Mini Project in Advanced Probabilistic Machine Learning, Group: 11

Fredrik Jonasson **Aikaterini Manousidou** **Amanda Seger**

Abstract

This project involves focuses on predicting the outcome of a game, no matter the sport. A popular model, developed by `Microsoft`, is called TrueSkill and can be used to predict the skill of players based on previous performance. This model includes a Bayesian method that uses random variables, which are drawn from Gaussian distributions, to predict the skill and ranking of players. This model increases the skill of winning players while at the same time decreasing the skill of losing players. Assumed Density Filtering was implemented to analyse the players rankings. Thereafter, Gibbs sampling and Message Passing were implemented in order to calculate the posterior distribution of the players. The model was tested with statistics of teams from an Italian football league season *(Serie A)*, where the prediction rate was 62.50%, if not considering draws. Then, the model was evaluated on a new data set, containing statistics of teams from the Swedish hockey league *(SHL)*, where the performance of the model was 55.22%. An extension experiment, introducing weights to the model in order to increase the accuracy of the model was performed with satisfactory results, to a prediction rate of 57.69% for the SHL - data set. The testing and the evaluation produced an acceptable accuracy.

Introduction

Ranking teams or players with algorithms and predictions is a way to make the game more interesting by matching players with similar skill against each other. The TrueSkill [1] model developed by Microsoft is a way to rank players with Bayesian probabilistics. In this project, the TrueSkill algorithm is used to rank teams after their skill with data from the Italian football league Serie A.

Modeling

The model, for the Trueskill Bayesian for one match between two player, is a joint distribution of the four random variables given in Equation (1). The model consists of three random Gaussian variables s_1, s_2, presenting the skills of the two players, and t with $\mu_t = s_1 - s_2$ for the outcome of the game. The result of the model is then defined by the discrete random variable $y = sign(t)$.

$$p(s_1, s_2, t, y) = p(s_1|s_2, t, y)p(s_2|t, y)p(t|y)p(y) \tag{1}$$

$$p(\mathbf{s}, t, y) = p(\mathbf{s}|t, y)p(t|y)p(y) \tag{2}$$

$$p(s_1) \sim \mathcal{N}(s_1, \mu_{s_1}, \sigma^2_{s_1}) \tag{3}$$

$$p(s_2) \sim \mathcal{N}(s_2, \mu_{s_2}, \sigma^2_{s_2}) \tag{4}$$

$$p(t|s_1, s_2) = p(t, \mathbf{s}) \sim \mathcal{N}(t, \mu_{t|\mathbf{s}}, \sigma^2_{t|\mathbf{s}}) \tag{5}$$

where $\mathbf{s} = [s_1, s_2]$, $\mu_{t|\mathbf{s}} = s_1 - s_2$. Therefore, the hyper parameters are μ_{s_1}, μ_{s_2}, $\sigma^2_{s_1}$, $\sigma^2_{s_2}$ and $\sigma^2_{t|\mathbf{s}}$

$$p(y|t) = \delta(y, t) = \delta(y - sign(t)) = \begin{cases} 1, \text{ if } y = \pm 1 \\ 0, \text{ otherwise} \end{cases} \tag{6}$$

Conditional Independence

According to the problem description, the Gaussian random variable t seems to only be dependent on $\mathbf{s}$. Similarly, the discrete random variable y seems to solely be dependent on t. Assuming that t is an observed variable then the relation between the variables is according to Equation (7). Thus, $\mathbf{s}$ and y are conditionally independent, meaning $(s \perp\!\!\!\perp y|t)$.

$$p(\mathbf{s}, y|t) = \frac{p(\mathbf{s}, y, t)}{p(t)} = \frac{p(\mathbf{s}|t, y)p(y|t)p(t)}{p(t)} = p(y|t)p(\mathbf{s}|t) \tag{7}$$

