.0	93.0
Number of Backers	34.3	34.5	44.7	39.6

For the Linear Regression implementation we have used the following attributes of datasets:
Inputs:

- **General[6]:**Goal,Category,MainCategory,Country,Currency,Duration as shown in Figure-9
- **Content[7]**: Number of rewards, Goal, Duration, FB-friends,FB-shares, Number of Projects Created, Number Of Projects Backed, Number Of Videos Uploaded , Images Uploaded, No.of words in description,No.of words in risk and 67 challenges, No.of FAQs, Currency,Category,MainCategory, FB-Connected, Has Video, has Website as shown in Figure-10
- **Description[8]**: Description of Project, Goal, Category, MainCategory, Country, Currency, Duration as shown in Figure-11

- **Rewards[9]**: Goal,reward levels,Duration,Main Category,Category,Min Reward, Max Reward ,Mean Reward as shown in Figure-12

The Target Output: Amount Pledge and Backers.

Following are the screen images of the web-interface for Linear Regression:

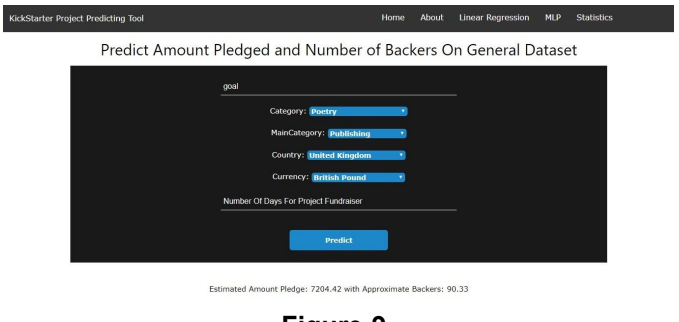

Figure-9

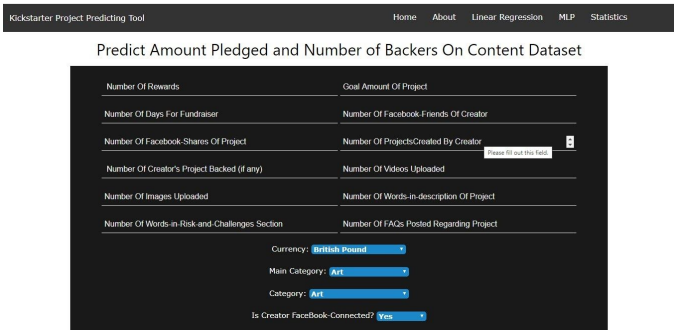

Figure-10

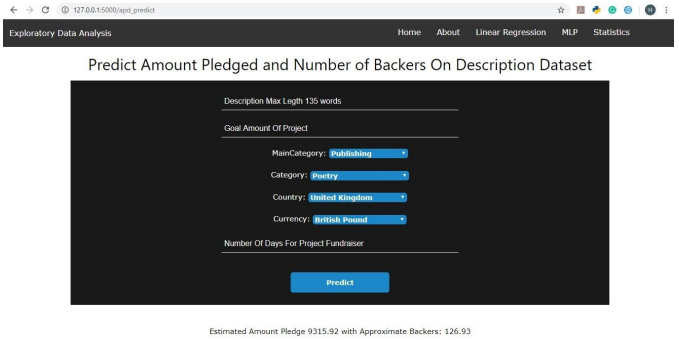

Figure-11

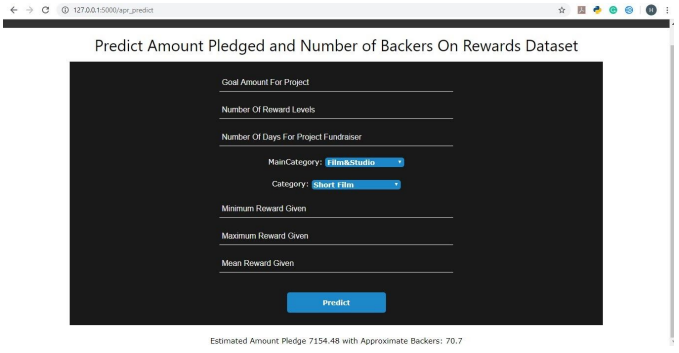

Figure-12

3. Comparison of different classifiers

We have calculated the accuracy by counting all outputs where actual value is equal to predicted value and have

divided that count with the total number of rows to get the percentage, that is, Accuracy.

