Machine Learning for Public Policy: HW3 fixed Final Report

The Prediction of Donor Choices using data from DonorsChoose.org

Ta-Yun Yang

1. Introduction

In most of the countries or states, besides the importance of culture, institutes and passion of instructors, budget is definitely one of the most important determinant to the quality of education. More and better teachers can be hired and teaching materials can be invested if the school or project fund is sufficient. School donations are one of the main focusing area for an online charity DonorsChoose.org, and the purpose of this analysis is to determine what are the school characteristics that is related to donor choices when they are selecting projects to fund.

2. Data

The data we are using contains information of the projects between 2012 to 2013 which includes the geographical and educational characteristics of a school and the dates when the projects are posted and funded. First of all, we define our outcome as whether a project will receive a donation in 60 days after it was posted. A project will be labelled as 1 if the project was not funded in 60 days, and as 0 otherwise. In the prediction, the project with higher probability to be classified as 1 should be considered as the projects which have higher risk not being funded in 60 days. Most of the school characteristics are saved as the features used in the training model, we only drop the variables which contains too many categories like ID to maintain the basic data size of each categories. The threshold here is set to be 51 to prevent dropping the state variable.

In order to increase the validity of the analysis, we use rolling window split as our strategy in the separation of training and testing datasets. The projects which are posted after 01/01/2012 and before 07/01/2012 will be defined as the first training set, followed by 60days gap for the outcome data to be collected and observed. We then define the half of year after the gap as the testing set. Projects of every half of the year will be added to the training sets, and the corresponding time interval of the testing subsets will be moved according to the end date of the training subset with 60days gap. In the end, we will have three combination of training and testing subsets.

Due to the fact that the features contain missing values which are not observed, we create a "unknown" category for categorical variables and impute mean for continuous variables in each combination of training and testing subsets. After scaling continuous variable and transform categorical variables to binary variables, the data is normalized and ready to be trained. Notice that the imputation, feature generation and training will be done with the training subsets respectively and

be passed to the corresponding testing subsets for analysis and transformation. We are not using data in testing subset for any training purpose.

3. Analysis

Multiple models and evaluation methods are included in the pipeline in making predictions from the data. Decision Tree, Random Forest, Logistic Regression, KNN, Naïve Bayes, Support Vector Machine, Bagging, Boosting, and Extra Trees are included in the used models. Accuracy, precision, recall, and area under sure are used as evaluation methods in this analysis. In this analysis, we want to know the performance of the prediction from the specific model and evaluation method, specifically we want to know the performance of the models which are targeting the 5% of the riskiest projects. In this case, we should observe the row of performance records where its thresholds are set to be 95 percentiles (where 5 percent of the outcome is classified as 1).

The performance table related to it is presented as the following using the largest training subset (the last combination of subset). Due to the time constraint, I only analysis the models with the highest priority, which includes random forest, decision tree and logistics regressions. For every model, only the combination of parameters which perform the best will be presented. We can observe from the following table that random forest models perform the best given our concern on the 5% riskiest population with evaluation of precision.

