Elo-R Draft

December 17, 2015

The Elo rating system sets out to assign a quantitative measure of the skills of players in a particular activity, such as chess, a video game, or an athletic sport. These ratings can be used to rank players, as well as forecasting the outcome of future matches.

To make things concrete, let's label the players $1, \ldots, N$. The original Elo system deals with 1v1 zero-sum games with binary (win/lose) outcomes. From Player i's history against other players, Elo derives a rating r_i, a real number capturing some measure of the player's skill. r_i increases when player i wins a match, and decreases when they lose. The magnitude of the increase or decrease depends on how "surprising" the result was. To determine the level of surprise, we need some model of the likelihood of match outcomes between players of known ratings. Typically, the logistic function is used.

Now we describe the setting for the new Elo-R system. Instead of 1v1 matches with win/lose outcomes, we assume a large number of players compete at the task, and the outcome is a total ranking of all the players in the match. Player i's performance in the match can be viewed as a number p_i, drawn from a logistic distribution centered at s_i. While we can't measure s_i directly, it's possible to get a fairly accurate reading on p_i by looking at the ratings r_j of players that win and lose against i. This suggests the following two-phase approach: upon a new match, estimate p_i for each player. Then, update the rating r_i to some sort of collective average of past p_is.

1 Performance Averaging

Let's focus on the second phase for now: having computed the latest p_i for each player, we want to combine it with past values to get a more precise estimate. To clarify the notation in this section, we add an extra subscript j, ranging from 1 to the number of matches M in which player i has competed, so that $p_{i,j}$ denotes the performance of player i on their jth match, and $r_{i,j}$ denotes their rating at this time. The sample mean is perhaps the most obvious statistic to take. That is, let

$$r_{i,M} = \frac{1}{M} \sum_j p_{i,j} = \left(1 - \frac{1}{M}\right) r_{i,M-1} + \frac{1}{M} p_{i,M} \tag{1}$$

In addition to being simple, this method has the advantage of being a simple function of the previous rating $r_{i,M-1}$ and the new performance $p_{i,M}$, so there is no need to remember the performance history explicitly. This memorylessness also implies that players who enter a match with the same rating and same number of matches can expect to be treated similarly.

However, there are two key problems here. The first is that, as a player accumulates more experience, the relative weight of a new performance approaches 0. This is because all performances

are weighted equally, even ones which are too outdated to be informative about a player's current skill. It corresponds to a model in which a player's true skill is an eternal constant, never changing, and each $p_{i,j}$ is a measurement of this constant. To properly account for a player's lifelong evolution, we wish to place less weight on the oldest performances. The second problem is that the mean is not robust to outliers. A single extreme performance, perhaps due to a power failure if the competition takes place over a web server, will drastically affect a player's rating. Thus, we wish to reduce the weight of outliers. Before we proceed, note that the above equation can be rewritten as

$$\sum_j 1(p_{i,j} - r_{i,M}) = 0 \tag{2}$$

We say that $r_{i,M}$ is a *weighted mean* of the elements $\{p_{i,j}\}_{j=1}^{M}$, each of whom has equal weight, say 1. If the $p_{i,j}$ are thought of as independent measurements of the present true skill $s_{i,j}$, with Gaussian noise variances of τ_j^2, treating $p_{i,0}$ as a prior, then the posterior belief over the true skill is a Gaussian centered at $r_{i,M}$ given by the following equation:

$$\sum_j \frac{1}{\tau_j^2}(p_{i,j} - r_{i,M}) = 0 \tag{3}$$

All we did here is set the weight of $p_{i,j}$ to $\frac{1}{\tau_j^2}$. If the weights (or equivalently, the $\tau_j s$) form a geometric sequence, we maintain the memoryless property. We'll derive the weights later; for now, it's simplest to imagine them decreasing by a constant ratio for each match backward in time.

In order to bound the effect of any individual outlier, we also require that its weight approach 0 as it approaches $\pm\infty$. We still want to treat non-outliers in essentially the same way as before, so their weight should be close to $\frac{1}{\tau_j^2}$ for performances close to the weighted mean, which is $r_{i,M}$ by definition. Thus, we opt for a weight of $\frac{\tanh \frac{p_{i,j}-r_{i,M}}{\tau_j}}{\tau_j(p_{i,j}-r_{i,M})}$. This choice will receive additional justification later: essentially, it has to do with assuming the heavy-tailed logistic distribution in place of the Gaussian noise discussed previously. The final equation is:

$$\sum_j \frac{1}{\tau_j} \tanh \frac{p_{i,j} - r_{i,M}}{\tau_j} = 0 \tag{4}$$

It is monotonic in $r_{i,M}$, which can hence be solved for using binary search. A straightforward linear approximation shows that in the limit as the performances vary much less than any of the τ_j, the rating behaves like a mean with weights $\frac{1}{\tau_j^2}$. In the limit as the performances vary much more than any of the τ_j, the rating behaves like a median with weights $\frac{1}{\tau_j}$.

