Bayesian Inference to Predict La Liga Football Results

Harrison Zhu
Department of Mathematics, EPFL
June 20, 2018

Project Outline

- Motivation
- Data
- Modelling Approaches
  - Model 1: Bayesian Logistic Regression
  - Model 2: Bayesian Poisson Regression
  - Model 3: Modified Bayesian Poisson Regression
- Test Results
- Conclusion

Motivation

There are currently 3.5 billion football fans in the world [1] and the football betting industry is worth billions.

- As a result, advanced mathematical modelling techniques are used to predict football results.
- Publications began in the 1990s, where Moroney proposed Poisson and negative binomial models.
- Methods have involved: GLMs, Bayes filter, and more recently Bayesian neural networks.

Motivation

We will present 3 different Bayesian GLM approaches and 3 approximation methods. Namely, the **models**:

1. Logistic Regression
2. Poisson Regression
3. Modified Poisson Regression,

Motivation

We will present 3 different Bayesian GLM approaches and 3 approximation methods. Namely, the **models**:

1. Logistic Regression
2. Poisson Regression
3. Modified Poisson Regression,

all with Gaussian prior $f(\boldsymbol{\theta}) = \exp[-\frac{1}{2}||\boldsymbol{\theta}||^2]$ and the **approximation methods**:

Motivation

We will present 3 different Bayesian GLM approaches and 3 approximation methods. Namely, the **models**:

1. Logistic Regression
2. Poisson Regression
3. Modified Poisson Regression,

all with Gaussian prior $f(\boldsymbol{\theta}) = \exp[-\frac{1}{2}||\boldsymbol{\theta}||^2]$ and the **approximation methods**:

1. Laplace Approximation optimised with minibatch stochastic gradient descent (SGD),
2. Metropolis algorithm with backtracking on the coefficient of the uniform noise and a preemptive loop to find a good starting λ that gives an acceptance rate between $10 - 50\%$,
3. Gaussian Variational Method (GVA).

Motivation: Notes on the approximations

- **Laplace Approximation**:
  We approximate the posterior with $\tilde{f}(\boldsymbol{\theta}|D) \approx \exp\left[-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^T \Lambda (\boldsymbol{\theta}-\boldsymbol{\mu})\right]$ by

$$\boldsymbol{\mu} := \boldsymbol{\theta}^* = \arg_{\boldsymbol{\theta}} \max \tilde{f}(\boldsymbol{\theta}|D) = \arg_{\boldsymbol{\theta}} \max \ell(D;\boldsymbol{\theta}) = \arg_{\boldsymbol{\theta}} \min -\ell(D;\boldsymbol{\theta}),$$

$$\Lambda := -\frac{\partial \log \tilde{f}(\boldsymbol{\mu}|D)}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^T}.$$

Motivation: Notes on the approximations

- **Laplace Approximation**:
  We approximate the posterior with $\tilde{f}(\boldsymbol{\theta}|D) \approx \exp\left[-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^T \Lambda (\boldsymbol{\theta}-\boldsymbol{\mu})\right]$ by

  $$\boldsymbol{\mu} := \boldsymbol{\theta}^* = \arg_{\boldsymbol{\theta}} \max \tilde{f}(\boldsymbol{\theta}|D) = \arg_{\boldsymbol{\theta}} \max \ell(D;\boldsymbol{\theta}) = \arg_{\boldsymbol{\theta}} \min -\ell(D;\boldsymbol{\theta}),$$
  $$\Lambda := -\frac{\partial \log \tilde{f}(\boldsymbol{\mu}|D)}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^T}.$$

- **Metropolis**:
  Define the walk $\boldsymbol{\theta}_{prop} \leftarrow \boldsymbol{\theta}_n + \lambda_n \boldsymbol{U}_n$, where $\boldsymbol{U}_n \sim Uniform([-1,1]^d)$. We accept $\boldsymbol{\theta}_{n+1} \leftarrow \boldsymbol{\theta}_{prop}$ with probability p if $\min(p,1) = p$ or else repeat, where $p := \exp\left[\ell(D;\boldsymbol{\theta}_{prop}) - \ell(D;\boldsymbol{\theta}_n)\right]$. Note that by writing it like this the computation is more efficient.

