Learning To Predict A Student's Performance On Problem Sets

Josh Taylor

Project Goal

The goal of this project is to show that it is possible to create a system that can effectively predict whether a student will get a problem right or wrong, given the performance of other students on a set of problems which contains the target problem and given the students performance on the same set of problems minus the target problem. The reason I would find such a system useful is that I plan on making a web site that generates dynamic problems sets for my senior project. The idea is that the web site would use students history on problems to create customized optimal sets of problems for learning a particular topic. Predicting the probability that a student gets a problem right is key to developing such a system.

Data

For this project the Grade 7 math problems from the California SAT9 exam were used as the problem set. The data is composed of over 400,000 students, each of whom have taken 80 problems.

Choice of Error Measure

For the predictive system to be effective, it must accurately estimate the probability of a students success on a problem, as opposed to predicting a binary value for success or failure. This is because, it would be useful when creating dynamic problem sets, to be able to specify how challenging a problem set should. Thus rather than just calculating the average number of times a prediction is wrong, error should measure the standard deviation of the difference between the success and the predicted probability of success. A measure of average absolute value of differences promotes a binary prediction while using sum of squared errors promotes an accurate estimate of probability. The measure of error therefore will be:

$$\text{error} = \text{sqrt}\left(\sum i{:}m \sum j{:}n \left(yHat_{ij} - y_{ij}\right)\right)/(m*n))$$

where yHat is the predicted probability of getting target problem right, and y_{ij} is success on the ith problem on the jth test (1 if correct, 0 if wrong). The error is summed over n, because each of the n problems is used as a target and over m for each of the m students in the sample. For frame of reference the error generated by simply guessing the mean success of the target problem over all students in the training set is shown in the table below:

(Note: The fact that the generalized error was smaller for the smallest training sets is idiosyncratic to the training set used. If results were averaged over a large number of training sets, this would not have been the case.)

Training	Generalized	Size
0.4791	0.4898	50
0.4792	0.4876	100
0.4742	0.4951	1000
0.4763	0.4921	10000

Choice of Model

The focus of this project is feature selection rather than model selection. GDA, SVM and linear regression were considered for use as the standard model for the experiments. GDA was discarded since the data is not Gaussian and GDA is not recommended when the independent variables are categorical. For SVM with a linear kernel, and a Gaussian kernel, and a linear regression on the full set of problems the errors were comparable. However, since some of the feature selection techniques required manipulating the covariance matrix, linear regression was chosen as the standard model fore each feature selection approach taken.

Approach 1: Construct Covariance Matrix Using Only K Most Correlated Problems

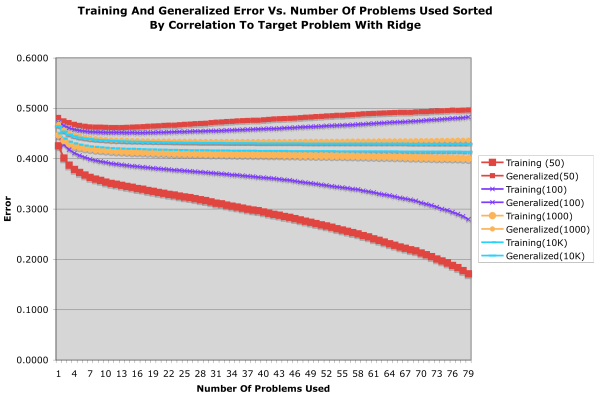
Motivation

The motivation behind using only a subset of problems to predict the success on a target problem is two fold. First, when the sample size is low, such as 50 or 100 students, using all other n-1 problems in the regression will over-fit the data. Secondly, without knowing anything about the data it seems possible that some of the problems, even for large sample sizes, would not have meaningful correlations to the target problem, and their inclusion in the regression would likely increase the generalized error.

Implementation

Approach 1 was implemented by calculating the correlation matrix from the covariance matrix using the formula: x'x where x is the m by n matrix of zeros and ones of n problems for m students after mean adjusting each of the n columns. A correlation matrix was then constructed form the covariance matrix, and for each target problem, the corresponding column of correlations was sorted to find the most correlated other problems to the target. The covariance matrix was then constructed using only the k most correlated problems and used to perform a linear regression.

Results



Num of Students	Min Generalized Error	Optimal Num of Problems
50	0.4616	12
100	0.4516	14
1000	0.4340	39
10000	0.4294	70

Not surprisingly, the results show that the optimal number of problems to use increases with the size of the training set, and of course the results improve as the training set size increases.

