

## 1. Classification vs Regression

Classification.

What we want to predict here is whether students will “pass” or “fail” the class.

So we chose column “passed” as a target label to train and test the model.

This target label (“passed” or “failed”) is not continuous but discrete value.

Problems about predicting discrete values are classification.

Therefore I decided to solve this problem as one of classification problems.

## 2. Exploring the Data

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

Done in the IPython Notebook.

## 4. Training and Evaluating Models

I chose these 3 models: Decision Tree, SVM, Gaussian Naive Bayes

Decision Tree:

The problem here is classification, especially binary classification (“yes” or “no”). So, with criterion “entropy” DT learner is supposed to work best because the information gain is calculated on log base 2.

strengths:

- able to handle both numerical and categorical data.

- simple to understand and to interpret.

- bigger classifier available (e.g. ensemble methods)

weaknesses:

- prone to overfit.

SVM:

The dataset here is small (395 data points) and SVM is said to be good at handling small datasets, so I chose this model.

strengths:

- Still effective in cases where number of dimensions is greater than the number of samples.

weaknesses:

If the number of features is much greater than the number of samples, the method is likely to give poor performances.  
 don't work well in very large datasets (because the training time happens to be cubic in the size of the datasets).

#### Gaussian Naive Bayes:

It is said that this model requires a small amount of training data to estimate the necessary parameters. So on the same reason as the SVM model, I chose this model.

Naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering.

#### strengths:

can be extremely fast compared to more sophisticated methods.  
 require a small amount of training data to estimate the necessary parameters.

#### weaknesses:

a bad estimator, so the probability outputs are not to be taken too seriously.

I've written strengths and weaknesses above according to the sklearn documentation and UDACITY's lecture videos.

#### Decision Tree

	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.003
Prediction time (secs)	0.000	0.000	0.000
F1 score for training set	1.000	1.000	1.000
F1 score for test set	0.732	0.773	0.803

#### SVM

	Training set size		
	100	200	300
Training time (secs)	0.001	0.004	0.007
Prediction time (secs)	0.001	0.002	0.005
F1 score for training set	0.848	0.887	0.876
F1 score for test set	0.779	0.792	0.784

## GaussianNB

	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.001
Prediction time (secs)	0.001	0.001	0.000
F1 score for training set	0.831	0.810	0.804
F1 score for test set	0.672	0.726	0.763

### 5. Choosing the Best Model

I chose the Decision tree model as the best model.

It has the highest F1 score for entire testing set size (0.803).

It also has the fastest prediction time (0.000 secs).

Although it doesn't have the fastest training time and have the second fastest training time (0.003 secs), I think the differences between its training time and that of the GaussianNB model (, which has the fastest training time here) is quite small and negligible. Because almost no disadvantage in times and have the highest F1 score for test set, I conclude that the DT model is the best model here.

The final model chosen above (DT model) is supposed to work as follows:

It separates the data points into two subsets and then separates each subsets into two sub-subsets and then separates each sub-subsets into sub-sub-subsets... until each subsets contains only one class (only "yes" or "no" in this case). In this separating process, it uses certain rules for each separation. It determines each rule so that it maximizes certain value (or criterion). Using this set of rules, DT model predicts whether given students will pass or not.

Imagine series of questions like "Does the student want to take higher education?", "Does the student have an extra educational support?", "Does the student study weekly more than 5 hours?", "Is the number of school absences of the student more than 50?", and so on. Then if the answer is "yes" "yes" "yes" "no", we might be able to predict that the student will pass. Or if the answer is "no" "no" "no" "yes", our prediction might be that the student will not pass.

The model's final F1 score is 0.787 for test set (lower than the default model's F1 score).