Motivation: Notes on the approximations

- **Laplace Approximation**:
  We approximate the posterior with $\tilde{f}(\boldsymbol{\theta}|D) \approx \exp\left[-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^T \Lambda(\boldsymbol{\theta}-\boldsymbol{\mu})\right]$ by
  $$\boldsymbol{\mu} := \boldsymbol{\theta}^* = \arg_{\boldsymbol{\theta}} \max \tilde{f}(\boldsymbol{\theta}|D) = \arg_{\boldsymbol{\theta}} \max \ell(D;\boldsymbol{\theta}) = \arg_{\boldsymbol{\theta}} \min -\ell(D;\boldsymbol{\theta}),$$
  $$\Lambda := -\frac{\partial \log \tilde{f}(\boldsymbol{\mu}|D)}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^T}.$$
- **Metropolis**:
  Define the walk $\boldsymbol{\theta}_{prop} \leftarrow \boldsymbol{\theta}_n + \lambda_n \boldsymbol{U}_n$, where $\boldsymbol{U}_n \sim Uniform([-1,1]^d)$. We accept $\boldsymbol{\theta}_{n+1} \leftarrow \boldsymbol{\theta}_{prop}$ with probability p if $\min(p,1) = p$ or else repeat, where $p := \exp\left[\ell(D;\boldsymbol{\theta}_{prop}) - \ell(D;\boldsymbol{\theta}_n)\right]$. Note that by writing it like this the computation is more efficient.
- **GVA**
  This a common variational technique in statistical learning. We maximise the evidence lower bound (ELBO) and obtain a Gaussian approximation of the posterior $\tilde{f}(\boldsymbol{\theta}|D) \approx \exp\left[-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^T e^{-2\boldsymbol{L}}(\boldsymbol{\theta}-\boldsymbol{\mu})\right]$, where $\boldsymbol{\mu}$ and $\boldsymbol{L}$ are obtained via optimisation of the ELBO.

Data

We take football results from the Spanish football league (La Liga) from season 1970/1971 to 2017/2018 from Kaggle [2]. We will use 2 different types of datasets, one with 20 teams and one with 4 teams. We will retreat analysing 4 teams for demonstration purposes and due to computational constraints.

1. Our 4-team-dataset contains 533 match scores, from 1970 to 2018: Athletico Madrid, Barcelona, Real Madrid and Valencia.
2. Our 20-team-dataset contains 1601 match scores, from 2010 to 2015: All the teams in 2017/2018 season.

Data

We take football results from the Spanish football league (La Liga) from season 1970/1971 to 2017/2018 from Kaggle [2]. We will use 2 different types of datasets, one with 20 teams and one with 4 teams. We will retreat analysing 4 teams for demonstration purposes and due to computational constraints.

1. Our 4-team-dataset contains 533 match scores, from 1970 to 2018: Athletico Madrid, Barcelona, Real Madrid and Valencia.
2. Our 20-team-dataset contains 1601 match scores, from 2010 to 2015: All the teams in 2017/2018 season.

Train-test split:

- 90% train and 10% test.

In practice, we should split the training set for parameter tuning e.g. K-fold cross validation, leave-1-out validation etc...

Model 1: Logistic, setup

Model 1: Logistic

Let the set we use for training be D, where $|D| = G$, say, denotes the cardinality of the set. Let:

1. Let 1 denote a win and 0 denote otherwise.
2. Let $r_{ij} \in \{0, 1\}$ be the result of team i vs team j when team i is at home.
3. Let Δ be the home advantage, α_i be the offensive index and β_i be the defensive index of team i.

Model 1: Logistic, setup

Model 1: Logistic
Let the set we use for training be D, where $|D| = G$, say, denotes the cardinality of the set. Let:

1. Let 1 denote a win and 0 denote otherwise.
2. Let $r_{ij} \in \{0, 1\}$ be the result of team i vs team j when team i is at home.
3. Let Δ be the home advantage, α_i be the offensive index and β_i be the defensive index of team i.

Our likelihood model is thus

$$R_{ij}|\theta \sim Bernoulli(p_{ij}),$$

where $logit(p_{ij}) = \Delta + \alpha_i + \beta_j$. In particular, we used the approximation $-\log(1 + e^{-t}) \approx -\max(0, -t)$ when we approximated the ELBO value as it is more stable.

