Build a Student Intervention System: Project Report

Shilpak Banerjee

May 7, 2016

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?

  Answer: This is a classification problem because the output we are predicting is discrete valued i.e. “passed” vs “failed”. In regression problems the output is continuous valued.

2 Exploring the Data

Can you find out the following facts about the dataset?

- Total number of students

  Answer: 395

- Number of students who passed

  Answer: 265

- Number of students who failed

  Answer: 130

- Graduation rate of the class (

  Answer: 67%

- Number of features (excluding the label/target column)

  Answer: 30

3 Preparing the Data

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

- Identify feature and target columns

  Answer: See code.

- Preprocess feature columns

  Answer: See code.

- Split data into training and test sets

  Answer: See code.

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

  Answer: See code.

4 Training and Evaluating Models

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

- Learning Model: *Decision tree classifier* [1]

  - What is the theoretical $O(n)$ time & space complexity in terms of input size?

    Answer: Training time of a binary tree is $O(n_{\text{features}} \cdot n^2_{\text{training set size}} \cdot \ln(n_{\text{training set size}}))$. Prediction time is $O(\ln(n_{\text{test set size}})$.

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

    Answer: Decision tree classifiers can be applied to a wide range of supervised learning classification problems with both numeric and non numeric features. Other strengths include little need for data preparation (inserting missing values), simple to understand, low running time of a trained tree. Weaknesses include their tendency to overfit the training data. So not suitable where the feature set is very large. Also in case of the ID3 algorithm we only select the best feature based on the entropy reduction in the next step. So trained tree may not be the optimal one.

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

    Answer: I find the Decision tree model easy to understand and hence a good first model for classification problems in general. In the training part of the student intervention data set the refined data frame consists of 48 features and 300 samples. So I did not think overfitting will be too big of a problem. (After running the test it did infact turned out to be a problem).

  - 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.

    Answer: See code.

  - Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

    Answer:

[1] I took help from `http://scikit-learn.org/` and `https://storage.googleapis.com/supplemental_media/udacityu/5414400946/ID3%20Algorithm%20for%20Decision%20Trees.pdf` for answering the questions about this model.

	Training set size		
	100	200	300
Training time (secs)	0.003	0.003	0.004
Prediction time (secs)	0.001	0.001	0.001
F1 score for training set	1.000	1.000	1.000
F1 score for test set	0.615	0.705	0.593

- Learning Model: *Support vector machine* [2]
  - What is the theoretical $O(n)$ time & space complexity in terms of input size?

    Answer: Theoritical training time is $O(n_{\text{features}} \cdot n^3_{\text{training set size}})$
  - What are the general applications of this model? What are its strengths and weaknesses?

    Answer: SVMs are used for classification problems. Strengths include effectiveness in higher dimensions, memory effective and also they can be improved a lot using Grid Search. Weakness include their ineffectiveness with an extremely large number of features and also they have a low training speed.
  - Given what you know about the data so far, why did you choose this model to apply?

    Answer: In our problem we do not have a large feature set compared to the sample size of the training set. So I think SVMs will make a good choice.
  - 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.

    Answer: See code.
  - Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

    Answer:

	Training set size		
	100	200	300
Training time (secs)	0.002	0.004	0.013
Prediction time (secs)	0.002	0.003	0.004
F1 score for training set	0.878	0.868	0.876
F1 score for test set	0.775	0.781	0.784

- Learning Model: *Naive Bayes with Gaussian NB* [3]
  - What is the theoretical $O(n)$ time & space complexity in terms of input size?

    Answer: Training time is $O(n_{\text{training set size}})$. (I am not sure)
  - What are the general applications of this model? What are its strengths and weaknesses?

    Answer: Naive Bayes is used in supervised learninng classification problems like spam filtration and document classification. Strengths include small requirement of training

[2] I took help from `http://scikit-learn.org/` and `https://storage.googleapis.com/supplemental_media/udacityu/5422370632/Kernel_Methods_and_SVMs.pdf` for answering the questions about this model.

[3] I took help from `http://scikit-learn.org/` and `https://storage.googleapis.com/supplemental_media/udacityu/5462070314/Bayesian%20Learning.pdf`

data and fast speed. Weakness include it beign a bad estimator of prediction probabilities.

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

  Answer: In our case our data set is not too big and we only require a classifier as opposed to predicting the probability of a student passing.

- 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.

  Answer: See code.

- Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.

  Answer:

	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	0.847	0.841	0.804
F1 score for test set	0.803	0.724	0.763

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?

  Answer: I choose the Support vector machine based on the test performed above.

  My test showed that F1 score for the training data is better for the SVM than the GaussianNB model. One may argue that the Naive Bayes has a marginally better F1 score than the SVM on the test data but my assumption at this point is that for a different test set this difference may not matter and since SVM is more customizable, I can better the F1 score of GaussianNB even on this test data set with a good SVM[4] improved by a gridsearch. Also I ruled out decision trees because an F1 score of 1.0 on the traing data indicated a massive overfitting problem. This is also reflected by the extremely poor performance on the test set.

  Another point that needed to be mentioned was the time factor. SVMs do take a longer time to train. But our training data set is not big enough for this to be a matter of concern.

  Best F1 score on training set: SVM @ 0.878

  Best F1 score on test set: GaussianNB @ 0.803

  Fastest training time: GaussianNB

  Fastest prediction time: GaussianNB

[4] I amanged to close in on the difference but could not better it.

- 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.

  Answer: I choose the SVM because our training data set is not very large. It is also a memory efficient model. There is a time penalty one needs to pay for SVM but our training time is not large enough (< 1 sec) to concern ourself with.

  One can make a strong case for the GaussianNB model also because of its performance on the test data set and also the SVM is only marginally better on the training set (In fact after a grid search the F1 score on the training data set drops to 0.832 from 0.876. So indeed it was overfitting and removal of the overfitting resulted in this drop and an increase in F1 score on the test data set from 0.784 to 0794. So the optimized SVM performed marginally worse on both the training set and the test set when compared to the GaussianNB.

- In 1-3 paragraphs explain to the board of supervisors in laymans 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).

  * *Answer:* Support vector machine (SVM) is an algorithm to classify a dataset into its target categories. We give a simple description of such an algorithm at work using a simple example.

  Consider a data set with only two features. This dataset can be easily plotted on the two dimensional plane. Assume this entire dataset is categorized into two categories, say, blue category and red category. Assume that we can draw a straight line on the plane which can divide the data points and the blue data points are on one side while the red data points are on the other side. Any such straight line correctly dividing the existing data set can be used to classify future data into red and blue category. And with any given data set, if there is one dividing straight line, then one can obtain infinitely many such lines by perturbing this line. Which one is the best choice to classify future data? Well, the SVM way is to choose a *separating* line which is furthest away from the red and the blue points. Note that choosing this best separating line only requires us to consider the red points that are closest to the blue cluster and blue points closest to the red cluster.

  Of course one can generalize such a technology to more general scenerios by considering complicated separating curves (and hypersurfaces for higher dimensions).

- 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.

  Answer: See code.

- What is the models final F1 score?

  Answer: 0.795