Approach 2: Interpolated Binning

Motivation
The motivation for binning is to try and gather as much information about a student into a minimal number of features. The previous results appeared to have a substantial difference in errors between the sample size of 50 and 10K. The problem may be that to get enough information to make an accurate prediction requires at least 10 to 15 problems, but by this time the regression is already over-fitting the data. Grouping students into bins according to the number successes they enjoyed on other problems has the advantage of being able to express a lot of information with a small number of features.

Implementation
In the interpolated binning approach, the features of the training set are bins that represent how many problems a student answered correctly. The bins receive weights according to which one or two bins straddle the number of successes the students enjoyed on the other problems. The weights always sum to one and no more than two will be non-zero.

Results

Bins: 20-40-60-80

Training	Generalized	Size
0.4303	0.4494	50
0.4364	0.4421	100
0.4211	0.4348	1000
0.4207	0.4337	10000

Bins: 20-80

Training	Generalized	Size
0.4386	0.4471	50
0.4418	0.4416	100
0.4244	0.4378	1000
0.4236	0.4368	10000

The above data show good results even for small training sets. For larger training sets more bin points are useful, while the smaller training sets benefit from fewer bins.

Approach 3: Eigenvector Analysis

Motivation:
Thus far, I have discovered one approach that appears to work well for large sample sizes and one that works well for small sample sizes. It would be nice, however to find an approach that works well for both. A good starting point would be to get a better sense of the factor structure of the data, and this can be done with an eigenvector analysis.

Implementation:
An eigenvector analysis was performed on the n x n correlation matrix to get a sense of the number of factors influencing correlation between problems.

Results And Analysis

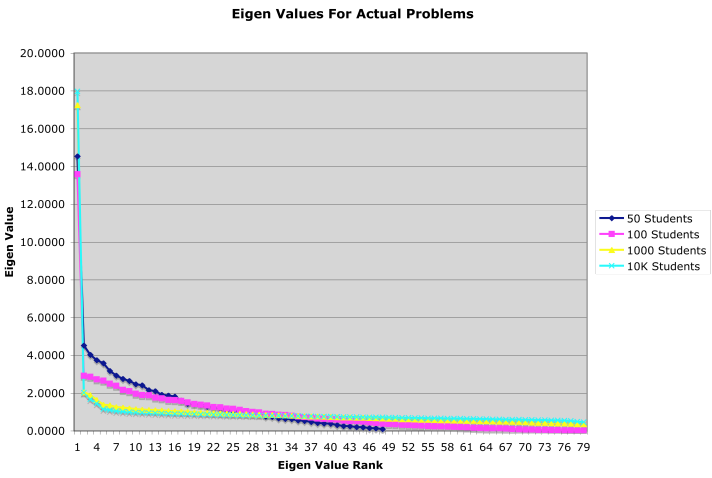


Fig. 1

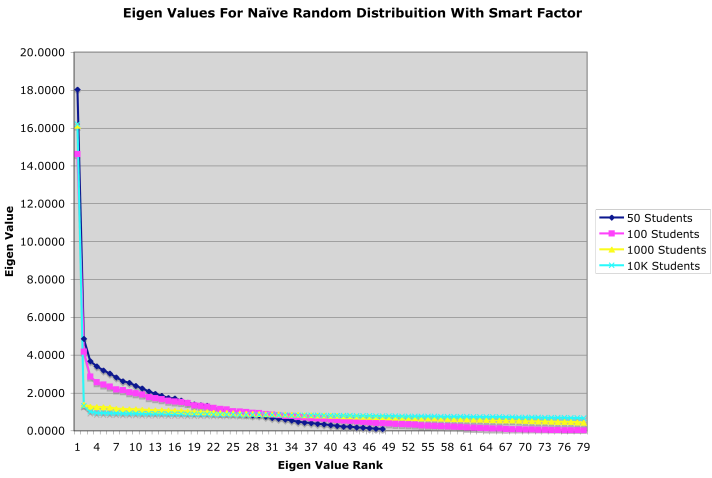


Fig. 2

From the data (Fig. 1) it would appear as though much of the substantive variance in the data can be explained by a few eigenvectors and the rest of the eigenvectors correspond to Eigen values that are difficult to distinguish from noise.

