Report

February 1, 2016

1 Project 2: Supervised Learning

Building a Student Intervention System

1.1 Classification vs Regression

Identifying students as needing intervention or not is a boolean decision with two discrete output labels making this a classification problem. A regression version would be to predict the dropout rate (a continuous variable) for a school given parameters on the school body.

1.2 Exploring the Data

- Total number of students: 395
- Number of students who passed: 265
- Number of students who failed: 130
- Number of features: 30
- Graduation rate of the class: 67.09%

1.3 Preparing the Data

1.3.1 Split data into training and test sets

Training set: 300 samples Test set: 95 samples

1.4 Training and Evaluating Models

1.4.1 Model 1 - K-Nearest Neighbors Classification

K-Nearest neighbors (KNN) is an instance-based learning mechanism that rather than finding an equation to classify data will instead base a classification on an average of the closest data points to it.

This has the strength that processing time for predictions can be very fast because only a small number K datapoints are required to make a prediction, not an analysis of the entire datbase. However, this means that all of the data have to be stored so that any prediction can be made.

This could be useful to model at-risk students because there are certainly many types of students but their classification may be complex, therefore rather than categorizing all students, we can base classification on just those students that are most similar.

Table of computation time and F1 statistics:

Training Size	Training Time (secs)	Prediction Time (secs)	F1 Train	F1 Test
100	0.0008	0.0017	0.8987	0.6457
200	0.0007	0.0022	0.8814	0.7328
300	0.001	0.0026	0.8951	0.736

1.4.2 Model 2 - Decision Tree Classification

A decision tree classifier learns a set of if-then-else rules to follow based on input parameters to predict data labels.

Speed is a strength of these models in that the time costs are logarithmic compared to others that are expnential or worse. However, it is possible to overfit the data and have a complex tree that does not generalize well. Decision tree algorithms can also generate very different trees given slight variations in training samples leading to multiple potential models.

A decision tree could be useful for this data because students of a certain type will likely share characteristics which a decision tree algorithm can find and group accordingly.

Table of computation time and F1 statistics:

Training Size	Training Time (secs)	Prediction Time (secs)	F1 Train	F1 Test
100	0.0006	0.0002	0.8444	0.6667
200	0.0009	0.0002	0.878	0.736
300	0.0014	0.0002	0.8593	0.6379

1.4.3 Model 3 - Support Vector Machine

Support vector classification (SVC) aims to find the ‘spaces’ between high dimensional data clouds that best separate labeled categories. It is a particularly useful method in high dimensional spaces where a decision boundary, also of high dimensionality, may be otherwise difficult to find. A downside to SVC is that models may not generalize well when trained on a sparsely filled feature space, i.e., it suffers greatly from the curse of dimensionality.

Table of computation time and F1 statistics:

Training Size	Training Time (secs)	Prediction Time (secs)	F1 Train	F1 Test
100	0.0012	0.0008	0.8193	0.805
200	0.0031	0.0011	0.8606	0.8101
300	0.0063	0.0017	0.8547	0.8267

1.5 Choosing the Best Model

Decision Tree classification is the fastest method available, in part due to its simplicity, which would minimize implementation costs. However its F1 scores for test data are the worst among the three models. KNN has a marginally lower training time requirement, because it is basically only storing information, however, this comes with the cost of high memory requirements as the model has to store the entire dataset and not not simply a method to predict labels for new data. SVC is performs even slower as it has more high-dimension matrix operations to perform.

Given the limited resources, the decision tree model will make a reasonable prediction with the least amount of processing time. This saves time and money which could be put towards helping those students identified as needing intervention. The SVC model does perform quite well with only 100 training samples and with only moderate increases in computation time required. However, the tuned SVC model (included in the python notebook) actually performs worse with cross validation which lends more support for a decision tree model.

The final decision tree will be generated by succesively dividing students using factors, e.g. gender, such that the two groups come closest to resembling a pass/no-pass split. The process will continue for each resulting subset until there are no factors that can divide students into groups of increasing uniformity. Other constraints can be used to influence the ‘shape’ of the tree and its ability to generalize. For example, the model is tuned by testing varying numbers of features that can be used, the minimum number of students that are required in a subset, and how many total subsets are allowed. These all affect how well the model can generalize to new data because it it easy to create a decision tree with training data that has a single student in a group as a result of many, many splits, but fails to generalize to test data because the paths to classification are too specific and only match the students used to train the model. This ability to generalize is captured in the F1 statistic which balances the models ability to correctly classify and not misclassify.

To then make a prediction, all that is needed is to follow the path in the decision tree mandated by how each factor is split. The final group will have a predicted label of either passing or not passing.

Tuning the model using a GridSearch allows many models to be computed using various levels of several parameters to find the best setting based on the model's F1 score. Each decision in the tree effectively adds a layer of depth and complexity. We can prevent overfitting by imposing a limit on the Maximum Depth of decisions (2 through 6 for this analysis). This forces the tree to stop growing in a way that only benefits the training set and and makes wrong predictions on test data.

The F1 score for the final decision tree is 0.7597 which is an improvement from 0.736 in the best initial model. This is on par with the initial F1 test scores of the KNN model and within range of the computationaly intensive SVC model (slightly worse than the initial sample size trials, but better than the tuned model).