Tim Salimans: How I won the Deloitte/FIDE Chess Rating Challenge

This year, from February 7 to May 4, a prediction contest was held at Kaggle.com/c/ChessRatings2 where I ended up taking first place. The goal of the contest was to build a model to forecast the results of future chess matches based on the results of past matches. This document contains a description of my approach; the code can be found at http://people.few.eur.nl/salimans/chess.html.

Contents

The base model

The basic model underlying my approach was inspired strongly by the TrueSkill model, as well as by the winner and runner-up of an earlier chess rating contest on Kaggle. The final model was programmed in Matlab, but some of the early experimentation was done using the Infer.NET package, which is definitely worth having a look at. Warning: The discussion in this section is somewhat technical.

The basic statistical model assumed for the result of a match between a white player A and a black player B is the familiar ordered probit model:

$$d = s_A - s_B + \gamma + \epsilon, \epsilon \sim N(0, \sigma^2)$$

if d> 1 : A wins

if -1 < d< 1 : A and B draw

if d< -1 : B wins

Here $d$can be seen as the *performance difference* between A and B in this match, $s_A$and s_B as the skills of player A and B, γas the advantage of playing white, and ϵas a random error term.

Given a set of match results, we will infer the skills $s_i$of all players by means of factorized approximate Bayesian inference. $\gamma$and $\sigma^2$are estimated using approximate maximum likelihood.

We specify an independent normal prior for each player's skill s_i, having mean $\mu_i$and a variance of 1 (determined by cross validation). The means $\mu_i$can be initialized to 0 and will be set to a weighted average of the posterior means of the skills of each players' opponents at each iteration. The effect of this is to shrink the skills of the players to those of their opponents, as was first done by Yannis Sismanis.

Given a set of match results r, the skills have the following posterior density

$$p(s|r) \propto \prod_{i=1}^{players} \phi(s_i - \mu_i) \prod_{j=1}^{matches} \psi_j(s_{W_j}, s_{B_j})$$

where $\phi()$is the standard normal distribution function, $\psi_j()$is the likelihood term due to the match result r_j, and $W_j$and $B_j$identify the white and black player in match j. This posterior distribution is of a very high dimension and is not of any standard form, which makes it intractable for exact inference. Approximate Bayesian inference can solve this problem by approximating the above density by a product of univariate normal densities.

$$\tilde{p}(s|r) \propto \prod_{i=1}^{players} \phi(s_i - \mu_i) \prod_{j=1}^{matches} \phi([s_{W_j} - m_{j,1}]/\sqrt{v_{j,1}})\phi([s_{B_j} - m_{j,2}]/\sqrt{v_{j,2}})$$

There exist various ways of obtaining the mean and variance terms ($m_{j,1}$, $v_{j,1}$, $m_{j,2}$, $v_{j,2}$) in this pseudo-posterior. The two methods I tried were expectation propagation, as is used in the TrueSkill model and Laplace approximation, as used here in a similar context. For the current model and data set the results for both methods were practically the same. The advantage of the Laplace approximation is that it is easier to apply when we change the ordered probit specification to something else like a (ordered or multinomial) logit model. However, since the ordered probit specification provided the best fit, my final submission was made using this specification in combination with expectation propagation. Both methods can only be applied directly when we know which of the two players was playing white. This wasn't the case for part of the data. My solution to this problem was to calculate the likelihood terms for both the case that the first player is white as well as the case that the second player is white, after which I weight the likelihood terms of both cases by their respective posterior probabilities. This is the natural thing to do when using the Laplace approximation, but it also works well with expectation propagation.

