Report coursework 2

4F13 - Probabilistic Machine Learning

November 18, 2021

Candidate no.: J902C

Word count: 1000

UNIVERSITY OF CAMBRIDGE
Department of Engineering

Trumpington Street
Cambridge CB2 1PZ
Tlf.: +44 01223 332600
http://www.eng.cam.ac.uk/

Table of contents

1. Question a)

After completing the code in `gibbsrank.py` (figure 1.1), Gibbs sampling can be executed to sample player skills. Some of them are plotted in figure 1.2.

```python
# Jointly sample skills given performance differences
m = np.zeros((M, 1))
for p in range(M):
    # TODO: COMPLETE THIS LINE
    m[p] = np.dot(t.transpose(), (p == G[:,0]).astype(int) - (p == G[:,1]).astype(int))

iS = np.zeros((M, M))
for g in range(N):
    # TODO: COMPLETE THIS
    iS[G[g,0],G[g,0]] += 1
    iS[G[g,1],G[g,1]] += 1
    iS[G[g,0],G[g,1]] -= 1
    iS[G[g,1],G[g,0]] -= 1
```

Figure 1.1: Completed code in `gibbsrank.py`

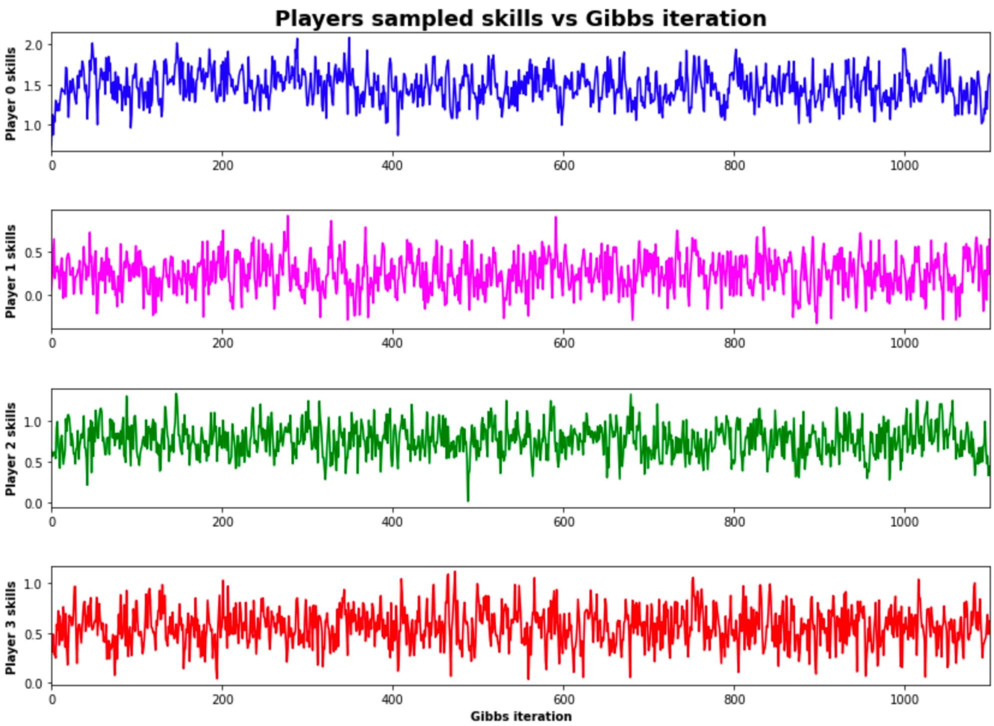
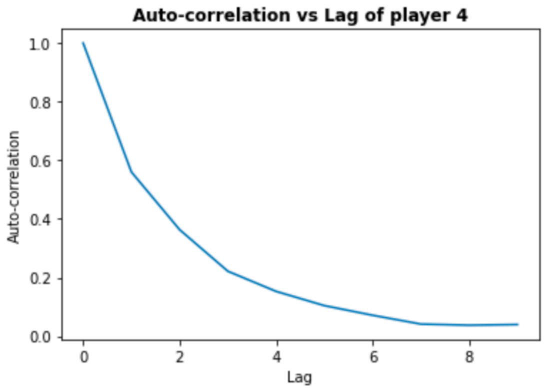
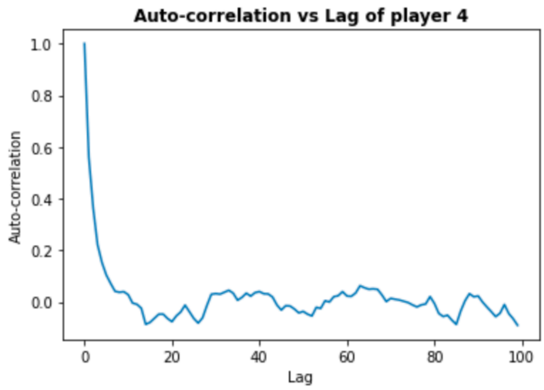


