STUDENT PERFORMANCE EVALUATION

PROJECT REPORT

Team Members :

Girija Godbole gsg160130

Vidya Gopalan vxg170530

Srividya Venkipurampathangi sxv161930

Introduction and problem description:

The project involves determining the rate of learning of students, using their interactions with an intelligent tutoring system. The dataset used to train the machine learning model contains logs of such interactions where students' responses to mathematical problems (algebra specifically) are recorded. Using these logs, the algorithm will attempt to learn the concept or the mathematical problem. The tutor's evaluation of the students' responses in the training data helps the system learn the concept and later predict if the student's attempt at later problems are correct or not.

Various features such as duration of time taken to solve the problem, number of hints asked for, number of incorrect responses to a problem, etc, are used to build the model. The dataset contains the responses of various students to various kinds of problems. Out system aims to predict the correctness of a particular student's solution to problems based on his previous performances. It classifies the result for each student and then makes generalizations as to what problems are generally considered hard and also how the rate of improvement varies between students.

The "knowledge component" of the actual algebraic concept contained in the dataset is used by the system to identify similarities between problems, and decide on what knowledge or skill can be commonly employed to solve different problems (e.g, multiplication can be used to find both area of a rectangle and area of a circle. Therefore, a student who correctly answers questions regarding finding area of rectangles would know multiplication well, thereby, leading to a prediction that he would mostly answer questions regarding area of circles correctly. In this case, more weightage would be given to the feature "skill" that will contribute to the overall prediction). Several such knowledge components will typically be involved in solving a problem, leading to us considering all of them separately and assigning appropriate weights to each.

Our model therefore, aims to infer the skill requirements of a problem by analyzing the data containing students' performances. Achieving this would be of great practical importance, as it would lessen human efforts to analyze such tasks and also make the process of evaluation speedier.

Related Work:

Machine Learning is majorly concerned with classification of data and class prediction for new data. Much attention has been given to predicting student performance in problem solving or completing courses. Progressive prediction of student performance in college programs, done by information sciences institute, predicts KNN yields the highest accuracy. Early Prediction of Students Performance using Machine Learning Techniques, done by students of St.Xavier's college, Kolkata applied feature selection algorithms to reduce the number of features and predicted that decision tree algorithms gave the best result. Predicting Postgraduate Students' Performance Using Machine Learning Techniques done by Department of Informatics, Ionian University said that Naïve Bayes and 1-NN achieved the best prediction results.

Dataset description: (features and attributes)

Features: The total number of features in our dataset is 19. The features are as follows:

1. Row: the row number Update (04-20-2010): for challenge data sets, the row number in each file (train, test, and submission) is no longer taken from the original data set file. Instead, rows are renumbered within each file. So instead of 1…n rows for the training file and n+1..m rows for the test/submission file, it is now 1…n for the training file and 1…n for the test/submission file.

2. Anon Student Id: unique, anonymous identifier for a student

3. Problem Hierarchy: the hierarchy of curriculum levels containing the problem.

4. Problem Name: unique identifier for a problem

5. Problem View: the total number of times the student encountered the problem so far.

6. Step Name: each problem consists of one or more steps (e.g., "find the area of rectangle ABCD" or "divide both sides of the equation by x"). The step name is unique within each problem, but there may be collisions between different problems, so the only unique identifier for a step is the pair of problem_name and step_name.

7. Step Start Time: the starting time of the step. Can be null.

8. First Transaction Time: the time of the first transaction toward the step.

9. Correct Transaction Time: the time of the correct attempt toward the step, if there was one.

10. Step End Time: the time of the last transaction toward the step.

11. Step Duration (sec): the elapsed time of the step in seconds, calculated by adding all of the durations for transactions that were attributed to the step. Can be null (if step start time is null).

12. Correct Step Duration (sec): the step duration if the first attempt for the step was correct.

13. Error Step Duration (sec): the step duration if the first attempt for the step was an error (incorrect attempt or hint request).

14. Correct First Attempt: the tutor's evaluation of the student's first attempt on the step - 1 if correct, 0 if an error.

15. Incorrects: total number of incorrect attempts by the student on the step.

16. Hints: total number of hints requested by the student for the step.

17. Corrects: total correct attempts by the student for the step. (Only increases if the step is encountered more than once.)

