CAPP 30254: Machine Learning for Public Policy
Rachel Ker

Predicting projects on DonorsChoose that will not be fully funded within 60 days of posting

This report uses data from Kaggle about projects on DonorsChoose from 2012 to 2013 to predict project that will not be fully funded within 60 days of posting. 969 classifier models were built to model this prediction in total, including k-nearest neighbors, decision trees, support vector machines, logistic regressions, and random forest, boosting and bagging classifiers. Table 1 in Appendix shows the different parameters used for the classifiers build.

Validation Methodology

These classifiers were built only on the training set data and then validated on the testing set. The data was split into 3 training and testing sets by time period with the number of observations in each set specified below, with the last 60 days in the training set only providing labels to prevent data leakage:

	Training Set	**Training set size**	**Testing Set**	**Testing set size**
1	Jan 2012 - Jun 2012	21,423	Jul 2012 – Dec 2012	32,676
2	Jan 2012 – Dec 2012	50,100	Jan 2013 – Jun 2013	21,585
3	Jan 2012 – Jun 2013	74,359	Jun 2013 – Dec 2013	43,836

Variables Used

The following are the variables used to build the classifier models, before discretization and generation of dummy variables:

- State the school is in
- Whether the school is in urban, rural or suburban area
- Whether the school is a charter school
- Whether the school is a magnet
- Teacher’s prefix (Ms, Mrs, Mr)
- Main subject for which project materials are intended
- Main subject area for which project materials are intended
- Secondary subject for which project materials are intended
- Secondary subject area for which project materials are intended
- Type of resource requested by a project
- School’s poverty level (measured by percentage of free and reduced lunch)
- Grade level for which project materials are intended
- Project cost including optional tip
- Number of students impacted by a project if funded
- Whether a project was eligible for a 50% off offer by a corporate partner

In this analysis, discretization was done by dividing number of students impacted by a project and project cost into 3 equal bins in the train set and the same bins were used to discretize the test set. Missing values were naively replaced with their mode since the variables are categorical variables and the descriptive statistics of the other variables were not sufficient to give more information about the value that should be imputed. Training set data were imputed with the mode of that of the training set only, and the testing set was imputed with the mode of the testing set only. The total number of missing observations for each variable is given in Table 2 in Appendix. Dummy variables were also created on the train set and the same dummy variables were generated in the test set. Any continuous variables would be scaled to a value between 0 and 1.

Model Performance

The metrics we evaluated the models on were precision and recall at 1%, 2%, 5%, 10%, 20%, 30%, thresholds and the area under the curve. All models were also compared to the baseline metrics when the model simply just predicts all observations as positives. Results of the best models for each metric are presented in Table 3 in the Appendix. The baseline for each train-test sets are as follows:

Train-test Set	**1 (Jul-12)**	**2 (Jan-13)**	**3 (Jul-13)**
Baselines	25.68%	31.49%	28.48%

Overall, there was no model that performed consistently well through all three train-test sets. The linear classifier models (i.e. logistic regression and support vector machines) did well in the first two train-test sets and tree ensemble classifiers (i.e. random forests and extra trees) on the last two train-test sets. All of the best models for each metric also did better than the baseline.

To recommend a model to identify 5% of the posted projects to intervene would depend on goal of the model. If we are looking to maximize recall or precision, I would recommend using either the **Logistic Regression Model (LR) (c=1, penalty=l2)** or **Extra Trees Classifier (ET) (criterion: entropy, max_depth=10, max_features=sqrt, min_samples_split=50, n_estimators=100)** for deployment and field testing. The precision-recall threshold curves for each of these models can be seen below in Figures 1 and 2.

- The LR model has the highest precision at 5% overall in the dataset with a precision of 50.8% and a recall of 8.1%. This means that among the top 5% of the projects predicted not to be fully funded within 60 days of posting 50.8% of the projects are actually not funded within 60 days, and 8.1% of these projects that actually did not get funded would be identified in this prediction set.