Figure 1.2: Some sampled player skills as a function of Gibbs iteration

Usually when executing Gibbs sampling (GS), the initial sequence of samples is discarded, until the chain has converged to the desired distribution. This is called **burn-in**. In the current problem, skill samples are converging quite fast (check visually the first iterations of figure 1.2), so the burn-in period will be low, around **20-30 iterations**. Question b (section 2) will explain the concept of *convergence*.
Additionally, to get less dependence between samples, GS is often run for a long time, and samples are **thinned** by keeping only every m-th sample.



(a) Auto-correlation plotted against lag



(b) Auto-correlation plotted against lag (greater)

Figure 1.3: Auto-correlation vs Lag for a certain player

After plotting the auto-correlation of samples of a certain player against the lag, shown in figures 1.3, it's reasonable to keep only **10-th sample** to obtain **(pseudo)-independent samples**. If 100 is a reasonable number of samples after burn-in and thinning, then **GS should be executed for about 1100 iterations**.

2. Question b)

Gibbs sampling method tries to obtain a set of independent samples to describe the (intractable) joint distribution. Therefore, convergence is achieved when the majority of samples stay steady for plenty of iterations. Plotting sample mean μ and sample standard deviation σ can confirm that the sample distribution has converged quite quickly, as shown in figure 2.1. So, taking into account *burn-i*n, *thinning* and *convergence velocity*, **1100 iterations is still a good amount of iterations for GS**.

In message passing (MP), the mean and variance of each player skill are computed to estimate games outcomes. Every iteration of MP the estimated means and variances are updated, and convergence is achieved when the changes are really small (less than a certain tolerance). In the figure 2.2 the skill means and variances for 3 players are plotted vs EP iteration. The point where the estimated object has changed less than $tol = 10^{-3}$ 10 iterations in a row is represented. As 2.2 shows, **50 iterations** are sufficient to achieve convergence for MP algorithm, although fewer iterations could be used, but since MP is really fast, 50 iterations is a reasonable number.

