Learning to Test

Michael Munie
Department of Computer Science
Stanford University
Stanford, CA 94305
munie@stanford.edu

1 Introduction

Our motivation for this project was that we wanted to be able to quickly tell if the CS qualifying exams or the EE qualifying exams are more likely correctly pass or fail a student. This question is difficult to answer without a formal model, so we will first describe how the exams are actually conducted, and then propose our formal model. In the Electrical Engineering Department a given student would go from professor to professor; each professor would examine him/her independently, and at the end of the day the judgments of the professors would be aggregated into an overall score. In contrast, in the AI quals in the Computer Science Department, the student would be examined by a group of professors who would meet as a committee. In the course of our research we have found that finding the expected utility for either model is NP-Hard and even simply determining which method's utility (without computing the difference) is greater is NP-Hard. [1]

2 Formal Model

Our formal model is based on Bayesian networks. In this paper we consider only binary networks, in which each node can take on exactly two possible values, true and false (which we'll sometimes denote 1 and 0). In the single-agent case the network has the following structure. The nodes in the network are partitioned into three disjoint sets, S, U, and the singleton set R. Intuitively, U is the set of hidden knowledge nodes; each node captures whether the student does or does not know a particular topic. S is the set of questions one can ask, and R is the decision (or result) node determining whether the student should pass the exam. Consistent with this intuition, we have the following constraints on the network. Among nodes in U there can be arbitrary correlations; for example, the node representing the student's knowledge of probability theory might influence the likelihood that the student knows machine learning. The parents of nodes in S are restricted to the set U; thus, the nodes in S are all independent given U and the nodes in S have no children. R is a special node, and represents our criterion for passing, or failing, the student. It is the child of exactly every node in U and expresses a set function $\{0,1\}^{|U|} \mapsto \{0,1\}$ (that is, R has a deterministic CPT). For this paper R will be 1 (the student will pass) if more than θ percent of the nodes in U are 1, and R will be 0 otherwise.

This is a complete description of the student's knowledge, and whether he ought to pass, for any given choice of questions and answers to those. Figure 1 shows an example of such a network.

To complete the model we must specify constraints on the questions asked. To this end we associate with each node a cost. Every $s_i \in S$ has an associated cost c_i, which will be assumed to be the unit cost unless explicitly mentioned. All nodes in U and R have cost ∞. Finally, we assume an given budget b; such that the aggregate cost of the questions asked will not be allowed to exceed b.

[1] The motivation could be set up equally well in terms of sensor networks, but we stick to the exam story both because it was our motivation.

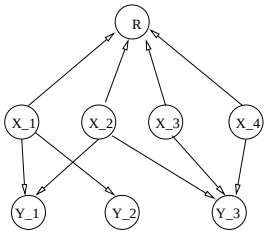

Figure 1: Sample single-agent network

In this paper we will focus on learning which of two models, the SMA (i.e. EE) and SMAC (i.e. CS), will perform better.

We capture the multi-agent (SMA and SMAC) models by duplicating each observable node as many times as there are professors, attaching a separate CPT to each node, and assigning each copy to a distinct professor. Each of these duplicated nodes is said to belong to the same question. The intuition behind this construction is that these duplicates represent a specific question that any of the professors could ask, but that they each draw different conclusions based on the answers (the individualized CPTs mean that the assumption that they all have access to the same set of questions is w.l.o.g.). Each professor has a budget of $\frac{b}{|P|}$, where P is the set of professors. In the SMA (Sequential Multi-Agent) mechanism, each professor i first selects a node, then based on the value of the node observed, the professor makes another observation until he fills his individual fraction of the budget. Each professor makes the observations independently. The results of the observations made by all of the professors are then used to compute the posterior on R, and make a pass/fail decision.

