

Machine Learning Engineer Nanodegree
Capstone Project

Bharadwaz Mahankali

July 11, 2018

Definition

Project Overview

DonorsChoose.org receives hundreds of thousands of project proposals each year for classroom projects in need of funding. A large number of volunteers is needed to manually screen each submission before it's approved to be posted on the DonorsChoose.org website.

Next year, DonorsChoose.org expects to receive close to 500,000 project proposals. As a result, there are three main problems that need to be solved:

1. How to scale current manual processes and resources to screen 500,000 projects so that they can be posted as quickly and as efficiently as possible.
2. How to increase the consistency of project vetting across different volunteers to improve the experience for teachers.
3. How to focus volunteer time on the applications that need the most assistance.

Problem Statement

Predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved, using the text of project descriptions as well as additional metadata about the project, teacher, and school. DonorsChoose.org can then use this information to identify projects most likely to need further review before approval.

This is a classification problem. Gradient Boosted Trees that perform best will be used for classification based on which decision whether proposal has to be approved can be made. Since essays written by teachers might play significant role in decision making, NLP techniques(provided by sklearn) will be used to effectively analyze essays and other text data. Goal is to bring Area under the ROC curve score, which is used for evaluation, closer to 1.

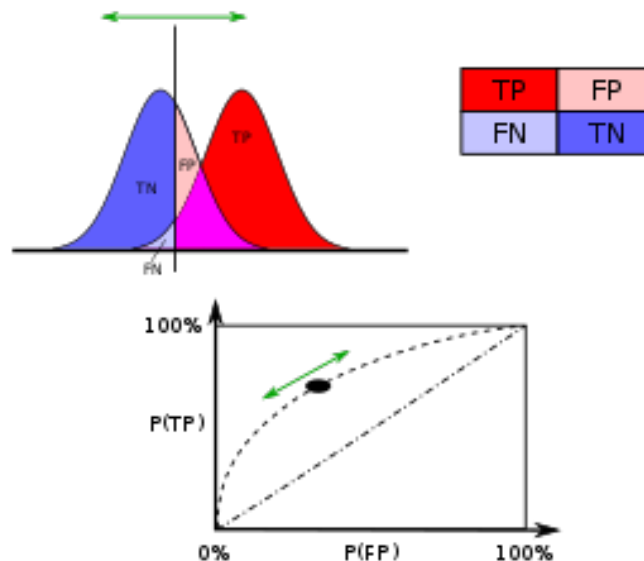
Metrics

Model will be evaluated on area under the ROC curve between the predicted probability and the observed target. Area under the ROC is used in classification analysis in order to determine which of the used models predicts the classes best. This problem boils down to a binary prediction. ROC curve is a very useful metric for such predictions as it is very easy to create confusion matrix which is used for calculating score.

- We predict 0 while we should have, the class is actually 0: this is called a **True Negative**.

- We predict 0 while we should have, the class is actually 1: this is called a **False Negative**.
- We predict 1 while we should have, the class is actually 0: this is called a **False Positive**.
- We predict 1 while we should have, the class is actually 1: this is called a **True Positive**.
- True positive rate (**TPR**), aka. sensitivity, hit rate, and recall, which is defined $TP / (TP + FN)$. Intuitively this metric corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points. In other words, the higher TPR, the fewer positive data points we will miss.
- False positive rate (**FPR**), aka. fall-out, which is defined as $FP / (FP + TN)$. Intuitively this metric corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points. In other words, the higher FPR, the more negative data points will be misclassified.

ROC curve is a plot of False positive rate (vs) True positive rate.



Analysis

Data Exploration

The competition dataset contains information from teachers' project applications to DonorsChoose.org including teacher attributes, school attributes, and the project proposals including application essays. Your objective is to predict whether or not a DonorsChoose.org project proposal submitted by a teacher will be approved.

File descriptions

train.csv - The training set.

test.csv - The test set.

resources.csv - Resources requested by each proposal; joins with test.csv and train.csv on id.

sample_submission.csv - A sample submission file in the correct format.

