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 supervised machine learning problem corresponds to classification because we want to predict a label or category (in this case if the student will pass the final exam or not). If the outcome of this problem was a yearly average grade or the grad from the exam (a continuous value) this problem would correspond to regression.

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: 130
- Graduation rate of the class (%): 67.09%
- Number of features (excluding the label/target column): 30

3. Preparing the Data

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

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

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

4. Training and Evaluating Models

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

The 3 supervised learning I chose were Support Vector Machines, Decision Trees and Naïve Bayes.

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

Support Vector Machines

This model is generally applied to classification problems were a decision boundry (represented by a hyperplane) classifies each object into one of two categories. There also exists implementations for Regression (SVR) and Clustering (SVC) which are all based from the initial implementation of SVMs a classification algorithm. SVM can be applied to a wide

range of classification problems such as classifying images, classifying proteins and cancer diagnosis in medical science.

Strengths:

- Works well in complicated domains were there exists a clear marginal separation.
- It has one regularization parameter (C) which makes it easy for users to avoid over-fitting.
- It is defined as an optimization problem so there is no local minima.

Weaknesses:

- It doesn't work well in big datasets (particularly if we have lots of features) because the training time may be to long.
- It doesn't work well with lots of noise.
- It doesn't provide a probability function for the classification.
- Different implementations need to be done to work on multiclass classification.

Decision Trees

Decision Trees are used in a wide variety of fields, in machine learning the two main applications are classification and regression. This model finds a set of rules to classify each observation based on a process called recursive partitioning. It can be used in a wide range of applications.

Strengths:

- Really easy to use and implement.
- Very easy to understand the results based on its tree schema.
- Can build different classifiers based on ensemble methods.
- Doesn't need a lot of data preprocessing.

Weaknesses:

- Likely to over fit to training data (be careful with the parameter fit).
- Can get stuck of local minima.

Naïve Bayes

This model is based on probabilistic inference from Bayes Rule. It is widely used on text categorization (for example detecting if an email is spam or legitimate) using the word frequencies as the features. It is considered Naïve because in text mining it doesn't consider the word order.

Strengths:

- Easy to implement and simple to run.

Weaknesses:

- Easy to break. For example, phrases that encompass different words with different meanings (for example Chicago Bulls) may not work that well.
- It may need to have a big data set in order to make reliable estimations. Small data sets may show low precision and recall.
- The assumption of independence between features may be to constraining.

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

At first I wanted to start using Decision Trees because of the easiness to run the model and to get a quick glimpse of which were the important variables. After a quick implementation I noticed the F score for the training sample was equal to 1 which meant training overfitting.

I that sense I decided to start working with SVM. I had two main reasons to do this:

1. The data set was not extremely high dimensional so I thought SVMs wouldn't have a hard time training.
2. I could avoid the overfitting that appeared in the Decision Trees just by handling C.

I chose Naïve Bayes as the third model because in theory it is the one that should use the least amount of computational resources which it also appears as an important factor for the client. I wanted to check if the performance of Naïve Bayes was similar to the performance of the other models. If the performance is similar, Naïve Bayes would be a strong candidate for best model based on performance and costs.

- 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.

Note: You need to produce 3 such tables - one for each model.

Sample Size	*Training Time (s)*	*F1 Train*	*Test Time (s)*	*F1 Test*
300	0.007249	0.869198	0.001686	0.758621
200	0.004737	0.86901	0.001355	0.781457
100	0.001254	0.847059	0.000676	0.786667

Support Vector Machines

Sample Size	*Training Time (s)*	*F1 Train*	*Test Time (s)*	*F1 Test*
300	0.003873	1	0.000234	0.717949
200	0.001344	1	0.000162	0.75

100	0.001019	1	0.000188	0.650794

Decision Tree Classifier

Sample Size	*Training Time (s)*	*F1 Train*	*Test Time (s)*	*F1 Test*
300	0.001139	0.808824	0.000297	0.75
200	0.001175	0.75502	0.000333	0.721311
100	0.00088	0.414634	0.000337	0.423529

Naïve Bayes

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.

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).

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?

The best model I chose for this task are Support Vector Machines. SVM present the most consistent values of the F1 score (0.75-0.78 on test samples) where Decision Trees and Naïve Bayes change their F1 score values based the training / test sample splits (and can go from 0.42 to 0.78). In this sense, SVM performs consistenly better on the test sample which shows that the model has good generalization.

Although SVM present higher computational costs, between 2 and 7 times the training times vs Decision Trees and Naïve Bayes and between 2 and 16 times the test prediction times, their computational time is so small for this sample sizes that it becomes irrelevant. For our computational needs we can get CPU prices for 0.4 USD per hour. This cost becomes irrelevant if we compare it to the cost of making a mistake classifying incorrectly a student (the cost of classifying a student in the 'pass' category when he actually needs assistance is much higher).

What SVMs actually do is try to find a boundary where students on one side pass the final exam and students on the other side fail the exam. This boundary is found using the history (student's characteristics and their academic performance) and should be able to accurately predict the future. When we wish predict if a student is going to pass or fail the final exam, we should input the student's characteristics and SVM will evaluate at which side of the boundary the student is.

The final F1 score achieved by the fine-tuned model is 0.7586.