Probabilistic Programming and PyMC3

Peadar Coyle*†

Abstract—In recent years sports analytics has gotten more and more popular. We propose a model for Rugby data - in particular to model the 2014 Six Nations tournament. We propose a Bayesian hierarchical model to estimate the characteristics that bring a team to lose or win a game, and predict the score of particular matches.

This is intended to be a brief introduction to Probabilistic Programming in Python and in particular the powerful library called PyMC3.

Index Terms—MCMC, monte carlo, Bayesian Statistics, Sports Analytics, PyMC3, Probabilistic Programming, Hierarchical models

1 INTRODUCTION

Probabilistic Programming or Bayesian Statistics [DoingBayes] is what some call a new paradigm. The aim of this paper is to introduce a Hierarchical model for Rugby Prediction, and also provide an introduction to PyMC3. Readers who are unfamiliar with Hierarchical models are advised to either read a more thorough exposition online or turn to the excellent textbook on multilevel modelling [Multilevel].

Since I am a rugby fan I decide to apply the results of the paper Bayesian Football to the Six Nations. Rugby union is a contact sport that consists of two teams of fifteen players. The objective is to obtain more points than the opposition through scoring tries or kicking goals over eighty minutes of playing time. Play is started with one team drop kicking the ball from the halfway line towards the opposition. The rugby ball can be moved up the field by either carrying it or kicking it. However, when passing the ball it can only be thrown laterally or backward. The opposition can stop players moving up the field by tackling them. Only players carrying the ball can be tackled and once a tackle is completed the opposition can compete for the ball. Play continues until a try is scored, the ball crosses the side line or dead-ball line, or an infringement occurs. After a team scores points, the non-scoring team restarts the game at the halfway with a drop kick towards the opposition. The team with the most points at the end wins the game.

Within the Bayesian framework, which naturally accommodates hierarchical models [DoingBayes], we use here the result proved in [Biao] that assuming two conditionally independent Poisson variables for the number of points scored, correlation is taken into account, since the observable variables are mixed at an upper level. Moreover, since we are employing a Bayesian framework, the prediction of the outcome of a new game under the model is provided by the posterior predictive distribution. This predictive distribution is approximated by a Monte Carlo method.

2 MODEL

My model is based upon the model proposed in [Biao] this is a Hierarchical model for inferring the strength of each rugby team in the Six Nations from the data we have about scoring intensity.

Let me introduce some data which we’ll need for the model.

data_csv = StringIO('''
home_team,away_team,home_score,away_score
Wales,Italy,23,15
France,England,26,24
Ireland,Scotland,28,6
Ireland,Wales,26,3
Scotland,England,0,20
France,Italy,30,10
Wales,France,27,6
Italy,Scotland,20,21
England,Ireland,13,10
Ireland,Italy,46,7
Scotland,France,17,19
England,Wales,29,18
Italy,England,11,52
Wales,Scotland,51,3
France,Ireland,20,22''')

One of the strengths of probabilistic programming is the ability to infer latent parameters. These are parameters that can’t be measured directly. Our latent parameter is the strength of each rugby team in the Six Nations from the data we have about scoring intensity.

We model these parameters according to a formulation that

\begin{align*}
  y_{gj} &\sim \text{Po}(\theta_{gj}) \\
  \theta_{gj} &\sim \text{Poisson}(\theta_{g})
\end{align*}

where the theta parameters represent the scoring intensity in the g-th game for the team playing at home (j=1) and away (j=2), respectively.

The vector of observed counts \( y = (y_{g1}, y_{g2}) \) is modelled as independent Poisson: \( y_{gj} \sim \text{Poisson}(\theta_{gj}) \) where the theta parameters represent the scoring intensity in the g-th game for the team playing at home (j=1) and away (j=2), respectively.