Our accuracy testing method is Precision, where if the actual value is equal to the predicted value then it is a true positive. We then divide this count with the total, where Total is equal to sum of correct values and incorrect values.

The table below provides a comparison table of the accuracies of six classifiers, that is, MultiLayer Perceptron (MLP) developed in this project along with sklearn's predefined classifiers such KNN, Naive Baiyes, Logistic Regression, Random Forest and Support Vector Machines(SVM) against the 4 datasets used in this project. The performance of the classifiers is evaluated in terms of their accuracies.

Classifier	General [6]	Description[8]	Content[7]	Rewards [9]
MLP	59.85	63.05	79.05	67.96
KNN	61.36	56.8	74.31	62.85
NaiveBaiyes	53.86	54.57	56.37	60.13
Logistic Regression	56.65	59.9	80.18	67.18
Random Forest	58.66	59.03	77.05	63.46
SVM	57.92	58.93	63.61	65.46

To get a more clear idea on these comparisons, we have inserted bar graphs of each of the four datasets. Each bar graph consists of accuracy plotted against the different classifiers.

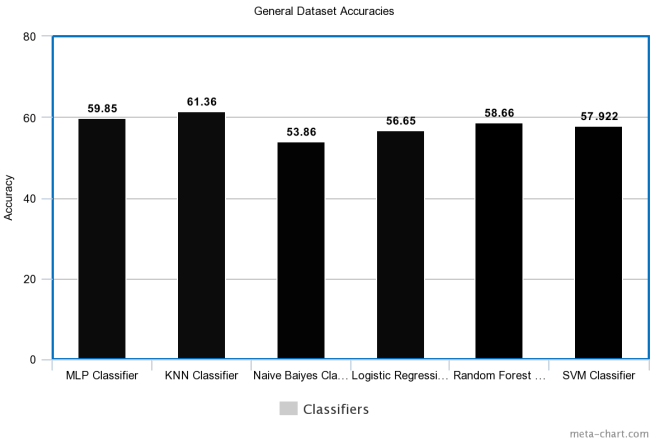

Figure 13: Bar Graph for General Dataset

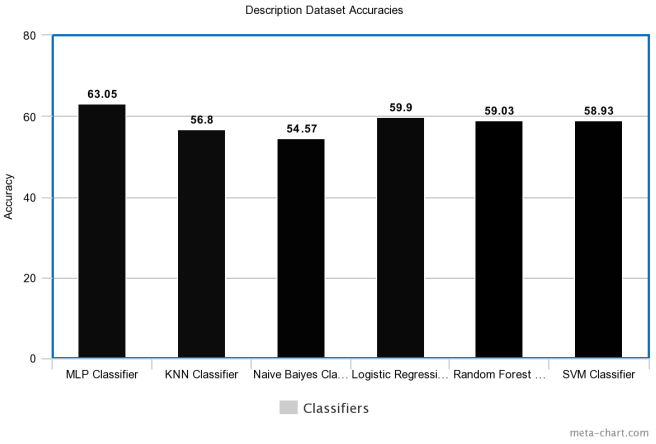

Figure 14: Bar Graph for Description Dataset

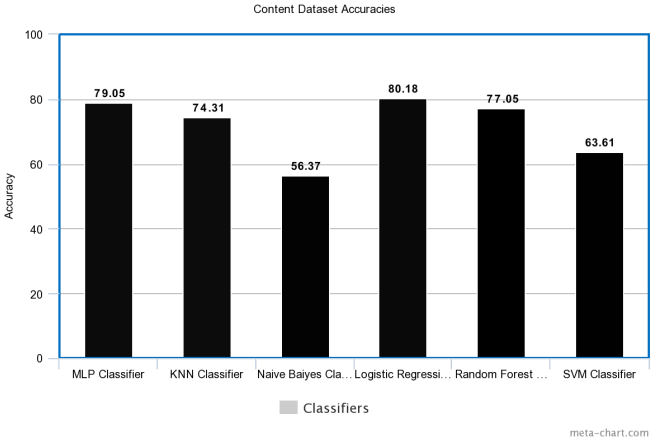

Figure 15: Bar Graph for Content Dataset

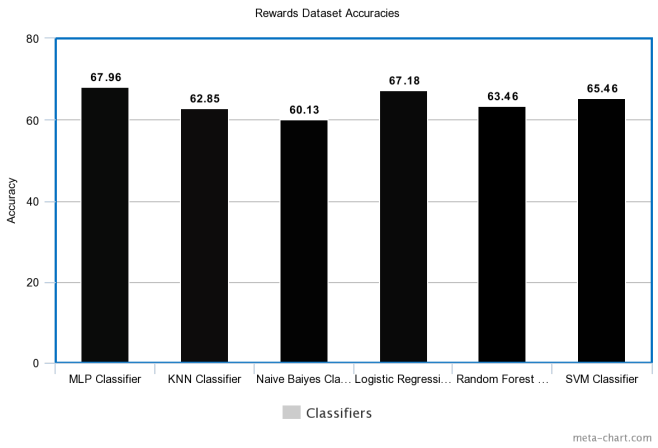

Figure 16: Bar Graph for Rewards Dataset

V. Conclusion

Since we know that every 2/3rd of the projects uploaded fail to receive their funding goal till they reach their deadline on Kickstarter, we have successfully designed a web-based application to help entrepreneurs. It contains:

- Classification, where the entrepreneurs can know whether with the current attributes, will their project be successful or not.
- Linear Regression, where if the entrepreneur wishes to know the value of the amount pledged and number of backers(attributes of a project that are realised at the end of the project), they can put the values of the current attributes into the Linear Regression model to predict those values.