Data fields

test.csv and train.csv:

- id - Unique id of the project application.
- teacher_id - Id of the teacher submitting the application.
- teacher_prefix - Title of the teacher's name (Ms., Mr., etc.).
- school_state - US state of the teacher's school.
- project_submitted_datetime - Application submission timestamp.
- project_grade_category - School grade levels (PreK-2, 3-5, 6-8, and 9-12).
- project_subject_categories - Category of the project (e.g., "Music & The Arts").
- project_subject_subcategories - Sub-category of the project (e.g., "Visual Arts").
- project_title - Title of the project.
- project_essay_1 - First essay*.
- project_essay_2 - Second essay*.
- project_essay_3 - Third essay*.
- project_essay_4 - Fourth essay*.
- project_resource_summary - Summary of the resources needed for the project.
- teacher_number_of_previously_posted_projects - Number of previously posted applications by the submitting teacher.
- project_is_approved - Whether DonorsChoose proposal was accepted (0="rejected", 1="accepted"); (train.csv only).

* Note: Prior to May 17, 2016, the prompts for the essays were as follows:

- project_essay_1: Introduce us to your classroom.
- project_essay_2: Tell us more about your students.
- project_essay_3: Describe how your students will use the materials you're requesting.
- project_essay_4: Close by sharing why your project will make a difference.

Starting on May 17, 2016, the number of essays was reduced from 4 to 2, and the prompts for the first 2 essays were changed to the following:

- project_essay_1: Describe your students: What makes your students special? Specific details about their background, your neighborhood, and your school are all helpful.

- `project_essay_2`: About your project: How will these materials make a difference in your students' learning and improve their school lives?
- For all projects with `project_submitted_datetime` of 2016-05-17 and later, the values of `project_essay_3` and `project_essay_4` will be NaN.

resources.csv:

Proposals also include resources requested. Each project may include multiple requested resources. Each row in `resources.csv` corresponds to a resource, so multiple rows may tie to the same project by id.

- `id` - Unique id of the project application; joins with `test.csv` and `train.csv` on `id`.
- `description` - Description of the resource requested.
- `quantity` - Quantity of resource requested.
- `price` - Price of resource requested.

`train.head(3).transpose()`

	0	1	2
id	p036502	p039565	p233823
teacher_id	484aaf1125708 9a66cfedc9461 c6bd0a	df72a3ba80894 23fa8a94be880 60f6ed	a9b876a9252e08a55e3d89 4150f75ba3
teacher_prefix	Ms.	Mrs.	Ms.
school_state	NV	GA	UT
project_submitted_datetime	2016-11-18 14:45:59	2017-04-26 15:57:28	2017-01-01 22:57:44
project_grade_category	Grades PreK-2	Grades 3-5	Grades 3-5
project_subject_categories	Literacy & Language	Music & The Arts, Health & Sports	Math & Science, Literacy & Language
project_subject_subcategories	Literacy	Performing Arts, Team Sports	Applied Sciences, Literature & Writing
project_title	Super Sight Word Centers	Keep Calm and Dance On	Lets 3Doodle to Learn
project_essay_1	Most of my kindergarten students come from low...	Our elementary school is a culturally rich sch...	Hello;\r\nMy name is Mrs. Brotherton. I teach ...

project_essay_2	I currently have a differentiated sight word c...	We strive to provide our diverse population of...	We are looking to add some 3Doodler to our cla...
project_essay_3			
project_essay_4			
project_resource_summary	My students need 6 Ipod Nano's to create and d...	My students need matching shirts to wear for d...	My students need the 3doodler. We are an SEM s...
teacher_number_of_previously_posted_projects	26	1	5
project_is_approved	1	0	1
resources_total_mean	449.97	400	469.99
resources_quantity_count	2	1	1