- The ET model has the highest recall at 5% overall in the dataset with a precision of 47.4% and a recall of 8.3%. This means that among the top 5% of the projects predicted not to be fully funded within 60 days of posting 47.4% of the projects are actually not funded within 60 days, and 8.3% of these projects that actually did not get funded would be identified in this prediction set.

Figure 1: Precision-recall on threshold curves for Logistic Regression Model (LR) (c=1, penalty=l2**)** over the 3 training set – (left) training set 1, (center) training set 2, (right) training set 3

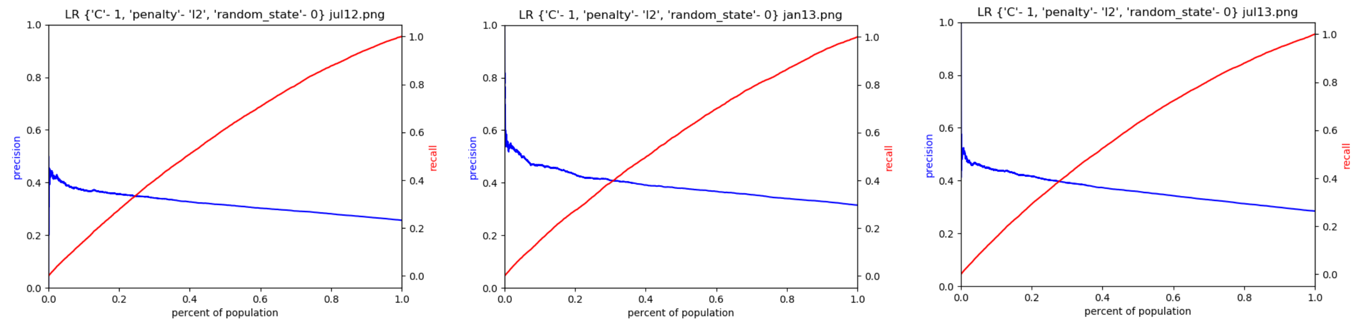
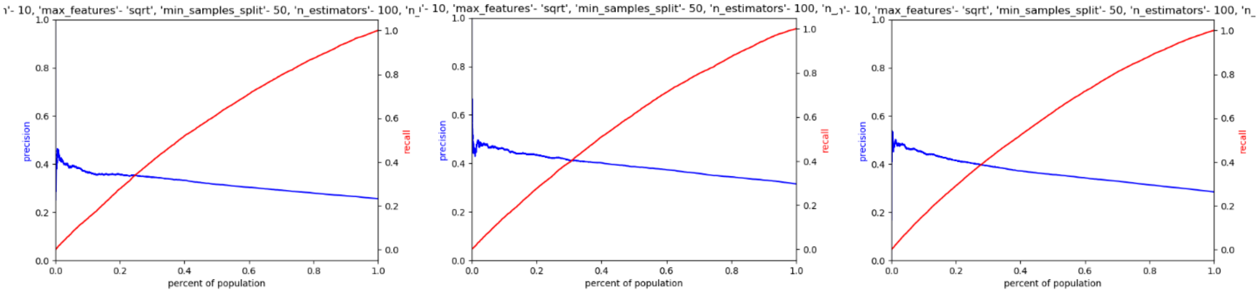


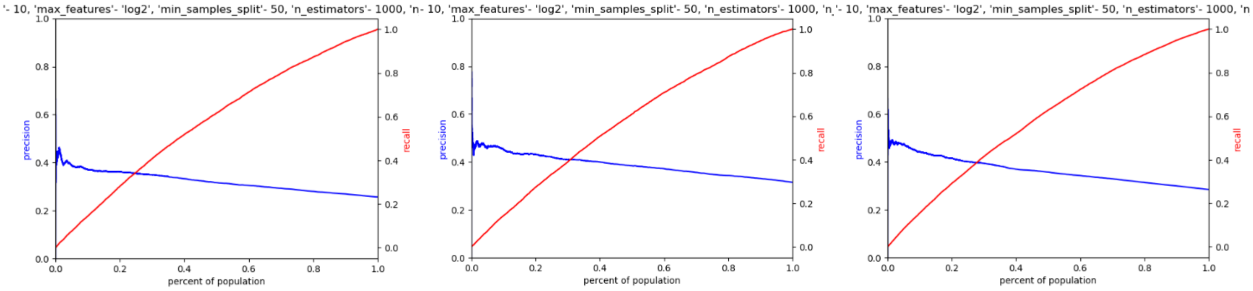
Figure 2: Precision-recall on threshold curves for Extra Trees Classifier (ET) (criterion: entropy, max_depth=10, max_features=sqrt, min_samples_split=50, n_estimators=100) over the 3 training set – (left) training set 1, (center) training set 2, (right) training set 3



