Comparison of the Outcome of Different Machine Learning Classifiers: Using the Datasets on School Funding Requests

Introduction.

Our client, which is a school district administration, is interested in the question that what types of schools, as well as what kinds of funding requests, are likely to get the funding within 60 days. They provide us with a full dataset on school funding requests, which consists of information on school name, school location, requested resource type, the poverty level of the area where the school is in, request date, date when the school gets funding, etc.

Our approach.

The data is from January 1st, 2012 to Dec 31st, 2013. As it is a time-series data, we decide to use temporal validation which creates training and testing datasets over time. As a result, we have three training-testing pairs. For the machine learning models, we include bagging classifier, random forest classifier, logistic regression classifier, support vector machine, gradient boosting classifier, decision tree classifier, k-nearest neighbor classifier, and Naïve Bayes model. Moreover, in order to guarantee the objectivity of our studies, we also use different parameters for each model we use; for instance, in logistic regression classifier models, the regularization methods include Lasso Regression (or 'L1' regularization) and Ridge Regression (or 'L2' regularization), so we tried both regularization parameters in our studies. We choose 'school_metro', 'school_charter', 'school_magnet', 'primary_focus_area', 'secondary_focus_area', 'resource_type', 'poverty_level', 'grade_level', 'total_price_including_optional_support', 'students_reached', 'eligible_double_your_impact_match' to be the features in our machine learning models.

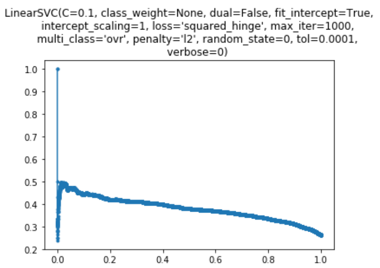
Findings.

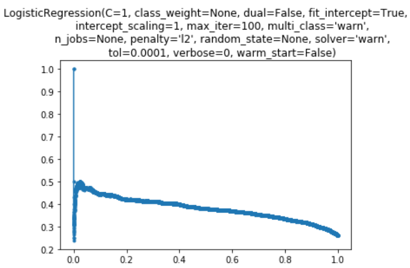
In our first test (training set: data from 2012/01/01 to 2012/06/30; testing set: data from 2012/09/01 to 2013/02/18), our auc-roc scores range from 0.538 to 0.667. The situations are similar in the second test and third test.

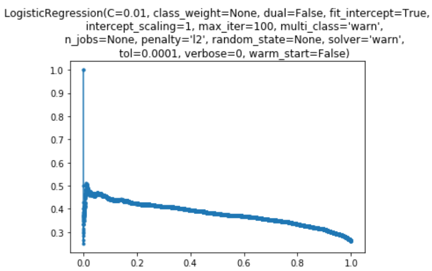
The result of the first test (the first 10 records sorted by auc-roc in descending order):

	model_type	clf	parameters	auc-roc	p_at_5	p_at_10	p_at_20	r_at_5	r_at_10	r_at_20	f1_at_5	f1_at_10	f1_at_20
39	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 1, 'penalty': 'l2'}	0.666884	0.44997	0.432858	0.410517	0.086309	0.166104	0.31511	0.144837	0.240081	0.356541
35	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 0.01, 'penalty': 'l2'}	0.666869	0.451183	0.430737	0.409911	0.086542	0.16529	0.314645	0.145227	0.238904	0.356015
38	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 1, 'penalty': 'l1'}	0.666868	0.450576	0.432555	0.41082	0.086425	0.165988	0.315343	0.145032	0.239913	0.356804
37	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 0.1, 'penalty': 'l2'}	0.666852	0.44997	0.433162	0.410668	0.086309	0.166221	0.315226	0.144837	0.240249	0.356673
34	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 0.01, 'penalty': 'l1'}	0.666734	0.449363	0.428615	0.405364	0.086193	0.164476	0.311155	0.144642	0.237727	0.352066
36	SVM	LinearSVC(C=1, class_weight=None, dual=False, ...	{'C': 0.1, 'penalty': 'l1'}	0.666687	0.44997	0.433162	0.409456	0.086309	0.166221	0.314296	0.144837	0.240249	0.35562
29	LR	LogisticRegression(C=10, class_weight=None, du...	{'C': 0.1, 'penalty': 'l2'}	0.666601	0.451789	0.431343	0.409001	0.086658	0.165523	0.313947	0.145423	0.23924	0.355225
28	LR	LogisticRegression(C=10, class_weight=None, du...	{'C': 0.1, 'penalty': 'l1'}	0.666588	0.450576	0.431949	0.409304	0.086425	0.165755	0.314179	0.145032	0.239576	0.355488
32	LR	LogisticRegression(C=10, class_weight=None, du...	{'C': 10, 'penalty': 'l1'}	0.666553	0.446938	0.432555	0.409759	0.085728	0.165988	0.314528	0.143861	0.239913	0.355883
33	LR	LogisticRegression(C=10, class_weight=None, du...	{'C': 10, 'penalty': 'l2'}	0.666524	0.447544	0.431949	0.409608	0.085844	0.165755	0.314412	0.144056	0.239576	0.355752
31	LR	LogisticRegression(C=10, class_weight=No	{'C': 1, 'penalty': 'l2'}	0.666497	0.44997	0.432252	0.409608	0.086309	0.165872	0.314412	0.144837	0.239744	0.355752

The precision-recall curves of the SVM model and Logistic Regression models in the first test:







Conclusion.

To conclude, SVM and Logistic Regression models perform well in the first test, while the Gradient Boosting model and Random Forest model do a better job in the second and third tests. In the first test, even if the auc-roc scores are relatively high, which is approximately 0.67, it does not necessarily indicate that our models do a good job in prediction. If you look at the precisions and recalls at 5% level, all of them are below 0.6. It means that the possibility that the model successfully predicts if a school receives the funding that they want within 60 days is below 60%, which does not meet our client goals of “identifying 5% of the posted projects to intervene with”. The results might indicate that our assumption on the features might not reflect the real-life scenarios.

Recommendations.

In order to build better machine learning models, we recommend our client to provide us with raw data that contains more useful information, such as the teacher's qualifications, the school's ratings within its school district, the donor's wealth, etc. More information would be useful for our analysis. Also, as for the models that we could go forward with and deploy, we would not suggest Logistic Regression Classifier. If we incorporate more features into our machine learning models, the decision boundaries might not be linear; the Logistic Regression might underperform in such situations. The models that we would recommend include Random Forest Classifier and Gradient Boosting, because they are not only performing slightly better in our studies, but also robust to large and noisy training data.