Promoting Student Success in Online Courses

Chuan Yu Foo CYFOO@STANFORD.EDU
Yifan Mai MAIYIFAN@STANFORD.EDU
Bryan Hooi BHOOI@STANFORD.EDU
Frank Chen FRANKCHN@STANFORD.EDU

1. Introduction

Online education has become popular as an effective method of imparting knowledge to a wide audience efficiently and at low cost. Unlike traditional "offline" courses, online courses open up the possibility of fine-grained tracking and monitoring of students' activities, progress and performance. In turn, this creates the possibility of of analyzing the factors involved in student success, retention and progress. An understanding of these factors could allow online educators to better tailor their courses to promote student success.

In this project, we will examine these factors in the context of the Machine Learning and Databases online courses (ml-class and db-class) that Stanford University is offering in Fall 2011. Specifically, we aim to examine the factors behind student success and student retention in these two courses.

2. Methods

2.1. Tasks

For this project we looked at three different tasks pertaining to our online education platform.

2.1.1. Student Success

We attemtped to predict students' midterm examination scores based on the data collected for the Databases course. This task was done using L1-regularized linear regression.

2.1.2. Student Retention

We attempted to predict whether students will continue on (defined by submitting at least half of the quizzes for that week) in the Machine Learning course in week 6 given their activities in weeks 1 - 5. This task was done using L1-regularized logistic regression from the LIBLINEAR library. (2).

2.1.3. Automatic Tagging

We attempted to predict tags on forum topics with related keywords, allowing users to more easily search for infomration on a particular topic, simply by selecting particular keywords.

Automatic tagging can be modelled as either a supervised or an unsupervised learning problem. In the supervised scenario, the algorithm learns in an online setting, in which the algorithm suggests tags whenever a forum post is made and receives feedback when the user actually selects a tag. In the unsupervised scenario, the algorithm attempts to automatically cluster forum posts and tags each cluster with a keyword that is representative of the cluster.

We have used both supervised and unsupervised learning methodologies for our project.

2.2. Data collection

Over the course of 10 weeks, students taking the online Machine Learning and Databases classes engage in a number of learning and non-learning related activities on the respective course websites. These include watching lecture videos, completing review quizzes, exercises, and programming assignments. Students may also ask questions and reply to questions on a Question and Answer (Q&A) forum on the website.

For the purposes of this project, we collected data the following activities of individual students: the times at which the student watches a lecture video, attempts a quiz, exercises or programming assignment; how frequently the student switches between video speeds during lecture; pause and seek events during lecture; attempts and scores on quizzes, exercises, and programming assignments; content of forum postings on the integrated Q&A forum; how many views the student's questions received; how many votes the student's questions and replies received; when the student registered and what track

(Basic or Advanced, only applicable for the Machine Learning course) the student is on.

2.3. Preprocessing

Before passing the data into the learning algorithms, the following preprocessing steps were taken.

2.3.1. Student Success and Retention

For the student success task, students who did not submit the midterm were removed. Data on student activities after the midterm began was also removed. After this, 213 features and 9537 examples remained.

For the student retention task, students who were deemed to dropped out before week 6 (i.e. had not submitted the one quiz for week 5) were removed. Data on students' activities after the end of week 6 was also removed. After this, 196 features and 5751 examples remained.

Following this, the remaining examples for each of the tasks were divided into train, cross-validation, and test sets in the ratios 49% : 21% : 30%. The training data was then normalized for each feature by subtracting the feature mean, and dividing by the feature standard deviation. Missing features (e.g. where students did not attempt quizzes) were replaced by 0 or the mean for that feature as appropriate. The normalization values (means and standard deviations) were saved and re-used for testing on the cross-validation set. After cross-validation, the learning algorithm was retrained on the training and cross-validation set and tested on the test set.

2.3.2. Automatic Tagging

For the automatic tagging task, we first preprocessed the forum text using a standard stemming algorithm, replaced common symbols with special tokens, and removed the most frequent words as well as the words and tags that appear at most 5 times. This left 2538 words, distributed among 2765 forum topic titles and message bodies. The resulting input to the algorithm is a document-word matrix containing the frequencies of each word in each message body, and a separate document-word matrix containing the frequencies of each word in each topic title. The output of the algorithm is the list of tags it predicts for each document.