The choice between the two models depends on whether precision or recall is of a higher priority in the goal. If the cost of expanding resources on a project that would actually be fully funded in 60 days but is predicted to not be fully funded is calculated to be higher than the cost of not detecting a project that would not be fully funded in 60 days (i.e. maximize for precision), the linear regression classifier of the specified parameters should be chosen. If it is the reverse (cost of not detecting a project that would not be fully funded is higher so the goal is to maximize recall), then extra trees classifier of the specified parameters should be chosen.

However, the classifiers recommended above are not consistently the best in precision and recall in all 3 training and testing set. Hence if we are looking for the most stable model with high recall and precision, I would instead recommend using the **Extra Trees Classifier (ET) (criterion: entropy, max_depth=10, max_features=log2, min_samples_split=50, n_estimators=1000)** for deployment and field testing. Sorting the models by best precision and recall at the threshold of 5% and getting their average rank, this ET classifier tops the chart for both precision and recall at 5% with an average rank of 16.7, average precision at 5% of 44.1% and average recall at 5% of 7.7% over all three train test sets. This means that among the 5% of projects predicted to be not funded in 60 days by the model, on average 44.1% of the projects are actually not funded within the 60 days and 7.7% of the projects that are actually not funded within 60 days were identified among this prediction set. The precision-recall threshold curve this model can be seen in Figure 3.

Figure 3: Precision-recall on threshold curves for Extra Trees Classifier (ET) (criterion: entropy, max_depth=10, max_features=log2, min_samples_split=50, n_estimators=1000) over the 3 training set – (left) training set 1, (center) training set 2, (right) training set 3



In conclusion, the different models above have their merits and which model to recommend depends on what metric we are optimizing on, specifically in our recommendations we consider precision, recall or stability. Figures 4 and 5 below plots these different models for precision and recall at the threshold at 5% for comparison over the different train test sets for informed decision.