Computing with the Model

From the Bayesian Network in Figure 1a, it can be seen that the model in Equation (2) can be simplified to $p(s_1, s_2, t, y) = p(t|s_1, s_2)p(y|t)p(s_1)p(s_2)$. This, together with Equation (7), results in the following expression for $p(s_1, s_2|t, y)$, shown in Equation (8).

$$p(\mathbf{s}|t, y) = \frac{p(t|\mathbf{s})p(y|t)p(\mathbf{s})}{p(y|t)p(t)} = \frac{p(t|\mathbf{s})p(\mathbf{s})}{p(t)} = p(\mathbf{s}|t) \tag{8}$$

Since both $p(\mathbf{s})$ and $p(t|\mathbf{s})$ are Gaussian distributed, then $p(t)$ is also Gaussian distributed and its mean and variance can be determined as according to `Corollary 2`, see Equation (9), where $A = [1, -1]^T$.

$$\begin{aligned} p(t) &\sim \mathcal{N}(t, \mu_t, \sigma_t^2) \\ \mu_t &= A\mu_{\mathbf{t}|\mathbf{s}} = \mu_1 - \mu_2 \\ \sigma_t^2 = \sigma^2_{t|s} + A^T \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} A &= \sigma^2_{t|s} + \sigma_1^2 + \sigma_2^2 \end{aligned} \tag{9}$$

Using Bayes' theorem and knowing whether y is observed, thus the sign of t is known, will then result in a truncated Gaussian. $p(t|s_1, s_2, y)$ must be zero when the sign off t is not y and this can be expressed with a Dirac delta function, seen in Equation (10).

$$p(t|s_1, s_2, y) \propto p(t|s_1, s_2)\delta\left(y - \text{sign}(t)\right) \tag{10}$$

The marginal probability that Player 1 wins is $p(y = 1)$, which equals $p(t > 0)$ and is obtained by integrating the expression in Equation (9).

$$p(y = 1) = p(t > 0) = 1 - p(t = 0) \tag{11}$$

$$p(t > 0) = \int_0^\infty \mathcal{N}(t, \mu_t, \sigma_t^2) = \int_0^\infty \frac{1}{2\pi\sigma_t^2} e^{-\frac{(t-\mu_t)^2}{2\sigma_t^2}} = 1 - (0 - \frac{1}{2\pi\sigma_t^2} e^{\frac{\mu_t^2}{2\sigma_t^2}}) \tag{12}$$

Bayesian Network

The Bayesian network, as seen in Figure 1a, is constructed to depict the model given in this project. According to Figure 1a, s_1,s_2 and y are graphically conditionally dependent. However, if $\mathbf{s} = [s_1, s_2]$ then the model can be depicted as the Bayesian network seen in Figure 1b, where $\mathbf{s}$ and y are conditionally independent and a head-to-tail network can be seen.

(a) The Bayesian Network **(b)** The simplified Bayesian Network

Figure 1: The Bayesian Networks for this model.

Gibbs Sampler

With a Gibbs sampler algorithm one can estimate the skill of a player based on a previous game. In the sampling, the same initial skill was assumed for the players, the initial skill of the players was modeled by the same Gaussian distributions, $\mu_1 = \mu_2$ and $\sigma_1 = \sigma_2$.

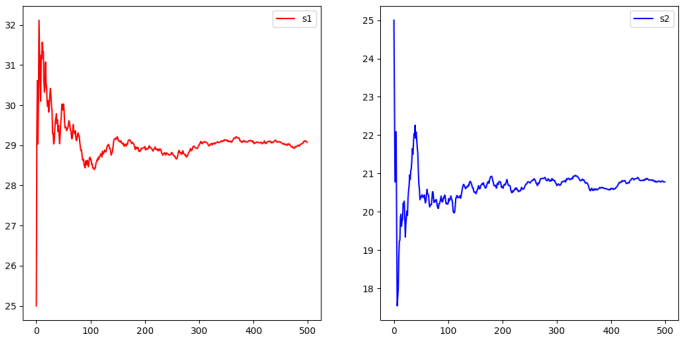


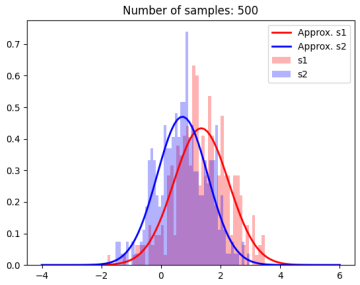
Figure 2: Figures over the means of s_1 and s_2.

