CMPE 256
PROJECT REPORT

DonorsChoose.org Application Screening

Submitted To:

Dr. Magdalini Eirinaki

Submitted By:

Team Samachar

Dixita Patodiya (011817182)

Sadab Qureshi (011486033)

Sai Teja Desu (012455287)

URL of Project: https://github.com/DixitaPatodiya/CMPE256

1. Introduction

1.1 Motivation:

We choose this dataset from the suggested Machine Learning Datasets provided in Canvas. We are motivated to work on DonorsChoose.org dataset because it is a live competition and good dataset for beginners to work on applying basics of machine learning concepts which we learnt as part of the course. Also, we believed that working on this competition can lead to habituate in solving live competitions.

The problem we focused on the selected dataset is to automate the application screening process where the organization gets more than 500,000 requests at any given point of time. Currently many volunteers are working on screening the applications manually which is equivalent to thousands of man hours. Organization believed that this manual process can be automated through machine learning algorithms to auto-approve at least some applications without taking much time. Such that the same volunteers can devote their time in other detailed process.

1.2 Objective:

Our objective is to work on predicting whether a given project application can be approved or not based on the information provided. Also, to implement efficient algorithms for training the model for better prediction with the help of standard evaluation techniques.

2. System Design & Implementation details

2.1 Algorithms Used:

The problem which we attempted to solve is based on text classification. Accordingly, we have selected following supervised learning algorithms:

- Logistic regression
- SVM
- Naive bayes
- Random forest
- K-nearest neighbors
- Decision tree and Gradient Boosting
    - Adaboost
    - Lightgbm
    - Xgboost

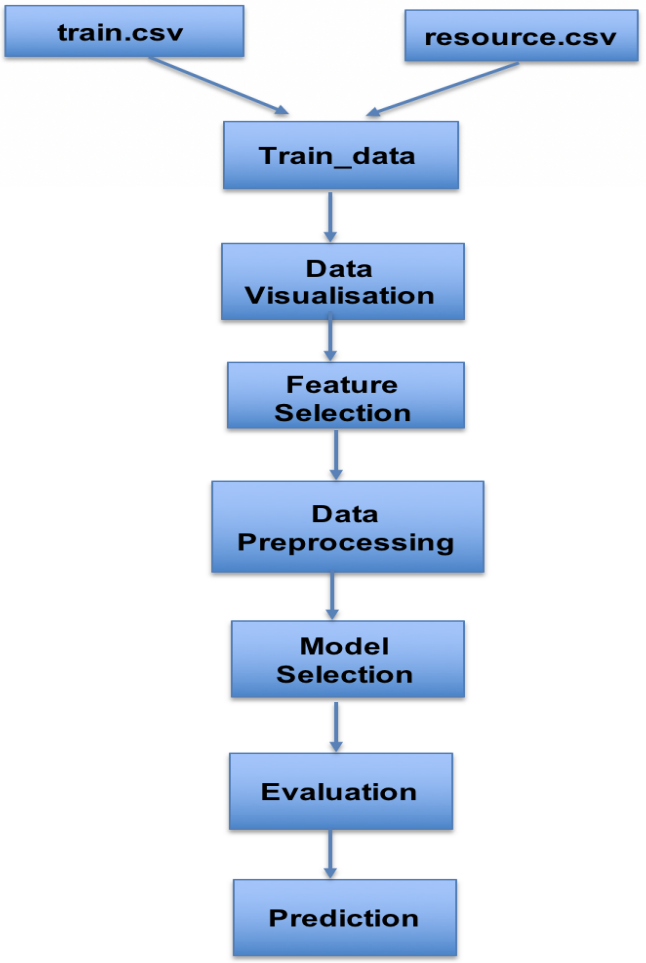
We have experimented above algorithms and based on the accuracy and the results each algorithm gave, we have finalized below four algorithms for better accuracy:

A. **Logistic regression:** It is simplest model used for classification problems. The accuracy we got by using this model was highest.
B. **Multinomial Naive bayes:** This classifier works well with text classification and our dataset also has multiple text attributes to work with.
C. **Adaboost, Xgboost and Lightgbm**: As part of trying different types of models we tried few of the decision-tree based models and these are popular for classifications and giving better accuracy scores.
D. **Random forest:** As part of the data exploration activity we observed the data imbalance in the label. Approved project percentage is 84.8%. So, we decided to try with Random forest algorithm.

2.2 Technologies & Tools used:

- Python: We used python as programming language as it is easy to work and has rich set of libraries for data exploration and analysis.
- Anaconda Navigator: Used for Jupyter notebook and also for easy installation of libraries used in python.
- Libraries used: scipy, numpy, pandas, nltk, sklearn, matplotlib, plotly, seaborn.
- Tableau: We used this software for better and easy visualization of dataset.