- General statistics using graphs and data visualizations for classified failed projects or for further improvements on current classified successful projects.
- Category-wise statistics using graphs and data visualizations for various attributes.

Hence our project has all these features packed together on our website for the entrepreneurs.Using this interface entrepreneurs can fine tune each and every parameter and increase the chances of success of projects that are likely to fail.

VI. Future Scope

The future work shall consist of improving the classification accuracy by finding datasets that have more correlation between attributes thus increasing overall accuracy of classification algorithms. It also includes the development of an app-based version of this system since most of the users carry out their major activities on mobile platforms.

VII. Acknowledgements

The success and final outcome of this project required a lot of guidance and assistance and we are extremely fortunate to have got this all along the completion of our project. We respect and thank Prof. Suchitra Patil, Assistant Professor, Department of Information Technology from K.J Somaiya College of Engineering, for providing us all the support and guidance which made us complete our project on time. We are extremely grateful to her.

VII. Refeerences

[1] Rausan Lin, Wang-Chien Lee, Chung-Chou H. Chang, “Analysis of Rewards on Reward-Based Crowdfunding Platforms”, in IEEE 2016/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM).

[2] Fahad Sarfaraz Ahmad, Devank Tyagi, Simran Kaur “Predicting Crowdfunding Success with Optimally Weighted Random Forests”, IEEE 2017.

[3] Zhiyuan TIAN, Lei GUAN, MeilinSHI,“The Key Factors of Successful Internet Crowdfunding Projects – An Empirical Study Based on Different Platforms” in IEEE 2018.

[4] Vishal Sharma, Kyumin Lee, “Predicting Highly Rated Crowdfunded Products”, in IEEE 2018/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM).