Model 1: Logistic, Laplace and Metropolis results

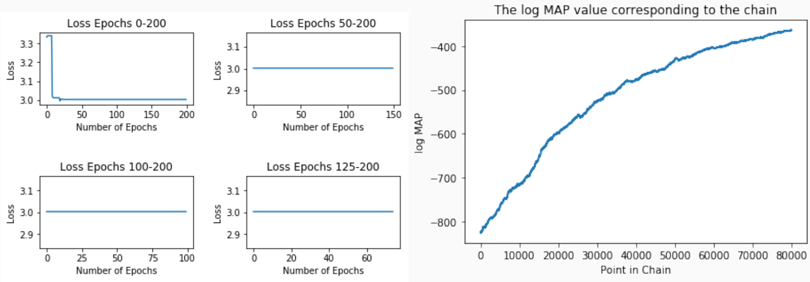

Figure 1: (Left) Laplace Approximation loss $-\log \tilde{f}(\boldsymbol{\theta}|D)$ during SGD training. (Right) The evolution of the $\log \tilde{f}(\boldsymbol{\theta}|D)$ as we increase the number of samples.

Comments:

1. The loss converges to about 3 for Laplace. For Metropolis, after 80,000 samples it is still converging to 0 slowly.
2. Difficult to tell what the burnin period is.
3. The Laplace approximation is very fast (10 minutes), whereas the Metropolis algorithms takes hours to compute.

Model 1: Logistic, Metropolis results

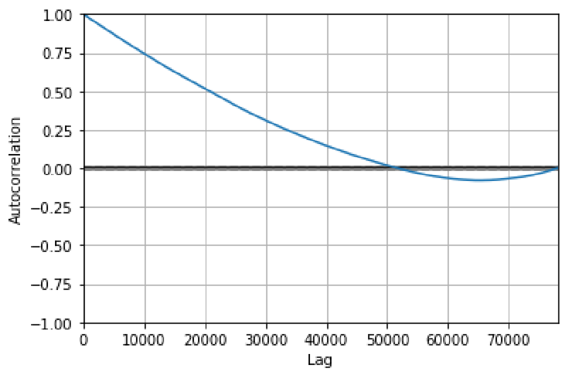

Figure 2: ACF of Markov chain.

Comments:

1. We can see that the auto-correlation falls to 0 after 50,000 samples.
2. Suggests that we should predict using the end points 50,000 - 80,000.

Model 1: Logistic, GVA

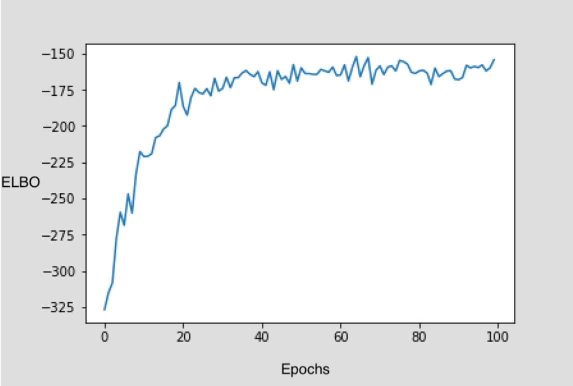

Figure 3: ELBO of Logistic model.

Comments:

1. The evidence lower bound (ELBO) seems to converge to around -150.
2. The computation takes a relatively long time, but much faster than Metropolis.

Model 2: Poisson
We are inspired by both Davison (2011) and Baio et al. (2010). We first construct a log-linear random effects model:

$$\mu_{ij}^{\text{home}} = \exp\left(\Delta + \alpha_i - \beta_j\right), \;\; \mu_{ij}^{\text{away}} = \exp\left(\alpha_j - \beta_i\right),$$

where $\alpha_i, \beta_i, \Delta$ correspond to our prior parameters of **attacking capability** of team i, **defensive capability** of team i and the **home advantage** (fixed for all teams). Our model is

$$Y^p_{h(i),a(i)} | \theta \sim Poisson(\mu^p_{h(i),a(i)}),$$

where p denotes the state in $\{\text{home}, \text{away}\}$, h, a are functions mapping the index i to its corresponding team number, which we set ourselves.

