Machine Learning Engineer Nanodegree

Capstone Proposal

Michael Feng
May 16th, 2017

Proposal

Education Data Mining Challenge

How generally or narrowly do students learn? How quickly or slowly? Will the rate of improvement vary between students? What does it mean for one problem to be similar to another? It might depend on whether the knowledge required for one problem is the same as the knowledge required for another. But is it possible to infer the knowledge requirements of problems directly from student performance data, without human analysis of the tasks?

We are gonna to predict student performance on mathematical problems from logs of student interaction with Intelligent Tutoring Systems.

In order to presents interesting technical challenges, has practical importance, and is scientifically interesting.

From a practical perspective, improved models could be saving millions of hours of students' time (and effort) in learning algebra. These models should both increase achievement levels and reduce time needed. Focusing on just the latter, for the .5 million students that spend about 50 hours per year with Cognitive Tutors for mathematics, let's say these optimizations can reduce time to mastery by at least 10%. One experiment showed the time reduction was about 15% (Cen et al. 2007). That's 5 hours per student, or 2.5 million student hours per year saved. And this .5 million is less than 5% of all algebra-studying students in the US. If we include all algebra students (20x) and the grades 6-11 for which there are Carnegie Learning and Assistment applications (5x), that brings our rough estimate to 250 million student hours per year saved! In that time, stu-

dents can be moving on in math and science or doing other things they enjoy.

From a scientific viewpoint, the ability to achieve low prediction error on unseen data is evidence that the learner has accurately discovered the underlying factors which make items easier or harder for students. Knowing these factors is essential for the design of high-quality curricula and lesson plans (both for human instructors and for automated tutoring software). So we have the potential to influence lesson design, improving retention, increasing student engagement, reducing wasted time, and increasing transfer to future lessons.

To this end, a model which accurately predicts long-term future performance as a byproduct of day-to-day tutoring could augment or replace some of the current standardized tests: this idea is called "assistment", from the goal of assessing performance while simultaneously assisting learning. Previous work has suggested that assistment is indeed possible: e.g., an appropriate analysis of 8th-grade tutoring logs can predict 10th-grade standardized test performance as well as 8th-grade standardized test results can predict 10th-grade standardized test performance (Feng, Heffernan, & Koedinger, 2009). But it is far from clear what the best prediction methods are; so, our algorithms may provide insights that allow important improvements in assistment.

Domain Background

If a student is correct at one problem (e.g., "Starting with a number, if I multiply it by 6 and then add 66, I get 81.90. What's the number?") at one time, how likely are they to be correct at another problem (e.g., "Solve for x: 6x+66=81.90") at a later time?

These questions are of both scientific interest and practical importance. Scientifically, relevant deep questions include what is the nature of human knowledge representations and how generally do humans transfer their learning from one situation to another. Human learners do not always represent and solve mathematical tasks as we might expect. You might be surprised if you thought that a student working on the second

problem above, the equation 6x+66=81.90, is likely to be correct given that he was correct on the first problem, the story problem. It turns out that most students are able to solve simple story problems like this one more successfully than the matched equation (Koedinger & Nathan, 2004; Koedinger, Alibali, & Nathan, 2008). In other words, there are interesting surprises to be found in student performance data.

Cognitive Tutors for mathematics are now in use in more than 2,500 schools across the US for some 500,000 students per year. While these systems have been quite successful, surprises like the one above suggest that the models behind these systems can be much improved. More generally, a number of studies have demonstrated how detailed cognitive task analysis can result in dramatically better instruction (Clark, Feldon, van Merriënboer, Yates, & Early, 2007; Lee, 2003). However, such analysis is painstaking and requires a high level of psychological expertise. We believe it possible that machine learning on large data sets can reap many of the benefits of cognitive task analysis, but without the great effort and expertise currently required.

Problem Statement

A step is an observable part of the solution to a problem. Because steps are observable, they are partly determined by the user interface available to the student for solving the problem. (It is not necessarily the case that the interface completely determines the steps: for example, the student might be expected to create new rows or columns of a table before filling in their entries.)

In the example problem above, the steps for the first question are:

- find the radius of the end of the can (a circle)
- find the length of the square ABCD
- find the area of the end of the can
- find the area of the square ABCD
- find the area of the left-over scrap

This whole collection of steps comprises the solution. The last step can be considered the "answer", and the others are "intermediate" steps.

A step record is summary of all of a given student's attempts for a given step. Training data are step records of different students on different exercises.

These data sets come from multiple schools over multiple school years. We gonna use thousands of student-step (a record of a student working on a step) to predict **Correct First Attempt**.

Correct First Attempt: our prediction value, a decimal number between 0 and 1, that indicates the probability of a correct first attempt for a student-step.

The target data is 0 or 1 makes the problem likely to be a classification problem, so we gonna use some ensemble machine learning model to make prediction based on training data. But in order to give a RMSE measure result, we also need to output CFA’s probability so that we can get RMSE for predicted result.