The burn in period is the amount of samples that is needed for the mean of the players skill to converge. One way to determine how many samples need to be discarded is to calculate the mean of all the previous samples for a given sample path. The mean will converge towards the true mean, however the burn-in affects how fast this is done. In Figure 2 the means are plotted. It is evident that both means seem to stabilize after 200 samples, therefore a burn-in of 200 samples was chosen.

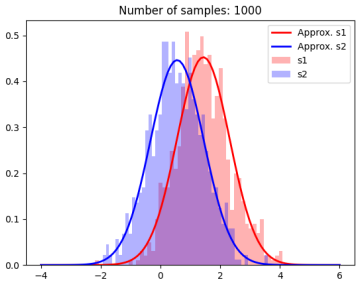
The number of samples was varied in order to find a sample size that produces good results while at the same time not emphasizing the computational cost. The numbers of samples tested were 500, 1000, 1500 and 2000 and the histograms and calculated posteriors for each of the cases can be found in Figure 3. The run-times for each of the sample sizes can be seen in Table 1.

In Figure 3a, it is evident that there still are a lot of variations in the posteriors. These variations seem to disappear as more samples are introduced, which can be seen in the rest of the figures. However, the differences between the histograms in Figures 3b, 3c and 3d are not as significant as the differences in the run-times for the respective cases. Therefore, a size of 1000 samples was used for the remainder of the project, which generates good results while not being too heavy computationally.

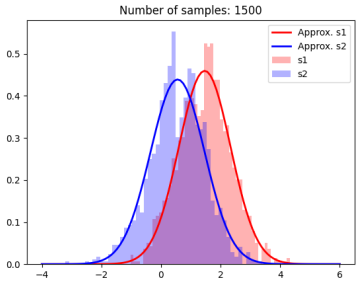
In Figure 4, the priors of the Gaussian random variables s_1 and s_2 are compared with their corresponding approximated posteriors. It can be concluded that the posterior for s_1 has shifted to the



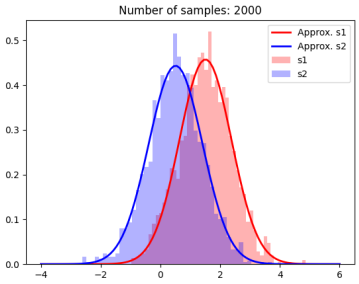
(a) 500 Samples



(b) 1000 Samples



(c) 1500 Samples



(d) 2000 samples

Figure 3: Figures over the distributions for different amounts of samples.

Table 1: Run-time for different number of samples

Samples	500	1000	1500	2000
Time [s]	0.15625	0.859375	1.796875	2.875

right, meaning that the mean has increased. Accordingly, the posterior for s_2 has shifted to the left, which means that the mean has decreased. Moreover, both posteriors seem to have slightly slimmer bell curves, meaning a reduction in variance. Therefore, it can be concluded that the skill of s_1 has risen, while the skill of s_2 has fallen, while both variances decrease since the model becomes more reliable.

