Click-through rate prediction using an ensemble of neural networks

Julian de Wit

Location : Hoek van Holland  
Email : [julian@dwssystemen.nl](mailto:julian@dwssystemen.nl)  
Competition : Criteo Labs display advertising challenge  
Date : 30 september 2014

# Summary

This document describes the 4th prize solution to the Criteo Labs display advertising challenge hosted by Kaggle.com. The solution consisted of two steps. First the data was preprocessed. Rare and unseen test-set categorical values were all encoded as one category. The remaining features were one-hot encoded or hashed so that we ended up with roughly 128K separate sparse features. The second step was to use deep neural networks to train a number of different models with variations coming from different network architectures, bagging and preprocessing parameters. The different models were averaged into one final solution.

# Features selection and extraction

At the start of the competition it quickly turned out that linear models using logistic regression would be very effective [1][2]. However, to improve the predictions one had to manually, or semi-automaticall, do feature engineering to find interesting tranformations or interactions. Our goal was to try to use a deep neural network which is potentially a more ‘potent’ model that could learn many interesting features automatically. This approach was reasonably succesful. However, some feature engineering was still necessary.

First we had to overcome technical problems. There were potentially more than 40M separate feature values and the first hidden layer of the network contained 128-512 units. This would result in a ~5 gigafloat weight matrix which would not fit on the GPU. Also, learning times would be unfeasible. We chose to do feature reduction by encoding all rare feature values with one code. After this some categories still had 100K+ distinct values. It was decided to hash these features to a smaller space [3] to guarantee memory usage boundaries. Features with less than 10K distinct values were One-Hot encoded because we hoped that would give better results than hashing. In the end, the traindata contained roughly 200K-250K features.  
  
Second, a solution had to be found for values in the test-test that were not in the train-set. Since the trained model had never these values before, it could not have a good estimation of the weights for these features. We recoded the unseen test values to ‘missing’ or ‘rare’. Basically this came down to giving an average weight . We do think however that more subtle approaches in estimating the unseen values, and especially unseen combinations of values, would give even better results.

Third, the numeric features were standardized. For variation longtail features were log-transformed. Both operations were not strictly necessary but standardizing helped greatly with convergence speed and the log-transformations helped for variation in the ensemble.

# Modeling techniques and training

2.http://fastml.com/vowpal-wabbit-eats-big-data-from-the-criteo-competition-for-breakfast/  
1.http://people.csail.mit.edu/romer/papers/TISTRespPredAds.pdf

3. Hashing trick