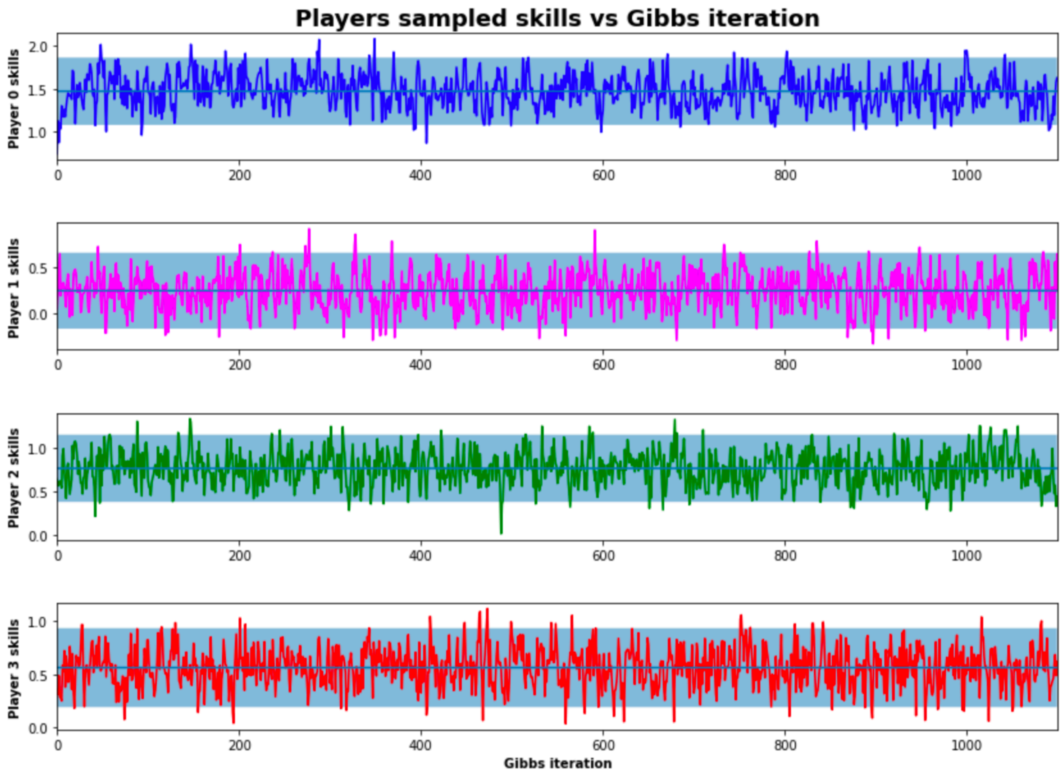


Figure 2.1: Plotting Gibbs samples, its mean μ and the shared area is $\mu \pm 2\sigma$

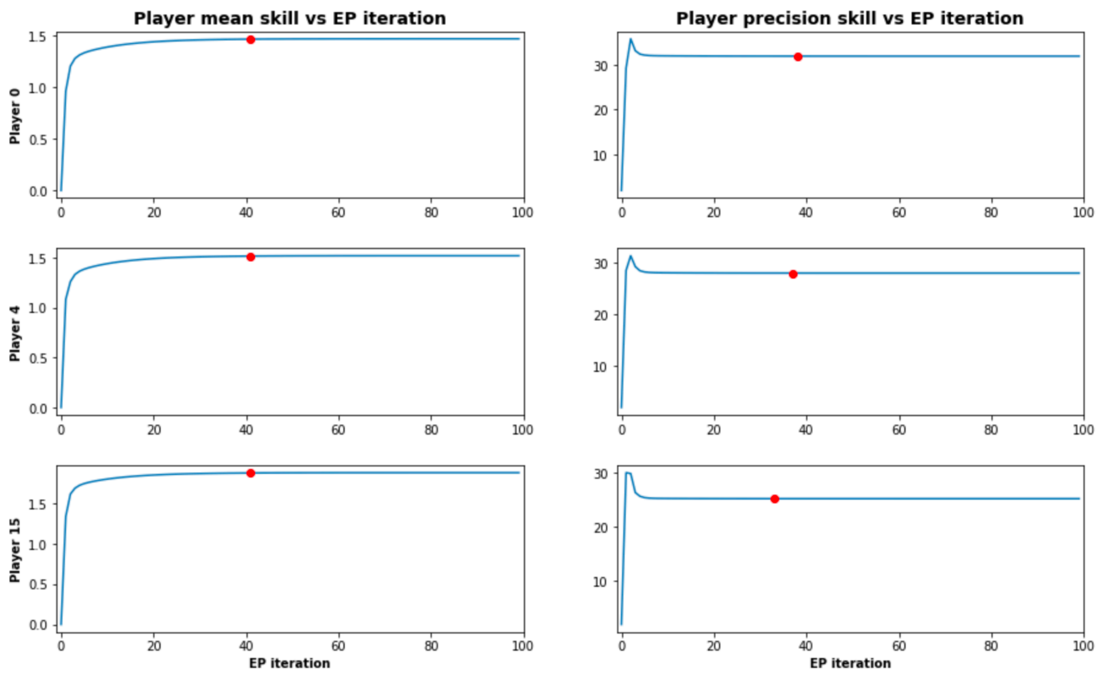


Figure 2.2: Skill mean and variance vs EP iteration for Nadal, Federer and Djokovic

3. Question c)

The 4 top players according to the ATP ranking when the data was collected is: Djokovic, Nadal, Federer and Murray.

