P2 Student Intervention System
Draft Submission for Feedback

1. Classification vs Regression

This problem is a classification problem as the output is a discreet value, in this case whether the student passes (label is “yes”) or they fail (label is “no”).

In comparison, regression problems relate to continuous values of data, such as length or temperature.

2. Exploring the Data

Can you find out the following facts about the dataset?

- Total number of students : 395
- Number of students who passed : 265
- Number of students who failed : 135
- Graduation rate of the class (%) : 67%
- Number of features (excluding the label/target column) : 30

3. Preparing the Data

See submitted iPython notebook

4. Training and Evaluating Models

- What is the theoretical O(n) time & space complexity in terms of input size?
- What are the general applications of this model? What are its strengths and weaknesses?
- Given what you know about the data so far, why did you choose this model to apply?
- Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F_1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant.

Model 1: Decision Tree

Decision trees can work by determining a set of decisions that can be made to predict the likely outcome of data. For example, if the biggest differentiator between passing and failing is the number of absences the student has, then the first decision about could be based on that feature, and so on through the remaining features.

An advantage of the DecisionTreeClassifier is that it it requires little data preparation in order to be used. Effectively the raw data can often be taken without complex normalisation. However, there can be issues with a tree that is biased to one particular outcome if one of the features dominates the dataset.

I chose this model to understand whether there are a number of features that are indicative of a student passing or failing, as if this was the case a decision tree should be able to create a good model based on these features. Decision trees work very well in supervised classification problems like this one.

Training set size ->	**100**	**200**	**300**
Training time (secs)	0.00079	0.00146	0.00161
Prediction time (secs)	0.00017	0.00013	0.00014
F1 score (training set)	1	1	1
F1 score (test set)	0.8244	0.9117	0.813

Model 2: Gaussian Naive Bayes

GaussianNB appears to make over-simplified assumptions on the data that they are provided with. They tend to work well in real world situations and often require a small amount of training data. This aligns well with a need to reduce the server resources for this project. In terms of disadvantages, the classifier works well but it is not known to be a good estimator of probability. In this project, the output is a classification so this model fits the requirements.

I chose this model as there are a large number of features and I wanted to understand whether a number of linear regression models, when merged together, would be a good prediction model. This is the model that was used in the house price example (albeit in a regression problem).

Training set size ->	100	200	300
Training time (secs)	0.00078	0.00058	0.00074
Prediction time (secs)	0.00035	0.00034	0.00025
F1 score (training set)	0.8407	0.7561	0.8
F1 score (test set)	0.7971	0.6607	0.752

Model 3: Support Vector Machines

The Support Vector Machine Classifier is very effective in high dimensional spaces, and it uses a subset of training points in the decision function. This means that it can be very memory efficient. However, there can be poor performance if the number of samples is much less than the number of features provided. This is not necessarily an issue in this case, but it is something to be aware of if the number of features increases or number of students that the model is trained with is reduced significantly.

Training set size ->	100	200	300
Training time (secs)	0.00395	0.00325	0.00561
Prediction time (secs)	0.00073	0.00111	0.00141
F1 score (training set)	0.8888	0.8859	0.8973
F1 score (test set)	0.8435	0.8378	0.8611

5. Choosing the Best Model

- Based on the experiments you performed earlier, in 1-2 paragraphs explain to the board of supervisors what single model you chose as the best model. Which model is generally the most appropriate based on the available data, limited resources, cost, and performance?
- In 1-2 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 make a prediction).
- 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 F_1 score?

Based on the experiments performed, I would choose the DecisionTree Classifier as the model to use. This is due to the fact that it achieves a very good f1 score on the test data, and consistently required low server resources in terms of CPU time.

This can be evidenced by comparing it's training time of 0.79ms for a training set size of 100 compared to 0.78ms (Gaussian NB) and 3.95ms (SVM). Although the training time is equivalent for Gaussian NB, the prediction time and accuracy tell a different story. Prediction time for the test set with a DecisionTreeClassifier took 0.17ms compared to 0.35ms (GaussianNB) and 0.73ms (SVM). The f1 score was slightly more favourable for the SVM classifier (0.8435 compared with 0.7971 for GaussianNB and 0.8244 for DecisionTree).

DecisionTreeClassifiers work by creating a model that predicts the value of a target variable (for example, whether a student will pass or fail) by learning simply decision rules inferred from the data features. For example, one of the rules could be that students have a greater chance of passing if there have a minimal number of absences from class. In creating the model, each of the features are used in order to infer the different rules.