1. Classification vs Regression

Your goal is to identify students who might need early intervention - which type of supervised machine learning problem is this, classification or regression? Why?

This is a classification supervised learning problem because of the binary output – to label whether a given student will pass the exam.

2. Exploring the Data

Can you find out the following facts about the dataset?

- Total number of students
- Number of students who passed
- Number of students who failed
- Graduation rate of the class (%)
- Number of features (excluding the label/target column)

Use the code block provided in the template to compute these values.

Total number of students: 395
Number of students who passed: 265
Number of students who failed: 130
Number of features: 31
Graduation rate of the class: 67.09%

3. Preparing the Data

Execute the following steps to prepare the data for modeling, training and testing:

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

Starter code snippets for these steps have been provided in the template.

```
Training set: 300 samples
Test set: 95 samples
```

4. Training and Evaluating Models

Choose 3 supervised learning models that are available in scikit-learn, and appropriate for this problem. For each model:

Decision Trees (Information gain)	Training set size		
	100	200	300
Training time (secs)	0.001	0.002	0.002
Prediction time (secs)	0.001	0.000	0.001
F1 score for training set	1.000	1.000	1.000
F1 score for test set	0.672	0.744	0.754

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

  General applications: Non-parametric supervised method for classification, regression. Using if-then-else rules, the model is easy for human to understand.

  Strengths:
  - Tree can be visualized. Easy to understand and interpret – this is important for this project because I have to explain the model to my client.
  - The cost of using decision tree is low (logarithmic to training)
  - Inclusive – can handle both numeric and categorical data.
  - Doesn't require much data preparation.
  - Non-parametric – don't have to worry about outliers or whether the data is linearly separable.

  Weaknesses:

  - Easy to over fit. In the above table, F1 score for training set == 1 signals strong overfitting.
  - Can be unstable as small variation in data creates completely different trees – This could be the reason why F1 score for test set oscillates when we change the size of the training set.
  - Can create biased trees if some classes dominate.

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

The given dataset is small and clean with both numeric (such as age and absences) and categorical (all the other attributes) data. Decision tree is good at handling dataset like such. Also, decision tree classifier takes little time to train – as the above table demonstrates, training time is nearly constant as the training size grows.

SVM - Linear SVC	Training set size		
	100	200	300
Training time (secs)	0.007	0.021	0.031
Prediction time (secs)	0.016	0.002	0.001
F1 score for training set	0.943	0.851	0.842
F1 score for test set	0.713	0.783	0.778

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

  General applications: Supervised method for classification, regression and outlier detection.

  Strengths:

  - Effective in high dimensional space – supports datasets with lots of features.
  - Versatility – can choose from different kernel functions for different learning goals.
  - Only use a subset of data points near the boundaries, as other data points are “turned off”.
  - Users are able to express their domain knowledge through the kernel they’re using.

  Weaknesses:

  - Linear relationship between training set size and training time – can take a lot of time to train if dataset is large.
  - Difficult to visualize and interpret as it gets to high dimension.
  - Provide poor performance when number of features is greater than the sample size of the data.

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

  The student dataset has lots of features while still much less than the number of observations – that’s a good fit for SVM. I pick linear SVC because the sample size is very small. According to the scikit-learn algorithm cheat-sheet, the best first thing I should try is linear SVC.

  http://scikit-learn.org/stable/tutorial/machine_learning_map/

Naïve bayes	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.001
Prediction time (secs)	0.000	0.000	0.001
F1 score for training set	0.654	0.812	0.811
F1 score for test set	0.413	0.724	0.733

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

  General applications: Supervised method for classification. It has many real world application such as recommender system, real-time predication and sentiment analysis.

  Strengths:

  - Extremely fast – scalable.
  - Great historical performance in many real-world applications.
  - When its assumptions of conditional independence holds, only need a little data to be able to generalize.
  - Users are able to express their domain knowledge through the kernel they're using.

  Weaknesses:

  - Poor performance if the attributes are far from conditionally independent.
  - Unseen attributes in the test set can spoil the whole probability chain – this can be solved by smoothing.
  - Provide poor performance when number of features is greater than the sample size of the data.

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

  According to the scikit-learn algorithm cheat-sheet, the best Naïve Bayes is the best fit for this data set because it is small and involves text. Also, Naïve Bayes allows the users to make inferences on a subset of the attributes, which is very useful in the context of student intervention system – the clients may ask questions on some particular subsets.

  http://scikit-learn.org/stable/tutorial/machine_learning_map/

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 recored to make your case.

Classifier	Naïve bayes			Decision Trees			SVM		
Training set size	100	200	300	100	200	300	100	200	300
Training time (secs)	0.001	0.001	0.001	0.001	0.002	0.002	0.007	0.021	0.031
Prediction time (secs)	0.000	0.000	0.001	0.001	0.000	0.001	0.016	0.002	0.001
F1 score for training set	0.654	0.812	0.811	1.000	1.000	1.000	0.943	0.851	0.842
F1 score for test set	0.413	0.724	0.733	0.672	0.744	0.754	0.713	0.783	0.778

As you can see from the above graph, cells that are colored green means better relative performance within the same row. Looking at the graph, we can quickly conclude that Naïve Bayes is the most time efficient model as it takes almost no time to train (0.001) and predict (0.000), and it will be highly scalable since the time does not linearly increase as the sample size increases. Decision Tree is also relatively time efficient but the training time doubles when training size doubles. SVM, in contrast, takes a lot of time to training, and its training time is linearly related to the training size.

All three models have comparable test F1 score at training size of 200 and 300 with SVM being the winner (test F1 score of 0.783 at training size of 200). However, when comparing with the training F1 score, Decision Tree completely overfits no matter what training size is used; SVM overfits at the training size of 100 and its training F1 score decreases afterwards as the training size increases. The only model has positive correlation between its training and test F1 score is Naïve Bayes.

Based on the experiments I performed, Naïve Bayes seems to be the winner, as it only has the best time efficiency but also generates comparable test F1 score at training size of 300. In additional, Naïve Bayes is very cheap and thus ideal for the given situation where data availability is small, resources are limited and cost needs to be low. Naïve Bayes can also provide flexible when the client is ready to scale in the future.

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 Naïve Bayes classifier will first turn the dataset into a frequency table. For example, for the attribute "gender", it will count all the male and females. Then, it will generate a likelihood table from the frequency table by calculate all the conditional probabilities of each attribute. Then, the algorithm will apply the Naïve Bayes formula to calculate the posterior probability. The class (Not Passed/Passed) with the highest posterior probability is the output.

	Not Passed	Passed	
Gender	Male: 5	Female:6	= % of male
…	% of Not Passed	% of Passed	

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?

Since Naïve Bayes has no parameter, I ran Bernoulli, Multinomial, and Gaussian with differe nt training size to come up with a final F1 score of 0.791366906475 at Bernoulli training size of 266.

I also used GridsearchCV to tune the max_depth parameter of decision tree, and didn't find a ny high F1 score.