18. KC(KC Model Name): the identified skills that are used in a problem, where available. A step can have multiple KCs assigned to it. Multiple KCs for a step are separated by ~~ (two tildes). Since opportunity describes practice by knowledge component, the corresponding opportunities are similarly separated by ~~.

19. Opportunity(KC Model Name): a count that increases by one each time the student encounters a step with the listed knowledge component. Steps with multiple KCs will have multiple opportunity numbers separated by ~~

The class label used is Correct First attempt. The learners used predict this value that is then checked against the given value.

Number of instances: We would be using 2357 instances for training and 590 instances as testing data.

Pre-processing techniques:

We removed the columns "Step Start time" and "Step End time" as they are consolidated in the column "Step duration" and were redundant. The column "First Transaction time" is also removed as it seems to have no correlation to the output. This was done using R.

The class label column (which is "Corrected First attempt") is then converted as the last column of the dataset (for ease of operations during the coding phase).

We selectively removed the rows containing null values for the columns excluding Correct Step Duration, Error step duration and Correct Transaction Time. This was done using R.

In the columns "Correct step duration", "Error step duration" and "correct transaction time", the null values were replaced with a 0 as the values in these columns were blank for a reason. The correct step duration was missing whenever the step was not correct. Similarly, the error step duration was missing whenever the step was correct. The correct transaction time was blank for an erroneous step. Therefore, removing the entire row for these missing values would result in loss of crucial data. The blank cells were replaced with zero at the excel sheet itself.

We next converted the categorical or "factored" features into unique numeric ones for ease of operations. This was done using the as.numeric function (after factoring) in R.

The features "KC" or Knowledge Component and "Opportunity" were tricky as they contained multiple values. They contains the different steps taken by a student to solve the problem (with the steps separated by '~~') and number of times each step was encountered (this is Opportunity) respectively. It is handled by assigning a unique integer value for each step (and opportunity) and concatenating it with the succeeding steps' integers. This was done using Java.

Eg, if the KC is : "Placing coordinate points~~Identifying Units", the value assigned would be say 23. And if another KC is "Define variable~~ Identifying Units~~ Placing coordinate points", the value assigned would be 432. We finally sort the digits of a number so that 123 and 321 are finally the same (as they technically indicate the same steps in a different order).

As a last step, the data is normalized to scale. This was done using appropriate R functions.

Finally, the pre-processed data we get is free of redundant features and is clean of rows that had missing values. The data obtained is numeric and appropriately scaled, ready to be fed into the learners.

Proposed solution and methods: (theoretical and practical)

We have decided from preliminary results (where we tested the dataset using classifiers Naïve Bayes, SVM-using a Lagrangian multiplier, Deep learning and Ada Boosting) that Deep Learning and Ada Boosting give the best results for this particular dataset.

We are hence, including details of various experiments done using deep learning techniques and ada boosting, both of which did exceptionally well. These two were excellent contestants for our proposed solution. We however, have decided that deep learning technique (with 5 layers-with number of nodes as-6,5,4,3,2) is our best model as the accuracy is the best with this particular combination. The MSE value obtained is 0.000024 and we observe that almost all of the test data get correctly classified.

While getting very good results with ada boosting as well, based on the time taken to run, the error value obtained and the weight given to each feature (this was consistent in terms of proportion when using deep learning and helped us make conclusions regarding which feature was more important in predicting) we have decided on the deep learning model as our final solution.

The features which are the most influential in predicting whether the student gets the answer correct or wrong are: Error step duration, correct step duration, Hints and Incorrects. This conclusion is consistent with our intuition as well, as the time taken (duration) to get a question right or wrong is a good predictor as to whether a student would get a future question correct(if time taken to get a question right is less, then the student has a higher chance of getting the same question right in future). Similarly, if number of incorrects for a question is more, then there's less chance of the student doing it correctly in future. Hints is important because if more hints are provided there are higher chances of making a correct first attempt.

We were also able to make generalizations as to which problems were the hardest to solve by the students based on how many questions of each problem were incorrectly solved by the students. The hardest problem turned out to be "L5FB02". A separate java code was used to determine the hardest problem. The dataset used for this (the one gotten after prediction) contains just the problem name and the class label after removing those rows where an incorrect prediction was made. This code is attached with the folder.