Figure 4: Precision at 5% over the 3 train test sets for selected models

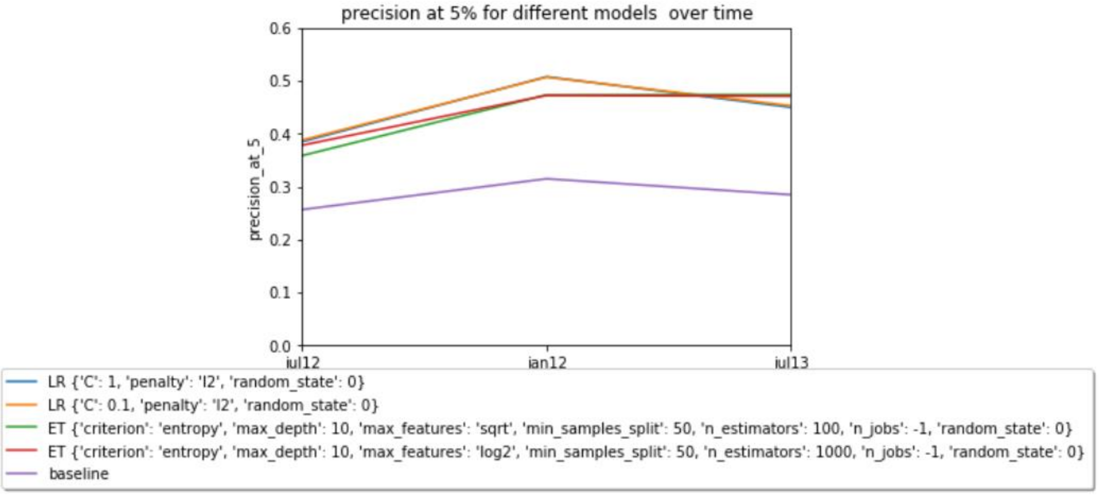
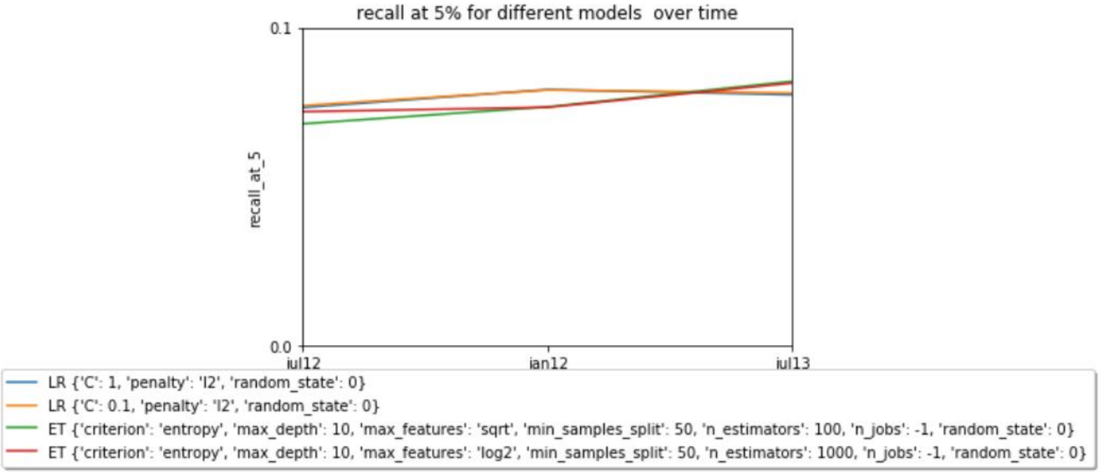


Figure 5: Recall at 5% over the 3 train test sets for selected models



Appendix

Table 1: Parameters used for the classifiers

	K-nearest neighbors Classifier	Decision Tree Classifier	Support Vector Machine	Logistic Regression	Random Forest Classifier	Extra Trees Classifier	Gradient Boosting Classifier	AdaBoost Classifier	Bagging Classifier	Naïve Bayes
n_neighbors	1,10,25,50									
weights	uniform, distance									
algorithm	auto, ball_tree, kd_tree							SAMME, SAMME.R		
criterion		gini, entropy				gini, entropy				
max_depth		1,5,10,20			1,5,10,20	1,5,10,20	1,5,10,20			
max_features					sqrt, log2	sqrt, log2				
min_samples_split		2, 10, 50			2, 10, 50	2, 10, 50				
c			0.01, 0.1, 1, 10	0.01, 0.1, 1, 10						
penalty				l1, l2						
n_estimators					10, 100, 1000	10, 100, 1000	10, 100	10, 100	10, 100, 1000	
n_jobs					-1	-1			-1	
learning_rate							0.001, 0.01			
subsample							0.1, 0.5			
alpha										1,10
fit_prior										True, False

Appendix

Table 2: Table of number of missing observations for each variable in the data

	index	missing count
0	projectid	0
1	teacher_acctid	0
2	schoolid	0
3	school_ncesid	9233
4	school_latitude	0
5	school_longitude	0
6	school_city	0
7	school_state	0
8	school_metro	15224
9	school_district	172
10	school_county	0
11	school_charter	0
12	school_magnet	0
13	teacher_prefix	0
14	primary_focus_subject	15
15	primary_focus_area	15
16	secondary_focus_subject	40556
17	secondary_focus_area	40556
18	resource_type	17
19	poverty_level	0
20	grade_level	3
21	total_price_including_optional_support	0
22	students_reached	59
23	eligible_double_your_impact_match	0
24	date_posted	0
25	datefullyfunded	0
26	60daysafterpost	0
27	notfullyfundedin60days	0