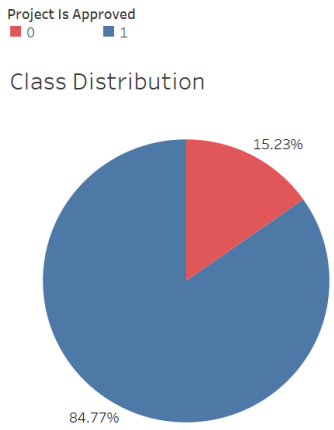
2.3 System workflow:

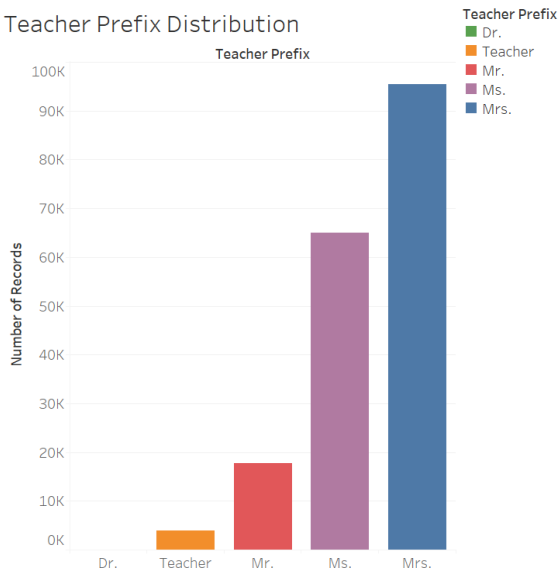


2.4 Visualization:

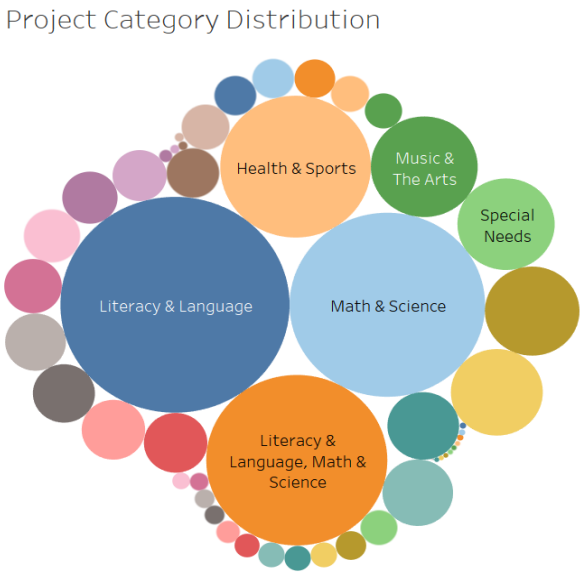
We have used data visualization for better understanding of relation between each attribute and we were able to estimate the importance of features:

- The bar graph below shown on right is generated to check the effect of teacher prefix on the project approval. We can observe that there are more female teachers submitting the applications. If we use this feature for training models, we may get biased model.
- The pie chart below shown on left depicts the proposals approved and rejected out of total number of applications. We can observe that there is an imbalance between each class.





- Below is the visualization is generated to check the distribution of project category. We can observe that projects of some categories are more, and some are very less.



3. Experiments

3.1 Dataset:

We have taken Dataset from Kaggle https://www.kaggle.com/c/donorschoose-application-screening/data

The size of zip file of this dataset is 203.9MB. It contains following files:

- Train.csv: It contains the data of 182080 proposals and has 16 features.
- Test.csv: test data for 78035 proposals and has 15 features.
- Resource.csv: data of resources requested, their quantity and price. It has 1541272 entries and have 4 features.

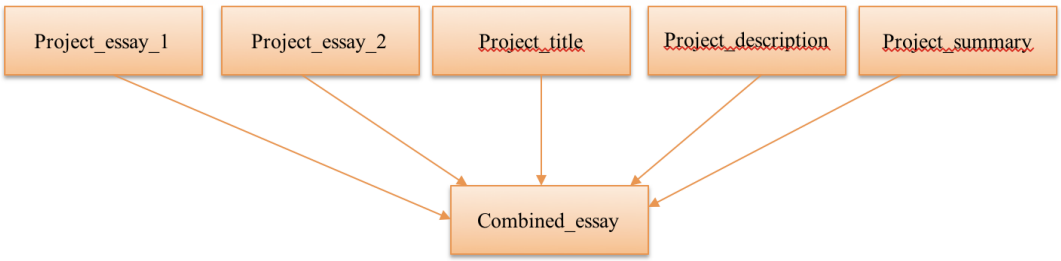
3.2 Data preprocessing:

- We merged train.csv and resource.csv based on id.
- Based on visualization and data exploration, we did feature extraction. The features which we selected to use were text based.
- Replacing null values: We checked for the null values in the data set and replaced those null values with valid string.
- Applied Tfidf vectorization on the text features of data set for removing special characters, stop words, stemming and for generating tokens.

