Advanced Probabilistic Machine Learning Project

Group 15: Sireesh Limbu, Vasileios Papadopoulos, Erik Bootsma

Introduction

In this project, probabilistic machine learning is used in an attempt to estimate the level of individual players or teams based on the results of matches. The model used is based on Microsoft's $Trueskill$.

Q.1: Formulating the Bayesian model

The Bayesian model is defined by three normally distributed random variables. Variables s_1 and s_2 represent the skill level of two opposing players or teams in a game. Variable t meanwhile represents the outcome of the game, resulted by the difference between skills s_1 and s_1. Finally we have one discrete random variable y, which denotes the result of a match. The probability distributions are:

$p(s_1) = \mathcal{N}(\mu, \sigma^2)$
$p(s_2) = \mathcal{N}(\mu, \sigma^2)$
$p(t|s_1, s_2) = \mathcal{N}(s_1 - s_2, \sigma_t^2)$
$y = sign(t) = \begin{cases} 1 & \text{, if } t > 0 \\ -1 & \text{, if } t < 0 \end{cases}$, there is no possibility of a draw in a match.

Inspired by Microsoft $Trueskill$[2], we have chosen the following values for the five hyperparameters: $\mu_{s\,1} = 25, \quad \mu_{s\,2} = 25\,, \quad \sigma_{s_1}^2 = 25/3\,, \quad \sigma_{s_2}^2 = 25/3\,, \quad \sigma_t^2 = 25/6$

Q.2: Computing with the model

Full conditional distribution of the skills

Let's compute the conditional probabilities of random variables s_1 and s_2 given that variables t and y are known. With Bayes Theorem we can change the terms in the model.

$p(s_1, s_2|t, y) = \frac{p(s_1,s_2,t,y)}{p(t,y)} = \frac{p(t|s_1,s_2)p(s_1)p(s_2)}{p(t)} = p(s_1, s_2|t)$

Corollary 1[1] is used to find this conditional distribution. We set $X_a = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$ and $X_b = t$. We know that $p(X_a) \sim \mathcal{N}(\mu_a, \sum_a)$, with $\mu_a = \begin{pmatrix} \mu_{s1} \\ \mu_{s2} \end{pmatrix}$ and $\sum_a = \begin{pmatrix} \sigma_{s_1}^2 & 0 \\ 0 & \sigma_{s_2}^2 \end{pmatrix}$, due to s_1 and s_2 being independent. Furthemore, $p(X_b|X_a) = \mathcal{N}(s_1 - s_2, \sigma_t^2) = \mathcal{N}(M_{x_a}, \sum_{b|a})$, with $M = (1 - 1)$. Now we can find our answer: $p(s_1, s_2|t) = \mathcal{N}(\mu_{s_1,s_2|t}, \sum_{s_1,s_2|t})$, with mean and variance:

- $\mu_{s_1,s_2|t} = \sum_{s_1,s_2|t} \left(\begin{pmatrix} 1 \\ -1 \end{pmatrix} \cdot (\sigma_t^2)^{-1} t + \begin{pmatrix} \sigma_{s_1}^2 & 0 \\ 0 & \sigma_{s_2}^2 \end{pmatrix}^{-1} \begin{pmatrix} \mu_{s_1} \\ \mu_{s_2} \end{pmatrix} \right)$
- $\sum_{s_1,s_2|t} = \left(\begin{pmatrix} \sigma_{s_1}^2 & 0 \\ 0 & \sigma_{s_2}^2 \end{pmatrix}^{-1} + (\sigma_t^2)^{-1} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \right)^{-1}$

Full conditional distribution of the outcome

Next we'll find the probability of variable t given knowledge of variables s_1, s_2 and y: $p(t|s_1, s_2, y) = \frac{p(t,s_1,s_2,y)}{p(s_1,s_2,y)} = \frac{p(t|s_1,s_2)p(y|t)p(s_1)p(s_2)}{p(s_1,s_2,y)}$. Due to proportionality and leaving out terms not dependent on t we can imply that $p(t|s_1, s_2, y) \propto p(y|t) \cdot p(t|s_1, s_2)$. As a result of the $p(y|t)$ term, this creates a Truncated Normal Distribution, i.e. a Normal Distribution where the Probability Distribution Function (pdf) is bounded on its sides by parameters a and b. This means that the domain of pdf is limited to [a,b]. Although we have one distribution, we're dealing with two pdf's. When y=-1 (player 2 wins), the domain is $(-\infty, 0]$. When y = 1 (player 1 wins), the domain is $[0, \infty)$. See Figure 6 for an example of this.

