Machine Learning Engineer Nanodegree
Capstone Proposal
Donia Robinson
April 17, 2017

KDD Cup 2010 Educational Data Mining Challenge

Domain Background

Mined data in the education domain has many possible uses. One such use is to predict future performance based on day-to-day tutoring methods. The 2010 KDD Cup uses the term "assistment", which means to assess performance while simultaneously assist learning. It would be useful if a student could ultimately reach the same level of proficiency, but do so with less learning hours. Those hours can then be spent doing something else. If tutoring is being paid for, this saves money as well. (As a parent of 3 junior-high and high-school children, the savings of time and money is always appreciated!)

In this project, student answers to algebra problems in online tutoring programs were recorded. These answers were recorded in such a way that analysis can be done to determine which portions of a problem a student has mastered or needs help with. These are logged as "transactions," and will be discussed in detail later on. This is similar to cognitive task analysis, which has been shown to result in improved methods of instruction (Clark, Feldon, van Merriënboer, Yates, & Early, 2007). Simply knowing if a student got a question right or wrong is a start, but knowing which parts of a question the student struggled on is much more valuable.

Clark, R. E., Feldon, D., van Merriënboer, J., Yates, K., & Early, S. (2007). Cognitive task analysis. In J. M. Spector, M. D. Merrill, J. J. G. van Merriënboer, & M. P. Driscoll (Eds.), Handbook of research on educational communications and technology (3rd ed., pp. 577-593). Mahwah, NJ: Lawrence Erlbaum Associates.

Problem Statement

The overall problem to be solved in this project is predicting whether or not a student got a step right (the first time) on a math problem. The prediction will be a numerical value between 0 and 1. At run time, each data set will be divided into a training and test set. The performance of the algorithm will be judged by comparing the predicted values for the test sets against the undisclosed true values. The difference is calculated using Root Mean Squared Error (RMSE). The smaller this number is, the better the algorithm is performing.

Datasets and Inputs

Because this project was originally a competition, the necessary data sets are well-defined. There are 5 data sets. There are a minimum of 19 inputs per data set, so I am not going to list

out each one (would simply be a listing exercise). However, there are a few groups of features that I anticipate being especially useful. The contents of the "Correct First Attempt" column becomes the labels for the data. Ultimately, this is what the algorithm is supposed to learn. "KC Model Name," which lists the skills used in that problem, will allow the problems to be connected via mathematical topic. "Step Name," if parsed out, may help with this as well. The "Hints" column will be interesting to look at. For example, does a student who requests a lot of hints generally go on to get the problem correct or incorrect? The time measurements may also provide insight.

Citation of data, per the KDD Cup website: "The project will use 5 data sets from 2 different tutoring systems. These data sets come from multiple schools over multiple school years. The systems include the the Carnegie Learning Algebra system, deployed 2005-2006 and 2006-2007, and the Bridge to Algebra system, deployed 2006-2007. The development data sets have previously been used in research and are available through the Pittsburgh Science of Learning Center DataShop (as well as this website)."

Solution Statement

The general solution (see "Project Design" for additional details) for the project will be to implement a supervised learning algorithm. In particular, care will need to be taken to accommodate a sparse data matrix, as well as handle the temporal relationships in the data (students improve over time).

As mentioned above in the "Problem Statement" section, the Root Mean Squared Error will be treated as the quantifiable result. It can be compared to the results on the KDD Cup 2010 leaderboard. (See note under "Evaluation Metrics" about how the test set will differ from the competition test set, though.)

Benchmark Model

One model of this problem that could be used as a benchmark involved the following steps: pre-process features; train sparse feature sets using logistic regression; condense features; apply random forest. By all appearances, this is a job for a well-tuned supervised learning algorithm.

The benchmark model result is the Root Mean Squared Error. The scores for the top teams in the 2010 competition are listed at: https://pslcdatashop.web.cmu.edu/KDDCup/results_full.jsp In particular, the winning team (mentioned above) ended up with a KDD "cup score" of 0.272952.

Evaluation Metrics

The evaluation metric that will be used to quantify the performance of the algorithm is Root Mean Squared Error. As mentioned in the "Datasets and Inputs" section, one column of the

data indicates if the student got the correct answer on the first try or not (0 or 1). (This is true for all data sets except the test sets under the "challenge data sets." It is possible to upload results to the competition website to have the algorithm scored. Rather than do that, I plan to split the training sets of the challenge data sets into training and testing portions, and evaluate that way.)

As previously mentioned, the prediction (y_pred) being made by the algorithm is a number between 0 and 1. Root Mean Squared Error will be calculated using the data from the "Correct First Attempt" column (y) and the prediction. RMSE = sqrt(mean((y-y_pred)^2))

Project Design

1. **Prepare Data** – The first step in my workflow would be to prepare/pre-process the data. The two columns representing a start and end time for an operation can be combined into a single column, length of time. Length of time may also need to be normalized. Additionally, I plan to take one of the data sets and look at information such as the min, max, mean, and standard deviation for certain features.
2. **Select Algorithm/Fit the Model** – Because a value between 0 and 1 should be returned, I would look into probabilistic classifiers first. A two-class logistic regression meets both of those requirements. I might also consider using PCA prior to the logistic regression, to simplify the input data slightly.
3. **Select Validation Method** – Three of the five data sets are already divided into training and test sets. As additional validation, however, these sets could be further broken down and k-fold validation applied.
4. **Use Fitted Model for Predictions** – This is the big moment, where the model will be run. The RMSE can be calculated at this point, to give a sense of how well the model is performing. That number alone can't help diagnose where issues in the model may be, though, so additional validation is necessary.
5. **Run Validation Method**
6. **Tweak Model** – I expect a lot of time to be spent here. Fine-tuning the model can make a huge difference in performance. As needed, I plan to generate graphs (such as biplots) to give graphical representations of how the features are related. For me, this can be more intuitive to understand than a list of numbers.
7. **Repeat steps 4-6 as needed**