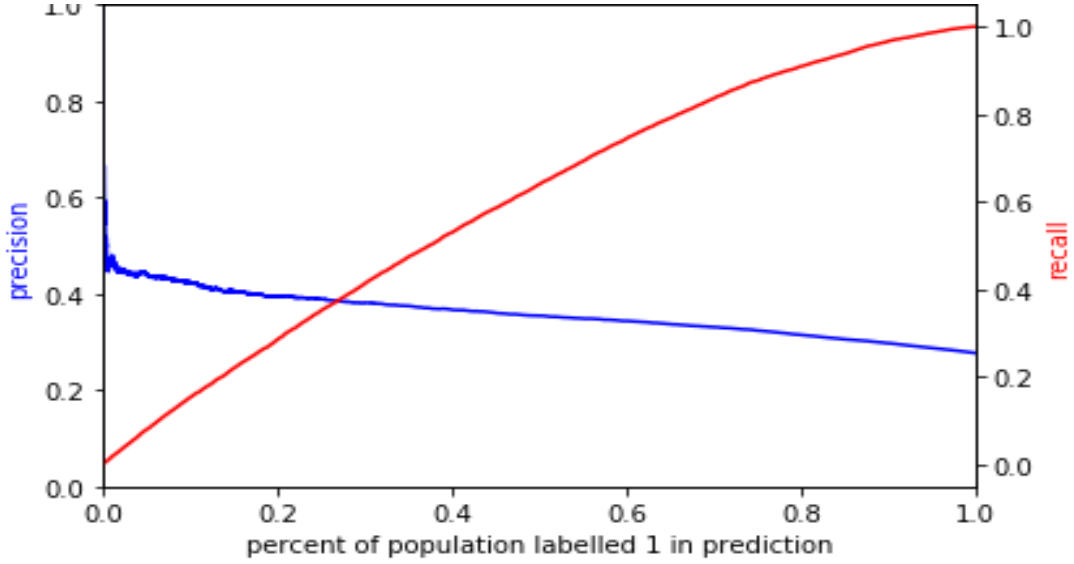
method	logistics
accuracy	0.715221
f1	0.129032
recall	0.0761655
precision	0.421818
AUC_ROC	0.564972
threshold	0.95
parameters	logistics with parameters : {'solver': 'warn',...
cross_validate_index	tmp_label2

method	decision_tree
accuracy	0.713433
f1	0.117592
recall	0.0689429
precision	0.399493
AUC_ROC	0.648682
threshold	0.95
parameters	decision_tree with parameters : {'min_samples_...
cross validate index	tmp label2

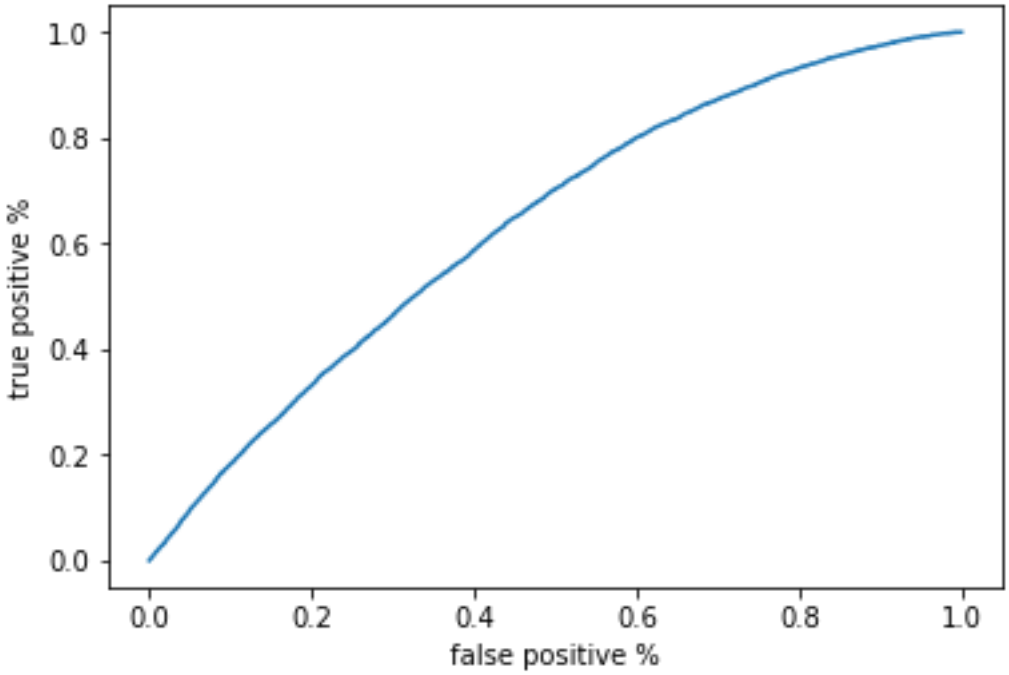
method	random_forest
accuracy	0.718525
f1	0.134241
recall	0.0787919
precision	0.453115
AUC_ROC	0.531754
threshold	0.95
parameters	random_forest with parameters : {'min_samples_...
cross_validate_index	tmp_label2