Marginal probability of Player 1 winning the game

We're looking for $p(y = 1)$, which implies that $p(t > 0)$. We already know the distributions of $p(t|s_1, s_2)$ and $p(s_1, s_2)$. By using the terms X_a and X_b like we did before, we can use Corollary 2[1] to calculate the probability of variable t: $p(t) \sim \mathcal{N}(\mu_t, \sum_t)$, with variance and mean:

$\mu_t = \mu_{s_1} - \mu_{s_2}, \sum_t = \sum_{t|s_1,s_2} + M \cdot \sum_a \cdot M^T = \sigma_t^2 + \sigma_{s_2}^2 + \sigma_{s_2}^2$

Using the marginal probability of variable t, we can find the probability of $p(t > 0)$ through a pdf.

Q.3: Bayesian Network graph

The Bayesian Network is presented in Figure 1. By using D-separation rules, we can make the some statements regarding independencies of our variables:

s_1 and s_2 are independent: $s_1 \perp\!\!\!\perp s_2|\theta$
s_1 and y are conditionally independent given t: $s_1 \perp\!\!\!\perp y|t$
s_2 and y are conditionally independent given t: $s_2 \perp\!\!\!\perp y|t$

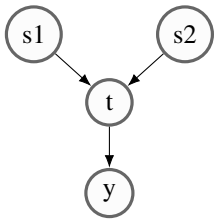


Figure 1: The Bayesian Network

Q.4: Gibbs sampler

Finding the distribution for $p(s_1, s_2|y)$ or $p(s_1, s_2, t|, y)$ is analytically very hard. Gibbs sampler helps with this problem by sampling from the posterior distribution (target distribution) through some conditional distributions.

Question 2 gives us these conditional distributions, namely $p(s_1, s_2|t, y)$ and $p(t|s_1, s_2, y)$. These span over the distributions of the required variables: s_1, s_2 and t. Furthermore, we know that s_1 and s_2 are **not independent** while conditioned on y. Thus, after we calculate the mean vector and covariance matrix from Question 2, we can start sampling for $p(s1, s2|t, y)$ by randomly drawing from this multivariate normal distribution. For $p(t|s1, s2, y)$ we draw random samples from the truncated normal distribution where the bounds of truncation depend on whether player 1 wins or not. We begin by assuming the initial values of s_1,s_2,t = 0.

Figure 2 features the plots for posterior values of s_1 and s_2 on the condition that s_1 wins ($y = 1$), without truncating the burn-in. After looking at these plots we realize the steady state is reached immediately after the first iteration. There is no burn-in period required as such. Furthermore, we see a higher distribution of s_1 than s_2, which makes sense as s_1 won the match. We can see from the plots in Figure 3 that after removal of the burn-in, a continuous pattern has been achieved with means of **27** and **23** and standard deviations of **2.44** and **2.33** for s_1 and s_2 respectively. We can use these values to plot a Gaussian approximation of the posterior distribution of the skills (see Appendix, Figure 7).

We plot the histograms together with Gaussian posteriors for four different number of samples - for **900** samples we get **2.75** seconds, for **800** we get **2.85** seconds, for **600** samples we get **2.65**seconds and finally for **400** samples we get **2.6** seconds. Thus, the optimum number of samples is 900 samples without affecting the accuracy.