Appendix

Table 3: Best models by metric and train-test set

traintest sets	metric	model	parameters	score
Jul-12	precision_at_1	AB	{'algorithm': 'SAMME.R', 'n_estimators': 10, 'random_state': 0}	0.478528
Jul-12	precision_at_2	RF	{'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.424196
Jul-12	precision_at_5	LR	{'C': 0.1, 'penalty': 'l2', 'random_state': 0}	0.388242
Jul-12	precision_at_10	LR	{'C': 0.1, 'penalty': 'l1', 'random_state': 0}	0.370064
Jul-12	precision_at_20	RF	{'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.359296
Jul-12	precision_at_30	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.345134
Jul-12	recall_at_1	AB	{'algorithm': 'SAMME.R', 'n_estimators': 10, 'random_state': 0}	0.018594
Jul-12	recall_at_2	RF	{'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.033015
Jul-12	recall_at_5	LR	{'C': 0.1, 'penalty': 'l2', 'random_state': 0}	0.075566
Jul-12	recall_at_10	LR	{'C': 0.1, 'penalty': 'l1', 'random_state': 0}	0.1441
Jul-12	recall_at_20	RF	{'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.279857
Jul-12	recall_at_30	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.403218
Jul-12	auc	LR	{'C': 0.01, 'penalty': 'l2', 'random_state': 0}	0.606524
Jan-13	precision_at_1	SVM	{'C': 0.1, 'random_state': 0}	0.562791
Jan-13	precision_at_2	SVM	{'C': 0.1, 'random_state': 0}	0.524362
Jan-13	precision_at_2	SVM	{'C': 1, 'random_state': 0}	0.524362
Jan-13	precision_at_5	LR	{'C': 1, 'penalty': 'l2', 'random_state': 0}	0.507878
Jan-13	precision_at_10	LR	{'C': 10, 'penalty': 'l1', 'random_state': 0}	0.464319
Jan-13	precision_at_20	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 10, 'n_jobs': -1, 'random_state': 0}	0.435488
Jan-13	precision_at_30	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.417143
Jan-13	recall_at_1	SVM	{'C': 0.1, 'random_state': 0}	0.017799
Jan-13	recall_at_2	SVM	{'C': 0.1, 'random_state': 0}	0.033245
Jan-13	recall_at_2	SVM	{'C': 1, 'random_state': 0}	0.033245
Jan-13	recall_at_5	LR	{'C': 1, 'penalty': 'l2', 'random_state': 0}	0.080612

Appendix

Jan-13	recall_at_10	LR	{'C': 10, 'penalty': 'l1', 'random_state': 0}	0.147396
Jan-13	recall_at_20	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 10, 'n_jobs': -1, 'random_state': 0}	0.276552
Jan-13	recall_at_30	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.397323
Jan-13	auc	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.611666
Jul-13	precision_at_1	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.511416
Jul-13	precision_at_1	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.511416
Jul-13	precision_at_1	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.511416
Jul-13	precision_at_2	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.494292
Jul-13	precision_at_5	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.474213
Jul-13	precision_at_10	RF	{'max_depth': 20, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.44992
Jul-13	precision_at_20	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.422722
Jul-13	precision_at_30	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.39635
Jul-13	recall_at_1	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.01794
Jul-13	recall_at_1	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.01794
Jul-13	recall_at_1	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.01794
Jul-13	recall_at_2	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.034679

Appendix

Jul-13	recall_at_5	ET	{'criterion': 'entropy', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.083213
Jul-13	recall_at_10	RF	{'max_depth': 20, 'max_features': 'log2', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.157937
Jul-13	recall_at_20	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.296812
Jul-13	recall_at_30	ET	{'criterion': 'entropy', 'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 10, 'n_estimators': 100, 'n_jobs': -1, 'random_state': 0}	0.417428
Jul-13	auc	RF	{'max_depth': 20, 'max_features': 'sqrt', 'min_samples_split': 50, 'n_estimators': 1000, 'n_jobs': -1, 'random_state': 0}	0.62698