Model 2: Poisson, Laplace and Metropolis results

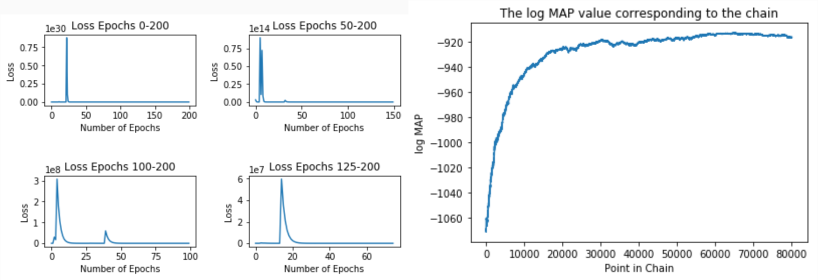

Figure 4: (Left) Laplace Approximation loss $-\log \tilde{f}(\boldsymbol{\theta}|D)$ during SGD training. (Right) The evolution of the $\log \tilde{f}(\boldsymbol{\theta}|D)$ as we increase the number of samples.

Comments:

1. The loss converges for both models. The loss for the Laplace approximation seems to converge to 0.
2. Again, we need to look at the ACF to see what the burnin period for the Markov chain algorithm is.

Model 2: Poisson, Metropolis results

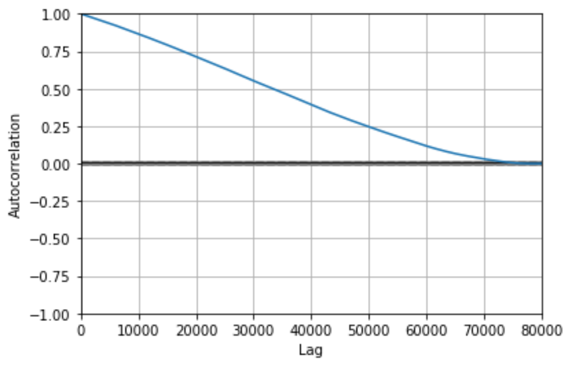

Figure 5: ACF of Markov chain.

Comments:

1. This time the ACF converges to 0 even slower, so we will just take 50,000 - 80,000 again for prediction.

Model 2: Modified Poisson, GVA

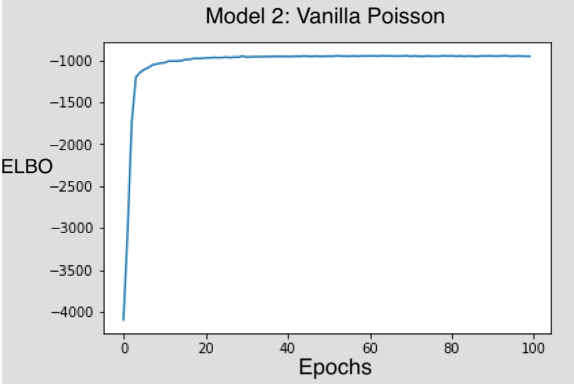

Figure 6: ELBO of Logistic model.

Comments:

1. The ELBO rapidly converges to around -980.
2. The ELBO is bounded above theoretically.

Model 3: Modified Poisson
We now change our link function to the soft-ReLU:

$$\mu_{ij}^{\text{home}} = \log\left(1 + e^{\Delta + \alpha_i - \beta_j}\right), \;\; \mu_{ij}^{\text{away}} = \log\left(1 + e^{\alpha_j - \beta_i}\right),$$

and so our model becomes

$$Y^p_{h(i),a(i)} | \theta \sim Poisson(\mu^p_{h(i),a(i)}),$$

where p denotes the state in $\{\text{home, away}\}$, h, a are functions mapping the index i to its corresponding team number, which we set ourselves.

Model 3: Modified Poisson, Laplace and Metropolis results

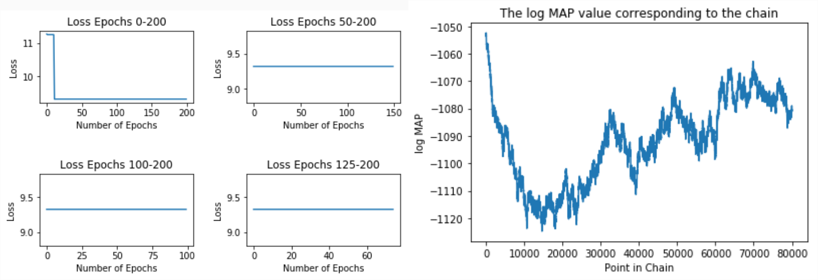

Figure 7: (Left) Laplace Approximation loss $-\log \tilde{f}(\boldsymbol{\theta}|D)$ during SGD training. (Right) The evolution of the $\log \tilde{f}(\boldsymbol{\theta}|D)$ as we increase the number of samples.