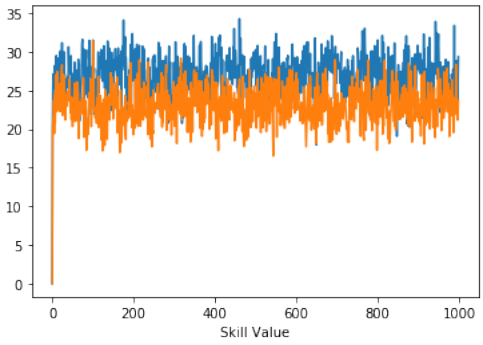


Figure 2: Posteriors of s_1 and s_2. Blue represents s_1 and orange represents s_2

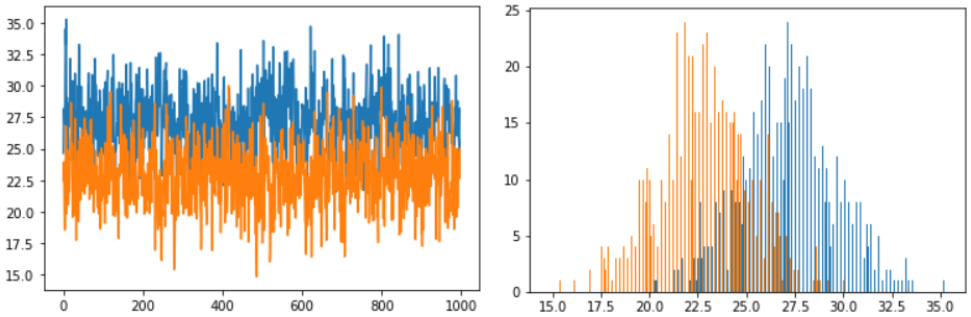


Figure 3: Posteriors after removal of the burn-in. Blue represents s_1 and orange represents s_2

Q.5: Assumed Density Filtering

In question 5 we use the posterior distribution calculated in the previous question as the prior for the first match and the posterior of the first as the prior of the seconds and so forth. We remove all the draws from the data set before processing the data.

The final ranking for the 20 teams can be seen in Figure 4. We discovered that if the sequence of matches is changed, that the final ranking will be different as well. This means that Assumed Density Filtering (ADF) is sensitive to the order of data processing. This might be because the posterior distribution at any time is always dependent on the current distribution at hand. The way in which all distributions are achieved seems to be changing the probability of the error in approximation.

Q.6: Application in production

For application of the predictor function we chose the ADF function that we've come up with in the above (Q5) question. In a simple application of ADH we create a function that takes in the Mean vector, Standard deviation vector and 't' value, which is a sample from the truncated normal distribution and then takes out 1000 samples for each of s_1 and s_2 from the multivariate normal distribution and subtracts the mean of both the samples to get 'meanT'. If this value (meanT) is greater than 0, the function gives an output of '1', else it outputs '-1'.

Furthermore, in the case where team 1 has won and our prediction is wrong, we update the mean skill of that particular team by adding a factor of **0.05x** where 'x' is the difference between the scores of the two teams. Likewise, in the case where team 2 has won and our prediction is wrong, we update the mean skill of that particular team by subtracting a factor of **0.05x**. In the latter case the difference 'x' is negative, hence the subtraction ends up being an addition. We get a prediction of **51%** which is slightly better than random guessing of 1/2 probability.

Teams	Mean Skill Value	Standard Deviation
Inter	29.073	1.154
Atalanta	26.806	1.262
Napoli	26.544	1.315
Milan	26.310	1.238
Torino	26.264	1.180
Roma	25.889	1.321
Bologna	25.689	1.256
Empoli	25.155	1.407
Udinese	24.706	1.151
Lazio	23.630	1.314
Spal	23.187	1.277
Juventus	22.672	1.313
Sampdoria	22.157	1.439
Sassuolo	21.958	1.155
Cagliari	20.291	1.132
Parma	20.101	1.125
Chievo	19.862	1.169
Frosinone	19.782	1.325
Genoa	18.770	1.341
Fiorentina	17.261	1.146

Figure 4: Final ranking

Question 7

We take the Bayesian Network graph in Figure 1 and convert it into a Factor Graph. An additional variable w was added, which represents the difference of skill between the players. The factor graph in Figure 5 describes the probabilistic relationships between the five variables. Due to its tree structure, we can use belief propagation to find distributions of probabilities through known variables.