In this section, clearly describe the problem that is to be solved. The problem described should be well defined and should have at least one relevant potential solution. Additionally, describe the problem thoroughly such that it is clear that the problem is quantifiable (the problem can be expressed in mathematical or logical terms) , measurable (the problem can be measured by some metric and clearly observed), and replicable (the problem can be reproduced and occurs more than once).

Datasets and Inputs

Data sets	Students	Steps	File
Algebra I 2008-2009	3,310	9,426,966	algebra_2008_2009.zip

Stamper, J., Niculescu-Mizil, A., Ritter, S., Gordon, G.J., & Koedinger, K.R. (2010). *[Data set name]. [Challenge/Development]* data set from KDD Cup 2010 Educational Data Mining Challenge. Find it at http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp

The data sets are provided for familiarizing ourselves with the format and developing our learning model.

For a description of the format of the data, we can see the official Data Format page.

We will split the train data into 3 part: training set, validation set and test data set. Use training set to train our model, validation set to validate model, then test set to know model's performance on unidentified data.

Solution Statement

Based on existing data given by the competition, we only have target feature **Correct First Attempt**. Which only contains value 1 and 0. So we can use classifier to predict the target result. We will train a linear classifier on training data set with validation data set together.

But the competition ask for probability of student's probability of a correct first attempt. So it will transform the problem from classification to regression problem.

We proposed replacing each categorical feature with a numerical one by using the "correct first attempt rate" (CFAR)

Take the student name feature as an example. The CFAR of a specific student name "sid" is defined as

$$\mathrm{CFAR} \equiv \frac{\#\text{steps with student name} = \text{sid and CFA} = 1}{\#\text{steps with student name} = \text{sid}}.$$

For each step whose student name is sid, the above CFAR would be used as the corresponding feature value. This setting directly connects a feature and CFA, which is now the target for prediction. We will consider CFARs for student name, step name, problem name, KC, (problem name, step name), (student name, problem name), (student name, unit name) and (student name, KC). But not practice on every combination.

So that the classification problem then can be transformed to regression problem.

Therefore, we can measure our model on test data with RMSE.

Which can be represented as follow:

y': Predicted target probability of First Correct Attempt(numpy ndarray)
y: Target's original CFAR(numpy ndarray)
n: Number of data records

RMSE = np.sqrt(np.sum(np.square(y' - y)) / n)

Our goal is to minimize the RMSE on test data set which comes from portion of training data set.
The test data set in competition's original data is used to predict and submit to competition's leader board so that attendants can achieve their ranks. But in our experiments, there is no leader board. So we only use portion of training data as test data. That's important to know.

Benchmark Model

To avoid overfitting, we will Cross Validation(CV) skill to shape our model. But for this situation, CV is likely not a suitable approach due to the temporal property of the data. So we will generate a validation pair (V, V′) with a distribution similar to the distribution of the original data training set.

Initial development was conducted on an internal split of training data for training and validation. For each unit name, We gonna collected the last problem from units with the same name to form a training subset.

V′: Generation of the validation set

V： An internal training set other than V′

So we will minimize the RMSE both on V′ and test set as low as possible. The benchmark model's performance could be measured by training output. We can print the RMSE for each training process to measure our model's performance.

Evaluation Metrics

Since the data comes from KDD CUP 2010 competition. Therefore, we don't have an unidentified portion to use for validation. In order to validate our data. We will only train on the training portion of each data set, meanwhile, use part of training portion data as validation set, and will then be evaluated on our performance at providing **Correct First Attempt** values for the test portion(part of training data).

We will compare the predictions we provided against the undisclosed(test dataset) true values and report the difference as **Root Mean Squared Error (RMSE).**

The square root of the mean/average of the square of all of the error.

The use of RMSE is very common and it makes an excellent general purpose error metric for numerical predictions.

Compared to the similar Mean Absolute Error, RMSE amplifies and severely punishes large errors.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

We will dedicate our best to acquire the low-

est RMSE value as possible.

Project Design

Firstly, we gonna to load data and prepare data.

Secondly, do one-hot encoding and regularization on numeric features for train data.

Thirdly, we gonna visualize data into chart, so that we can see trends and scatters and etc. Removes scatters and add potential feature to data. We gonna use GMM to cluster data into different clusters so that we can add additional features.

Finally, we will use ensemble algorithms model to train data(train set and validation set). For e.g., XGBoost, LightGBM, GBDT or etc. And tuning hyper parameters. Then predict result on test data set. Compute **RMSE**.

The above procedure will go through V(training set), V'(validation set), and test set.

Noticing: Using notebook to present details would be a good practice.

LightGBM is a fast, distributed, high performance gradient boosting (GBDT, GBRT, GBM or MART) framework based on decision tree algorithms, used for ranking, classification and many other machine learning tasks. It supports multi CPU cores' computation. My laptop a MBP(2015 Mid) only contains an AMD GPU which is not supported well by NVIDIA's cuDNN. So I'd like to use some algorithms such as LightGBM or xGBoost so that computation could be accelerated by multi cpu cores.

There are some features with string type data but may have highly dependency with prediction result. Such as: unit name, section name, problem name, step name and knowledge components. These features may be considered using some clustering tricks or feature combining skills to process.