Comments:

1. The loss falls and converges for the Laplace approximation.
2. Burnin period of the Metropolis algorithm appears to be large again.

Model 3: Modified Poisson, Metropolis results

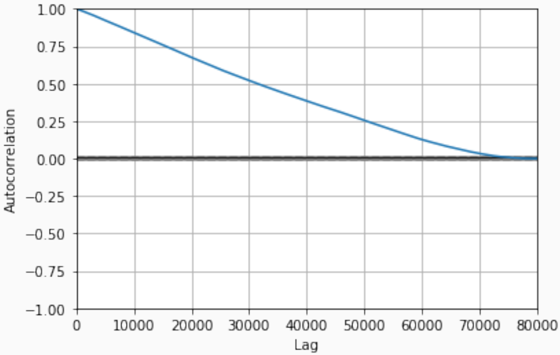

Figure 8: ACF of Markov chain.

Comments:

1. This time the ACF converges to 0 slow as well, and so take 50,000 as the cutting point again.

Model 3: Modified Poisson, GVA

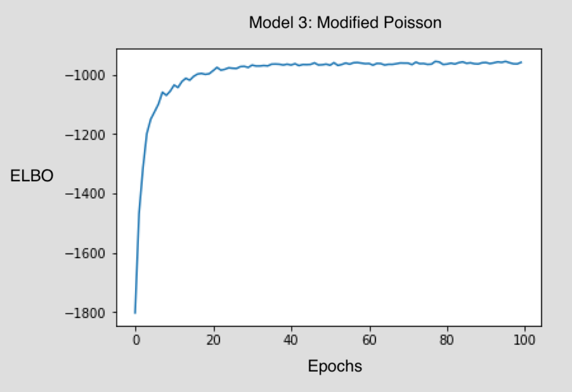

Figure 9: ELBO of modified Poisson model.

Comments:

1. The ELBO converges to around -980 again.
2. Convergence is less rapid than the vanilla Poisson model.

Test Results

For variational and optimisation methods, once we have approximated the posterior distribution, we will calculate the expected scores via:

$$E(Y_k|D) = \int_{\Omega_\theta} \int_{\Omega_{Y_k}} y_k f(y_k|\boldsymbol{\theta}) f(\boldsymbol{\theta}|D) dy_k d\boldsymbol{\theta}.$$

To do this, we will use sampling again. Note that for Metropolis, we have already obtained samples from the posterior distribution.

Test Results

For variational and optimisation methods, once we have approximated the posterior distribution, we will calculate the expected scores via:

$$E(Y_k|D) = \int_{\Omega_\theta} \int_{\Omega_{Y_k}} y_k f(y_k|\boldsymbol{\theta}) f(\boldsymbol{\theta}|D) dy_k d\boldsymbol{\theta}.$$

To do this, we will use sampling again. Note that for Metropolis, we have already obtained samples from the posterior distribution.
Algorithm:

1. Take n_1 samples of $\boldsymbol{\theta}$ from the posterior distribution.
2. For each $\boldsymbol{\theta}$, take n_2 samples of y_k from the likelihood distribution.
3. Take the sample mean $\frac{1}{n_1 n_2} \sum_{i=1}^{n_1 n_2} y_k$, which by the Ergodic Central limit theorem or LLN converges to the $E(Y_k|D)$.

Test Results: Model 1, Logistic

To quantify our findings for the Logistic model, where we predicted the probabilities of the home team winning, we use the empirical Cross-entropy loss:

$$H(p,q) = -\sum_{x \in \mathcal{T}} p(x) \log q(x) + (1 - p(x)) \log (1 - q(x)),$$

where x is a outcome, p is the empirical probability of a win, q is the predicted probability and $\mathcal{T}$ is the test set.

Test Results: Model 1, Logistic

To quantify our findings for the Logistic model, where we predicted the probabilities of the home team winning, we use the empirical Cross-entropy loss:

$$H(p,q) = -\sum_{x \in \mathcal{T}} p(x)\log q(x) + (1-p(x))\log(1-q(x)),$$

where x is a outcome, p is the empirical probability of a win, q is the predicted probability and $\mathcal{T}$ is the test set. We aim to minimise this. Results:

1. Laplace: 17.89
2. Metropolis: 26.89
3. GVA: 18.87

The Laplace approximation gives the better result, but the GVA is not far behind.