	Train		Test	
Model	±0	±1	±0	±1
Unweighted LR	17.6%	48.4%	17.6%	47.3%
LWLR	19.0%	51.3%	16.2%	47.0%
Neural network	17.7%	48.3%	16.0%	47.6%

Figure 1: Model accuracies for unweighted linear-regression (Unweighted LR) and locally-weighted linear regression (LWLR) (percentage of scores predicted to within indicated error) for the student success task

3. Results

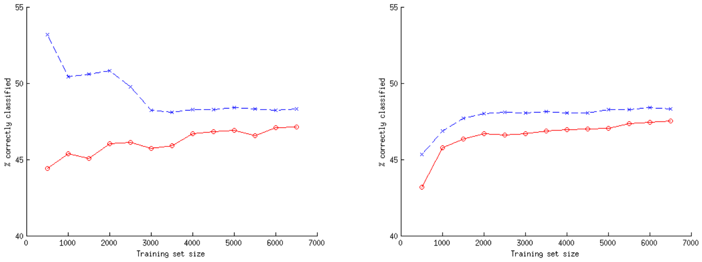
3.1. Student success

For the student success task, our goal was to predict students' midterm scores based on the features described above. The results for this task are summarized in Figure 1. The learning curves can be seen in Figure 2.

The first model we tried was **L1-regularized linear regression**. The regularization parameter $\lambda = 1000$ was chosen using cross-validation. The model successfully predicted 16.6% of students' midterm scores exactly, and 47.3% of students' midterm scores to within ±1 (this includes the students' whose scores were predicted exactly). The accuracies on the training set were similar, with 17.6% of scores predicted exactly, and 48.36% predicted to within ±1. This suggested that the algorithm was suffering from high bias.

As it was not possible to obtain more features, we attempted to address this issue by using a more powerful model – **regularized locally-weighted linear regression**. The regularization parameter $\lambda = 500$ and the bandwidth parameter $\tau = 10$ were chosen using cross-validation. Performance on the test set was still poor, with 16.2% of scores predicted exactly, and 47.0% predicted to within ±1. Performance on the training set was similar, with 19.0% of scores predicted exactly, and 51.3% predicted to within ±1. This suggested that the high bias problem was still unresolved.

In another attempt to resolve the high bias problem, we used a **regularized neural network** to model the data. The neural network we used had 213 input units, one for each feature; 50 hidden units, and 1 continuous-valued output unit for the predicted score. The hidden layer used a sigmoid activation function, while the output layer was linear. The regularization parameter $\lambda = 300$ was chosen using cross-validation. Compared to unweighted lin-



(a) Unweighted linear regression (b) Neural network

Figure 2: Learning curves for two of the three models in the Student Success task. The graphs plot training (blue line) and test (red line) accuracy (number of scores predicted to within ±1) against number of training examples.

ear regression, the neural network did not perform much better. On the test set, 16.0% of scores were predicted exactly, and 47.6% predicted to within ±1. Training accuracy was similar, with 17.7% of scores were predicted exactly, and 48.3% predicted to within ±1. This suggested that the high bias problem was still not resolved.

In the light of this, we conclude that the features we have are not sufficiently predictive of students' midterm grades. Interestingly enough, since our features included assessment data, including the number of time students attempted each quiz and the minimum, mean and maximum scores they obtained for each quiz, this might suggest that the current assessment methods are not sufficiently discriminative of students' understanding of the material (under the assumption that the midterm is a ground truth indicator of students' understanding of the course material).

3.2. Student retention

Model	Train	Test
Logistic regression	71.1%	68.8%
SVM with Gaussian kernel	81.7%	68.8%

Figure 3: Model accuracies for the student retention task

For the student retention task, our goal was to predict whether students would continue in the course or drop out of the course in week 6, based on features collected regarding their activities in weeks 1 - 5. The results for this task are summarized in Figure 3. The learning curves can be seen in Figure 4 above.