Our other metric for comparison was the ROC curve. This curve turned out to great for both techniques as well.

Experimental Results and analysis:

Sr No.	**Classifier**	**Parameters tested**	**Best set of parameters**	**Accuracy**	**Error**
1	**Neural Networks**	**1.Hidden layer=1** **2. Hidden unit=1**	**1.Hidden layer=1** **2. Hidden unit=1**	**92.8**	**0.0720608698703** **MSE= 0.0002170512721**
2.	**Deep Learning**	**Hidden layers=2** **Hidden units(5,3)**		**99.51**	**0.004927779420** **MSE=0.00003462320789**
3.	**Deep Learning**	**Hidden layers=2** **Hidden units(5,2)**		**99.12**	**0.008867927028** **MSE=0.000005844022231**
4.	**Deep Learning**	**Hidden Layers= 3** **Hidden units(5,4,3)**		**99.5**	**0.005077164688** **MSE= 0.00002403638696**
5.	**Deep Learning**	**Hidden Layers=5** **Hidden Units(6,5,4,3,2)**	**Hidden layers=5** **Hidden units(6,5,4,3,2)**	**99.99**	**0.00006005222162** **MSE= 0.00002403638696**
6.	**Deep Learning**	**Hidden Layers=7** **Hidden units(7,6,6,5,4,3,2)**		**99.99**	**0.0000600002312** **MSE= 0.00002403638696**
7.	**Boosting**	**weights= 3.453377389**		**98**	**0.02**

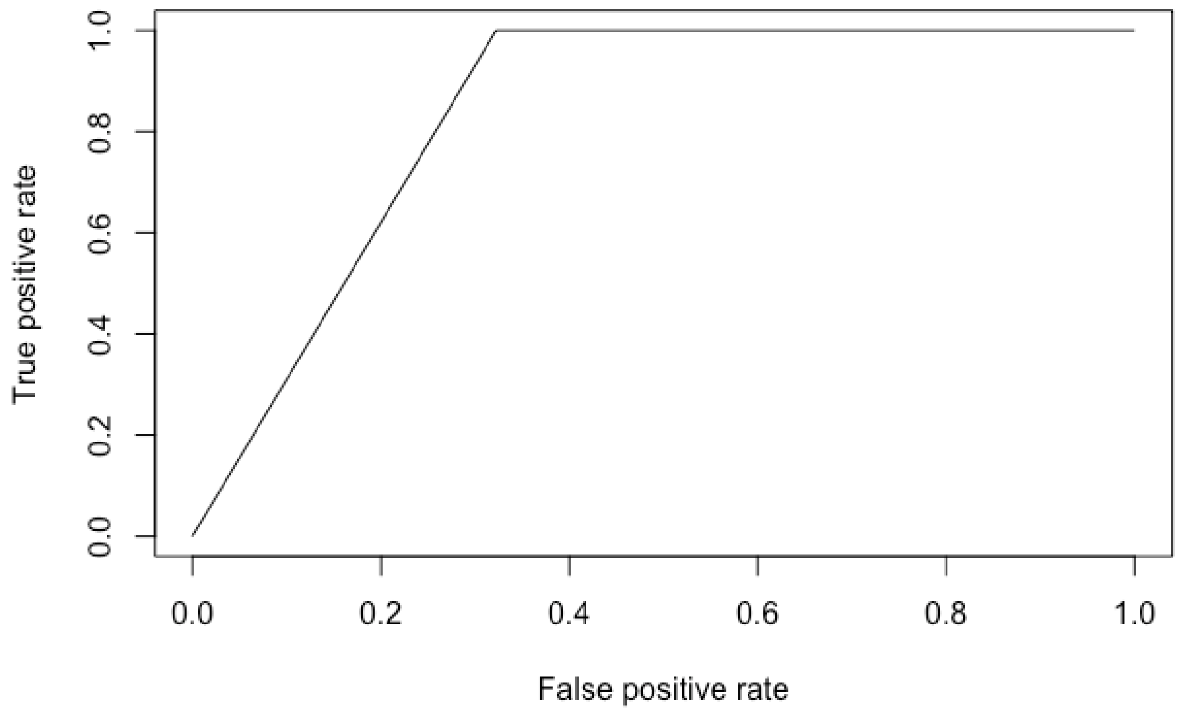

Fig 1 – ROC curve for boosting.