[5] Long-Sheng Chen, En-Li Shen, “Finding the Keywords Affecting the Success of Crowdfunding Projects”, in IEEE 2019.

[6] M. Mouille, “Kickstarter Projects” 2018. [Online]. Available:

https://www.kaggle.com/kemical/kickstarter-projects/activity

[7] Taki, “Kickstarter Dataset” 2018. [Online]. Available: https://www.kaggle.com/tayoaki/kickstarter-dataset

[8] A. Noca, “Kickstarter Projects” 2018. [Online]. Available: https://www.kaggle.com/antonionoca/kickstarter2018nlp

[9] P. Arienza, “Kickstarter” 2017. [Online]. Available: https://www.kaggle.com/parienza/kickstarter

[10] S.Bansal,C.Agarwal, “textstat 0.6.0” 2020. [Online]. Available: https://pypi.org/project/textstat/

[11] Lukkarinen, A, Teich, J. E., Wallenius, H. and Wallenius, J, (2016)."Success drivers of online equity crowdfunding campaigns". Decision Support Systems 87, 26–38.

[12] J.-H. Kim, P. W. Newberry, and C. Qiu, “An empirical analysis of a crowdfunding platform,” Available at SSRN, 2015.

[13] V. Rakesh, J. Choo, and C. K. Reddy, “Project recommendation using heterogeneous traits in crowdfunding,” in ICWSM, 2015.

[14] J. Solomon, W. Ma, and R. Wash, “Don’t wait!: How timing affects coordination of crowdfunding donations,” in CSCW. ACM, 2015, pp.547–556
.

[15] Brown, T., Boon, E., Pitt, L. (2016). Seeking funding in order to sell: Crowdfunding as a marketing tool, Business Horizons.

[16] Hobs, J., Grigore, G. and Molesworth, M. (2016) "Success in the management of crowdfunding projects in the creative industries, Internet Research, Vol. 26 No. 1, pp. 146-166.

[17] T. Kim, M. H. Por, & S.-B. Yang, “Winning the crowd in online fundraising platforms: The roles of founder and project features,” Electronic Commerce Research and Applications ,Vol. 25, 2017, pp.86-94.

[18] G. M. Thomaz, A. A. Biz, E. M. Bettoni, L. Mendes-Filho, & D.Buhalis, “Content mining framework in social media: A FIFA world cup 2014 case analysis,” Information and Management, vol. 54, no. 6,2017, pp. 786-801.

[19] W. Wang, K. Zhu, H. Wang, & Y.-C. J. Wu, “The impact of sentiment orientations on successful crowdfunding campaigns through text analytics,” vol. 11, no. 5, 2017, pp. 229-238.

[20] Q. Du, Z. Qiao, W. Fan, M. Zhou, X. Zhang, & G. Wang, “Money Talks :A Predictive Model on Crowdfunding Success Using Project Description,” Twenty-first Americas Conference on Information Systems, AMCIS, 2015, pp. 1–8.

[21] M. Zhou et al., “Project description and crowdfunding success: an exploratory study” Information Systems Frontiers, Springer, December 7, 2016, pp. 1-16.

[22] J.Chung and K. Lee, “A Long-Term Study of a Crowdfunding Platform: Predicting Project Success and Fundraising Amount”, HT ’15, Guze- lyurt, Northern Cyprus: September 1-4, 2015.

[23] D. Cumming, G. Leboeuf, A. Schwienbacher, “Crowdfunding models: Keep-it-all vs. All-or-nothing,” January 2017, [Online]. Available: https://ssrn.com/abstract=2447567

[24] T. Tran, K. Lee, N. Vo, and H. Choi, “Identifying on-time reward delivery projects with estimating delivery duration on kickstarter,” in ASONAM,2017.

[25] D. Frydrych, A. J. Bock, T. Kinder, B. Koeck, “Exploring entrepreneurial legitimacy in reward-based crowdfunding,” Venture Capital, vol. 16, pp. 247-269, March 2016.