Senior Project

A Bayesian Approach to find Optimal Strategies in Twenty20 Cricket through Monte Carlo Simulations

Supervisors: Amin Hussain & Farooq Naseer

Farwa Ismail 18020109

Asad Hussain 18100013

Table of Contents

Introduction

Twenty20 Internationals are being played since 2004 and they have revolutionised the way in which cricket is played. The shorter format of the game is extremely fast paced with every agent trying to maximize its goal – the batsmen wanting to score maximum runs, bowlers wanting to get every player out before the end of the innings and fielders wanting to prevent every run from being scored.

Different strategies come at play when a team goes down to play its innings. One of the most common dilemmas facing the team is whether to play a fast, hard-hitting game and get out before the twenty overs or to play a smooth and long innings by playing all twenty overs. Which strategy helps you score more runs?

In this paper, we try to answer this question by simulating matches where every player follows their personal probability distribution which is reflective of their past performance and 'playing style'. The total score of the game would tell us what kind of strategy given every player's capabilities is best suited for that innings. Every delivery is an independent generalized Bernoulli trial and the probability distributions are obtained using Bayesian inference. In order to assess if the total score is optimal, only second innings are taken into consideration since we know what the target score to achieve is. The simulation of the matches gives us individual playing strategies, that are, the run-rates at which the batsmen should play in order to maximize the winning probability of the game. There are around 500 T20 internationals played between 2005 and 2017 included in the data with ball-by-ball data for every match.

Literature Review

Considerable amount of analytical work has been in the field of cricket which dates back to 1945 when simple geometric distributions were fitted to the number of runs scored in

test cricket (Elderton & Wood). Those were argued by Kimber and Hansford (1993) since ducks and dots were difficult to fit in a geometric distribution. Bracewell and Ruggiero (2009) proposed to use beta distribution for ducks and dots and geometric distribution for others.

Duckworth-Lewis method is widely known to predict winners in weather interrupted games. It was first used in Malaysia in 1997 (BBC Sport , 2007). Dyte (1998) simulated test matches by only using career batting and bowling averages as inputs, Clarke and Norman (2003), Swartz, Gill, Beaudoin and deSilva (2006), and Norman and Clarke (2010) suggested decision making adjustments to the game, and Bailey and Clarke (2006) in one-day cricket and Scarf, Shi and Akhtar (2011) in test cricket made run scoring models. Winning and Score Predictor (WASP) is another tool that is being used since 2012. The model of Cricviz is also based on it. It is a tool that predicts scores and results of limited format games (Wikipedia).

More relevant work in terms of playing strategies has been demonstrated by Clarke (1988), Clarke and Norman (1999), Preston and Thomas (2000), Swartz, Gill and Muthukumarana (2009), Davis, Perera, and Swartz (2015), and Stevenson and Brewer (2017). Work of Davis, Perera and Swartz (2015) specifically uses Bayesian inference to estimate probability distributions of players, however, it uses bowler, number of overs consumed and number of wickets lost as parameters instead of required run-rate (explained ahead in Methodology) as it is in our case.

Data

Ball-by-ball data of 572 T20 internationals between June 2005 and July 2017 was obtained from cricsheet.org in the yaml format. It was converted to csv format using pandas and was eventually used in Python and Stata. Figure 1 shows the match

information provided by crcicsheet.org before every match and figure 2 shows the information provided for a particular delivery.