2 Approximate Bayesian Derivation (needs a total rewrite)

The kernel of our Bayesian belief on s_i will be a product of logistic pdfs, which can be thought of as a prior times a set of independent measurements centered on s_i with different variances.

The players take part in a ranked event such as a video game challenge or programming contest. We model this as follows: each player i performs at level p_i, drawn independently from a logistic distribution centered at s_i. The ranking we observe is a total ordering on the players based on their

performances: i outranks j, written $i \succ j$, iff $p_i > p_j$. According to this model, ties occur with probability zero; we will treat actual ties as half a win and half a loss.

Fix i. Let e be the evidence consisting of, for each j, whether or not i beat j. That is, we ignore the relative ordering of player pairs which don't include i. Our goal is to approximate the posterior distribution of s_i given e:

$$f(s_i \mid e) \propto f(s_i) \Pr(e \mid s_i) = f(s_i) \int \Pr(e \mid p_i) f(p_i \mid s_i) dp_i \tag{5}$$

Since the integral is hard to evaluate, we will treat $\Pr(e \mid p_i)$ as a delta-function that spikes at the *maximum a posteriori* (MAP, a.k.a. posterior "mode") estimate of p_i. This is justified as $N \to \infty$, because the evidence e would overwhelmingly concentrate p_i into a narrow range. Having fixed p_i, Equation (1) simplifies to

$$f(s_i \mid e) \propto f(s_i) f(p_i \mid s_i) \tag{6}$$

Our update algorithm thus divides into three phases: first, it must determine the player's performance p_i in the contest. Then, it will use this value to update the belief distribution on s_i. Finally, the belief distribution is summarized by finding the point r_i that achieves its maximum.

2.1 Performance estimation

To compute the MAP of p_i, we must maximize

$$f(p_i \mid e) \propto f(p_i) \Pr(e \mid p_i) \tag{7}$$

p_i can be written as the sum of two logistic random variables. If we replace these by normal random variables with the same mean and variance, then their sum is also normal, and we approximate it by a logistic with the same mean and variance. Thus if $\alpha = \pi/\sqrt{3}$ and $\tau = \sigma_i^2 + \delta^2$,

$$f(s_i) = \frac{2e^{2(s_i - r_i)/\sigma_i}}{\sigma_i \left(1 + e^{2(s_i - r_i)/\sigma_i}\right)^2} \tag{8}$$

$$f(p_i \mid s_i) = \frac{2e^{2(p_i - s_i)/\delta}}{\delta \left(1 + e^{2(p_i - s_i)/\delta}\right)^2} \tag{9}$$

$$f(p_i) \approx \frac{2e^{2(p_i - r_i)/\tau}}{\tau \left(1 + e^{2(p_i - r_i)/\tau}\right)^2} \tag{10}$$

Let $W_i = N - Rank_i$ denote the number of players i won against. Then

$$\Pr(e \mid p_i) = \prod_{j \succ i} \Pr(p_j > p_i) \prod_{j \prec i} \Pr(p_j < p_i) \tag{11}$$

$$\approx \prod_{j \succ i} \frac{1}{1 + e^{2(p_i - r_j)/\tau}} \prod_{j \prec i} \frac{e^{2(p_i - r_j)/\tau}}{1 + e^{2(p_i - r_j)/\tau}} \tag{12}$$

$$\propto \frac{e^{2W_i p_i/\tau}}{\prod_{j \neq i} 1 + e^{2(p_i - r_j)/\tau}} \tag{13}$$

$$C_1 + \ln f(p_i \mid e) = C_2 + \ln f(p_i) + \ln \Pr(e \mid p_i) \tag{14}$$

$$\approx \ln \frac{2}{\tau} + (p_i - r_i)\frac{2}{\tau} - 2\ln\left(1 + e^{2(p_i - r_i)/\tau}\right) + W_i p_i \frac{2}{\tau} - \sum_{j \neq i} \ln\left(1 + e^{2(p_i - r_j)/\tau}\right) \tag{15}$$

Differentiate w.r.t. p_i and set to zero:

$$0 = \frac{2}{\tau}\left(1 - \frac{2e^{2(p_i - r_i)/\tau}}{1 + e^{2(p_i - r_i)/\tau}} + W_i - \sum_{j \neq i} \frac{e^{2(p_i - r_j)/\tau}}{1 + e^{2(p_i - r_j)/\tau}}\right) \tag{16}$$

$$Rank_i = \frac{2}{1 + e^{2(p_i - r_i)/\tau}} + \sum_{j \neq i} \frac{1}{1 + e^{2(p_i - r_j)/\tau}} \tag{17}$$

Use binary search to solve for p_i. This is the *performance* of player i in the match.