The first model we tried was **L1-regularized logistic regression**. The regularization parameter, $\lambda = 1$, was chosen using cross-validation. The model was not successful, accomplishing an accuracy of 68.8% on the test set, which is at chance for the data set (the entire data set, and the test set in particular, both contain dropped out and continuing students in the ratio $1 : 2$, making chance performance about 66.6%). Accuracy on the training set was similarly low at 71.1%, suggesting that the logistic regression model might be suffering from high bias.

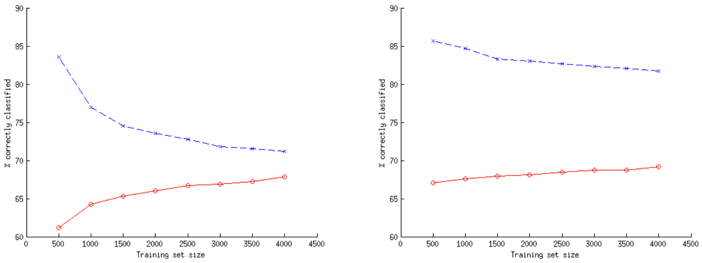
Since it was not possible to obtain additional features for the dataset, we attempted to address the high bias problem by using a more powerful model, the **support vector machine (SVM) with Gaussian kernels**. However, performance was still poor, with the SVM yielding an accuracy of 68.8% on the test set and 81.7% on the training set, suggesting that the high bias problem was still not resolved.

In the light of this, we conclude that the features we have are not sufficiently predictive of whether students drop out of the course or not. This could be explained by the fact that students may drop out of the course for many external reasons such as loss of interest, lack of time, other commitments, and so on, reasons which are not captured by the features in our dataset.

3.3. Automatic Tagging – Supervised

3.3.1. Modifications to Standard Naive Bayes

For the tag prediction task, we use the Naive Bayes algorithm with two significant modifications.



(a) Logistic regression

(b) SVM with Gaussian Kernel

Figure 4: Learning curves for the two models in the Student Retention task. The graphs plot training (blue line) and test (red line) accuracy against number of training examples.

Incorporating message titles and bodies: We assumed titles and bodies are conditionally independent given a class, then we can use the Naive Bayes assumption to model probability of title text and body text separately. As such, we pick the class

$$i$$

which maximizes the posterior probability of the text, such that:

$$\operatorname{argmax}_i \left(p(y_i)p(T|y_i)p(B|y_i)\right) =$$

$$\operatorname{argmax}_i \left(p(y_i)\prod_j p(T_j|y_i)\prod_j p(B_j|y_i)\right)$$

where y_0 is the untagged class, y_1 is the tagged class, and T and B are the title and body vectors for the message we are currently processing: specifically, T_j, the jth entry of T, is 1 if the jth word appears in the message title and 0 otherwise, and B_j, the jth entry of B, is 1 if the jth word appears in the message body and 0 otherwise. Justifying this formula requires an extension to the Naive Bayes assumption: we need to assume that the message titles and bodies are conditionally independent given the class of the message, which we feel is a reasonable assumption.

Modified Laplace Smoothing: In our version of Laplace smoothing, we estimated $\phi_{j|y=1}$, the conditional probability of the jth word being present given the class $y = 1$, as follows:

$$\phi_{j|y=1} = \frac{\sum_{i=1}^{m} 1\{x_j^{(i)} = 1 \cap y^{(i)} = 1\} + \delta}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} + 2\delta}$$

and similarly for $\phi_{j|y=0}$; here $y^{(i)}$ is the class of the ith training example and $x_j^{(i)}$ is 1 if the jth word appears in the ith training example, and 0 otherwise. δ is a variable constant which would be 1 in regular Laplace smoothing; in our case, we chose δ by cross-validation to minimize test error. This modification was performed because we observed that the sparseness of the data meant that the ones added during Laplace smoothing seemed to be overwhelming (or at least adversely affecting) the sparse frequency data that was actually collected, and thus we felt that a smaller parameter δ might be more appropriate. It turns out that this modification to Laplace smoothing can in fact be justified as a linear interpolation between a maximum likelihood estimator and a uniform prior. (3)

3.3.2. Training Results

For each tag, the vast majority of the forum posts belong in the 'untagged' category. As such, we measured the performance of our algorithm using precision and recall, combined into the F-measure score.