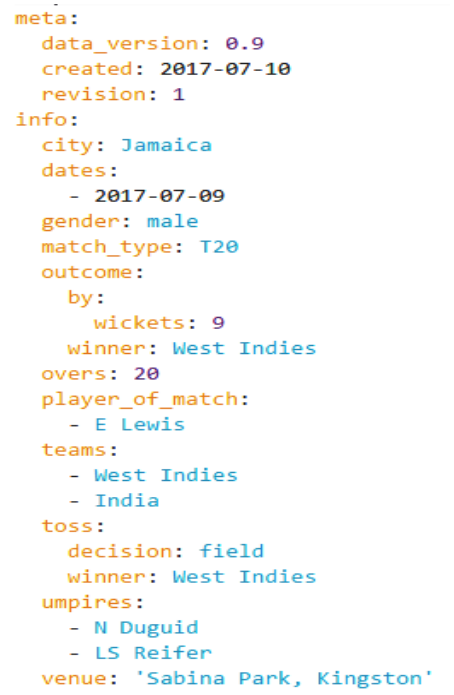

Figure 1

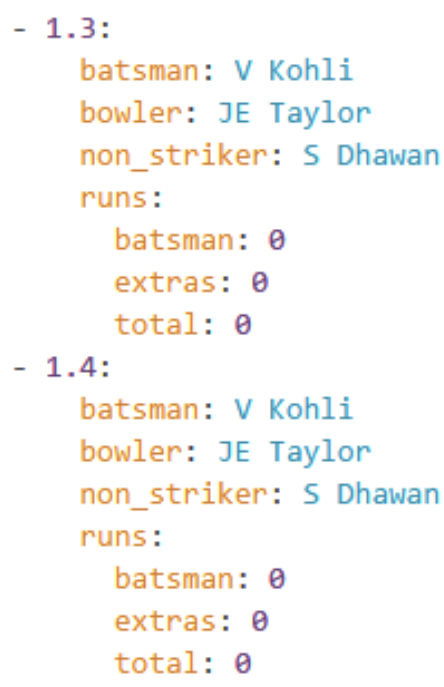

Figure 2

Player information was scrapped from espncricinfo.com. Both datasets were merged to make three main tables – all players, all matches and all deliveries. Following are the list of variables created for all players: PlayerId, PlayerName, Full name, Team, Batting style, Bowling style, Fielding position, Current age, Height, Playing role, Born, Also known as, Died, Education, Major teams, Nickname, Other, Relation, In a nutshell, TotalRuns, and MatchesPlayed. Figure 3 below provides a snapshot of the table.

	PlayerId	PlayerName	Full name	Team	Batting style	Bowling style	Fielding position	Current age	Height	Playing role	...	Also known as
0	0	Izatullah Dawlatzai	Izatullah Dawlatzai	Afghanistan	Right-hand bat	Right-arm Fast	NaN	26 years 274 days	NaN	Bowler	...	Izatullah Khan
1	1	Shoaib Malik	Shoaib Malik	Pakistan	Right-hand bat	Right-arm Offspin	NaN	36 years 7 days	NaN	Middle-order batsman	...	NaN

Figure 3

List of variables created for all matches: File, Team1, Team2, Venue, City, Dates, TossWinner, TossDecision, MatchType, Winner, ByWickets, ByRuns, Overs, PlayerOfTheMatch, Umpires, Gender, BattingFirst, BattingSecond, MatchId, and Target. Figure 4 and 5 below present snapshots of the table.

	File	Team1	Team2	Venue	City	Dates	TossWinner	TossDecision	MatchType	Winner	ByWickets	ByRuns
0	T20/1001349.yaml	Australia	Sri Lanka	Melbourne Cricket Ground	NaN	2017-02-17	Sri Lanka	field	T20	Sri Lanka	5	0
1	T20/1001351.yaml	Australia	Sri Lanka	Simonds Stadium, South Geelong	Victoria	2017-02-19	Sri Lanka	field	T20	Sri Lanka	2	0

Figure 4

MatchType	Winner	ByWickets	ByRuns	Overs	PlayerOfTheMatch	Umpires	Gender	BattingFirst	BattingSecond	MatchId	Target
T20	Sri Lanka	5	0	20	DAS Gunaratne	['MD Martell', 'P Wilson']	male	Australia	Sri Lanka	0	168
T20	Sri Lanka	2	0	20	DAS Gunaratne	['SD Fry', 'SJ Nogajski']	male	Australia	Sri Lanka	1	173

Figure 5

List of variables in all deliveries: Innings, Delivery, Batsman, Bowler, Runs, NonStriker, Extras, Wicket, Fielder, PlayerOut, ExtraType, BatsmanRuns, MatchId, InningsId, TotalScore, BatsmanScore, NonStrikerScore, BattingTeam, WicketsDown, Target, RequiredRR, CurrentRR, LegalDeliveries, Bowling style, Legal, BowlingTeam, BatsmanBalls, Winner, BattingTeamWon, and RRR. Figure 6 and 7 below present snapshots of the table.