The table 3.1 contains the probabilities that the skill of one player is higher than the other. To calculate the probability that the player i is greater than that of player j the following reasoning is applied: w_i is skill player i, w_j skill player j, $p(w_i > w_j) = p(w_i - w_j > 0)$. Assuming $w_i \sim N(\mu_i, \sigma_i^2)$ and $w_j \sim N(\mu_j, \sigma_j^2)$, then:

$$p(w_i - w_j > 0) = \int_0^\infty N(w_i - w_j; \mu_i - \mu_j, \sigma_i^2 + \sigma_j^2) = 1 - \int_{-\infty}^0 N(w_i - w_j; \mu_i - \mu_j, \sigma_i^2 + \sigma_j^2) \quad (3.1)$$

In python this can be computed by using ($\star$):

```
mu=meanSkills[i]-meanSkills[j]
var=1/precisionSkills[i]+1/precisionSkills[j]
probSkill(i,j)=1-scipy.stats.norm.cdf(0, mu, np.sqrt(var))
```

	Novak-Djokovic	Rafael-Nadal	Roger-Federer	Andy-Murray
Novak-Djokovic	---	0.9398	0.9089	0.9853
Rafael-Nadal	0.0602	---	0.4272	0.7665
Roger-Federer	0.0911	0.5728	---	0.8108
Andy-Murray	0.0147	0.2335	0.1892	---

Figure 3.1: Probabilities that the skill of one player is higher than the other

The table 3.2 contains the probabilities of one player winning the other one. Since the predicted game outcome depends on the skill difference plus some noise ϵ with mean 0 and variance 1, then the equations are quite similar, we just have to add the noise variance:

$$p(w_i - w_j + \epsilon > 0) = 1 - \int_{-\infty}^0 N(w_i - w_j + \epsilon; \mu_i - \mu_j, 1 + \sigma_i^2 + \sigma_j^2) \quad (3.2)$$

In python this can be computed as in 1 by just changing:

```
var=1.0+1/precisionSkills[i]+1/precisionSkills[j]
```

The **main difference between theese tables** is that the winning probabilities account for the possible **noise** that a tennis game can have (player motivation, terrain, fan support) and not just the skill difference.

	Novak-Djokovic	Rafael-Nadal	Roger-Federer	Andy-Murray
Novak-Djokovic	---	0.6554	0.638	0.7198
Rafael-Nadal	0.3446	---	0.4816	0.5731
Roger-Federer	0.362	0.5184	---	0.5909
Andy-Murray	0.2802	0.4269	0.4091	---

Figure 3.2: Probabilities of one player winning the other one

4. Question d)

GS algorithm returns a set of sampled player skills. The skills of two players can be compared in different ways:

First, a player skill w_i can be approximated by considering $w_i \sim N(\mu_i, \sigma_i^2)$, where μ_i is the **skill mean** for player i and σ_i^2 is the **skill variance**. Then, the probability that the skill of one player is higher than other can be computed exactly as done in 3.1. In table 4.2a the skills of Nadal and Djokovic are compared after using GS and approximating their marginal skills by Gaussians.

Another method to compare two players skills is to consider their joint distribution:

$$\begin{bmatrix} w_i \\ w_j \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_i \\ \mu_j \end{bmatrix}, \Sigma\right), \tag{4.1}$$

where Σ is the covariance matrix between w_i and w_j. The probability that the skill of one player is higher than other can be computed by calculating the area under the 2D-gaussian pdf limited by the plane $\{x = y, z \in \mathbb{R}\}$, as shown in figure 4.1.