The SMAC model is a particular way to capture the operation of committees. In this model, all the professors are put into a committee. Each professor is still given the budget of $\frac{b}{|P|}$. Inside the committee, observations must be made by question group, not by individual node. What we mean by this is that all the nodes in the same question group that belong to professors inside the committee must be observed simultaneously, or none of the nodes observed at all. A committee will observe question groups until its budget is exhausted. After all the nodes in a particular question group have been observed, the choice of the next question group may be conditioned on the results of previous observations.

The intuition is that although fewer question groups will be examined in aggregate than in the SMA model, the total professor time spent will be the same.

The following will be stated without proof, but provide us our motivation to try and find under what conditions the SMA mechanism beats the SMAC mechanism.

Theorem 1

- *Given any budget b and set of researchers P, There are networks for which the best GSMAC mechanism outperforms the best SMA mechanism: the best SMAC yields an expected utility of 1, the highest possible, and the best SMA yields the expected utility of $.5 - \frac{|P|}{2^{b-|P|+1}}$.*
- *Given any budget b and set of researchers P, There are networks for which the best SMA mechanism outperforms the best SMAC mechanism, and it does so by the widest possible margin: the best SMA yields an expected utility of 1, the highest possible, and the best SMAC yields the expected utility of the trivial mechanism which decides based on the prior of R.*

Theorem 2 *Deciding if, on a specific network, $EU(SMAC) \geq EU(SMA)$ is NP-hard.*

3 Implementation

The data generation code is written in Python, and the machine learning code is in Matlab. The data generation is implemented as follows. We first create a random directed graph and reject the graph

if cycles are present. This graph will represent the set of U nodes. Then we populate the probability tables of the links with random values. After this is complete, we add a R node with a random θ value ranging from .5 to 1. For each node in U we add a group of question nodes with random probability tables, one for each professor, corresponding to a question group.

Now we have a random student network. We would like to know which mechanism, SMA or SMAC, has the higher expected utility on this network. Using the Bayesian network, we compute exactly the utilities of the SMA and SMAC mechanisms and output the result to a file.

In addition, for each network, we record the following features.

- The number of simply connected components in the U network.
- The diameter (Largest shortest path between any two nodes) of the U network.
- The number of edges in the U network
- The actual average link strength in the entire network
- θ
- Prior on R
- Maximum of R's prior
- Number of nodes with [0,1,2,3...] outgoing links
- Number of nodes with [0,1,2,3...] incoming links
- Number of nodes with [0,1,2,3...] total links

4 Results

Overall, the results were fairly inconclusive. Using SVM implemented with the SMO algorithm we were able to get 24% classification error on the test set, and using logistic regression we were able to get 27% error. We also tested with GDA, but the error was over 40% which seems to suggest that the data is not Gaussian. The test data had 27% of the examples with a true negative classification, so the results of our machine learning are not much better than guessing uniformly according to the prior. Both SVM and Logistic Regression performed almost as well on the test set as they did on the training set. It was difficult to find informative features so we also ran SVM with a quadratic kernel to get more features, but the error rate was still relatively high at 25%. The testing and training were made more difficult by the fact that generating enough training data was very slow, so for all of these examples we were training on data sets of size 200 and testing on data sets of size 85. However, since the error rate was similar on the test and training sets, it is more likely that our features are our problem, not the small number of training examples.

Below is a selection of our results with the error on the test set.

Algorithm	5 nodes and 2 observations	5 nodes and 4 observations
SVM	.2471	.2706
Logistic Regression	.2706	.2588

We graphed the outputs of both logistic regression and SVM according to the true classification of the data to see if there was any natural break, but the output looks very hard to classify and in fact may be independent of the labels. The data is presented in figures 2, and 3.

5 Conclusion

The initial motivation for applying machine learning to this problem was that in other NP-Hard problems like k-SAT, there appears to be a natural phase transition when measured according to the number of variables per clause. Instances that have a ratio less than this phase transition are very likely to be satisfiable, and instances that have a ratio greater than this phase transition are very likely to be unsatisfiable. Ideally, we would be able to find this phase transition via automated methods, but it is very dependent on the feature. In the particular problem we worked on, it doesn't appear that we have found the right feature, if one exists.