MatchId	InningsId	Innings	Delivery	Batsman	Bowler	Runs	NonStriker	Extras	Wicket	Fielder	PlayerOut	ExtraType	BatsmanRuns	TotalScore
0	0	1	0.1	AJ Finch	SL Malinga	0	M Klinger	0	NaN	NaN	NaN	NaN	0	0
0	0	1	0.2	AJ Finch	SL Malinga	0	M Klinger	0	NaN	NaN	NaN	NaN	0	0
0	0	1	0.3	AJ Finch	SL Malinga	1	M Klinger	0	NaN	NaN	NaN	NaN	1	1
0	0	1	0.4	M Klinger	SL Malinga	2	AJ Finch	0	NaN	NaN	NaN	NaN	2	3
0	0	1	0.5	M Klinger	SL Malinga	0	AJ Finch	0	NaN	NaN	NaN	NaN	0	3
0	0	1	0.6	M Klinger	SL Malinga	3	AJ Finch	0	NaN	NaN	NaN	NaN	3	6

Figure 6

PlayerOut	ExtraType	BatsmanRuns	TotalScore	BatsmanScore	NonStrikerScore	BattingTeam	WicketsDown	Target
NaN	NaN	0	0	0	0	Australia	0.0	169.0
NaN	NaN	0	0	0	0	Australia	0.0	169.0
NaN	NaN	1	1	1	0	Australia	0.0	169.0
NaN	NaN	2	3	2	1	Australia	0.0	169.0
NaN	NaN	0	3	2	1	Australia	0.0	169.0
NaN	NaN	3	6	5	1	Australia	0.0	169.0

Figure 7

Model

The idea is to create a model which gives you the best 'strategy' that maximizes winning probability of the team given every player's individual probability distributions. We take required run-rate (RRR) as the situation parameter since we can model a player's playing style depending on how they respond to different RRRs. In order to use RRRs, only second innings of the matches are taken into consideration.

Several other situation parameters can be used instead of RRR, such as number of overs left and number of wickets lost. One can also use multiple situation parameters at a time. Davis, Perera and Swartz (2015) use multiple parameters to sort their players. We use RRR as our parameter also because it gives us a positive correlation with the probability of wicket (Figure 8). Probability of wicket being a negative thing, we get a perfect trade off which can be used to create an optimization problem. Intuitively, when the RRR of the match increases, players start taking more risks in order to score more runs, hence, they end up increasing their probability of wicket.

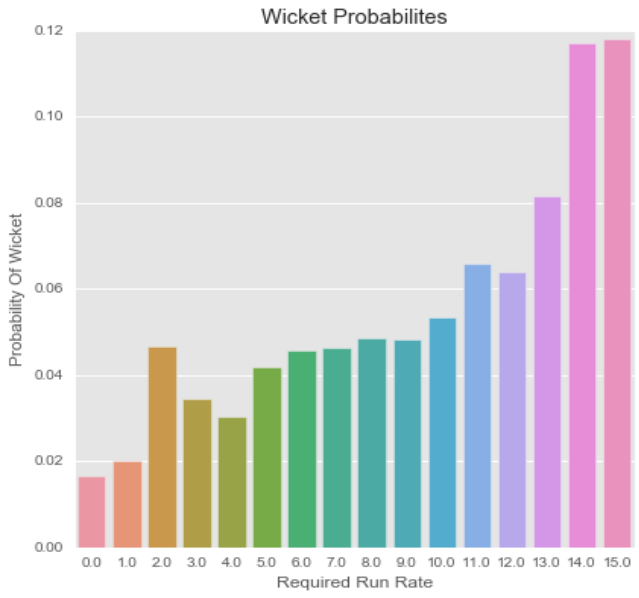

Figure 8

Every ball is an independent Bernoulli trial but since there are eight outcomes (dot, single, double, three, four, five, six, wicket) on every delivery, we are going to use generalized Bernoulli trials. From the data available, a new table is constructed which is sorted by player and by required run-rate (RRR) with outcome counts ${C^b}_{i,r}$. Here, b labels the batsman, i labels the outcome and r labels the RRR that the batsman is trying to play at. The RRRs are discretized by rounding down to the nearest integer. Figure 9 shows how the new table looks like. It shows the frequency distribution of delivery outcomes.