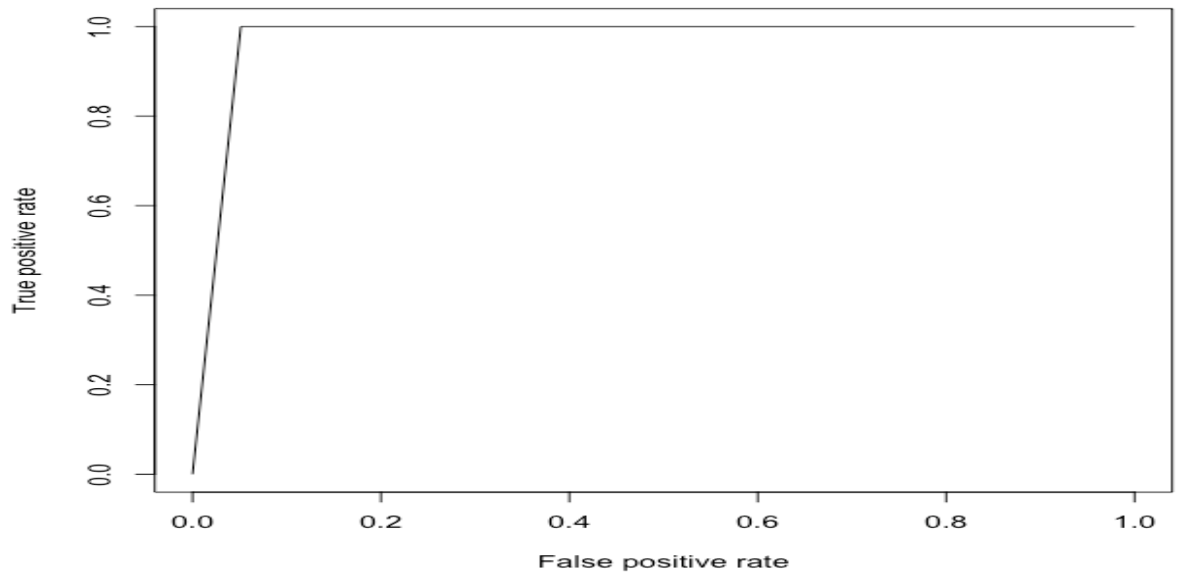

Fig 2 – ROC curve for deep learning.

