Protein Matching With Custom Neural Network Objective Functions

David S. Vogel
MEDai / AI Insight
University of Central Florida
Orlando, FL
dvogel1@cfl.rr.com

Eric Gottschalk
MEDai / AI Insight
Orlando, FL
egottschalk@medai.com

Morgan C. Wang
Dept. of Statistics & Actuarial Science
University of Central Florida
Orlando, FL
cwang@mail.ucf.edu

ABSTRACT

This 2004 KDD Cup presents a perfect case where the usual neural network objective functions do not apply. While the contest problem consisted of 4 different entries with 4 different objective functions, this paper will focus on the solution optimizing GRMSE (Grouped Root Mean Squared Error). It will be shown that the more typical objective functions (including RMSE) cannot be as effective at meeting this criteria. While this objective function may be specific to this problem, and the reader may never see this exact function again in his/her lifetime, the idea behind this paper is applicable in many situations. Too often neural networks are used to minimize SSE (sum of the squares of the errors) or Cross Entropy, when the true measure of success for the model may require a small coding change to the Neural Network objective function. It is shown in this paper that a few small coding changes can make a big difference on a model's performance.

Keywords

KDD Cup, KDD, machine learning, objective functions, protein, homology, classification, neural networks, MITCH.

1. INTRODUCTION

Task II of the 2004 KDD Cup [1] was a test of one's ability to predict homologous protein sequences within native sequences of proteins (approximately 1000 proteins per native sequence). 153 sequences were provided as the training set. 150 sequences were specified to be the test set. Many of these native sequences had only a single homology. We held out one third of the training set (stratified by homology count) prior to any analysis for validation purposes. Each native sequence will be referred to as a "block."

Meeting the "Top 1" criteria (the top prediction for each block must be a true positive) was similar to finding a needle in a haystack. The MEDai / A.I. Insight team scored highest in this category with 92% correct (only 12 false positives out of 150 blocks). The overall strategy for all 4 evaluation criteria was to increase the number of features by detecting interaction variables, and then to feed the data set into a low-complexity MITCH neural network.

Beyond the overall strategy, small changes were made to customize the model to meet the 4 different criteria: Sum of Top 1, Average rank of last homology, average accuracy, and average RMSE. The winning strategy for the "Top 1" optimization was an ensemble of 4 different models. This was effective because the homologies were extremely sparse (0.9% overall, but as little as 0.1% in the most difficult native sequences). We observed from our validation set that a high probability was likely not to be a false positive, but that any given model would score some blocks with all low probabilities, indicating that the homology was not detected. By using a simple maximum of the 4 sub-models, we were able to increase our validation set accuracy by about 3%. Assuming that this improvement held true on the test set, the ensemble was the difference between 1st place and approximately 7th place.

2. BLOCK BIAS

Certain variables had bias in their relationship with the outcome. These were the raw scores for local alignment: predictors #10, 20, 30, 40, 50, 60, and 70. By dividing each of these variables by the mean for their block to obtain relative values, their predictive value increases significantly. Figures 1 and 2 illustrate how the predictive relationship depends on the block average for each variable, and how normalizing (dividing by the block average) improves their predictive power. Table 1 displays these improvements.

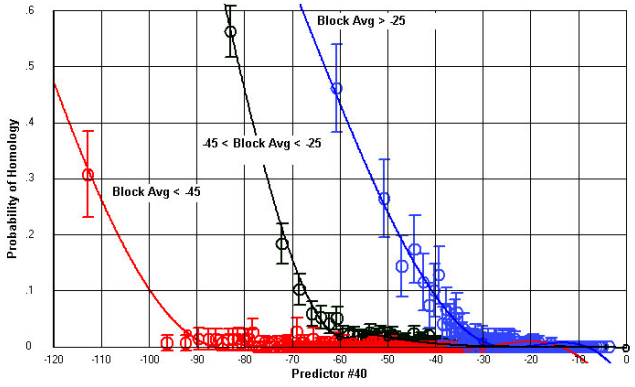

Figure 1: Predictor #40 is an example of where the predictive patterns vary based on the block averages. Using this variable in it's original state would introduce inter-block noise, lessening the effectiveness of its predictive potential.

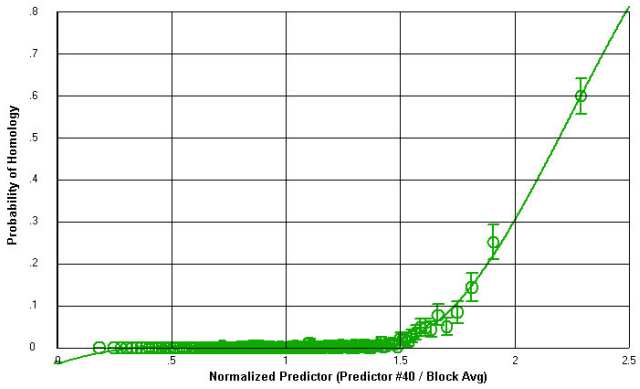

Figure 2: Normalizing the predictor using the block average value strengthens it's predictive power.

Predictor Number	Original Correlation	Transformed Correlation
60	.56	.58
10	.32	.42
40	.14	.25
30	.12	.19
50	.10	.19
70	.06	.15
20	.01	.07

Table 1: Predictive value of local alignment raw scores before and after dividing by their block mean. Predictive value here is measured as the Pearson correlation coefficient between the predictor and the outcome.