	batsman	rrr	dots_	ones_	twos_	threes_	fours_	fives_	sixes_	out
106	AB de Villiers	4.0	19.0	19.0	3.0	1.0	3.0	0.0	2.0	1.0
107	AB de Villiers	5.0	13.0	18.0	3.0	1.0	4.0	0.0	2.0	1.0
108	AB de Villiers	6.0	40.0	33.0	7.0	0.0	8.0	0.0	0.0	4.0
109	AB de Villiers	7.0	24.0	23.0	5.0	0.0	11.0	0.0	2.0	4.0
110	AB de Villiers	8.0	33.0	18.0	5.0	0.0	8.0	0.0	1.0	5.0
111	AB de Villiers	9.0	40.0	28.0	11.0	0.0	15.0	0.0	5.0	4.0
112	AB de Villiers	10.0	19.0	22.0	3.0	2.0	3.0	0.0	4.0	5.0

Figure 9

Using the tallies $C^b{}_{i,r}$ of every batsman, we form a probability distribution that shows the probability of scoring every outcome for a given RRR by simply normalizing the frequencies (figure 10). However, as we can see in figure 10 and otherwise, there are a lot of tallies that are zero for a given outcome. This is due to the small amount of T20 internationals that have been played and the fact that players who have played fewer innings haven't faced all or enough situations. The zero tallies are not realistic since it is senseless to say that Sarfraz Ahmed is never going to get out on a RRR of 5.0. Therefore, we use Bayesian inference with a 'smart' prior that gets rid of this problem. See Bayesian Inference topic ahead for more details.

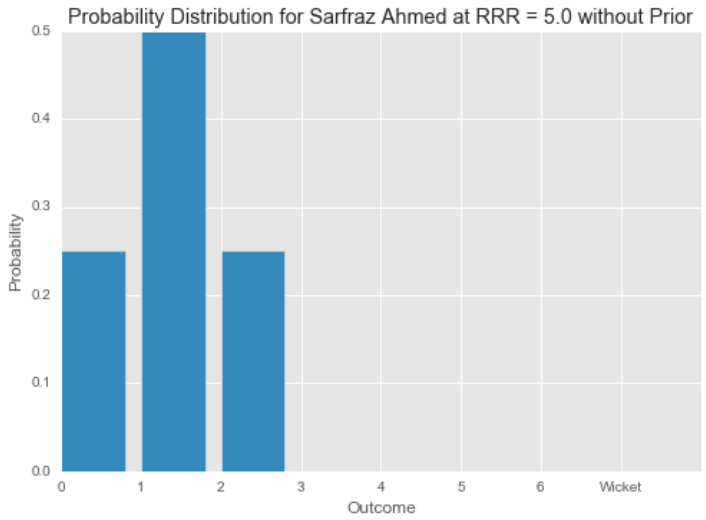

Figure 10

Once we get our updated probability distributions, we simulate every match a 1000 times. At this point, we set up a team according to the batting order and give them the run-rates[1] we want them to play at to compute the winning probability of the team at those run-rates. Every player can be given a different run-rate to play at. All combinations of

[1] From here on, 'run-rates' refers to the run-rate assigned to each batsman by the model at the beginning of the game. The RRR is simply just the required run-rate of the match at that particular delivery. These two are not necessarily the same.

the assigned run-rates would be run by the simulator with every combination being run a 1000 times. Since there are 36 possible run-rates and there are 11 players in a team, 36^{11} will be the number of combinations. This model is going to extremely tedious to run. Therefore, we converted the current model to a hybrid one where the first three players would be given a situation parameter and the rest of the players would be 'free agents' meaning that they would play according the RRR of the match at that time. The model would try all possible combination of run-rates and output the strategy that would give the highest winning probability. Before we present the results, we will talk about how the probability distributions were inferred.

