Thomas Hepner
2/19/2015
Build a Student Intervention System

1) Classification vs Regression

The goal is to identify students who might need early intervention. This supervised machine learning problem is a classification problem as we are predicting whether each student either might needs early intervention or not. We should attempt to solve this problem as a classification task since our target prediction for each student is a binary, not numerical outcome.

2) Exploring the Data

- **Total number of students:** 395
- **Number of students who passed:** 265
- **Number of students who failed:** 130
- **Graduation rate of the class (%):** 67.09%
- **Number of features (excluding the label/target column):** 30

3) Preparing the Data

- Identify feature and target columns
- Preprocess feature columns
- Split data into training and test sets

See attached iPython notebook for each step.

4) Training and Evaluating Models

3 supervised learning models were chosen from scikit-learn. They were **(A) KNeighborsClassifier, (B) DecisionTreeClassifier, and (C) SVC.**

(A) KNeighborsClassifier:

- **What are the general applications of this model? What are its strengths and weaknesses?**

K-Nearest Neighbors classification is a simple algorithm that works well for basic classification problems. Its strengths are (1) no computational cost in the training phase, (2) robust to noisy data, and (3) is an effective classifier if the size of the training data is large. Its weaknesses are that (1) it is a *lazy learner,* meaning that the model doesn't learn in the training phase so prediction is computationally expensive, (2) the model cannot be interpreted since no concepts are learned while training, and (3) its performance depends on the dimensionality of the underlying data.

- **Given what you know about the data so far, why did you choose this model to apply?**

I chose this model because it is (1) a classification model, (2) simple and computed quickly, and (3) determines whether or a student needs early intervention based on their *proximity* or *similarity* to other students that needed early intervention; in other words, the model intuitively makes sense.

- **Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant. Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.**

Model = KNN	Training Time (ms)	Prediction Time (ms)	F1 Training Score	F1 Test Score
N = 100	0.58522	1.52295	0.84	0.76
N = 200	0.47411	1.5401	0.84	0.76
N = 300	0.43418	1.40851	0.84	0.76

(B) DecisionTreeClassifier:

- What are the general applications of this model? What are its strengths and weaknesses?

The model is used to model a decision tree from data. Its strengths are (1) easy to visualize and interpret, and (2) very fast computation time even as data analyzed becomes larger, and (3) it is a white box model where any given observation's prediction can easily be explained with Boolean logic.

Its weaknesses are (1) that it is prone to overfitting, and (2) a globally optimal solution is not guaranteed.

- Given what you know about the data so far, why did you choose this model to apply?

I chose this model because (1) it is a classification model, and (2) it would be really easy to explain the results and actions to take to management.

- Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant. Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

Model = Decision Tree	Training Time (ms)	Prediction Time (ms)	F1 Training Score	F1 Test Score
N = 100	1.67143	0.16794	1.0	0.7
N = 200	1.39827	0.10522	1.0	0.69
N = 300	1.42643	0.11469	1.0	0.64

(C) SVC:

- What are the general applications of this model? What are its strengths and weaknesses?

Support Vector Classifiers are used to build models that assign new data into one category or another. Its strengths are (1) its use of a mathematical function, *or kernel,* to create nonlinear decision boundaries to classify data, and (2) its superior predictive performance compared to most other classification algorithms. Its weaknesses are (1) it is incredibly computationally expensive, and (2) choosing an appropriate kernel is challenging and overfitting the data is easy.

- Given what you know about the data so far, why did you choose this model to apply?

I chose to apply this model because of its superior predictive ability for many classification tasks.

- Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant. Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

Model = Support Vector Classifier	Training Time (ms)	Prediction Time (ms)	F1 Training Score	F1 Test Score
N = 100	5.14484	1.32941	0.86	0.8
N = 200	4.97511	1.28461	0.86	0.8
N = 300	4.95258	1.27104	0.86	0.8

5) Choosing the Best Model

- Based on the experiments you performed earlier, in 2-3 paragraphs explain to the board of supervisors what single model you choose as the best model. Which model has the best test F1 score and time efficiency? Which model is generally the most appropriate based on the available data, limited resources, cost, and performance? Please directly compare and contrast the numerical values recorded to make your case.