In estimating the skills of the players we would like to assign more importance to matches that have occurred recently than to matches that were played long ago. The main innovation in my approach is to do this by replacing the pseudo posterior above with a weighted version:

$$\tilde{p}_w(s|r) \propto \prod_{i=1}^{players} \phi(s_i - \mu_i) \prod_{j=1}^{matches} \phi([s_{W_j} - m_{j,1}]/\sqrt{v_{j,1}})^{w_{j,1}}\phi([s_{B_j} - m_{j,2}]/\sqrt{v_{j,2}})^{w_{j,2}}$$

for weights $w_{j,1}$ and $w_{j,2}$ between zero and one. Since the normal distribution is a member of the exponential family this does not change the functional form of the posterior. Because of this, the weights can be incorporated quite naturally into the expectation propagation algorithm. An advantage of using this weighting scheme in combination with factorized approximate inference is that each match may now have a different weight for each of the two players. This is not possible using more conventional weighting methods like the one used to win the first Kaggle chess competition.

The use of a weighted likelihood in a Bayesian framework is an ad hoc solution, but can be viewed as a way of performing approximate inference in a model where the skills vary over time according to some stochastic process. An alternative solution would be to assume that this stochastic process is a (possibly mean-reverting) random walk, in which case we could use a forward-backward algorithm similar to the Kalman filter. However, for this particular problem the weighting approach performed slightly better.

After trying multiple options, the weight function chosen was $w_{j,1} = \exp(-0.012 * l_{j,1} - 0.02 * t_j - 0.1 * q_j)$, with $l_{j,1}$ the number of matches played by this player between the current month and the end of the sample, t_j the number of months in the same period, and q_j an indicator variable equal to one if match j is from the tertiary data set, which was of lower quality. The coefficients in this function were determined by cross-validation. There were other weighting schemes that showed some promise, such as overweighting those matches with players close in skill level to player B, when estimating the skill of player A for predicting his/her result against B. Alternatively, we could overweight those matches containing players that regularly played against B, as we can be more certain about their strength in relation to B than for players that have never played against this player. Due to time constraints I was unable to explore these possibilities further. The combination of approximate inference with likelihood weighting may be an interesting topic for future research.

Post-processing

The predictions of the base model scored very well on the leaderboard of the competition, but they were not yet good enough to put me in first place. It was at this time that I realized that the match schedule itself contained useful information for predicting the results, something that had already been noticed by some of the other competitors. In chess, most tournaments are played according to the Swiss system, in which in each round players are paired with other players that have achieved a comparable performance in earlier rounds. This means that if in a given tournament player A has encountered better opponents than player B, this most likely means that player A has won a larger percentage of his/her matches in that tournament.

In order to incorporate the information present in the match schedule, I generated out-of-sample predictions for the last 1.5 years of data using a rolling 3-month prediction window. (i.e. predicting months 127-129 using months 1-126, predicting months 130-132 using months 1-129 etc.) I then performed two post-processing steps using these predictions and the realized match outcomes, the first using standard logistic regression

and the second using a locally weighted variant of logistic regression. These post-processing steps used a large number of different variables as can be seen in the code below, but the most important variables were:

- the predictions of the base model
- the posterior means of the skills of A and B
- the number of matches played by these players
- the posterior means of the skills of the opponents encountered by A and B
- the variation in the quality of the opponents
- the average predicted win percentage over all matches in the same month for these players
- the predictions of a random forest using these variables

By comparing the quality of the opponents of A and B in a given tournament we may predict the result of the match between A and B. However, the data only indicated the month in which each match was played and not the tournament, and some players appear to have played in multiple tournaments in the same month. What finally pulled me ahead of the competition may have been the addition of a variable that weighs the skills of the opponents of B by their rooted pagerank to A on the match graph. The idea behind this was that if a player C, who played against B, is close to A on the match graph, the match between B and C most likely occurred in the same tournament as the match between A and B.

In order to make the locally weighted logistic regression computationally feasible, the post-processing procedure first allocates the matches in the test set into a small number of cluster points for which the logistic regression is performed. For the final prediction we can then use local interpolation between the parameter estimates at the cluster centers using Gaussian processes. Using this approach the post-processing was sufficiently fast to allow me to quickly try many different settings and variables.

Conclusions

I would like to thank both the organizers and the competitors for a great competition. Much of the contest came down to how to use the information in the match schedule. Although interesting in its own right, this was less than ideal for the original goal of finding a good rating system. Despite of this I hope that the competition, and my contribution to it, was useful and that it will help to advance the science of rating systems.