We care about the precision in this case since we want care about how accurate we are in predicting the riskiest projects which are potentially not funded from all the projects we predicted to be risky. I

will suggest the deployment of random forest models with the corresponding parameters since it not only have the highest precision, but also higher recall comparing to other models in the given threshold. We can have better efficiency and coverage by using this method. In the precision and recall curve presented below, we can observe that if we classified all the projects to be risky, the baseline precision will be close to 0.2. Our prediction with the riskiest 5% population contains precision with higher than 0.4 which provides the improvement of the base prediction. The natural caveat of all the models will be the recall which is lower than 0.2, which means that although we have higher efficiency in predicting, we only cover less than 20% of projects that really need helps.



The following ROC curve depict the smooth positive correlation between true positive and false positive rate draw from the random forest model.



We then check if the performance of these methods are valid over time, and we can discover that

from the time table below, there is some fluctuation when we are training across different training subsets. However, in general precision of all the models are still higher than 0.4 in all of the case.

	method	accuracy	f1	recall	precision	AUC_ROC	threshold	parameters
63	random_forest	0.730881	0.133578	0.079604	0.414868	0.638209	0.95	random_forest with parameters : {'min_samples_...
133	random_forest	0.681820	0.128485	0.074438	0.469027	0.638187	0.95	random_forest with parameters : {'min_samples_...
203	random_forest	0.717039	0.134594	0.079448	0.440000	0.636547	0.95	random_forest with parameters : {'min_samples_...

We then move on to the training of the full grid to validate the previous conclusion. We can discover that the best performance in extra tree, adaboost, bagging, KNN and SVM are clearly worse than the best performance in random forest. We can observe the high performance in precision in gradient boosting methods, but its f1 score and recall are significantly worse than other case. The selection of random forest model will be preserved since it provides balance between efficiency and coverage with relatively good performance comparing to other models.

```
method                                                        ET
accuracy                                                0.707038
f1                                                      0.104004
recall                                                  0.061392
precision                                                   0.34
AUC_ROC                                                 0.563977
threshold                                                   0.95
parameters            ET with parameters : {'n_jobs': -1, 'class_wei...
cross_validate_index                                  tmp_label2
```

```
method                                                  adaboost
accuracy                                                0.713584
f1                                                      0.124027
recall                                                 0.0732108
precision                                               0.405455
AUC_ROC                                                 0.634503
threshold                                                   0.95
parameters            adaboost with parameters : {'n_estimators': 10...
cross_validate_index                                  tmp_label2
```

```
method                                               gradientboost
accuracy                                                0.723131
f1                                                     0.00283812
recall                                                 0.00142263
precision                                               0.565217
AUC_ROC                                                  0.52255
threshold                                                   0.95
parameters            gradientboost with parameters : {'n_estimators...
cross_validate_index                                  tmp_label2
```

```
method                                                    bagging
accuracy                                                0.714615
f1                                                      0.127178
recall                                                 0.0750711
precision                                               0.415758
AUC_ROC                                                 0.566985
threshold                                                   0.95
parameters            bagging with parameters : {'base_estimator__C'...
cross_validate_index                                  tmp_label2
```

method	KNN
accuracy	0.707886
f1	0.1066
recall	0.0629241
precision	0.348485
AUC_ROC	0.544768
threshold	0.95
parameters	KNN with parameters : {'n_neighbors': 10, 'lea...
cross_validate_index	tmp_label2

method	SVM
accuracy	0.702431
f1	0.0899147
recall	0.0530751
precision	0.293939
AUC_ROC	0.503502
threshold	0.95
parameters	SVM with parameters : {'decision_function_shap...
cross_validate_index	tmp_label2

Name: 0, dtype: object