Donor Application Screening

Introduction

- Donorchoose.org is an organisation which funds school level projects.
- Teachers from different schools request for the material they require for the students.
- Each year they receive thousands of applications for which they require lot of manpower and time to screen this applications.
- So, to make it more easier they want to automate this application screening process.

Objective

- The goal here is to predict whether the proposal would be approved or not based on the description provided in application.

- Use efficient algorithms for training the model for predicting accurately.

Dataset

- Dataset we worked on is taken 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.

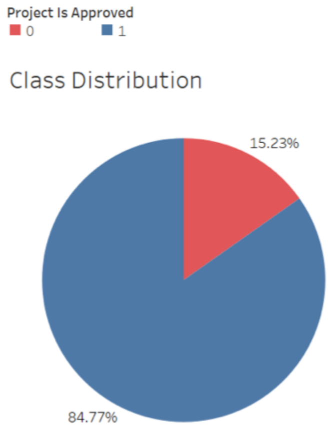
- Test.csv: test data for 78035 proposals and has 15 features.

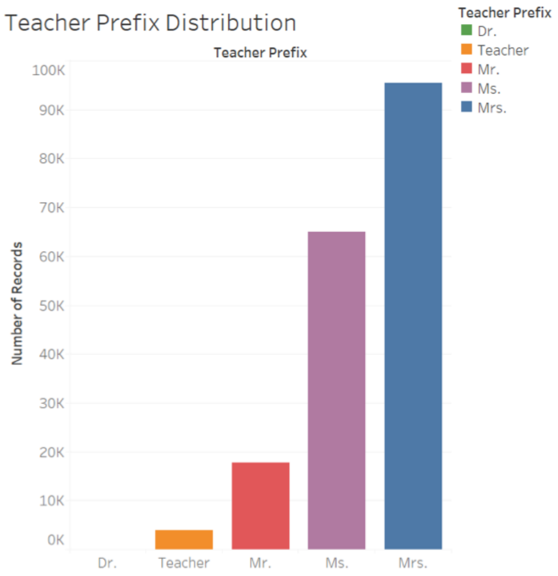
- Resource.csv: data of resources requested, their quantity and price. It has 1541272 entries and have 4 features.

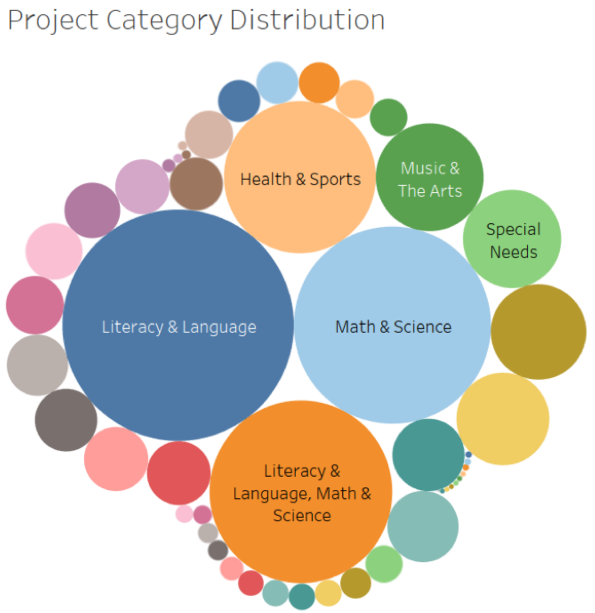
Technologies Used

- Anaconda Navigator for jupyter notebook
- Python as Programming Language
- Tableau for better visualization
- Libraries used: scipy, numpy, pandas, nltk, sklearn, matplotlib, plotly, seaborn

Visualization







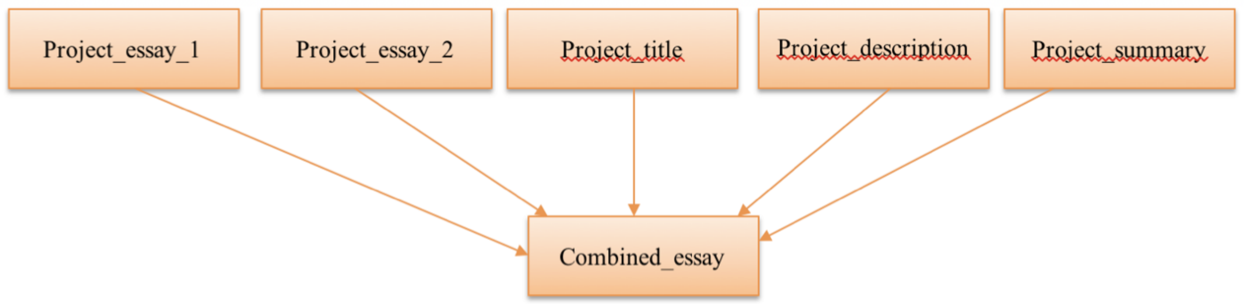
Data Preprocessing

- We merged train.csv and resource.csv based on id.
- Based on visualisation 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 some string.
- Applied Tfidf vectorization on the text features of data set for removing special characters, stop words, stemming and for generating tokens.