`train.describe().transpose()`

	count	mean	std	min	25%	50%	75%	max
teacher_number_of_previously_posted_projects	182080.0	11.2370	28.0160	0.00000	0.000	2.00	9.0000	451.0
project_is_approved	182080.0	0.84768	0.35933	0.00000	1.000	1.00	1.0000	1.0
resources_total_mean	182080.0	257.936	423.186	1.75216	50.215	126.66	306.122	12344.0

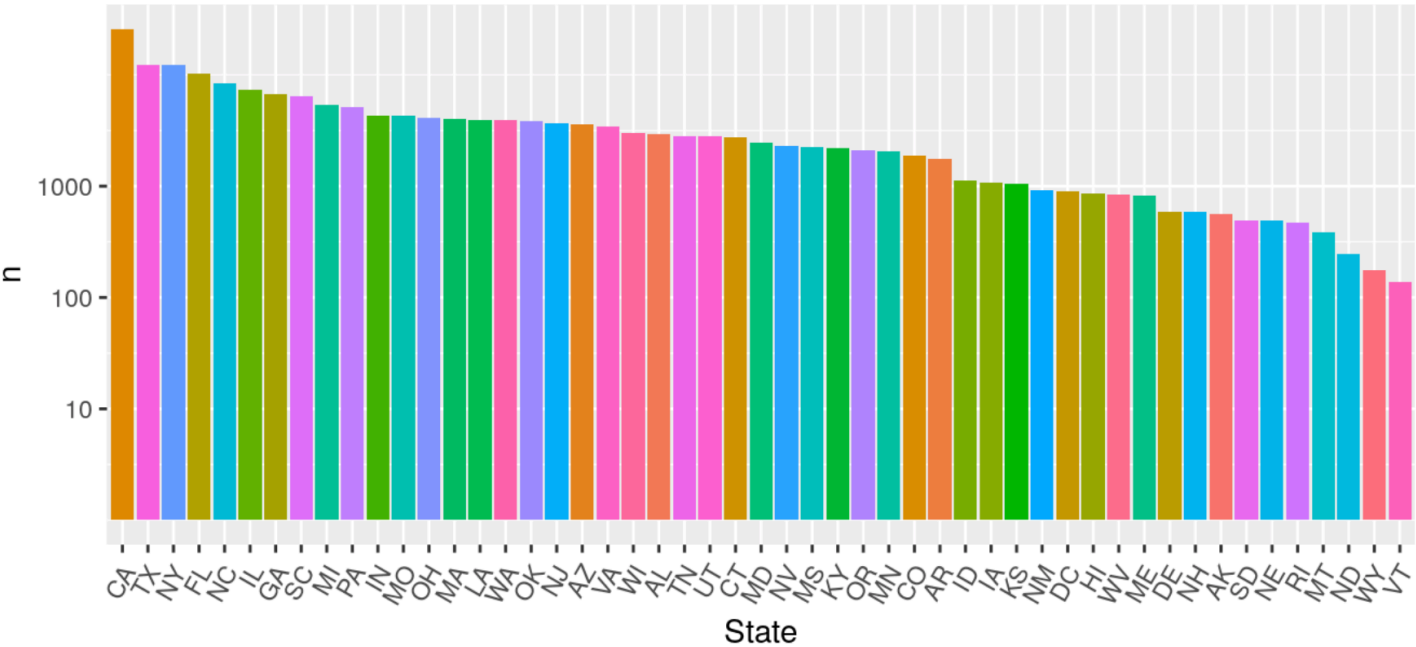
resources_quantity_count	182080.0	5.941505	8.349074	1.000000	1.000	3.00	7.0000	100.0
--------------------------	----------	----------	----------	----------	-------	------	--------	-------

```
print(train.shape, test.shape)
```