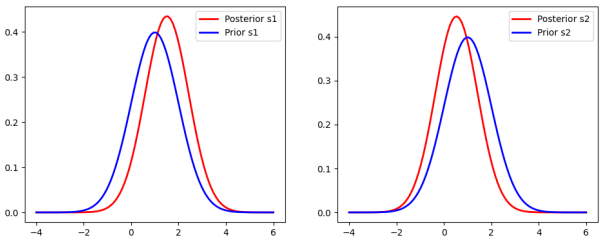


Figure 4: The prior $p(s_1)$ versus the posterior $p(s_1|y=1)$, respective prior $p(s_2)$ versus the posterior $p(s_2|y=1)$

Assumed Density Filtering

In this section, Assumed Density Filtering was implemented in order to process the stream of matches between the teams while using the posterior distribution of the previous match as the prior distribution for the current match. This was performed both when the matches were ordered (the initial data) and

when the matches were randomly shuffled. The teams start with mean $\mu = 25$ and variance $\sigma = \frac{25}{3}$ [1]. The results and the final ranking can be seen in Table 2.

Table 2: Run-time for different number of samples

Ordered Team Names				Shuffled Team Names			
Team	Skill	St. Deviation	Rank	Team	Skill	St. Deviation	Rank
Torino	31.891047	1.755752	1	Juventus	32.048216	1.905635	1
Juventus	31.741627	2.108421	2	Torino	31.681313	2.134662	2
Milan	30.447879	1.777098	3	Milan	29.207301	2.305139	3
Napoli	30.269304	2.170079	4	Napoli	29.100913	1.745115	4
Atalanta	28.779017	1.955832	5	Inter	28.253576	1.758871	5
Roma	28.713603	1.874686	6	Atalanta	28.102441	1.530777	6
Inter	28.455179	1.731672	7	Roma	27.927912	1.865664	7
Sampdoria	26.824184	1.796849	8	Lazio	25.816467	1.531834	8
Lazio	25.760704	1.802561	9	Sampdoria	25.640691	2.001874	9
Bologna	24.914862	1.756016	10	Bologna	24.679258	1.838560	10
Udinese	23.486190	2.130260	11	Udinese	23.223301	1.838934	11
Empoli	23.100523	2.016000	12	Fiorentina	23.040708	1.893416	12
Fiorentina	22.546451	1.958640	13	Cagliari	22.410705	2.062775	13
Cagliari	22.303006	1.695036	14	Empoli	22.294303	1.881577	14
Parma	22.243202	1.967670	15	Genoa	22.128341	1.563842	15
Sassuolo	22.229554	1.794753	16	Sassuolo	22.097743	1.582902	16
Genoa	21.808070	1.853396	17	Parma	21.909478	1.969877	17
Spal	20.768789	1.732461	18	Spal	21.305187	1.650730	18
Chievo	18.807105	1.805011	19	Frosinone	19.142703	1.882927	19
Frosinone	18.142503	2.382927	20	Chievo	17.681578	1.942790	20

Since Gibbs sampling is a stochastic method, the random variables generated will differ from run to run and as a consequence the results of a run may differ. However, the outcome should converge to a solution given that a large enough number of samples is used.

Predicting

Using an Assumed Density Filtering (ADF) prediction model, the calculated posterior from the last game is used as a priori for the next prediction. The player which has the highest skill, the majority of the time, is predicted to win. This prediction method gives $62,5\%$ if the matches that end in a draw are discarded. By randomizing the match ordering, the prediction rate changes to $52,3\%$, which still gives a better result than random guessing (which is 50%).

Factor Graph

In Figure 5 the factor graph of the model, together with all the messages, is illustrated. The factors in Equation (13) are obtained by Equations (3), (4), (5) and (6), respectively. Furthermore, the messages in Equation (14) are obtained from Figure 5 together with the factors in Equation (13).