3. BLOCK VARIABLES / INTERACTIONS

We calculated the means for each of the 74 predictors within each block. These means were then entered as predictors into the data set. It was not likely that these new predictors would contribute very much on their own, but very likely that they would have interactions with other variables. Table 1 is just one example of an interaction between initial variables and group mean variables. In fact, it is also a good example of a non-multiplicative interaction. Most interactions that we used were multiplicative because they don't require individual attention. With approximately 700 highly significant interactions that didn't require any transformations, there were concerns that investigating the hundreds of non-multiplicative interactions would be time consuming and end up over-fitting.

Using the 74 block averages and the count of sequences in each block, a simple regression model was created on a block level to predict how many homologies existed in each block. This could be done with a fairly high degree of accuracy as the LOOCV (Leave one out cross validation) R-squared was about 0.6. This prediction was used as a predictor for the classification model.

4. MODIFYING THE NEURAL NETWORK

C. Bishop [2] discusses various neural network objective functions. This idea was motivation to optimize a new objective function specific to this problem: GRMSE. Although binary outcomes are typically modeled using the Bernoulli objective function, we will compare the formulation of the GRMSE objective function to the more commonly used RMSE objective function. Minimization of RMSE is numerically identical to the minimization of MSE (Mean Squared Error), which is equivalent to minimizing SSE (Sum Squared Error) since these measures are all increasing functions of each other. Within the neural network, the objective function must be repeatedly calculated:

$$SSE = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 = \sum_{g=1}^{G} \sum_{i \in I_g} (\hat{y}_i - y_i)^2$$

$$GRMSE = \sum_{g=1}^{G} \sqrt{\sum_{i \in I_g} (\hat{y}_i - y_i)^2 / N_g}$$

N = Number of records in the training sample.

N_g = Number of records in group G.

$\hat{y}_i$ = Predicted Value of the i^{th} record.

y_i = Actual Dependent Variable Value of the i^{th} record.

G = Number of groups in the training sample.

I_g = The set of indices within a group g.

These formulas are simple, yet extremely computationally intensive and code intensive within a neural network program. However, given that one already has the code to evaluate SSE, only two function calls (a square root and a division) need to be inserted into the code for evaluation of the new objective function.

The second calculation needed for the optimization of a neural network is the gradient, which is distinct for each objective function.

$$\nabla SSE = 2 \sum_{g=1}^{G} \nabla_g$$

$$\nabla GRMSE = \sum_{g=1}^{G} \left[\left(N_g \sum_{i \in I_g} (\hat{y}_i - y_i)^2 \right)^{-1/2} \nabla_g \right]$$

where we define for convenience the shorthand notation for each group's component of the gradient:

$$\nabla_g = \sum_{i \in I_g} (\hat{y}_i - y_i) \nabla \hat{y}_i$$

Again we find that the code is almost entirely the same between the two gradients, the difference being a multiplicative constant for each group's component to the gradient. These constants happen to be a function of the same summations which were previously obtained during the calculation of the objective function. Therefore there is no significant increase in the number of calculations when modifying the program to calculate the gradient for the GRMSE objective function. Most importantly, the amount of coding necessary is miniscule.

5. RESULTS AND DISCUSSION

The initial submission was based on a neural network model minimizing SSE using 854 predictors. The GRMSE error was 0.03805. By making the small alterations necessary for the Neural Network to optimize GRMSE directly, the result improved to a GRMSE error of 0.03602. This is more than a 5% improvement. The in-sample GRMSE's were 0.035 and 0.028, respectively. This is a little surprising, as the GRMSE minimization appears to have more of a tendency for over-fitting. For this experiment, all 854 predictors were used for both neural networks. However, more research should be done to reduce the variables by varying degrees to compare the sensitivities of both results to such experiments.

6. ACKNOWLEDGMENTS

Our thanks to KDD Cup 2004 organizers Rich Caruana and Thorsten Joachims at Cornell University for the interesting and well run competition. We would also like to thank Ron Elber at Cornell University for the donation of the data for this task. Finally, we acknowledge A.I. Insight, Inc. and MEDai, Inc. for the use of their predictive modeling technology MITCH (Multiple Intelligent Tasking Computer Heuristics).

7. REFERENCES

[1] ACM *SIGKDD KDD Cup* 2004 homepage http://kodiak.cs.cornell.edu/kddcup.

[2] Bishop, C. M. Neural Networks for Pattern Recognition. Oxford University Press, Oxford UK, 1995.

About the authors:

David S. Vogel is the Senior Scientist at A.I. Insight, where he has spent over 6 years leading the development of their modeling tool MITCH (Multiple Intelligent Tasking Computer Heuristics). David has been the winner of the KDD Cup in 3 different topics as well as honorable mentions on 3 additional topics, attesting to the versatility of his approach to predictive modeling. David's research interests include development of innovative modeling techniques, scalable algorithms, and new applications for predictive models.

Eric Gottschalk is recently joined A.I. Insight as a Research Associate, where he assists in the development of MITCH technology. Eric's research interests are in Wavelets and their applications to data mining, and also non-linear time series analysis. Eric studied applied math and computer science at the University of Colorado, and earned a MS in Data Mining from the University of Central Florida.

Morgan C. Wang is Professor in the Department of Statistics and Actuarial Science of the University of Central Florida. He and his colleague established a SAS Data Mining Certificate Program in the Fall of 2000 and started a Master's degree in Data Mining in the Fall of 2001. Every graduate of the program has served an internship with an eminent Orlando business partner in the aerospace, entertainment, hospitality or automobile service industry. Moreover, all faculty have established consulting relationships with industrial clients inspiring relevant research directions, student employment opportunities and enhanced curriculum case studies.