2.2 Proportional rating update

Our approximate formula for the posterior $f(s_i \mid e)$ takes the prior $f(s_i)$ and multiplies it by a new logistic pdf $f(p_i \mid s_i)$. The general form for our posterior will be proportional to a product of normal and logistic pdfs. Since a product of normals is proportional to another normal pdfs, wlog we assume there is only one normal in the product:

$$e^{-(s_i - \mu_0)^2/\tau_0^2} \prod_k \frac{e^{2(s_i - \mu_k)/\tau_k}}{\left(1 + e^{2(s_i - \mu_k)/\tau_k}\right)^2} \tag{18}$$

Differentiate its logarithm w.r.t. s_i and set to zero:

$$0 = \frac{2(\mu_0 - s_i)}{\tau_0^2} + \sum_k \frac{2}{\tau_k}\left(1 - \frac{2e^{2(s_i - \mu_k)/\tau_k}}{1 + e^{2(s_i - \mu_k)/\tau_k}}\right) \tag{19}$$

$$0 = \frac{\mu_0 - s_i}{\tau_0^2} + \sum_k \frac{1}{\tau_k} \tanh \frac{\mu_k - s_i}{\tau_k} \tag{20}$$

Solve for s_i with binary search, and use its value as $r_{i,new}$. $1/\sigma_i^2 = \sum_k 1/\tau_k^2$.

Define the *information* contained in this distribution by $I = \sum_k 1/\tau_k^2$. The sum occurs by analogy to Gaussians: if we were to replace the logistic pdfs by Gaussian measurements with the same mean and variance, then their product would be a Gaussian with variance $(\sum_k 1/\tau_k^2)^{-1}$.

However, now we have to add noise to account for changing skills. Let σ^* denote a limit. Solve the fixpoint equation:

$$\frac{1}{(\sigma^*)^2} = \frac{1}{(\sigma^*)^2 + \nu^2} + \frac{1}{\delta^2}(\tag{21}$$

$$(\sigma^*)^2 = \tag{22}$$

We say m is a weighted average of $\{x_i\}$ with weights $\{w_i\}$ if $\sum_i w_i(m - x_i) = 0$. In this sense, s_i is a weighted average of μ_k with weights $\tanh(\mu_k - s_i)/(\mu_k - s_i)$.

2.3 Proportional Rating Update

If we approximate these by normal pdfs, their product is again normal, and we assign the corresponding mean and variance to our posterior belief over s_i:

$$r_{i,new} = \frac{\sigma_1^2 p_i + \sigma_2^2 r_i}{\sigma_3^2} \qquad \sigma_{1,new} = \frac{\sigma_1^2 \sigma_2^2}{\sigma_3^2} \tag{23}$$

Between contests, we model changes in a player's true skill by adding to s_i a Gaussian noise with mean 0 and variance proportional to σ_1^4/σ_3^2. Again approximating the product of pdfs by analogy to Gaussians, the noise conveniently resets our uncertainty σ_1 to its original value.

2.4 Alternative rating update based on the MAP of s_i

$$C + \ln f(s_i \mid e) \approx \ln f(s_i) + \ln f(p_i \mid s_i) \tag{24}$$

$$= \frac{s_i - r_i}{\sigma_1} - \ln \sigma_1 - 2 \ln \left(1 + e^{(s_i - r_i)/\sigma_1}\right) + \frac{s_i - p_i}{\sigma_2} - \ln \sigma_2 - 2 \ln \left(1 + e^{(s_i - p_i)/\sigma_2}\right) \tag{25}$$

Differentiate w.r.t. s_i and set to zero:

$$0 = \frac{1}{\sigma_1} \left(1 - \frac{2e^{(s_i - r_i)/\sigma_1}}{1 + e^{(s_i - r_i)/\sigma_1}}\right) + \frac{1}{\sigma_2} \left(1 - \frac{2e^{(s_i - p_i)/\sigma_2}}{1 + e^{(s_i - p_i)/\sigma_2}}\right) \tag{26}$$

$$= \frac{1}{\sigma_1} \tanh \frac{r_i - s_i}{2\sigma_1} + \frac{1}{\sigma_2} \tanh \frac{p_i - s_i}{2\sigma_2} \tag{27}$$

Solve for s_i with binary search, and use its value as $r_{i,new}$. Note that if $\sigma_1 < \sigma_2$, then

$$|r_{i,new} - r_i| < 2\sigma_1 \operatorname{artanh} \frac{\sigma_1}{\sigma_2} \tag{28}$$

In other words, this method enforces an upper bound on the rating change per event.

The first method can be thought of as setting r_i to the player's average historical performance, with exponentially decaying weights to place the emphasis on recent events. In contrast, the second method does something similar when the performances are consistent, but puts much less weight on distant outliers. Maybe this method can be made more accurate by remembering p_i values from the last 10 or so matches and computing the MAP on all 10 matches simultaneously with exponentially decaying weights, using the rating from 10 matches ago as the prior mean. The behavior induced by this modification can be likened to TopCoder's volatility measure: one very strong performance won't change the rating much, but the second or third consecutive strong performance will have a larger effect. Unlike on TopCoder, a very weak performance following a very strong perforance will *not* have a large effect.