James Kelly
Machine Learning Nanodegree P2
Student Interventions Classifier

1. Regression based machine learning makes a prediction that is typically a continuous numerical value. The output for a classification problem, on the other hand, is a Boolean or categorical value (if more than two categorical options). Given that the output of this machine learning problem will be binary (intervene or not intervene) this problem is best addressed using classification techniques.
2.
    a. Total number of students: 395
    b. Number of students who passed: 265
    c. Number of students who failed: 130
    d. Graduation rate: 67.09$
    e. Number of features: 30
3. The feature and target columns were identified. The feature columns were then preprocessed such that non-numeric columns, including those with non-binary values, were converted into binary values with either a 0 or 1. Lastly, the data was split into training and testing sets. The training set has 300 values while the test has 95.
4.
    a. The first model selected is a random forest classifier. This is an example of an ensemble learning method in which a number of relatively naïve hypotheses are aggregated up to create a more robust hypothesis. In this case, a number of individual decision trees are generated, and subsequently aggregated to create a "forest" of decision trees. This algorithm is computationally intensive and should use a relatively large amount of computing power and is somewhat prone to overfitting. On the other hand, it can be used to generate very good classifiers if techniques (such as bagging) are used to "prune" the forest so it generalizes better. This model was chosen to see if a computationally intensive algorithm is viable on a small data set given the budget constraints.

Random Forest			
Training Size	100	200	300
Training Time	9.685	9.913	10.769
Prediction Time (Test)	0.001	0.002	0.39
F1 Score-Training	0.955	0.949	0.945

F1 Score-Test	0.748	0.791	0.857

b. The second model selected is a support Vector Machine (SVM). SVM's are less computationally intensive, but can be somewhat limited if the data is non-linearly separable. However, SVM's may make data linearly separable in a higher dimension space by using the kernel trick. In such cases, SVM's may provide a relatively low cost algorithm that still provides good performance. An SVM was chosen because it is computationally light and may provide good performance.

Support Vector Machine			
Training Size	100	200	300
Training Time	0.08	0.144	0.265
Prediction Time (Test)	0.01	0	0
F1 Score-Training	0.773	0.792	0.846
F1 Score-Test	0.841	0.841	0.833

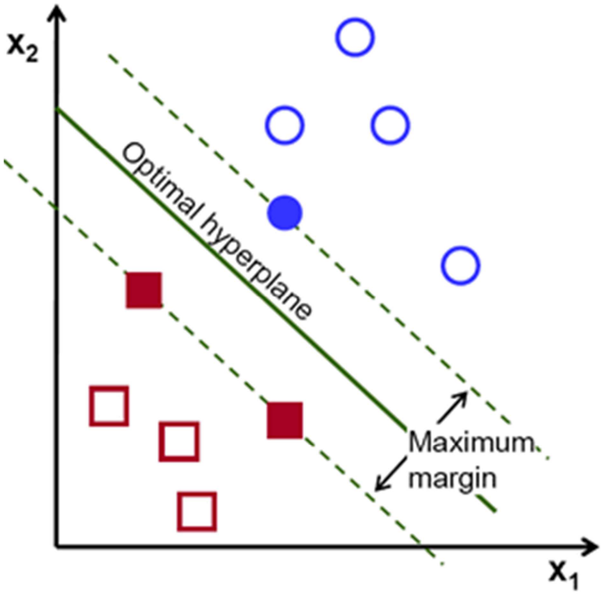
c. The third model selected is a naïve Bayes classifier. Naïve Bayes makes use of Bayesian statistics in which conditional probabilities are used. More specifically, Bayesian statistics uses a prior belief to inform how statistically likely a given outcome is. This differs from frequentist statistics in which the likelihood of an event occurring is determined by the maximum likelihood estimator. Bayesian classifiers are computationally light, highly scalable, and perform well when the features are independent. It was chosen because it should perform well with minimal resources if the features are independent.

Gaussian Naïve Bayes			
Training Size	100	200	300
Training Time	0.001	0.001	0.001
Prediction Time (Test)	0.001	0	0
F1 Score-Training	0.494	0.811	0.786

F1 Score-Test	0.408	0.75	0.791

5. After evaluating three machine learning algorithms (Random Forest, Support Vector Machines and Naive Bayes), it was determined that support vector machines (SVMs) afford superior performance for this classification problem.

   In simple terms, SVMs work on data that is linearly separable, meaning that it creates a line (hyperplane) that separates data into groups; in this case the groups are "intervene" and "not intervene". The optimal hyperplane is one that maximizes the margin between the two groups (see illustration below). While some datasets may not appear linearly separable at first blush, data can be made linearly separable using the kernal trick. The kernel trick can make the data separable in a higher dimension space.



A SVM was selected for three reasons:

1. The SVM does not appear to over fit the data- Here, I tried to employ the techniques that are formalized by the thresholdout method; if there was a significant disparity in the f1 score between the training and test set, I assumed the lower score was correct. If the scores were relatively close, I assumed the higher score. Thresholdout formalizes this so the tendency to over fit data is abstracted away. While I did not use thresholdout directly, I used its principles as a small heuristic when judging model performance. In the case of random forest, for example, the training score was quit good while the test score was considerably lower. Naive bayes had similar scores

for both the training and the test set, but both were relatively low. This suggests the model didn't over fit and may actually have too much bias. The SVM had strong performance on both training and test data and was therefore selected.

2. Less computational power is needed- The first algorithm that I tested was a random forest classifier. Random forests are an example of ensemble learning, and they tend to be very computationally intensive. Given that computational resources appear to be limited, SVM's should afford greater scalability in the event that the training and test sizes increase. Naive bayes requires little computational power but it's performance was not as strong as the SVM.
3. The SVM does not require a significant amount of data to make a reasonable prediction. As training size increases, the SVM performance remains relatively constant. The random forest classifier, on the other hand, is more complex and requires significant amounts of data to improve performance. This is a problem given that computational power is a constraint.

All told, the SVM affords superior performance in terms of avoiding both false positives and false negatives, while taking the computational budget into account. **The final F1 score after tuning the parameters with GridSearchCV was 0.841.**

Resources:

http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

http://starcourse.blogspot.com/2015/08/thresholdout-stunning-paper-in-science.html

Udacity discussion forums