Model	Train	Test
NB	0.214	0.0212
NB+Titles	0.239	0.0332
NB+Titles+Smoothing	0.912	0.0734

Figure 5: F1 score for the tag prediction task

Naive Bayes: In our first attempt, we used Naive Bayes with the usual Laplace smoothing and only using the message bodies as input. As can be seen from the table, the predictor's training and test F-measures were both quite poor. Due to the large difference between training and test errors we hypothesized that overfitting was occurring. As no more data is readily available, however, we decided to incorporate the topic titles into our predictor as

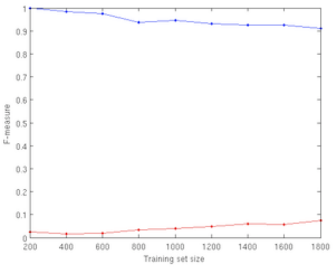


Figure 6: Learning Curves for the final tag prediction algorithm. The graphs plot training (blue line) and test (red line) F-measure against number of training examples.

described in our methodology, hoping to give our algorithm more information to work with.

Naive Bayes (with Titles): Therefore, for our second attempt, we incorporated the topic titles using the modified prediction formula described in our methodology. As can be seen from the table, this led to a small improvement in both training and test performance. Note, however, that this modification does not directly tackle the overfitting problem: it does provide more data, but simultaneously enlarges the space of possible models (since the prediction model for each tag now includes separate conditional probability vectors for the title and message body). This is consistent with the observation that the training F-measure also improved, whereas normally adding more data would be expected not to improve training performance.

Naive Bayes (with Titles and Modified Smoothing): On our third attempt, we used the variable smoothing constant $\delta = 0.00001$ chosen using cross-validation. As a result, the algorithm did significantly better in both training and test settings. The large difference between training and test performance that still remains (as can be seen from the learning curve) indicates that the algorithm is still significantly overfitting. Considering that even the most frequent tags appear in only about 30 messages, this is not completely unexpected, as the algorithm does not have enough data in a particular tag's 'tagged' class to build up accurate prior and conditional probabilities for that class.

3.4. Automatic Tagging – Unsupervised

For the unsupervised automatic tagging task, we used Latent Dirichlet allocation (LDA) model proposed by Blei et. al. (2003). Specifically, we used the LDA-c implementation by Blei to process our data. The topics produced were able to distinguish between admin-related, theory-related, and implementation-related content. We give some examples below.

Admin Topics week, date, slide, post, due, schedul, note, materi, miss, got, solut, accuraci, submiss, review, incorrect, check, accept, temp

Theory Topics valid, test, model, cross, curv, real, high, sampl, select, overfit, layer, output, nn, hidden, paramet, classifi, unit, digit, given

Implementation Topics plot, gnuplot, after, call, window, close, paus, execut, matlab, window, command, instal, undefin, script, near, version, linux, type

4. Conclusion

For student success and retention, quiz scores and other features are not very predictive. Therefore, we conclude that student retention and success is hard to predict with the set of features we have. We have identified the errors to be poor accuracy due to high bias and we plan to acquire more features such as student survey and time spent on quiz, etc... in the future. In addition, we could collect more features by initiating voluntary student surveys throughout future courses in order to understanding any external factors influencing their performance and success rates throughout the class.

For text identification, we again found that the results were not as good as we had expected. We identified this as severely overfitting due to the large difference between training and test F-scores. We will attempt to get more forum data over the next iteration with more classes and try different algorithms (such as regularized Naive Bayes) as we move on with the project.

5. Acknowledgments

We would like to thank Professor Andrew Ng for giving us the opportunity to work on the Online Education project. We would also like to thank Jiquan Ngiam for his generous assistance.

References

[1] D. Blei, A. Y. Ng, M. I. Jordan. Latent Dirichlet Allocation. Journal of MA-

chine Learning Research 3 993-1022. http://www.cs.princeton.edu/ blei/papers/BleiNgJordan2003.pdf

[2] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9(2008), 1871-1874. Software available at http://www.csie.ntu.edu.tw/ cjlin/liblinear

[3] M. D. Smucker and J. Allan. An Investigation of Dirichlet Prior Smoothing's Performance Advantage. http://www.mansci.uwaterloo.ca/ msmucker/publications/Smucker-Smoothing-IR319.pdf