MLND P2 submission

Note: ANS are in RED font

Q1. 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?

ANS

Classification. We will be determining two sets of students:

- who are likely to pass and don't require intervention.
- The other that may require intervention to pass the class.

Q2. Classification Vs Regression

Now, 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

ANS

Total number of students: 395
Number of students who passed: 265
Number of students who failed: 130
Number of features: 30
Graduation rate of the class: 67.09%

Q4. Training and Evaluating Models

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

- 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.

Produce a table showing training time, prediction time, F_1 score on training set and F_1 score on test set, for each training set size.

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

Sample	Training set size		
	100	200	300
Training time (secs)			
Prediction time (secs)			
F1 score for training set			
F1 score for test set			

ANS Below

1. SVC - Support Vector Classification

- What is the theoretical O(n) time & space complexity in terms of input size?
    - Time complexity: $O(n^4)$
    - Space complexity: O(n) (I have programmed a SVM into an embedded device. The space required was the vector that was used to separate the data (n)).
    - Reference: http://scikit-learn.org/stable/modules/svm.html#complexity
- What are the general applications of this model? What are its strengths and weaknesses?
    - SVM's work great when the data is separable using a hyperplane.
    - Strengths:
        - SVMs generate a classifier that is O(n) in space complexity and hence is portable and fast (can run on embedded systems)
    - Weekness:
        - When there are too many features, the data has to be ported into a higher dimensional space. Which can grow to become quite large.
        - When the data has to be updated and re-trained often, SVMs can be slower than some other options.
- Given what you know about the data so far, why did you choose this model to apply?
    - SVM's work great for two-class classifications (in our case, student passes or not). SVC also work great with fewer features(in our case, just 30).
- 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.

SVC	Training set size		
	100	200	300
Training time (secs)	0.001	0.003	0.007
Prediction time (secs)	0.001	0.002	0.004
F1 score for training set	0.877697841727	0.867924528302	0.876068376068
F1 score for test set	0.774647887324	0.781456953642	0.783783783784

2. Decision Trees

- What is the theoretical O(n) time & space complexity in terms of input size?
    - Time Complexity: O($n^2 log_n$)
    - Space Complexity: O(n^2)
    - Reference: http://scikit-learn.org/stable/modules/tree.html#complexity
- What are the general applications of this model? What are its strengths and weaknesses?
    - This model is used to split the data into multiple subclasses.
    - Strengths:
        - Great for binary classification
        - Has excellent classification for data already seen before (training set)
    - Weaknesses:
        - Can get really complicated , and will tend to overfit given the many features.
        - Has a high space complexity, thus is not very portable.
- Given what you know about the data so far, why did you choose this model to apply?
    - The data has a lot of binary features. So Decision tree would fit and create subclasses efficiently.
- 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.

	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.002
Prediction time (secs)	0.000	0.000	0.000
F1 score for training set	1	1	1
F1 score for test set	0.628099173554	0.753846153846	0.633333333333

3. Gaussian Naive Bayes

- What is the theoretical O(n) time & space complexity in terms of input size?
    - Time Complexity: O(n^2)
    - Space Complesity: O(n^2)
- What are the general applications of this model? What are its strengths and weaknesses?
    - Strengths:
        - Simple probabilistic model works well for binary features.
    - Weaknesses:
        - It assumes that the values are independent
- Given what you know about the data so far, why did you choose this model to apply?
    - Gaussian NB assumes the data is independent, which works well in our case. The F1 score is also as high (0.76). GaussianNB is also efficient for training (in terms of computation.
- 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.

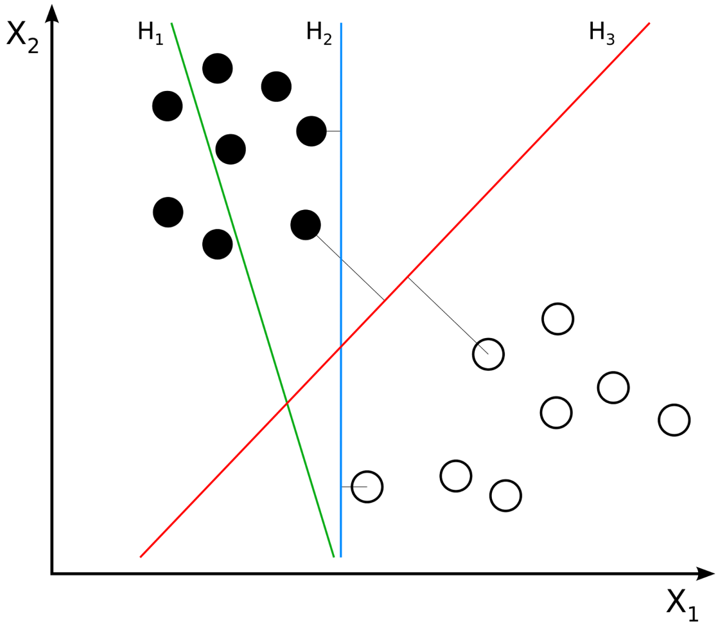
	Training set size		
	100	200	300
Training time (secs)	0.001	0.001	0.001
Prediction time (secs)	0	0	0
F1 score for training set	0.846715328467	0.840579710145	0.80378250591
F1 score for test set	0.802919708029	0.724409448819	0.763358778626

Q5. 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?
    - I would say that SVC is the best model. SVC has a higher F1 Score for the test set. The F-1 score for 300 samples is 0.783, which is more than 0.66 and 0.76 for decision trees and Naive bayes. The running time is not considered as the code can detect running time differences in 0.001 s increments. Which is not granular enough for an accurate comparison.
    - Although, with the complexity of O(n^4), with increasing input sizes, the running time is expected to increase exponentially. Something to keep in mind while deploying SVM's into production.
    - The scores are also a lot more stable with varying input sizes, which could prove as a more reliable algorithm when the amount of data increases. SVC also work great for two-class classifications (in our case, student passes or not). SVC also work great with fewer features(in our case, just 30).
- 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).

Training the SVM:

Support vector machines, look at the data and try to find a line that separates the data into two classes. The line looks much like the picture below



Margin: While finding this line that separates the data, the SVM has a parameter to find the most appropriate line. This parameter is the margin. The margin is the difference between the two closest points to the line in the different sets. In the above picture, we can see that the Red line best sperates the classes, and also maximizes the margin.

The SVM can then use this line to decide on the class for new data.

Kernel trick:

There is often a function that can be used to differentiate the data. For example a circle function can be used to differentiate points that are closed to the origin.

Support vectors:

These would be two separators that are closest to the data set.

- 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?
  - 0.817