I chose the Support Vector Classifier, SVC as the single best model. Its test F1 score was greater than the other models regardless of the training set size (100, 200, or 300 data points) while maintaining a very fast prediction time for every model constructed. The SVC model constructed with 300 data points in the training data had an F1 test set score of 0.8 compared to 0.64 for the DecisionTreeClassifier and 0.76 for the KNearestNeighbors model.

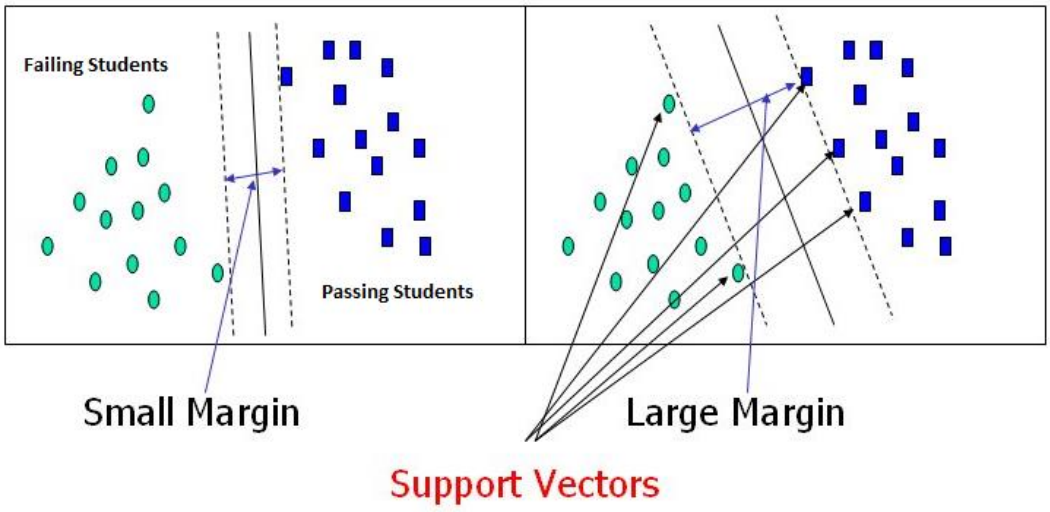
Model	N = 100	N = 200	N = 300
KNN	0.76	0.76	0.76
Decision Tree	0.7	0.69	0.64
Support Vector Classifier (SVC)	0.8	0.8	0.8

Using the full training set of 300 data points, the SVC model was constructed in 4.95 milliseconds compared to 1.43 milliseconds for the DecisionTreeClassifier and 0.43 milliseconds for the KNearestNeighbors model. The prediction time of the SVC model was 1.27 milliseconds compared to 0.11 for the DecisionTreeClassifier and 1.41 for the KNearestNeighbors model. The time differences between these models are very small and almost negligible for our performance and cost considerations.

The SVC model is clearly the best model choice given that its F1 score is substantially better than the other classifiers and is still able to construct the model in an almost negligible amount of time.

- In 1-3 paragraphs explain to the board of supervisors in layman's terms how the final model chosen is supposed to work (for example if you chose a decision tree or support vector machine, how does it learn to make a prediction).

The final model, a Support Vector Classifier, is constructed by using the characteristics of prior students such as gender, age, access to internet, absences from school, etc.; this is called the *training data*. The model separates between the students who passed and failed with boundaries. See below:



The model creates *decision* boundaries in the data that maximize the *margin, or* distance in the data, between the failing students and the passing students. These boundaries are simply lines that separate the two categories of students. In the next phase, the prediction phase, the model takes data about the students who we want to know whether they are likely to pass or not. The model uses the characteristics of these new students, and applies the same boundaries as created before, to determine whether a student is likely to pass or fail.

This algorithmic approach tends to yield predictive results that have superior predictive performance compared to other models.

- Fine-tune the model. Use gridsearch with at least one important parameter tuned and with at least 3 settings. Use the entire training set for this. What is the model's final F1 score?

After tuning with gridsearch, the model's final F1 score was 0.85 on the training set, and 0.81 on the test set.