Predicting Serious Credit Delinquencies

Leith McIndewar

ML Pipeline Improvement
May 6, 2016

1. Model Fitting

I used the GridSearchCV function to build classification models. This function generates all possible combinations of the parameter values in the grid and returns the model with the best scores. The best parameters are determined using, by default, a 3-fold cross validation.

2. Evaluation Metrics

I ran the following classifiers: K Nearest Neighbors, Decision Tree, SGD (including linear SVM), Logistic Regression, Naive Bayes, Ada Boosting, Random Forest, and Gradient Boosting.

The models all returned very close accuracy scores. Of the methods tested here, all of the ensemble models show very good precision and significantly better recall. GradientBoosting, AdaBoost, and RandomForest all have recall measures over 0.15 while none of the other models are above 0.05. The ensembles also have among the highest levels of precision of any of the models. These combined give the ensembles F1 scores that are five to six times as high as the other models.

The biggest draw back of the ensemble models is the time needed to train the models. GradientBoosting in particular took just over an hour (one hour and six minutes) to train. This was true despite reducing the range of parameters that were tested while fitting GradientBoost and RandomForest. During this implementation I only timed how long it took to fit the model and not the time to make predictions on the test set. Despite a very low train time for

Metric	**LR**	**AB**	**NB**	**DT**	**SGD**	**KNN**	**RF**	**GB**
Precision	0.583	0.537	0.349	0.286	0.066	0.557	0.587	0.576
Recall	0.039	0.186	0.020	0.001	0.008	0.016	0.157	0.201

Table 1.

Metric	**Run 1**	**Run 2**
Accuracy	0.933	0.933
AUC	0.500	0.512
Precision	0.285	0.559
Recall	0.001	0.261
F1 Score	0.002	0.050

Table 2. Metrics from two DecisionTrees.

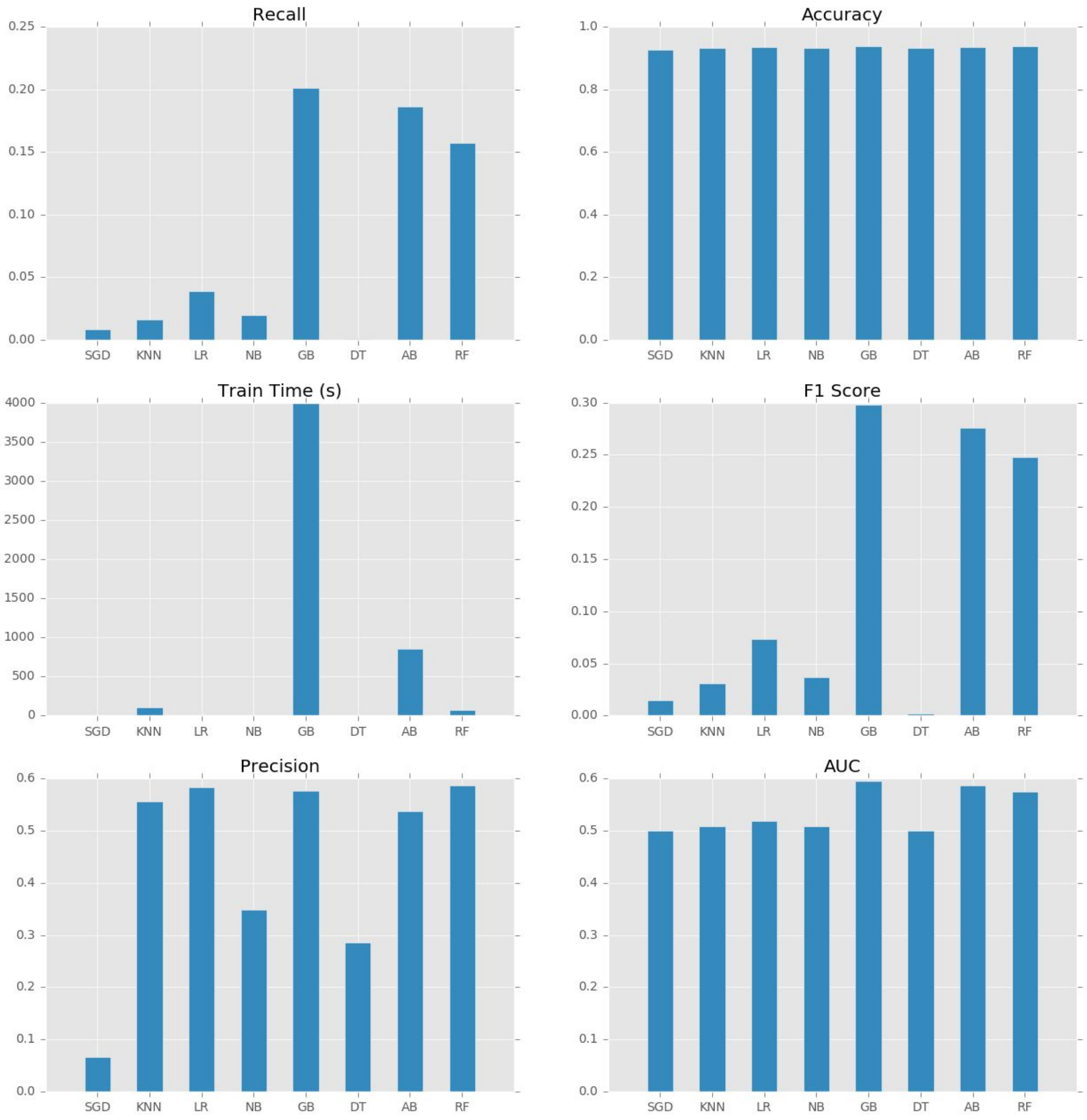
RandomForest; when included in the models to run list I found that execution took much longer. I will add a prediction time to the pipeline as that is another important consideration for some problems.

Note: I found Decision Trees to be very unstable. In the iteration shown in the comparison graphs, the Decision Tree did very poorly. I found in other instances the model to be significantly better on some of the metrics.

3. Conclusion

I would strongly recommend using one of the ensemble methods for this problem. In this case, GradientBoost provides enough of an advantage in recall to be the best choice. Recall should be favored above precision: it is more important to identify true positives (those who would experience a serious delinquency) than to have a high precision. Given the problem of predicting credit delinquencies, neither the train or prediction times for these models is a crucial consideration. A credit model would presumably only need to be updated periodically with new data. Integrating additional information and rerunning these models could become an issue as the dataset grew, but for now it does not pose a problem. Likewise, because credit decisions probably do not need to be made in a matter of seconds or even minutes, the train times observed here are not a problem.

Created with Madoko.net.


Recall
0.25
0.20
0.15
0.10
0.05
0.00
SGD
KNN
LR
NB
GB
DT
AB
RF
Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
SGD
KNN
LR
NB
GB
DT
AB
RF
Train Time (s)
4000
3500
3000
2500
2000
1500
1000
500
0
SGD
KNN
LR
NB
GB
DT
AB
RF
F1 Score
0.30
0.25
0.20
0.15
0.10
0.05
0.00
SGD
KNN
LR
NB
GB
DT
AB
RF
Precision
0.6
0.5
0.4
0.3
0.2
0.1
0.0
SGD
KNN
LR
NB
GB
DT
AB
RF
AUC
0.6
0.5
0.4
0.3
0.2
0.1
0.0
SGD
KNN
LR
NB
GB
DT
AB
RF