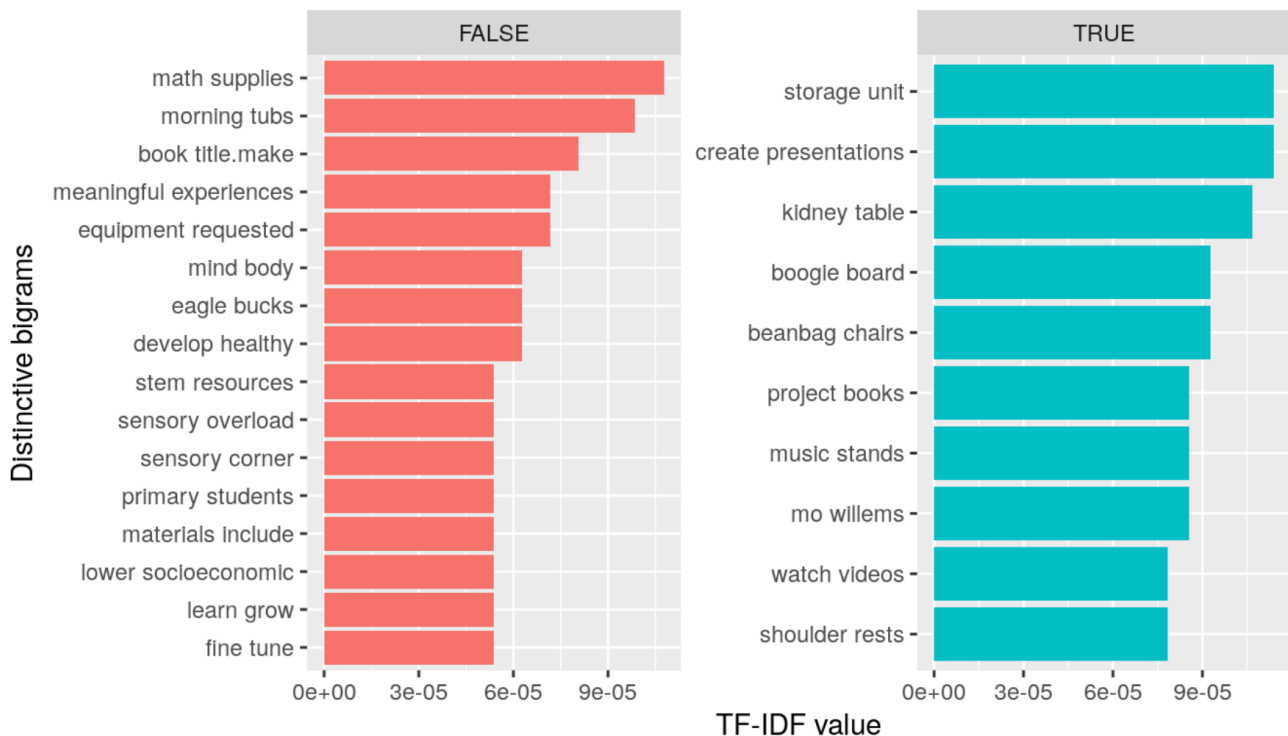
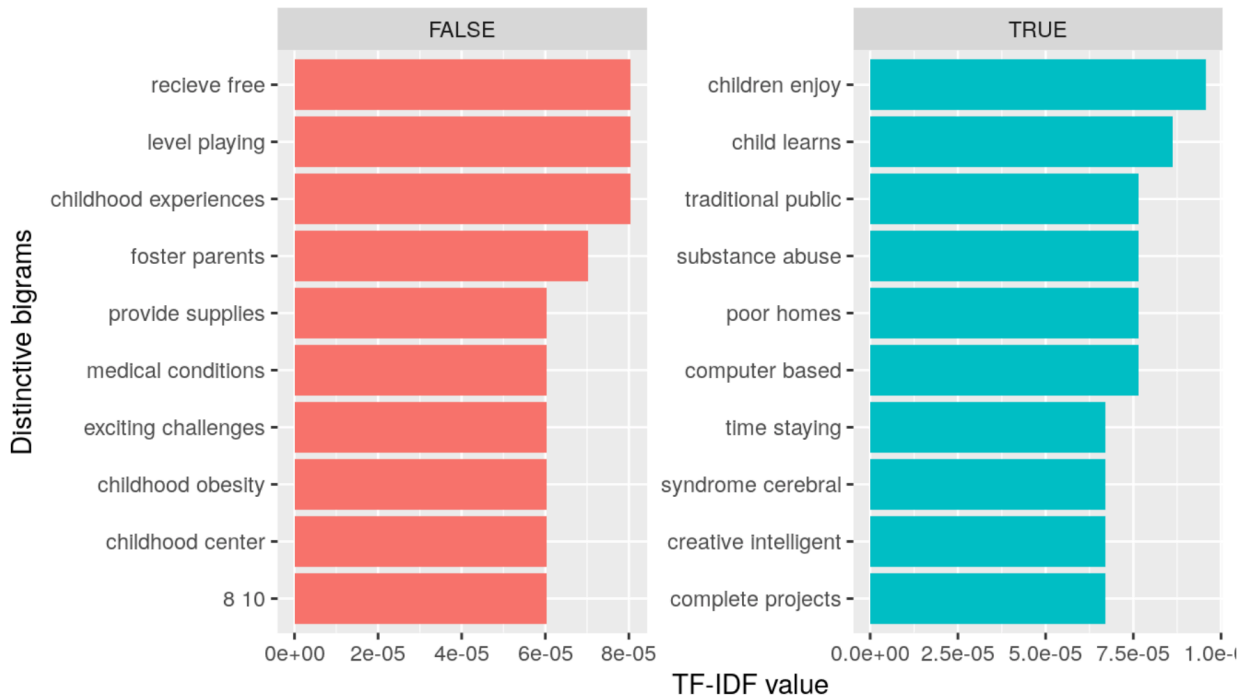
```
((182080, 18), (78035, 17))
```