Inferring the model parameters

The model we have constructed so far can be seen as a black box function $f(T, M, P^b{}_{i,r}, R)$ which outputs winning probabilities. Here, T, M, $P^b{}_{i,r}$ and R are defined in the following way:

- T is the Target for the 2nd innings which is to be simulated.
- $M = \{b1, b2, b3, \ldots b11\}$ where bi are strings denoting the player names, in the order in which they will come to bat.
- $P^b{}_{i,r}$ is the probability that a delivery faced by batsman b will result in an outcome $i$where $i \in \{0,1,2,3,4,5,6, Wicket\}$ given that the batsman chooses to play at the rate r. For a discussion on this refer to section (Model)
- $R(rrr, over, runs)$ is a function that encapsulates the batsman's strategy. The batsman will be given a strategy where he has to control his goal scoring rate $r$depending on the game situation. The dependence of a batsman's goal run rate, on the match situation is encapsulated by this function R where its inputs are

match situation variables such as required run rate, overs remaining and runs scored.

Given these inputs the model will perform monte carlo runs and output a probability of winning the match.

To connect this model to reality we need to supply our model with accurate estimates of ${P^b}_{i,r}$, from data. To do this we will use bayesian inference.

Bayesian Inference

In Bayesian inference one needs three ingredients:

- A Statistical model of a system with parameters we want to estimate: $M(\hat{\theta}_i)$where $\hat{\theta}_i$are the parameters of M
- Data from the system: C_l
- A prior estimate for the probability distributions of the parameters, so that $\hat{\theta}_i \sim D(\alpha_i)$such that $\hat{\theta}_i$are the parameters.

The PDF of $\hat{\theta}_i$are called the priors. We are essentially assuming that the parameters for our model are sampled from the distribution, $D(\alpha_i)$. We have set the $\alpha_i$ourselves in such a way that it best reflects our understanding of the ranges of $\hat{\theta}_i$, without having seen any data at all, as is the norm in Bayesian inference. It allows domain knowledge to enter into statistical inference in a very natural yet rigorous way. The α_i are called hyperparameters.

From these ingredients, Bayes rule will provide for us the updated probability distribution of $\hat{\theta}_i$, having seen the data. We can apply bayes rule by:

$$P(\hat{\theta}_i|c_l, \alpha_i) = P(c_l \mid \hat{\theta}_i)\, P(\hat{\theta}_i \mid \alpha_i)$$

This can be heuristically seen as:

Probability distribution of $\hat{\Theta}_i$ *having seen the data = (Probability that we see* c_l *assuming that* c_l *follows our model with parameters* $\hat{\Theta}_i$*) x (Probability distribution of* $\hat{\Theta}_i$ *having seen no data)*

$$\hat{\Theta}_i \,|c_l, \alpha_i \;\sim\; M(\,\hat{\Theta}_i) \;*\; D(\alpha_i)$$

Where $*$ defines pointwise multiplication between the underlying PDFs.

We will come back to this once we have to look at how we chose the prior we have chosen. For now, let us first formalize the model whose parameters we want to estimate.

The Delivery Model

Consider our particular model for the whole match, we have a black box function $f(T, M, {P^b}_{i,r}, R)$ which has 1 parameter (or matrix of parameters) that has to be fixed from data, namely,${P^b}_{i,r}$. We need to get estimates of these parameters given that we have seen data. To estimate these parameters we first note that our match function is constructed by simulating a lot of deliveries, the outcome of which are independent up to their correlation through match situation parameters.

We model every delivery as a sampling from a categorical distribution, also called a generalised Bernoulli trial, since it's like a Bernoulli trial, but instead of 2 outcomes, there are eight.

$$\hat{O}_{b,r} \;\sim\; \boldsymbol{Cat}\,(\,{P^b}_{i,r}\,)$$

So the outcome of a particular delivery $\hat{O}_{b,r}$(faced by batsman b, with a goal rate r) follows a categorical distribution with parameters ${P^b}_{i,r}$ such that:

$$Prob(\hat{O}_{b,r} \;=\; i) \;= {P^b}_{i,r}$$

Data for Bayesian inference

Now, what kind of data will allow us to infer about the parameters of the categorical distribution $\hat{O}_{b,r}$?