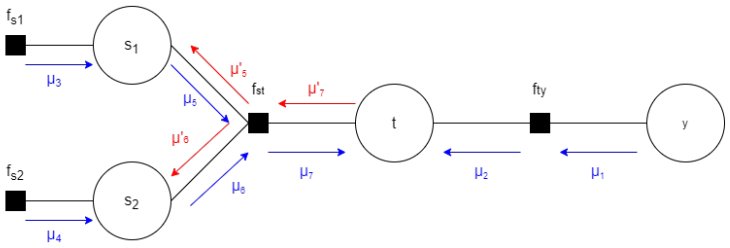


Figure 5: Schematic figure over the factor graph of the model.

$$
\begin{aligned}
f_{s_1}(s_1) =& \mathcal{N}(s_1, \mu_{s_1}, \sigma^2_{s_1}) \\
f_{s_2}(s_2) =& \mathcal{N}(s_2, \mu_{s_2}, \sigma^2_{s_2}) \\
f_{st}(\mathbf{s}, t) =& \mathcal{N}(s, \mu_{st}, \sigma^2_{st}) \\
f_{ty}(t, y) =& \delta(y - sign(t))
\end{aligned}
\tag{13}
$$

$$
\begin{aligned}
\mu_1 =& 1 \\
\mu_2 =& f_{ty} \\
\mu_3 =& f_{s_1} \\
\mu_4 =& f_{s_2} \\
\mu_5 =& \mu_3 \cdot 1 = f_{s_1} \\
\mu_6 =& \mu_4 \cdot 1 = f_{s_2} \\
\mu_7 =& f_{st} \cdot \mu_5 \cdot \mu_6 = f_{st} f_{s_2} f_{s_1}
\end{aligned}
\tag{14}
$$

The message μ_7 can be marginalized with the help of `Corollary 2`, see Equation (15), as previously shown in Equation (9).

$$
\mu_7(t) = \int f_{st}(\mathbf{s}, t) \cdot \mu_5 \cdot \mu_6 d\mathbf{s} = \mathcal{N}(t, \mu_1 - \mu_2, \sigma^2_{t|s} + \sigma^2_1 + \sigma^2_2) \tag{15}
$$

Message-passing Algorithm

The factor μ'_7 is not Gaussian, when applying the trick to multiply and divide by μ_7, a truncated Gaussian is obtained. Using `Moment matching` on the truncated Gaussian, an approximation of μ'_7 is obtained, seen in Equation (16).

$$
\mu'_7(t) = \mu_2 = \frac{\mu_2 \cdot \mu_7}{\mu_7} \approx \frac{\hat{q}(t)}{\mu_7} \approx p(t) = \mathcal{N}(t, \mu_t, \sigma^2_t) \tag{16}
$$

The final messages from *t* to *s* (μ'_5 and μ'_6), defined in Equation (17), are given by the marginal of μ'_7, μ_5 and μ_6. Finally, using `Corollary 2`, the marginal of s_1 and s_2 can be obtained by Equation (18).

$$
\begin{aligned}
\mu'_5(s_1) &= \int f_{st}(\mathbf{s}, t) \cdot \mu_6 \cdot \mu'_7 dt = \mathcal{N}(s_1, \mu_1 + \mu_2, \sigma^2_{t|s} + \sigma^2_1 + \sigma^2_2) \\
\mu'_6(s_2) &= \int f_{st}(\mathbf{s}, t) \cdot \mu_5 \cdot \mu'_7 dt = \mathcal{N}(s_2, \mu_1 - \mu_2, \sigma^2_{t|s} + \sigma^2_1 + \sigma^2_2)
\end{aligned}
\tag{17}
$$

$$
\begin{aligned}
p(s_1|t) &= \mu_3 \cdot \mu'_5 = \mathcal{N}(s_1, \mu_{s_1}, \sigma^2_{s_1}) \cdot \mathcal{N}(s_1, \mu_1 + \mu_2, \sigma^2_{t|s} + \sigma^2_1 + \sigma^2_2) \\
p(s_2|t) &= \mu_4 \cdot \mu'_6 = \mathcal{N}(s_2, \mu_{s_2}, \sigma^2_{s_2}) \cdot \mathcal{N}(s_2, \mu_1 - \mu_2, \sigma^2_{t|s} + \sigma^2_1 + \sigma^2_2)
\end{aligned}
\tag{18}
$$