Exploratory Visualization

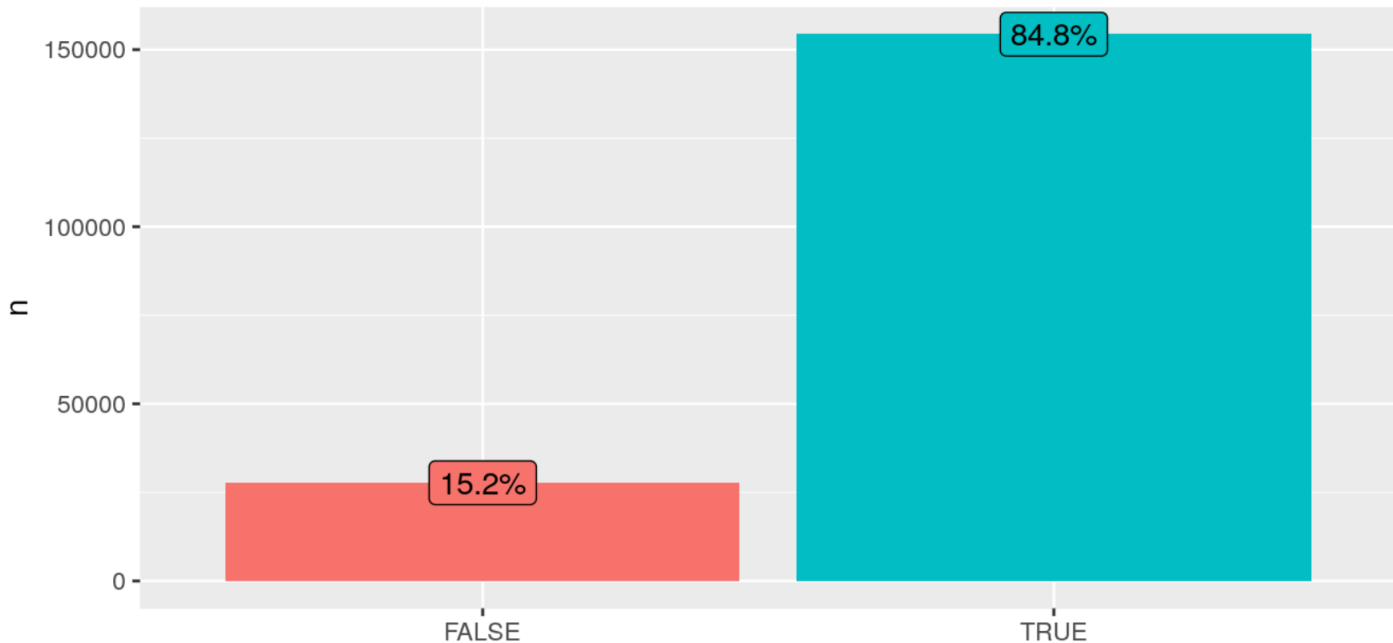
A plot with the number of submitted project proposals across the US states on a log-scale. As we can see, California dominates in the plot while Vermont stands at last place.