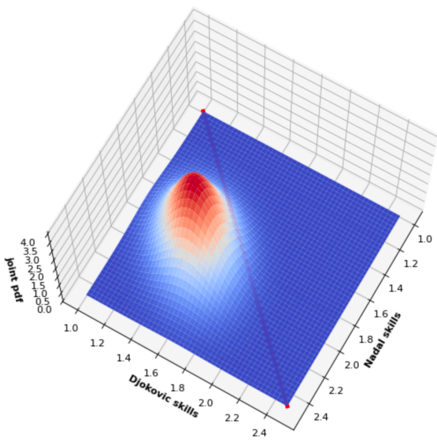


Figure 4.1: Probability that Djokovic skill is higher than Nadal's (around 94%)

	Novak-Djokovic	Rafael-Nadal
Novak-Djokovic	---	0.9192
Rafael-Nadal	0.0808	---

(a) Marginal skills

	Novak-Djokovic	Rafael-Nadal
Novak-Djokovic	---	0.9401
Rafael-Nadal	0.0599	---

(b) Joint skills

	Novak-Djokovic	Rafael-Nadal
Novak-Djokovic	---	0.9383
Rafael-Nadal	0.0617	---

(c) Samples

Figure 4.2: Probability that the skill of one player is higher than other one by approximating their skills with different approaches

The third technique is **counting** how many **skill samples** of player i are greater than the skill samples of player j and divide it by the total number of skill samples. In python:

```
prob=np.mean(skillSamples[i]>skillSamples[j])
```

To sum up table 4.2, all three methods predict that the **probability of Djokovic having more skill than Nadal is about 91-94%**. However, the method that approximates the skills by a joint Gaussian **takes into account the correlation between the skills** of two players (it isn't *isotropic* like the *marginal* approach). If the number of samples goes to *infinity*, the approximation by a joint Gaussian and the inference using just samples should give the same results, but since the no. samples tends to be considered finite, **the approximation by a joint Gaussian should be the preferred approach**.

	Novak-Djokovic	Rafael-Nadal	Roger-Federer	Andy-Murray
Novak-Djokovic	---	0.9401	0.8834	0.981
Rafael-Nadal	0.0599	---	0.3955	0.7542
Roger-Federer	0.1166	0.6045	---	0.8082
Andy-Murray	0.019	0.2458	0.1918	---

Figure 4.3: Probabilities that the skill of one player is higher than the other using Gibbs sampling

In figure 4.3 a 4 by 4 table for the skills is shown, and it can be compared to figure 3.1 resulted from using MP algorithm. It's easy to check that the **probabilities are quite similar using both methods**.

5. Question e)

The rankings of players can be compared using different methods of inference. In figure 5.1 **empirical games outcome averages** are plotted. They're computed by **counting how many games a player has won divided by the total played games**. There're players that have no victories, so their predicted outcomes are zero.