To test this hypothesis another eigenvector analysis was performed on the results of a randomly generated answer set that shared the same mean value of P for each problem as the test data set but had no other structural correlation in common. For each student, success on a problem was assumed to be uncorrelated to their performance on all other problems. This was simulated by sampling from a binomial distribution with the appropriate parameter P set to each problem's mean. Each student was assumed to be no different from any other.

The graphs of the Eigen values for the naïve random sampling look very similar to the tails of the Eigen value graphs for the actual data, suggesting that many of the eigenvectors with smaller Eigen values may simply be capturing spurious correlation.

Next a "smart factor" was added to the naïve random data by making some randomly generated students more likely to find success than others. This was accomplished by shifting the mean value of all P's for each student independently. The shift was drawn from a uniform distribution between -.4 and .4, with the resulting probabilities for each problem bounded by 0 and 1.

The graphs of the Eigen values for the naïve random distribution with the smart factor added (Fig. 2) and the actual data are astonishingly similar. It thus appears that a single factor is responsible for much of the predictability in a student's performance. In any event only the first few Eigen values are distinguishable from those from the randomly generated data, except for in the very largest data sets.

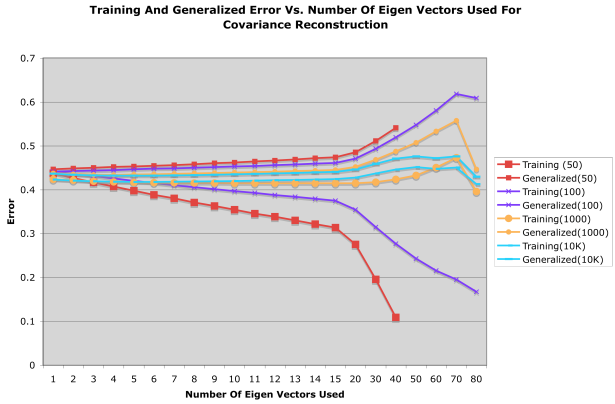
Approach 3a: Reconstructing Covariance Matrix From K Largest Valued Eigenvectors

Motivation:
Considering that many of the eigenvectors with small Eigen values are difficult to distinguish from random, it might be profitable to make predictions without incorporating the eigenvectors with small Eigen values into the model.

Implementation:
Excluding the small valued eigenvectors was accomplished by creating covariance matrices using subsets of the Eigen vectors in the following manner. First, find the eigenvectors and corresponding Eigen values of the covariance matrix X transpose X, sorted in descending order of Eigen values. Next set the Eigen values to zero for the k+1 to the nth eigenvectors, and reconstruct the covariance matrix using the formula $CE = U\Lambda U^T$, where U is the matrix of eigenvectors and Λ is the diagonal matrix of modified Eigen values. Finally, replace the diagonal of the matrix CE with the diagonal of the original covariance matrix.

There are two reasons for this last step. First, with out doing so, CE would not be invertible, because it was created with fewer that n eigenvectors and therefore could not have full rank. By replacing the diagonal of CE with one that comes from a matrix that is full rank, CE becomes invertible. It also makes sense to use the diagonal from the covariance matrix, since we are relatively confident in the variances along the diagonal. We can measure reasonably narrow confidence bounds on the diagonal, while off diagonal elements are much more volatile. This makes it difficult to be confident in resulting partial correlation coefficients without the use of large data sets. Finally standard linear regressions are performed using the resulting covariance matrices from each training set while varying the number of included eigenvectors.

Results:



Num Of Students	Min Generalized Error	Optimal Num Of Eigenvectors
50	0.4465	1
100	0.4413	1
1000	0.4338	4
10000	0.4321	4

The results show that a single eigenvector should be used for training sets of 50 or 100 students while the first four eigenvectors are useful for larger training sets.

Aside: Comparing Generalized Error of 20-80 Binning To Generalized Error of Regression Using Just The First Eigenvector

After the eigenvector analysis I hypothesized that the first eigenvector may represent a smart factor. We can essentially think of the 20-80 as a measure of intelligence, the closer a student is to 80 the more intelligent they are. The minimal difference in errors suggests that this hypothesis was not far off.