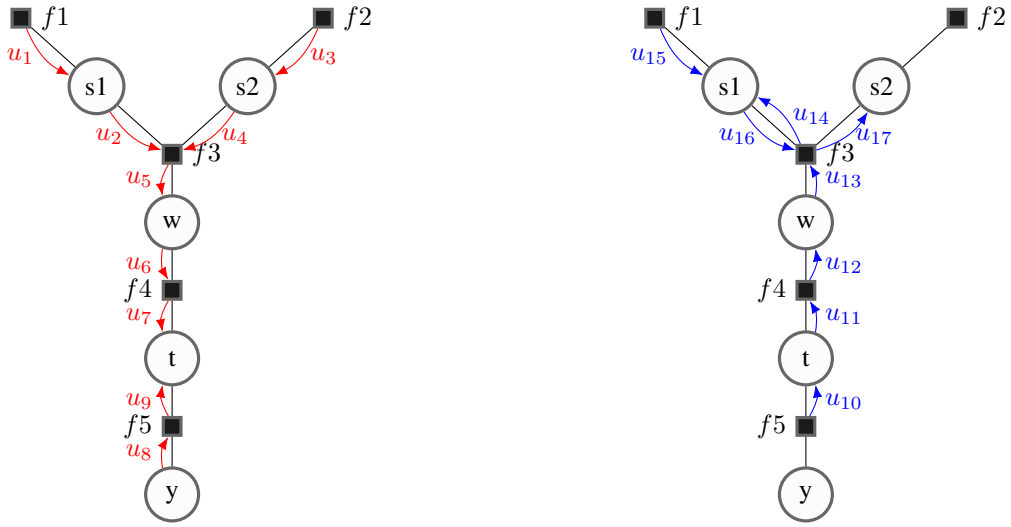


Figure 5: The Factor graph. First set of messages on the left, seconds set of messages on the right.

Concerning the factors:
$f_1(s_1) = \mathcal{N}(s_1; m_{s_1}, \sigma^2_{s_1})$
$f_2(s_2) = \mathcal{N}(s_2; m_{s_2}, \sigma^2_{s_2})$
$f_3(s_1, s_2, t) = \delta(w - (s_1 - s_2))$
$f_4(w, t) = \mathcal{N}(t; w, \sigma^2_t)$
$f_5(t, y) = \delta(y - sign(y))$

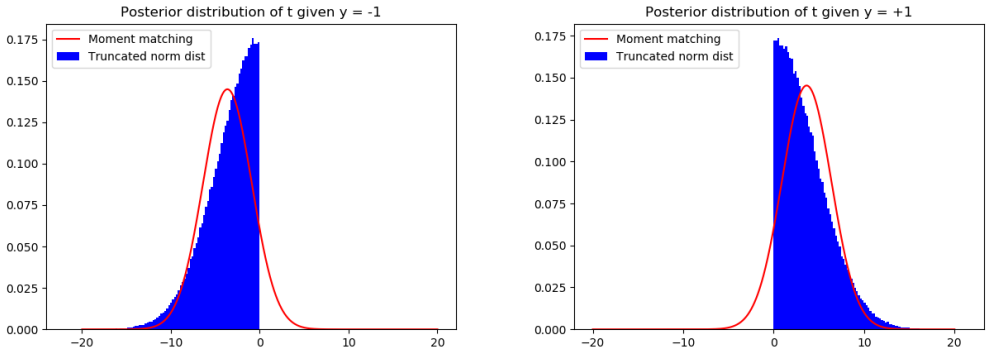


Figure 6: Truncated Normal Distribution for posterior of t. See left for y = -1, right for y = +1.

Three of the five factors are normally distributed. Factors f_3 and f_5 are distributed using a Dirac measure. This measure is a way to 'force' a deterministic relationship between variables into a probabilistic relationship. It only has a non-zero, positive value when its argument is equal to zero.

$u_1(s_1) = u_{f_1 \to s_1}(s_1) = \mathcal{N}(s_1; \mu_{s_1}, \sigma^2_{s_1})$
$u_2(s_2) = u_{s_1 \to f_3}(s_1) = u_1(s_1)$
$u_3(s_2) = u_{f_2 \to s_2}(s_2) = \mathcal{N}(s_2; \mu_{s_2}, \sigma^2_{s_2})$
$u_4(s_2) = u_{s_2 \to f_3}(s_2) = u_3(s_2)$

Message requires some more effort. However, a combination of Dirac measure tricks together with Gaussian shifting and Corollary 2[1] can be used to solve this. See the appendix for the full derivation.