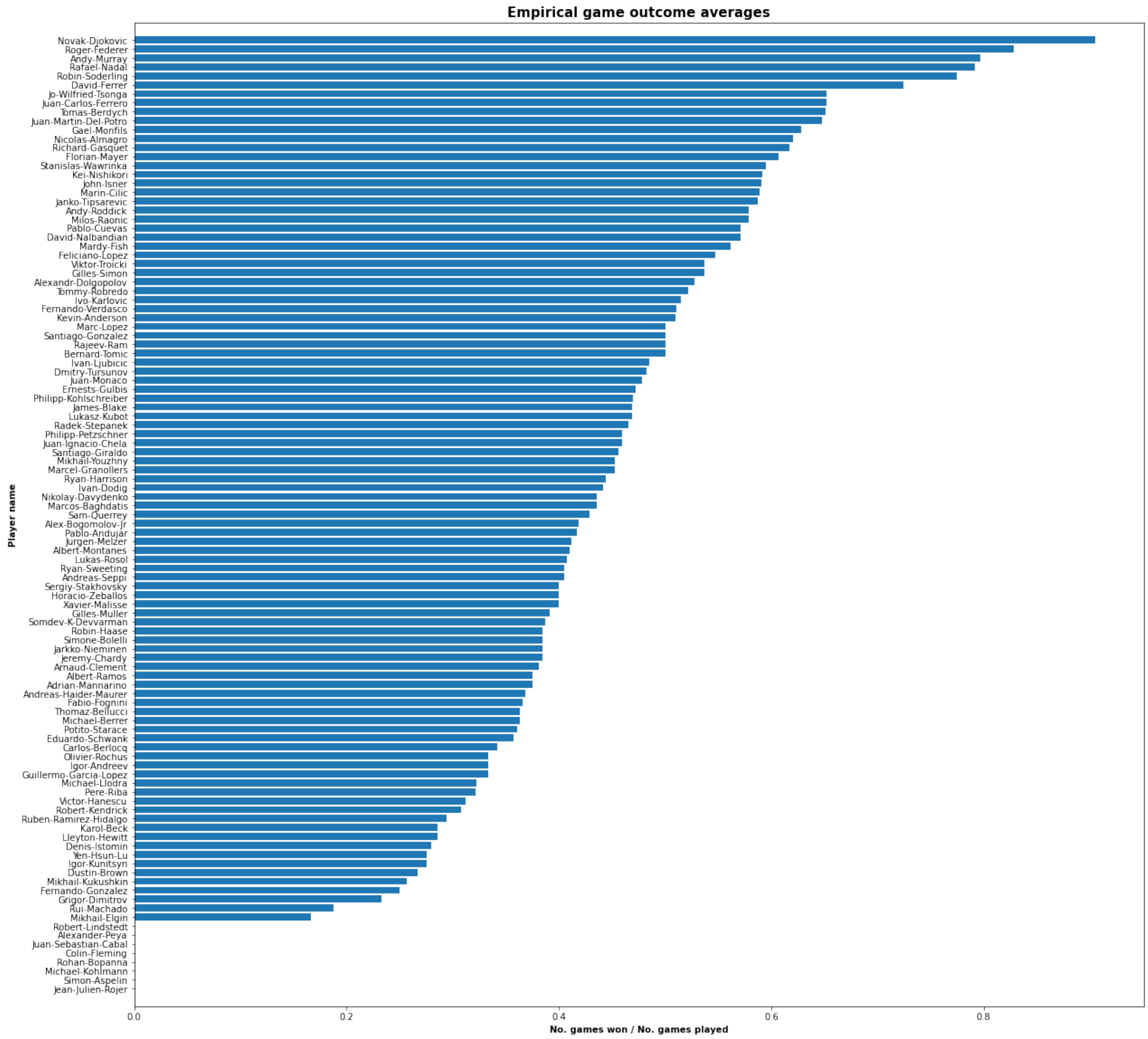


Figure 5.1: Empirical game outcomes averages

Gibbs samples can be used to predict game outcomes. This ranking is represented in figure 5.2. The plotted probabilities are the mean probability of winning for a certain player vs the rest. This probabilities have been calculated using equation 3.2 using the Gibbs samples as explained in section 4.

Similar to the previous ranking, **MP algorithm** can be used to get the ranking by computing the probabilities of one certain player winning the rest of players (figure 5.3), using equation 3.2 as done in section 3.