Data Preprocessing

Feature extraction:

- Created Text feature from using features



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

Data Preprocessing

Conversion of Categorical Features into Numeric Vectors

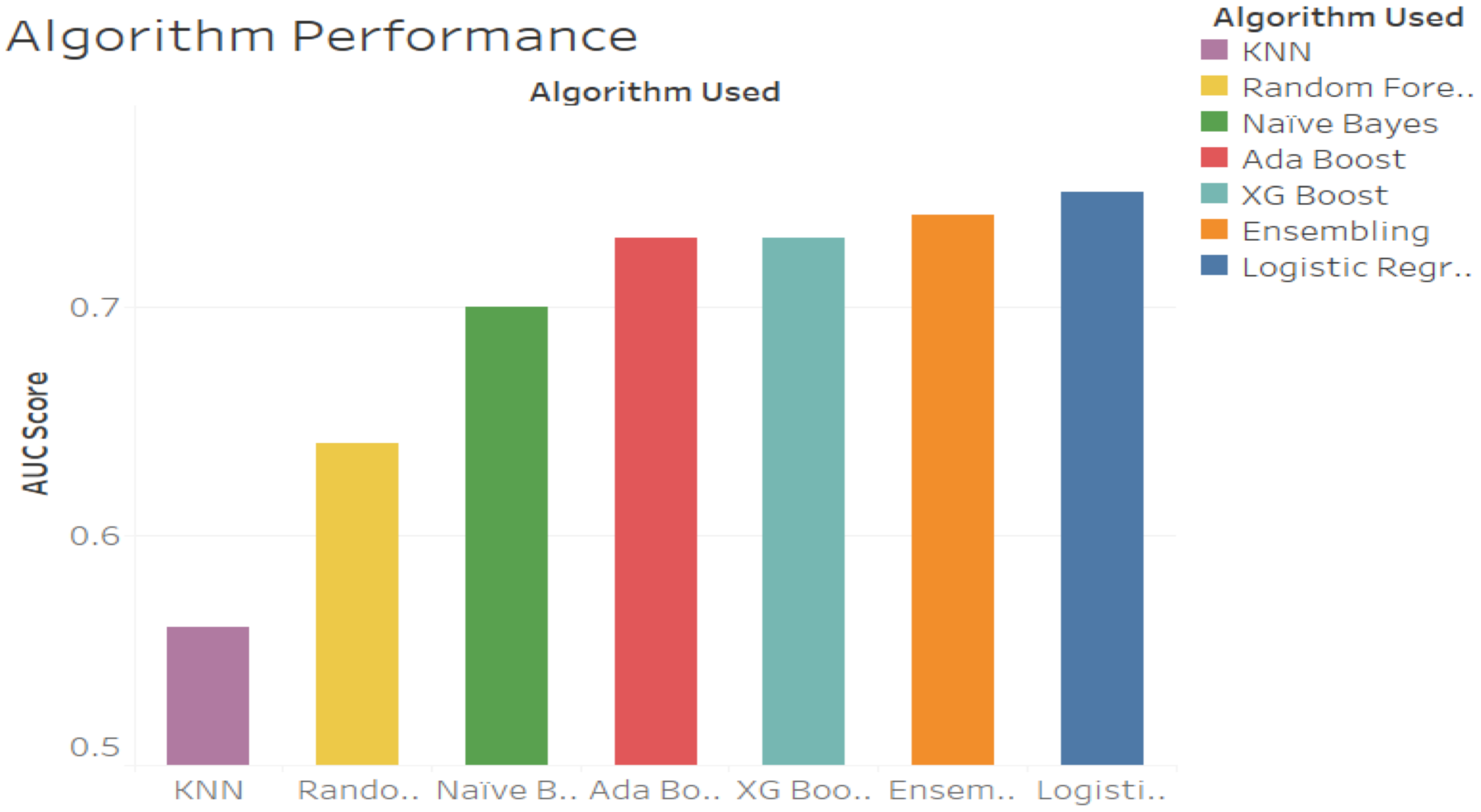
- school_state
- project_subject_categories
- project_subject_subcategories
- teacher_prefix

Feature Removal

- id
- project_title
- project_essay_3
- project_essay_4
- teacher_number_of_previously_posted_projects
- Categorical Features

Algorithms

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


Algorithm Performance
Algorithm Used
AUC Score
0.7
0.6
0.5
KNN
Rando..
Naïve B..
Ada Bo..
XG Boo..
Ensem..
Logisti..
Algorithm Used
KNN
Random Fore..
Naïve Bayes
Ada Boost
XG Boost
Ensembling
Logistic Regr..

Algorithms- worked for us

Based on the accuracy, selected the following algorithms for final model.

- Logistic regression
- Naive bayes
- Xgboost
- Random forest

Evaluation and Methodology

- We followed n-fold cross validation for value of n=5. Considered average as final validation score.
- We performed cross validation on all experimented algorithm and then selected the models which gave best accuracy.
- Ensemble learning : We used the combination of the algorithms which gave us best results by ensembling it.

Things that worked /Things that didn’t worked

- **Things that worked:**
- Referred kaggle’s beginner tutorial.
- All worked on preprocessing to make it faster and easier.
- Experimenting different models to get more accuracy for getting optimized results.

- **Things that didn’t work well:**
- Some algorithms were taking very long time to implement. Like SVM took a lot of time to implement.

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 works better than complex on some datasets.

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