For the Log linear Poisson and soft ReLU Poisson, because we predicted the scores of matches the most natural measure is the mean squared error

$$MSE = \sum_{i=1}^{\mathcal{T}} ||(\hat{y}_i^{home} - y_i^{home}, \hat{y}_i^{away} - y_i^{away})^T||^2,$$

where $\mathcal{T}$ is the test set, $\hat{y}_i^p$ are the predicted scores, y_i^p are the observed scores and $|| \cdot ||$ is the Euclidean norm.

Test Results: Models 2 and 3

For the Log linear Poisson and soft ReLU Poisson, because we predicted the scores of matches the most natural measure is the mean squared error

$$MSE = \sum_{i=1}^{\mathcal{T}} ||(\hat{y}_i^{home} - y_i^{home}, \hat{y}_i^{away} - y_i^{away})^T||^2,$$

where $\mathcal{T}$ is the test set, $\hat{y}_i^p$ are the predicted scores, y_i^p are the observed scores and $||\cdot||$ is the Euclidean norm. Results:

Approximation	Model 2: Log Linear	Model 3: Soft ReLU
Laplace	91.19	574.65
Metropolis	104.64	165.39
GVA	90.29	878.98

Conclusion

Modelling

- Computational complexity posed as a serious issue for all our models. Could exploit efficient gradient computing platforms such as PyTorch or TensorFlow.
- We could take this further by modelling the parameters as time-dependent, and thus we enter the domain of Time Series.
- We could also introduce hyperparameters to each of the parameters and thus have hierarchical models.

Conclusion

Modelling

- Computational complexity posed as a serious issue for all our models. Could exploit efficient gradient computing platforms such as PyTorch or TensorFlow.
- We could take this further by modelling the parameters as time-dependent, and thus we enter the domain of Time Series.
- We could also introduce hyperparameters to each of the parameters and thus have hierarchical models.

Approximations

- The Laplace approximation works well for all 3 models.
- The Metropolis algorithm universally is consistent but is computationally burdensome. We should explore packages such as NUTS, PyMC3 and STAN.
- The GVA works well too, and gives the best result for model 2, but does not work so well for model 3.

Conclusion: So, which model was the best?

	Offensive Capability α_i	Defensive Capability β_i
Athletico Madrid	-0.0237	-0.1794
Barcelona	**0.3185**	-0.0302
Real Madrid	0.1647	**-0.0170**
Valencia	-0.0688	-0.1640
$\Delta = 0.3139$		

Table 1: Capabilities/parameters estimated via the GVA of model 2.

Away | Home	Athletico Madrid	Barcelona	Real Madrid	Valencia
Athletico Madrid	-	2.16:1.03	2.00:1.02	1.51:1.19
Barcelona	1.41:1.58	-	**1.39:1.32**	1.28:1.54
Real Madrid	1.38:1.47	**1.79:1.24**	-	1.26:1.44
Valencia	1.62:1.12	2.11:0.94	1.95:0.93	-

Table 2: Expected Scores via the GVA of model 2. Scores given by home:away.

Conclusion: So, which model was the best?

Poisson regression with canonical log link function, approximated by the GVA!

- Relatively logical and realistic results.
- Simplicity of the model: uses the canonical link.
- GVA: Although slightly slower than Laplace, it gives a lower MSE
- Logistic: Very simple model and thus difficult to extract exact information form nust the probabilities. Also limited number of parameters and covariates involved in the model.

References

- McGowan, T., (2015). Google: Getting in the face of football's 3.5 billion fans. *CNN*, `https://edition.cnn.com/2015/02/27/football/roma-juventus-google-football/index.html`.
- Moya, R., (2017). Football Matches of Spanish League. *Kaggle*, `https://www.kaggle.com/ricardomoya/football-matches-of-spanish-league`.
- Dehaene G. (2018), *Bayesian Computation Course*, EPFL.
- Baio G., Blangiardo M. A. (2010), A Hierarchical model for Rugby prediction, *Journal of Applied Statistics, 37 (2), pp. 253 - 264.*

References

- Coyle P. (2017), A Hierarchical model for Rugby prediction, https://docs.pymc.io/notebooks/rugby_analytics.html.
- Wikipedia (2018), 2017–18 La Liga, https://en.wikipedia.org/wiki/2017âĂŞ18_La_Liga.
- Davison A.C. (2011), Statistical Models, *Cambridge University Press.*