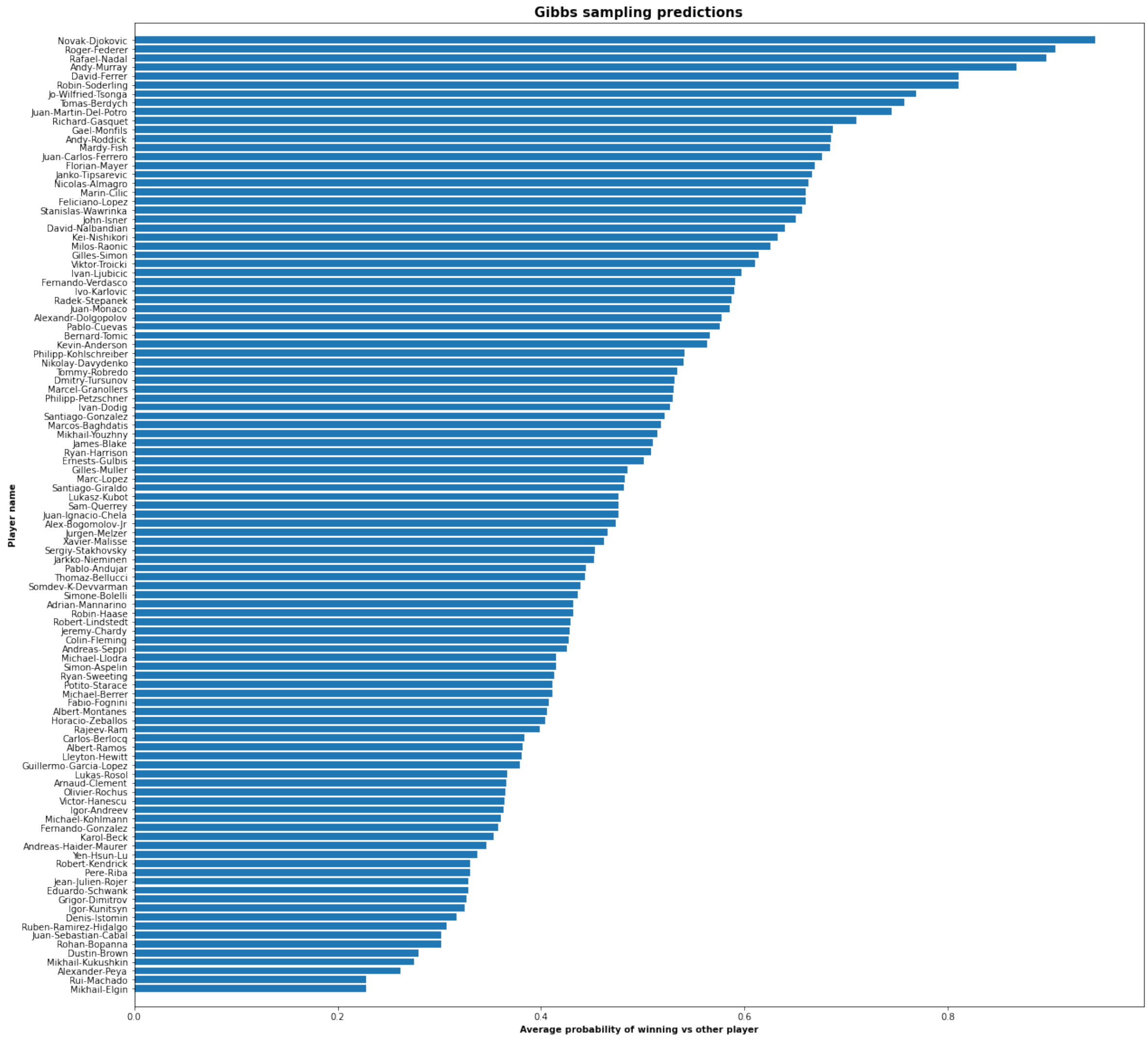


Figure 5.2: Predicted game outcomes using Gibbs sampling

To sum up, **rankings using GS and MP are quite similar**: both methods have been proven effective to tackle this problem. On the other hand, ranking using just game outcomes seems worse, since the no. games isn't large enough.