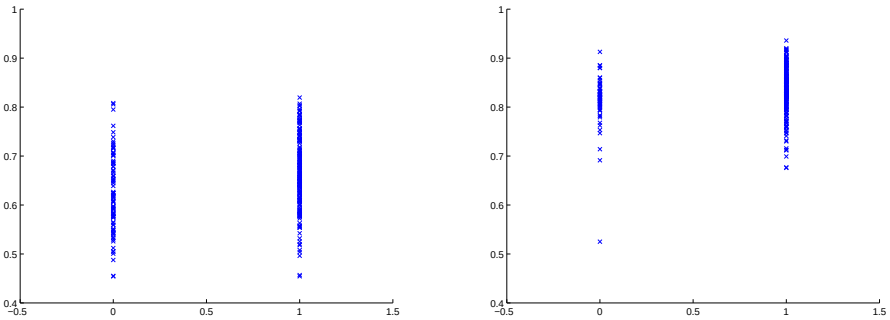

Figure 2: Logistic Regression on 5 node networks with budgets of 2 (left) and 4 (right)

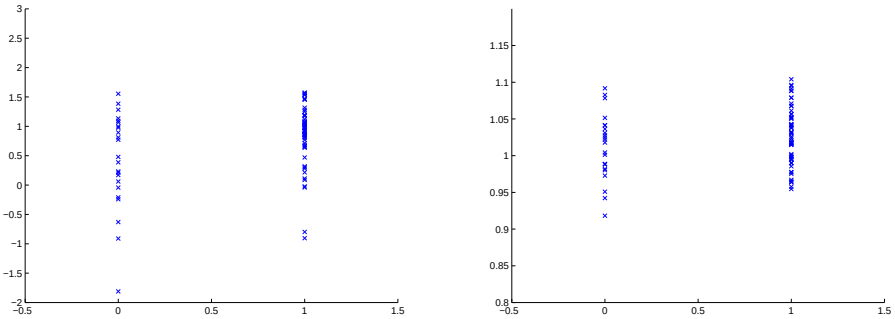

Figure 3: SVM on 5 node networks with budgets of 2 (left) and 4 (right)

The problem of predicting which structure of professors has greater expected utility proved to be a difficult problem to apply machine learning to. The results are better than simply going with the prior, but perhaps not good enough to be dismissed as more than a random variation. Through this project I personally have learned a lot about applying machine learning, and hopefully, with some more work, I can produce some more positive results on this problem.

6 Future Work

The initial results are a little unsatisfactory, but there are areas in which I can hopefully improve them. First, my data set was a little too small, and with more time I should be able to generate a larger set. Secondly, and more importantly, the features I have come up with are insufficient. Hopefully, through a little more work to gain insight into the structure of these networks, hopefully I can find a better set of features. I would also like to acknowledge that this work is based on joint work done with Yoav Shoham and with advice on the machine learning from Haidong Wang.

References

[1] Andreas Krause and Carlos Guestrin. *Optimal Nonmyopic Value of Information in Graphical Models - Efficient Algorithms and Theoretical Limits*. in *Nineteenth International Joint Conference on Artificial Intelligence,* 2005.

[2] Andreas Krause and Carlos Guestrin. *Near-Optimal Nonmyopic Value of Information in Graphical Models*. in *21st Conference on Uncertainty in Artificial Intelligence,* 2005.

[3] D. Kempe, J. Kleinberg and E. Tardos. *Maximizing the Spread of Influence through a Social Network*. in *SIGKDD,* 2003.

[4] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, *Introduction to Algorithms*, McGraw-Hill, Boston, MA, 2001.

[5] Robert J. McEliece, *The Theory of Information and Coding*, Cambridge University Press, Cambridge, United Kindom, 2002.