Thomas Hepner
2/16/2015
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):** 31

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?**

This model uses a vote of the *k*-nearest neighbors for classification predictions. Its strengths are its simplicity and quick computation time. Its weakness is that the model is highly sensitive to the local structure of the 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.**

```
----------------------------------------
Training set size: 100
Training KNeighborsClassifier...
Done!
Training time (secs): 0.000
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.001
F1 score for training set: 0.820512820513
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.811188811189
----------------------------------------
Training set size: 200
Training KNeighborsClassifier...
Done!
Training time (secs): 0.000
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.002
F1 score for training set: 0.812949640288
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.651515151515
----------------------------------------
Training set size: 300
Training KNeighborsClassifier...
Done!
Training time (secs): 0.001
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.004
F1 score for training set: 0.815347721823
Predicting labels using KNeighborsClassifier...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.619047619048
```

(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.

```
------------------------------------------
Training set size: 100
Training DecisionTreeClassifier...
Done!
Training time (secs): 0.001
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.000
F1 score for training set: 1.0
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.000
F1 score for test set: 0.586206896552
------------------------------------------
Training set size: 200
Training DecisionTreeClassifier...
Done!
Training time (secs): 0.002
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.001
F1 score for training set: 1.0
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.000
F1 score for test set: 0.691176470588
------------------------------------------
Training set size: 300
Training DecisionTreeClassifier...
Done!
Training time (secs): 0.001
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.001
F1 score for training set: 1.0
Predicting labels using DecisionTreeClassifier...
Done!
Prediction time (secs): 0.000
F1 score for test set: 0.625
```

(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) fast computation time, and (2) superior predictive performance compared to most other classification algorithms. Its greatest weakness is its interpretability as it is a black box model.

- 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.

```
------------------------------------------
Training set size: 100
Training SVC...
Done!
Training time (secs): 0.001
Predicting labels using SVC...
Done!
Prediction time (secs): 0.001
F1 score for training set: 0.805369127517
Predicting labels using SVC...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.822784810127
------------------------------------------
Training set size: 200
Training SVC...
Done!
Training time (secs): 0.003
Predicting labels using SVC...
Done!
Prediction time (secs): 0.002
F1 score for training set: 0.819875776398
Predicting labels using SVC...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.805031446541
------------------------------------------
Training set size: 300
Training SVC...
Done!
Training time (secs): 0.006
Predicting labels using SVC...
Done!
Prediction time (secs): 0.004
F1 score for training set: 0.804828973843
Predicting labels using SVC...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.8125
```

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 about 0.2 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 0.8125 compared to 0.625 for the DecisionTreeClassifier and 0.619 for the KNearestNeighbors model.

Using the full training set of 300 data points, the SVC model was constructed in 0.004 seconds compared to 0.001 seconds for the DecisionTreeClassifier and 0.004 seconds for the KNearestNeighbors model. The prediction time of the SVC model was 0.001 seconds compared to 0.000 for the DecisionTreeClassifier and 0.001 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 SVC model learns to make predictions by taking input features or variables and transforming them into a greater number of features by applying a mathematical function, referred to as a *kernel.* This method is commonly referred to as the *kernel trick.* By doing this, the algorithm is better able to distinguish boundaries between our classes or categories, in this case whether a student needs early intervention or not. This algorithmic approach typically yields predictive results that are much more accurate than those produced by other models, but at the expense of interpretability of the model.

- 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.

```
----------------------------------------
Training set size: 300
Training GridSearchCV...
Done!
Training time (secs): 39.622
Predicting labels using GridSearchCV...
Done!
Prediction time (secs): 0.002
F1 score for training set: 0.851528384279
Predicting labels using GridSearchCV...
Done!
Prediction time (secs): 0.001
F1 score for test set: 0.807947019868
```