Generalized	Generalized	Size
0.4465	0.4471	50
0.4413	0.4416	100
0.4375	0.4378	1000
0.4366	0.4368	10000

Approach 3b: Eigen Value Multiplier and Ridge:

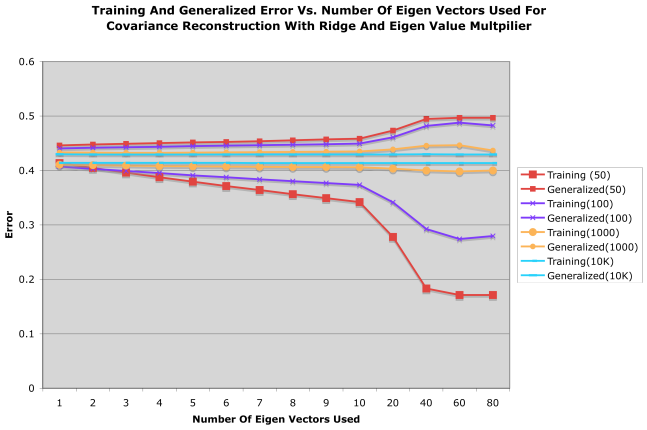
Motivation:

The later part of the graph of the errors for the reconstructed covariance matrix exhibits seemingly strange behavior. In particular, in all cases with greater that 50 tests in the training set, the regression preferred having all eigenvectors to having all but the last few. But the optimal strategy is to just keep the largest one or several. This result suggested that discontinuity in the treatment of eigenvectors of similar corresponding Eigen values was not favorable. It also suggested that even the last few eigenvectors had some useful information content. To address this, two approaches were tried: demotion of Eigen values after the kth eigenvector using a multiplier and promotion of the diagonal using a ridge multiplier. The ridge multiplier has a similar impact to demoting Eigen values, but was less discriminating in that it treats all components of off diagonal elements equally.

Implementation:

In this approach, the full values of the first k Eigen values are used, but a multiplier less than one that depends on the sample size is applied to Eigen values k+1 to n. Again the covariance matrix, CE, is constructed from the altered Eigen values and their corresponding vectors. Finally, a ridge multiplier is applied to the diagonal of the matrix.

Results



Ridge: 1.4

Num Of Students/ Multiplier	Min Generalized Error	Optimal Num of Eigenvectors
50 / .05	0.4458	1
100 / .15	0.4405	1
1000 / .4	0.4324	2
10000 / 1	0.4294	3

Using a single multiplier after the kth Eigen vector and either using a ridge of 1.4 or no ridge, generated comparable results and in both cases improved the results achieved by zeroing out Eigen values. Notice that the optimal multiplier when the training set is 10,000 is 1., meaning the only distortion of the original covariance is the ridging.

Conclusion

The best approach for all training set sizes was reconstructing the covariance matrix using one or several fully weighted eigenvectors and demoting the rest. The optimal values for both the demotion multiplier and the number of full valued eigenvectors used, makes sense in light of the eigenvector analysis. At small training set sizes the eigenvectors corresponding to small Eigen values are likely just capturing spurious correlation, while the structure captured by all the eigenvectors is meaningful for large sample sizes. It is not surprising then that the larger multipliers and larger number of fully weighted eigenvectors were optimal for the larger training sizes. Using this approach seems to be a very robust strategy, because it has mechanisms to incorporate the optimal amount of information, avoiding over-fitting the data at low sample sizes and under-fitting at large sample sizes. This approach would be especially useful for implementing a dynamic problem set system, since the sample sizes for problems is likely to range drastically in size.

The other two approaches were not nearly as robust. The sorted correlation approach was nearly as effective using 10000 students but was significantly worse at lower sizes. The binning approach on the other hand was nearly as effective at small sizes and worse at larger sizes. This makes sense since because the eigenvector approach essentially only uses the first eigenvector at small sizes, and that single eigenvector appears to be capturing the same information as the binning.

The results from this project are extremely promising, especially considering the difficult nature of making predictions using test data. Test problems are inherently orthogonal, since the objective of a test is to test as many points of knowledge as possible with the fewest test problems. Thus the problems are unlikely to be highly correlated, making accurate predictions harder. In fact, the tail of the eigenvector analysis, which looks very similar to a random naïve distribution, suggests that this is indeed the case. Furthermore, the SAT9 is a multiple-choice test, and the ability of a student to randomly get a problem right adds additional noise. Thus I am hopeful that when I implement these approaches on sets of math problems that are highly correlated and are not multiple choice, the errors will be significantly lower. It may be, of course that with a richer correlation matrix, other approaches would be even better. But the modified Eigen value approach seems likely to be a very good starting point.