'project_essays' are most important features in our dataset that give lot of information about the outcome. Since Essays 1 and 2 are dominant we just consider these for analysis. Below plot shows top bigrams after performing TF-IDF on essay 1 and essay 2.



Below plot shows percentage of approvals and rejections. This is based on the target: *project_is_approved*. As we can see approval percentage is much higher (~85).

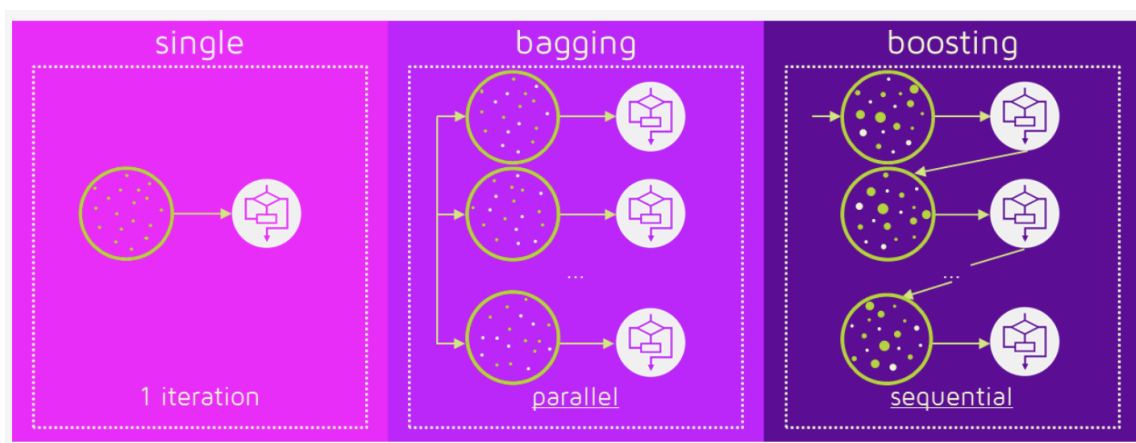


** Above visualizations are taken from Kaggle kernel - <https://www.kaggle.com/headsortails/an-educated-guess-donorschoose-eda>

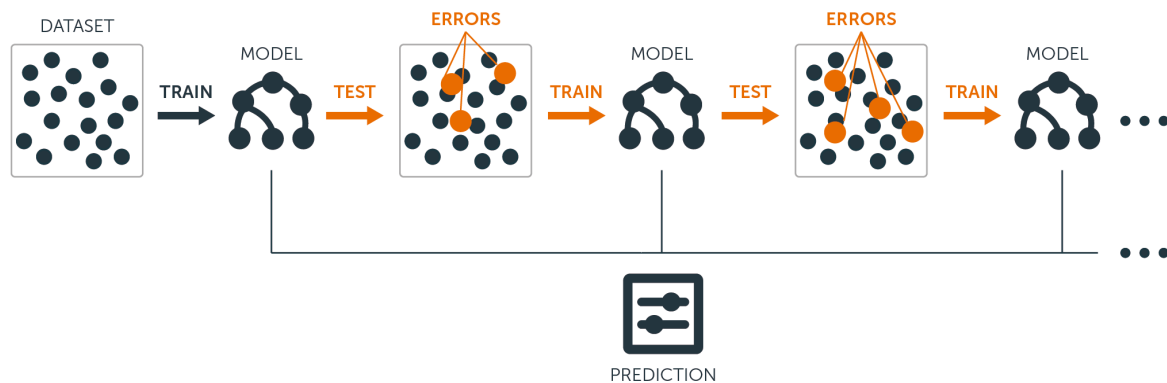
Algorithms and Techniques

Since this is a classification problem and we have multiple features, I choose to use gradient boosted trees to build learning model.