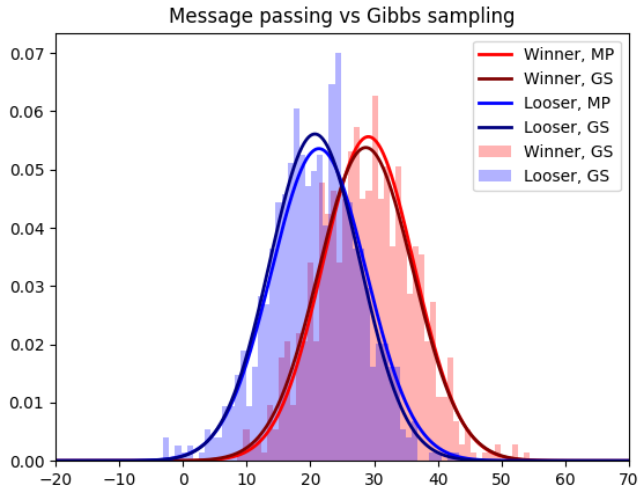


Figure 6: The priors calculated with Gibbs sampling and message passing.

Our own Data

One way to extend the project is to try the TrueSkill model on a new data set with results from the SHL season 2020-2021 [2]. The model evaluated was **ADF** with Gibbs Sampling. All matches in this data set end with a winner and a loser, the matches can go to overtime and Shoot-out but cannot end with a draw. On this data set, the model managed to predict 55.22% of the matches correctly. This is a better result than what the model managed to predict for the original data set of football matches, if the matches that end in a draw were also counted. With the matches that ended with a draw the model only managed to predict 47% of the matches correctly. One reason why the model performs better with the secondary data is because all matches end with a winner and a loser and no draws are possible.

In the Table 3 the predictions of the skills and the ranks of all teams in the 20-21 season of SHL, are demonstrated. Compared with the end table of the finished season, the top 4 teams that the model predicts with the highest skill finished top 4 in the original season. The same can be seen for the bottom 4 teams.

Table 3: Prediction for the SHL 20/21 season

Team	Skill	St. Deviation	Rank
Leksand	29.395472	1.321738	1
Skellefteå	28.446541	1.729087	2
Rögle	27.877047	1.922858	3
Vaxjö	27.534237	1.983769	4
Frölunda	27.385875	1.826358	5
Örebro	26.767322	1.619832	6
Färjestad	26.484391	1.601648	7
Luleå	25.995930	1.386205	8
Djurgården	24.392890	1.678789	9
Malmö	24.146039	1.175442	10
Oskarshamn	22.758793	1.282689	11
Linköping	22.617658	2.054682	12
HV71	22.074557	1.707033	13
Brynäs	22.067252	1.511484	14

Open-ended Project Extension

An extension that was implemented but deemed unsuccessful was a way to predict a draw in the games. This was done by controlling the absolute value of the variable t and comparing it to a fixed value. This extension resulted in worse results than the initial implementation and therefore it was not pursued further.

Another performance improvement would be to introduce weights to the model. This is implemented by updating the hyper-parameters depending on how large the score difference is between the teams. In this extension, the skill improves more when a team wins with a larger score difference. Similarly, the skill decreases more when the team loses with a large score difference. Different weights were tried in order to find the relationships generating the best prediction rates. The best parameters found was the following; if the score difference equals 2, the mean skill of the winner increases by a factor of 1.1 and the mean skill of the loser decreases by a factor of $\frac{1}{1.1}$. Furthermore, if the goal difference is equal or greater than 3, the winner's mean skill increases by a factor of 1.2 and the loser's decreases by a factor of $\frac{1}{1.2}$. This extension increased the prediction rates for both data sets, to 57.69% for the `SHL`-data set and to 49.47% respective 69.12% (if draws are disregarded) for the `SerieA`-data set.

References

[1] R. Herbrich, T. Minka, and T. Graepel, “TrueSkill: A Bayesian Skill Rating System,”

[2] “SHL table of 20/21 season.”