$u_5(w) = \mathcal{N}(w; \mu_{s_1} - \mu_{s_2}, \sigma^2_{s_1} + \sigma^2_{s_2})$
$u_6(w) = u_{w \to f_4}(w) = u_5(w)$ $u_7(t) = \int f_4(w,t) u_6(w) dw = \mathcal{N}(t; \mu_{s_1} - \mu_{s_2}, \sigma^2_{s_1} + \sigma^2_{s_2} + \sigma^2_t))$
$u_9(y) = u_{y \to f_5}(y) = \delta(y - y_{obs})$
$u_8(t) = u_{f_5 \to y}(t) = \delta(y_{obs} - sign(t))$

Q.8: Message-passing algorithm implementation

In the previous question we defined the first set of messages. See Figure 5 (left side) for an overview. We can use the message passing to propagate known information through the factor graph and find the posterior distribution of t, given y: $p(t|y = y_{obs})$. This probability is approximately equal to $u_7(t) \cdot u_8(t)$. However, its value needs to be approximated using moment matching. Of the approximation $\hat{p}(t|y = y_{obs})$ the mean and variance are called m_t and s_t respectively. Figure 6 illustrates this approximation for $y = -1$ and $y = +1$ respectively.

The 'second' round of messaging will go in the direction of variables s_1 and s_2 in order to find the posterior distributions of these variables, given the knowledge that variable y is known. The overview of messages is featured in Figure 5 (right side). We first 'update' the u_8 message, calling it u_{10}.

$u_{10}(t) = u_{f_5 \to t}(t) = \dfrac{\hat{p}(t|y = y_{obs})}{u_7(t)} = \mathcal{N}(t; m_{10}, s^2_{10})$, with mean and variance:

$$m_{10} = \frac{m_t \cdot (\sigma^2_{s_1} + \sigma^2_{s_2} + \sigma^2_t) - (\mu_{s_1} - \mu_{s_2}) \cdot s^2_t}{\sigma^2_{s_1} + \sigma^2_{s_2} + \sigma^2_t) - s^2_t}, s^2_{10} = \frac{(\sigma^2_{s_1} + \sigma^2_{s_2} + \sigma^2_t)) \cdot s^2_t}{\sigma^2_{s_1} + \sigma^2_{s_2} + \sigma^2_t) - s^2_t}$$

$u_{11}(t) = u_{t \to f_4}(t) = u_{10}(t)$

$$u_{12}(w) = u_{f_4 \to w}(w) = \int f_4(w,t) u_{11}(t) dt = \int \mathcal{N}(t; w, \sigma^2_t) \mathcal{N}(t; m_{10}, s^2_{10}) dt = \mathcal{N}(w; m_{10}, \sigma^2_t + s^2_{10})$$

$u_{13}(w) = u_{w \to f_3}(w) = u_{12}(w)$
$u_{14}(s_1) = u_{f_3 \to s_1}(w) = \int f_3(s_1, s_2, w) u_{13}(w) u_4(s_2) ds_2, w = \mathcal{N}(s_1; m_{10} + u_{s_2}, \sigma^2_t + s^2_{10} + \sigma^2_{s_2})$

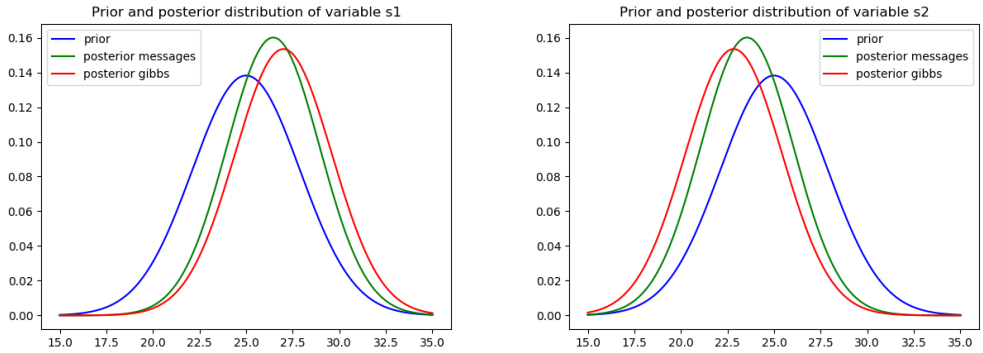


Figure 7: Prior and posteriors of $s1$ (left) and $s2$ (right) skill variables.