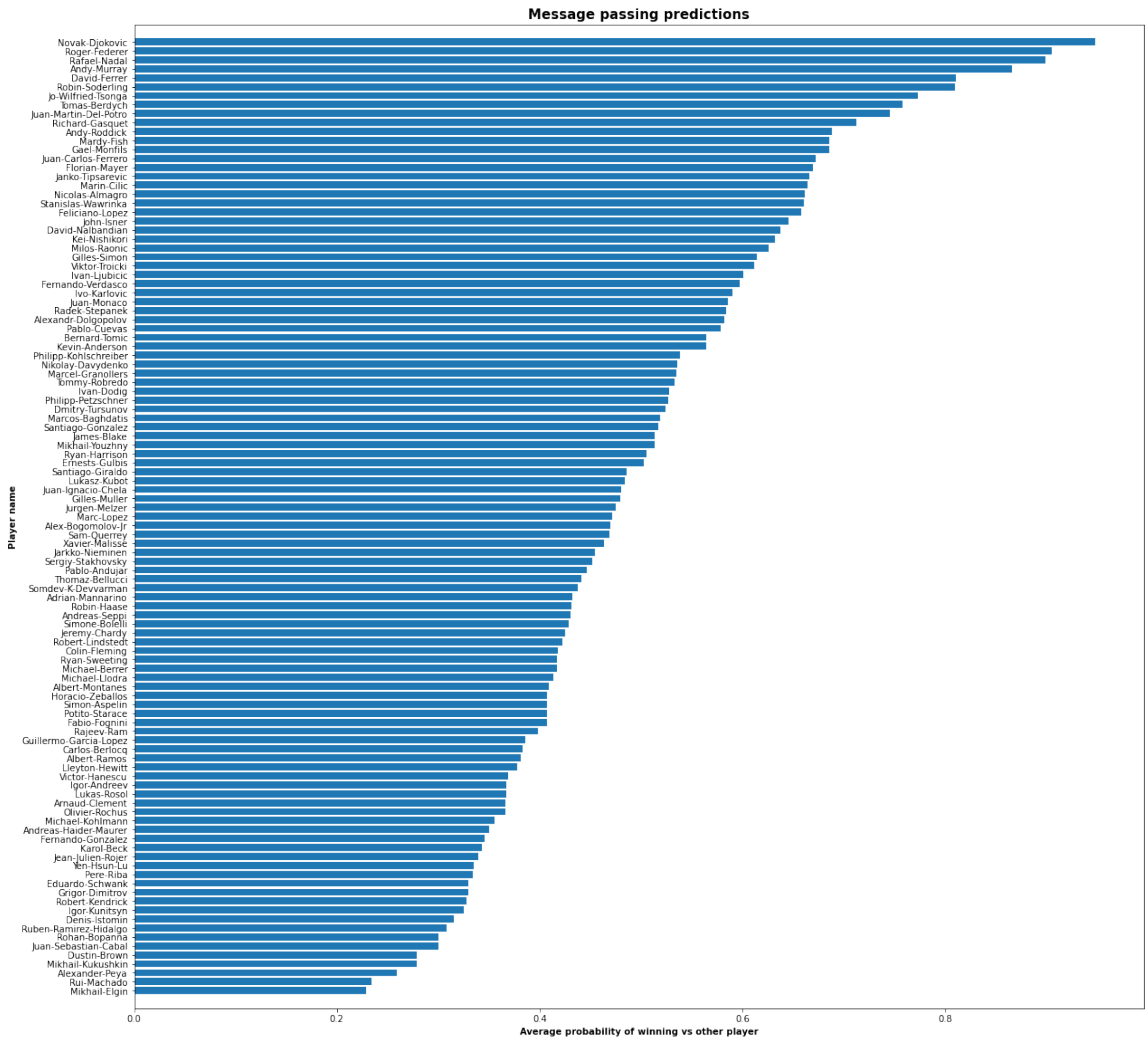


Figure 5.3: Predicted game outcomes using Message Passing

Bibliography

[1] Probabilistic Ranking slides (2021). Carl Edward Rasmussen, José Miguel Hernández Lobato, David Krueger. https://www.vle.cam.ac.uk/course/view.php?id=69021&sectionid=252511#section-7

[2] Herbrich, Ralf and Minka, Tom and Graepel, Thore. (2007). *TrueSkill™: A Bayesian Skill Rating System*. The MIT Press. https://proceedings.neurips.cc/paper/2006/file/f44ee263952e65b3610b8ba51229d1f9-Paper.pdf