Boosting is an ensemble technique in which the predictors are not made independently, but sequentially. Below visualization shows the how various ensemble techniques.



Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. Below visualization shows how it works.



XGBoost is one of the fastest implementations of gradient boosted trees. It looks at the distribution of features across all data points in a leaf and uses this information to reduce the search space of possible feature splits. It also helps us achieve regularization in model.

Benchmark

Kaggle DonorsChoose.org Application Screening competition leaderboard score at the time of proposal submission was 0.79 (Area under receiver operating characteristic curve). By benchmark was to reach score that stands at 30% of public leaderboard, which was 0.74.

Methodology

Data Preprocessing

First, I have processed resource information that was given in resources.csv and added the result to training and testing sets.

Then, I have encoded 'teacher_id', 'teacher_prefix', 'school_state', 'project_grade_category', 'project_subject_categories', 'project_subject_subcategories' features using LabelEncoder.

Challenging part of the project is to process text. I choose to use TF-IDF for processing text and extract useful information out of it. So as a next step, I have processed

'project_title', 'project_resource_summary', 'project_essay_1', 'project_essay_2' features using TF-IDF algorithm and added the result to training and testing set.

Finally, I have removed the original features 'id', 'project_is_approved', 'project_resource_summary', 'project_title', 'project_essay_1', 'project_essay_2', 'project_essay_3', 'project_essay_4', 'project_submitted_datetime', 'project_subject_categories', 'project_subject_subcategories' from training and testing sets as we have already processed them and using transformed results.

Implementation

I have built a basic model to start with. Values that I used are: eta= 0.05, max_depth= 8, test_size=0.3. Training and validations scored for this base model are: train-auc=0.90123, valid-auc=0.739

```
x_train, x_test, y_train, y_test =  
model_selection.train_test_split(train[col], train['project_is_approved'], test_size=0.3,  
random_state=18)  
params = {'eta': 0.05, 'max_depth': 8, 'objective': 'binary:logistic', 'eval_metric': 'auc', 'seed': 18,  
'silent': True}  
watchlist = [(xgb.DMatrix(x_train, y_train), 'train'), (xgb.DMatrix(x_test, y_test), 'valid')]  
model = xgb.train(params, xgb.DMatrix(x_train, y_train), 450, watchlist, verbose_eval=10,  
early_stopping_rounds=20)
```

Refinement

I ran the model with various values by tuning manually. Below are the results:

```
eta: 0.1, max_depth: 8, test_size=0.3 -> train-auc:0.90394    valid-auc:0.738839  
eta: 0.1, max_depth: 7, test_size=0.3 -> train-auc:0.87082    valid-auc:0.738927  
eta: 0.1, max_depth: 7, test_size=0.25 -> train-auc:0.882395    valid-auc:0.742017  
eta: 0.1, max_depth: 7, test_size=0.25 -> train-auc:0.896722    valid-auc:0.74973  
eta: 0.1, max_depth: 6, test_size=0.25 -> train-auc:0.878008    valid-auc:0.752383
```

Results

Model Evaluation and Validation

The final model was built with values, eta: 0.1, max_depth: 6, test_size=0.25. Kaggle ROC score after submitting the result generated by this model is 0.75496. Training and validation scores were 0.878008 and 0.752383 respectively. As we can see, Kaggle score and my validation score are almost similar. We do see some overfitting during training.