We model these parameters according to a formulation that has been used widely in the statistical literature, assuming a log-linear random effect model

\begin{align*}
  \log \theta_{g1} &= \text{home} + \text{att}_h(g) + \text{def}_a(g) \\
  \log \theta_{g2} &= \text{att}_a(g) + \text{def}_h(g)
\end{align*}

The parameter home represents the advantage for the team hosting the game and we assume...
that this effect is constant for all the teams and throughout the season. The scoring intensity is determined jointly by the attack and defense ability of the two teams involved, represented by the parameters $\text{att}$ and $\text{def}$, respectively. Conversely, for each $t = 1, \ldots, T$, the team-specific effects are modelled as exchangeable from a common distribution $\text{att}_t = \text{Normal}(\mu_{\text{att}}, \tau_{\text{att}})$ and $\text{def}_t = \text{Normal}(\mu_{\text{def}}, \tau_{\text{def}})$

Team strength is decomposed into attacking and defending strength components. A negative defense parameter will sap the mojo from the opposing team’s attacking parameter. I made two tweaks to the model. It didn’t make sense to me to have a $\mu_{\text{att}}$ when we’re enforcing the sum-to-zero constraint by subtracting the mean anyway. Also because of the sum-to-zero constraint, it seemed to me that we needed an intercept term in the log-linear model, capturing the average points scored per game by the away team. This we model with the following hyperprior. $\text{intercept} = \text{Normal}(0, 0.001)$

The hyper-priors on the attack and defense parameters are also flat $\mu_{\text{att}} = \text{Normal}(0, 0.001)$, $\mu_{\text{def}} = \text{Normal}(0, 0.001)$, $\tau_{\text{att}} = \Gamma(0.1, 0.1)$ and $\tau_{\text{def}} = \Gamma(0.1, 0.1)$

3 BUILDING AND EXECUTING THE MODEL

You can see the full code at [Peadar] but the important thing to note that in [PyMC3] the model is all contained in a context manager. I specified the model and the likelihood function. Fundamentally the Bayesian approach is about calculating posterior distributions. A conventional way to do this is to use a Monte Carlo sampler of which there are many see [DoingBayes]. I chose the No U Turn Sampler [NUTS] which is a modern sampler for this problem, and we used the Maximum A Posteriori algorithm to find the starting point for that sampler. The Maximum A Posteriori algorithm is a modern optimization approach to finding the starting point. Since convergence of samplers is strongly affected by which starting point is chosen. It is beyong the scope of this article to go into the technicalities but I recommend the following references as a starting point [NUTS], [DoingBayes] and [PyMC3]... and the references included in those articles.

4 RESULTS

We can use the model above to help us estimate the different distributions of attacking strength and defensive strength. These are probabilistic estimates and help us better understand the uncertainty in sports analytics.

Forest plot of the results

4 is an example of the type of figures that can be generated, which in this example is a forest plot of credible intervals (see [Biao], and [DoingBayes] for explanations on how to interpret credible intervals) The estimated ranking of teams is Wales for 1, France for 2, Ireland for 3, Scotland for 4, Italy for 5 and England for 6.

I have built a non-trivial model or generative story for exploring rugby data, I expect that these models can be easily adopted to other sports such as soccer or American Football. PyMC3 despite being at the time of writing in beta is a useful framework for building Probabilistic Programming models. I was able to show how to use modern MCMC (Markov Chain Monte Carlo) samplers to approximate a likelihood function (generally one which would be extremely difficult to calculate without numerical methods) and from this infer latent parameters - that is parameters that are not easy to measure directly. In this case it is team strength but there are numerous other applications such as Stochastic Volatility in Finance [PyMC3]. Also we were able to illustrate how uncertainty estimates such as ‘credibility intervals’ come out ‘for free’ from models such as this. I hope that this example and the references inspire you to build your own models and please submit these models to the documentation.

REFERENCES

[PyMC3] PyMC3 Development Team PyMC3, https://pymc-devs.github.io/pymc3/
[Biao] Gianluca Baio and Marta Blangiardo Bayesian hierarchical model for the prediction of football results, Journal of Applied Statistics, Volume 37, Issue 2, 2010
[DoingBayes] John K. Kruschke,*Doing Bayesian Data Analysis*, Academic Press / Elsevier, 2014
[Peadar] Peadar Coyle Github https://github.com/springcoil/TutorialPyMCRugby*, 2015
[NUTS] Hoffman, Matthew D.; Gelman, Andrew The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo, arXiv:1111.4246, 11/2011
[Multilevel] Hill, Jennifer; Gelman, Andrew Data Analysis Using Regression and Multilevel/Hierarchical Models, Cambridge University Press, 2007