Output for our model:

```
Anon_Student_id.to.1layhid1          -0.86060231772769
Problem_Hierarchy.to.1layhid1        -0.60063969592082
Problem_Name.to.1layhid1              0.13900484685715
Problem_View.to.1layhid1             -1.81234708248784
Step_Name.to.1layhid1                -0.60744563246814
Correct_Transaction_Time.to.1layhid1  0.92745887644699
Step_Duration.to.1layhid1            -4.34282153850028
Correct_Step_Duration.to.1layhid1    23.48638207295761
Error_Step_Duration.to.1layhid1     -25.27956023422727
Incorrects.to.1layhid1              -16.30906183461250
Hints.to.1layhid1                   -13.10815423400201
Corrects.to.1layhid1                  0.93218676678457
KC.to.1layhid1                       -0.40244766829201
Opportunity.to.1layhid1               0.36252002767406
Intercept.to.1layhid2                 0.68364969479305
Anon_Student_id.to.1layhid2           1.02221212098414
Problem_Hierarchy.to.1layhid2         2.06786045302102
Problem_Name.to.1layhid2             -0.39227825833576
Problem_View.to.1layhid2             -0.71854055094927
Step_Name.to.1layhid2                 0.00440233567020
Correct_Transaction_Time.to.1layhid2 -0.42013017707674
Step_Duration.to.1layhid2            13.64070078613724
Correct_Step_Duration.to.1layhid2   -20.59502594829238
Error_Step_Duration.to.1layhid2      22.36615026429222
Incorrects.to.1layhid2               15.73925107938117
Hints.to.1layhid2                    20.06971176973122
Corrects.to.1layhid2                 -2.16328714864762
KC.to.1layhid2                       -0.38912507814852
Opportunity.to.1layhid2              -1.56710975195270
Intercept.to.1layhid3                -0.92764660009747
Anon_Student_id.to.1layhid3           0.88752861406689
Problem_Hierarchy.to.1layhid3         0.27040758783398
Problem_Name.to.1layhid3             -0.23235793896989
Problem_View.to.1layhid3              1.00034697368452
Step_Name.to.1layhid3                -0.08363394384102
Correct_Transaction_Time.to.1layhid3 -0.10950950597537
Step_Duration.to.1layhid3            -5.47125729986615
Correct_Step_Duration.to.1layhid3    25.64596952394947
Error_Step_Duration.to.1layhid3     -23.80634903969010
Incorrects.to.1layhid3              -11.34215848283664
Hints.to.1layhid3                    -2.91538507145659
Corrects.to.1layhid3                  4.60825526178130
KC.to.1layhid3                       -0.55003996422436
Opportunity.to.1layhid3              -0.01373928926738
Intercept.to.1layhid4                -0.73537650067074
Anon_Student_id.to.1layhid4          -0.15430985395265
Problem_Hierarchy.to.1layhid4         0.66837345727877
Problem_Name.to.1layhid4              1.13503220399038
Problem_View.to.1layhid4             -0.50414775736393
Step_Name.to.1layhid4                 0.78568678246244
Correct_Transaction_Time.to.1layhid4 -0.32846824765244
Step_Duration.to.1layhid4            -7.83771020341336
Correct_Step_Duration.to.1layhid4    23.11948375932922
Error_Step_Duration.to.1layhid4     -24.85986888102028
Incorrects.to.1layhid4              -13.44615797368716
Hints.to.1layhid4                    -6.12542993459132
Corrects.to.1layhid4                  2.22131970481149
KC.to.1layhid4                       -0.56972636186193
Opportunity.to.1layhid4              -0.67513496384014
Intercept.to.1layhid5                 0.44717448623857
Anon_Student_id.to.1layhid5           0.17978269129999
Problem_Hierarchy.to.1layhid5        -1.08356644462319
```

```
Problem_Name.to.1layhid5              1.44772936430928
Problem_View.to.1layhid5             -0.07768529820593
Step_Name.to.1layhid5                -0.44510144136127
Correct_Transaction_Time.to.1layhid5  0.17379307548129
Step_Duration.to.1layhid5             6.89565182347607
Correct_Step_Duration.to.1layhid5   -23.26094200577451
Error_Step_Duration.to.1layhid5      24.33158563810958
```

The logs for all the experiments performed have been included in a **separate log file.**

Analysis:

The neural network with 1 hidden layer and 1 hidden unit gave us the accuracy of 92.8%. As we increased the number of hidden layers, we found that the accuracy went on increasing upto a certain number of hidden layers. It was around 99.99%. After a point, the accuracy remained constant. Hence, we decided to consider the neural net with 5 hidden layers and hidden units as (6,5,4,3,2) to maintain the minimum complexity. Boosting gave 98% accuracy.

Conclusion:

We therefore, conclude that the classification method of **deep learning** coupled with **ada boosting** gives a very good accuracy when predicting students' performances based on the dataset used and also in making generalizations regarding which problems are found harder compared to the rest. Using this particular ensemble mechanism reduces the variance and gives great results with high precision.

Contribution of team members:

Concepts and ideas were discussed and suggestions were evaluated on feasibility. The project was done as a group.

References:

- http://www.kdd.org/kdd-cup/view/kdd-cup-2010-student-performance-evaluation/Data
- Machine Learning - Thomas Mitchell
- https://cran.r-project.org/web/packages/adabag/adabag.pdf **-** Package adabag
- https://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf **-** Package neural net
- Predicting Student Performance: A Solution for the KDD Cup 2010 Challenge, Yongming Shen, Qiong Chen, Ming Fang, Qiuyong Yang, Tingting Wu, Lixiong Zheng, Zongfa Cai, Journal of Machine Learning Research.
- Early Prediction of Students Performance using Machine Learning Techniques, International Journal of Computer Applications (0975 – 8887) Volume 107 – No. 1, December 2014, Anal Acharya, Devadatta Sinha.
- Predicting Postgraduate Students' Performance Using Machine Learning Techniques Maria Koutina , Katia Lida Kermanidis
- Progressive Prediction of Student Performance in College Programs, AAAI Publications, *Jie Xu, Yuli Han, Daniel Marcu, Mihaela van der Schaar*