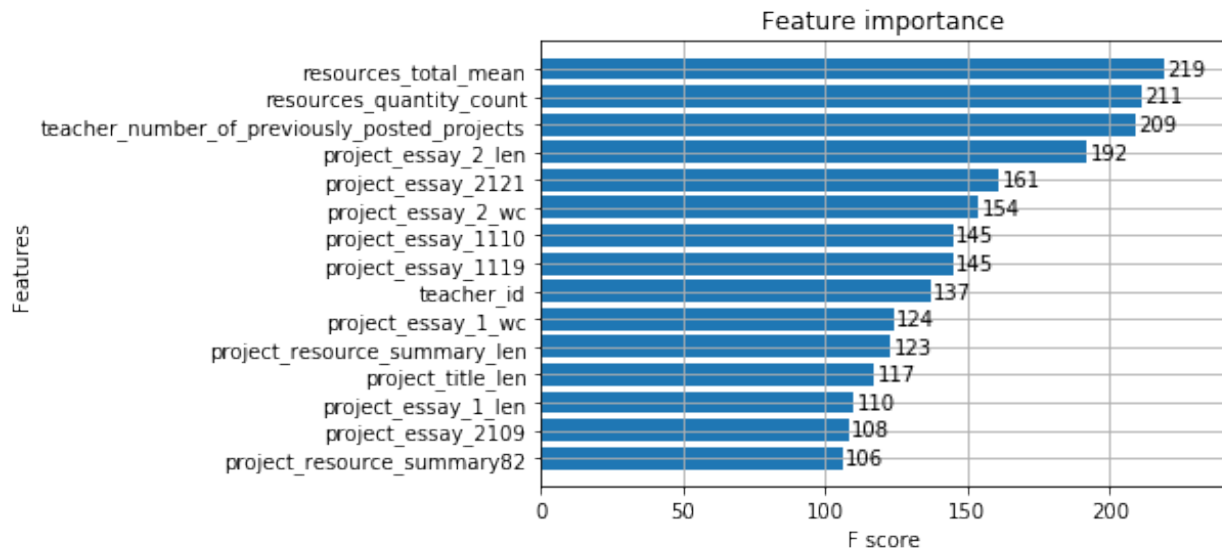
```
x_train, x_test, y_train, y_test =  
model_selection.train_test_split(train[col], train['project_is_approved'], test_size=0.25,  
random_state=18)  
params = {'eta': 0.1, 'max_depth': 6, 'objective': 'binary:logistic', 'eval_metric': 'auc', 'seed':  
18, 'silent': True}  
watchlist = [(xgb.DMatrix(x_train, y_train), 'train'), (xgb.DMatrix(x_test, y_test), 'valid')]  
model = xgb.train(params, xgb.DMatrix(x_train, y_train), 450, watchlist, verbose_eval=10,  
early_stopping_rounds=20)  
  
test['project_is_approved'] = model.predict(xgb.DMatrix(test[col]),  
ntree_limit=model.best_ntree_limit)  
test['project_is_approved'] = test['project_is_approved'].clip(0+1e12, 1-1e12)
```

Justification

Final model prediction score which is 0.75496 is better compared to the benchmark score of 0.74. Result can be further improved as discussed in conclusion.

Conclusion

Free-Form Visualization



Above visualization shows the feature importance of learned XGB model. We can see easily that project essay plays dominant role compared to all other features. It was very challenging to extract information from project essays for better predictions.

Reflection

I choose this problem to get a taste of NLP. Faced lot of challenges with text processing. Working on this made me realize how difficult NLP really is.

Using XGBoost has made it easy for me to train and evaluate result quickly. It was very easy to run multiple experiments as the training the model was fast.

I have completely ignored essay 3 and 4 in my model. As they looked less significant compared to other features.

Improvement

XGBoost performed well on this classification problem. We can further improve this by using GridSearchCV for parameter tuning to save time. Some advanced NLP techniques can be explored to extract more useful information from text during pre-processing.

Given more data, Deep Learning techniques can be used.