3.3 Feature Engineering:

3.3.1 Feature extraction:

- Created combined text feature from all the text attributes for easy implementation:



- Creating total price and total quantity feature from resource.csv

3.3.2 Conversion of Categorical Features into numeric Format:

- school_state
- project_subject_categories
- project_subject_subcategories
- teacher_prefix

3.3.2 Feature removal:

We considered below features are not important and not influencing the decision of the class.

- **id**

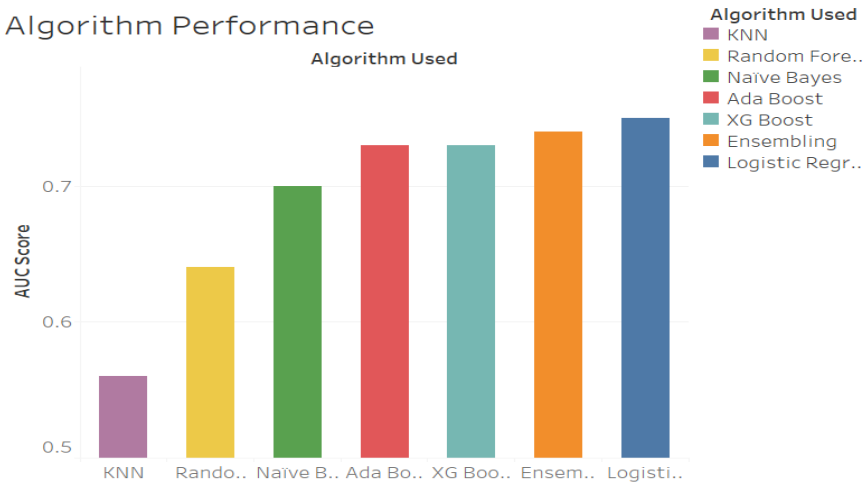
- **project_title**
- **project_essay_3**
- **project_essay_4**
- **text**
- **teacher_number_of_previously_posted_projects**
- **Categorical Features**

3.4 Methodology followed:

We followed n-fold cross validation with value of n=5. It randomly splits train data into train and test and does the validation. Final validation score is considered by calculating average of all five cross validation scores. We performed this method on all experimented algorithms and then selected the models which gave best accuracy and then again we tried to optimize results by changing some parameters in the model.

Ensemble learning: We tried combination of the different algorithms which gave us best results.

Below is the comparison of each algorithm performance



3.5 Analysis of results:

Based on comparative analysis of each algorithm, logistic regression model gave the best output and k-nearest-neighbor model gave the lowest accuracy. Also, after changing learning rate, other parameters and by trying different models, we were able to get more accuracy score. Initially, we got around ~55% accuracy and we were able to increase upto ~75% using ensemble learning.

4. Discussion & Conclusions

4.1 Decisions made:

- Deciding important features and removing unnecessary features.
- Based on the features we selected, we discussed different preprocessing techniques that can be applied.
- We discussed the algorithms that can be used and how to implement them to get good accuracy.
- Selected N-fold cross validation as the evaluation method.

4.2 Difficulties faced:

- The size of dataset is big and also there are many features, so initially it was difficult to understand it and decide how to apply different data exploration techniques.
- Deciding on which features could affect the prediction.
- Vectorizing the text features.
- In this data set the results were dependent on 2 files that is train.csv and resource.csv, so when we merged these two files it was taking a lot of time for preprocessing data.
- Initially the function for preprocessing data which we developed was taking a lot of time to preprocess the text and give us the desired output.
- Increasing accuracy after 70%.

4.3 Things that worked:

- We referred the beginner tutorial on the kaggle for getting started with datasets and algorithm and each did individual work so that we all understood the approach to solve the problem.
- Each team member worked on preprocessing to make it faster and easier.
- Experimenting different models to get more accuracy for getting optimized results.

4.4 Things that didn’t work well:

- Some algorithms were taking very long time to implement. Like SVM took a lot of time to implement.

4.5 Conclusion:

- Spending quality of time in preprocessing of data is necessary to get optimized results.
- Feature Engineering can make algorithms to work better.
- Simple algorithms work better than complex on some datasets.

5. Project Plan

5.1 Justification of Tasks:

Tasks were equally divided and everyone contributed to the project

- We all discussed the project ideas and selected the topic.
- Initially we started doing individual work as we all wanted to learn and get hands on with data. So, data exploration, visualization, analysis and preprocessing were performed by all then we merged our work into one.

Task	Subtask	Assignee
Project Topic		All
Discussion on topic		All
Data Analysis		All
Data Visualization		All
Data Preprocessing		All
Algorithms	KNN, Random forest and Naive Bayes	Saiteja Desu
	Logistic Regression, SVM, Xgboost, ensembling , cross validation	Dixita Patodiya
	Decision tree - Adaboost, Lightgbm and XGBoost	Sadab Qursehi
Report		All
Presentation		All

6.References:

- https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a
- https://www.youtube.com/watch?v=ZiKMIuYidY0&t=4931s
- https://www.kaggle.com/c/donorschoose-application-screening
- http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html