The posterior distribution of s_1: $p(s_1|y = y_{obs}) \propto u_1(s_1) \cdot u_{14}(s_1) \propto \mathcal{N}(s_1; m_{s_1}, s_1^2)$, with:

$$m_{s_1} = \frac{u_{s_1} \cdot (\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2) + (m_{10} - u_{s_2}) \cdot \sigma_{s_1}^2}{\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2 + \sigma_{s_1}^2}, s_1^2 = \frac{(\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2) \cdot \sigma_{s_1}^2}{\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2 + \sigma_{s_1}^2}$$

Proceed with message passing. First update message $u_1(s_1)$, which we'll call $u_{15}(s_1)$.

$u_{15}(s_1) = u_{f_1 \to s_1}(t) = \frac{\hat{p}(s_1|y = y_{obs})}{u_{14}(t)} = \mathcal{N}(t; m_{15}, s_{15}^2)$, with mean and variance:

$$m_{15} = \frac{m_{s_1} \cdot (\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2) - (m_{10} - u_{s_2}) \cdot_1^2}{\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2 - s_1^2}, s_{15}^2 = \frac{(\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2) \cdot s_1^2}{\sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2 - s_1^2}$$

$u_{16}(s_1) = u_{s_1 \to f_3}(s1) = u_{15}(s_1)$
$u_{17}(s_2) = \int f_3(s_1, s_2, w) u_{16}(s_2) u_{13}(w) ds_1, w = \mathcal{N}(s_2; m_{15} - m_{10}, s_{15}^2 + \sigma_t^2 + s_{10}^2)$

We finished defining the messages. For u_{14} and u_{17} we used the same tricks as with u_5 (see appendix). The posterior distribution of s_2: $p(s_2|y = y_{obs}) \propto u_{17}(s_2) \cdot u_3(s_2) \propto \mathcal{N}(s_1; m_{s_2}, s_2^2)$, with:

$$m_{s_2} = \frac{(m_{15} - m_{10}) \cdot \sigma_{s_2}^2 + u_{s_2} \cdot (s_{15}^2 + \sigma_t^2 + s_{10}^2)}{s_{15}^2 + \sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2}, s_2^2 = \frac{\sigma_{s_2}^2 \cdot (s_{15}^2 + \sigma_t^2 + s_{10}^2)}{s_{15}^2 + \sigma_t^2 + s_{10}^2 + \sigma_{s_2}^2}$$

If we state that $y = +1$ we can clearly define the posterior distributions derived above (see Figure 7 and Table 1 in the appendix). As you can see, the posteriors of variables $s1$ and $s2$ which were found using Gibbs sampling and message passing are quite similar. The distributions react to the knowledge that $y = +1$ in a logical way: the mean skill of s_1 increases due to a win, while the mean skill of s_2 decreases due to a loss. The variance of both variables decreases due to the fact that more knowledge about the player's skill is available.

Q.9: Your own data

The ADH predictor function was then applied to our Premier League's dataset [4]. It has a similar format of 20 teams and 380 matches, making it an ideal dataset for our prediction function. An accuracy of ~52% is achieved. This is justified, since this dataset is similar to the given dataset in Q6.

Q.10: Project extension

We have already implemented a prediction code that takes care of the difference in scores and not just the final victory into consideration. This is in itself an extension, on top of which we finally include the 108 draw matches into our ADH processing. Here, if the match happens to be a draw, we don't update the mean of those particular teams for that particular match. After the application we get a rough rate ~50% accuracy in the whole dataset including the draws.

References

[1] Thomas B. Schön and Fredrik Lindsten, *Manipulating the Multivariate Gaussian Density*. Division of Automatic Control Linköping University.

[2] Microsoft Trueskill (2005). Retrieved from: https://www.microsoft.com/en-us/research/project/trueskill-ranking-system/

[3] N. Wahlström, *Advanced Probabilistic Machine Learning lecture notes*. Department of Information Technology, Uppsala University, 2019

[4] English Premier League. Retrieved from: https://datahub.io/sports-data/english-premier-league

[5] C. Bishop, *Pattern Recognition And Machine Learning*. Springer, 2006.

Appendix

Q4: Gaussian approximation

Method	mean $s1$	variance $s1$	mean $s2$	variance $s2$
prior	25	8.33	25	8.33
posterior through Gibbs	27	6.76	22.83	6.76
posterior through messages	26.46	6.20	23.54	6.21

Table 1: Mean and variance for the priors of variables s_1 and s_2, as well as their posteriors.

Q7: Message derivation

Messages u_5, u_{14} and u_{17} were defined using a combination of tricks and theorems. Below follows the full derivation of message u_5. Messages u_{14} and u_{17} were derived the same way.

$$u_5(w) = \int f_3(s_1, s_2, w)u_2(s_1)u_4(s_2)ds_1, s_2 = \int\int \delta(w-(s_1-s_2)\mathcal{N}(s_1; \mu_{s_1}, \sigma^2_{s_1})\mathcal{N}(s_2; \mu_{s_2}, \sigma^2_{s_2})ds_1 ds_2$$

We can 'change' the argument of the Dirac measure, as long as we remember that is only has a non-zero, positive value if the argument is equal to zero.

$$\delta(w - (s_1 - s_2) = \delta(w - s_1 + s_2) = \delta(s_1 - w - s_2) = \delta(s_1 - (w + s_2))$$

We use this trick to solve the integration over s_1, after which Gaussian Shifting is used. In the third step we state $z = \mu_{s_1} - s_2$, and therefore, $s_2 = -z + \mu_{s_1}$. Finally, Gaussian Shifting is used again.

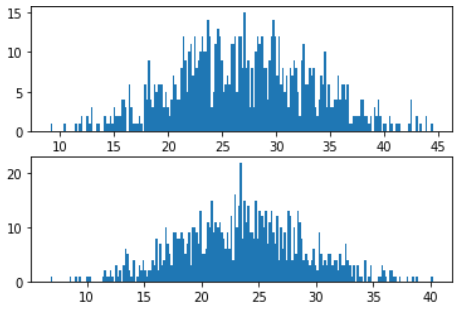
$$u_5(w) = \int \mathcal{N}(w + s_2; \mu_{s_1}, \sigma^2_{s_1})\mathcal{N}(s_2; \mu_{s_2}, \sigma^2_{s_2})ds_2 = \int \mathcal{N}(w; \mu_{s_1} - s_2, \sigma^2_{s_1})\mathcal{N}(s_2; \mu_{s_2}, \sigma^2_{s_2})ds_2$$

$$= \int \mathcal{N}(w; z, \sigma^2_{s_1})\mathcal{N}(\mu_{s_1} - z; \mu_{s_2}, \sigma^2_{s_2})dz = \int \mathcal{N}(w; z, \sigma^2_{s_1})\mathcal{N}(-z; \mu_{s_1} - \mu_{s_2}, \sigma^2_{s_2})dz$$

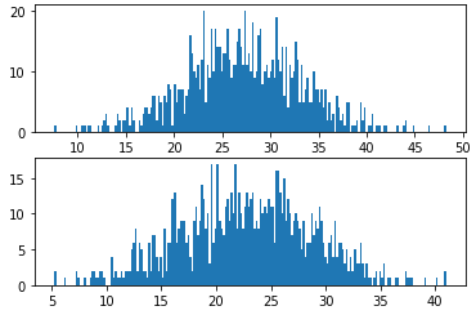
Now we can use Corollary 2[1] to do the integration over dz.

$$u_5(w) = \mathcal{N}(w; \mu_{s_1} - \mu_{s_2}, \sigma^2_{s_1} + \sigma^2_{s_2})$$

Q8: Posteriors of the skills



(a) Gaussian Approximation $s1$(above) and $s2$(below)



(b) P(s1|y=1)(above) and P(s2|y=1) (below)

Figure 8: Posterior distributions