This will be simply be data that counts the frequency of different outcomes $C_{i,b,r}$ that occurred in the dataset. Here $C_{i,b,r}$ counts the number of times in the data set the following event occurs: batsman $b$is batting, the required run rate is r and the outcome of the delivery is i.

Since our data set is rich enough that we have access to ball by ball outcome events, we can extract $C_{i,b,r}$ from them. A standard count for b = 'AB de Villiers' is shown in figure 9. Here, we can see that $C_{0,b,4} = 19$and $C_{4,b,7} = 11$for b = 'AB de Villiers'.

The Prior

To choose the prior we first had to choose a class of prior distributions. Often Bayesian updating is a complex procedure when the underlying PDF of $M(\hat{\theta}_i)$ is not known which is very often the case when $M(\hat{\theta}_i)$ is slightly complex. But luckily for us, the PDF of our $M(\hat{\theta}_i)$ has a closed form expression as a multinomial distribution with parameters $\hat{\theta}_i$.

$$PDF[M(\hat{\theta}_i)] \sim Multinomial(\hat{\theta}_i)$$

Now, the nice thing about the multinomial is that there exists a conjugate prior to the Multinomial called the Dirichlet Distribution. What that simply means is that if

$$PDF[\hat{\theta}_i] \sim Dirichlet(\alpha_i)$$

$$Dirichlet(\alpha_i + c_i) = Multinomial(\hat{\theta}_i \mid c_i)\ Dirichlet(\alpha_i)$$

Hence, the posterior distribution after looking at observations follows:

$$PDF[\hat{\theta}_i \mid c_l, \alpha_i] = Dirichlet(\alpha_i + c_i)$$

The mean of the Posterior distribution will be our new estimate after looking at the data c_i.

Since there are not enough deliveries to get a large amount of observations for each given batsman and run rate, we decided to keep our prior estimates ${P_0}^b{}_{i,r}$ to be independent of batsmen, and follow the principle: *"Every batsman is an average batsman until proven otherwise"*. So in our prior estimate, we simply take the distribution of outcomes by run rate only. Our data (figure 11) gives us:

	rrr	dots_	ones_	twos_	threes_	fours_	fives_	sixes_	out	dels
0	0.0	213.0	204.0	55.0	7.0	174.0	0.0	119.0	15.0	787.0
1	1.0	310.0	284.0	73.0	8.0	103.0	0.0	40.0	37.0	855.0
2	2.0	510.0	411.0	96.0	10.0	148.0	1.0	62.0	61.0	1299.0
3	3.0	897.0	720.0	155.0	18.0	225.0	0.0	99.0	109.0	2223.0
4	4.0	1298.0	988.0	202.0	19.0	323.0	1.0	110.0	105.0	3046.0
5	5.0	2090.0	1654.0	326.0	23.0	533.0	2.0	142.0	195.0	4965.0
6	6.0	3461.0	2617.0	499.0	36.0	890.0	1.0	239.0	342.0	8085.0
7	7.0	4139.0	3017.0	611.0	57.0	952.0	4.0	272.0	424.0	9476.0
8	8.0	3873.0	3031.0	593.0	56.0	932.0	0.0	264.0	415.0	9164.0
9	9.0	3265.0	2791.0	516.0	31.0	765.0	1.0	257.0	399.0	8025.0
10	10.0	2347.0	2076.0	426.0	28.0	533.0	1.0	231.0	347.0	5989.0
11	11.0	1203.0	1220.0	249.0	13.0	259.0	0.0	129.0	210.0	3283.0

Figure 11

Which clearly gives us more data points to work with. We will use these to construct our prior, by simply taking their proportions in the dataset. For example, our prior estimate for the probability of a wicket at a required run rate of 7 would simply be:

$$ {P_0}^b{}_{Wicket,7} = \frac{Nwickets}{Ndeliveries} = \frac{424}{9476} \simeq 0.0447 $$

We can do it similarly for the rest.

However, to give the prior distribution we need the $\boldsymbol{\alpha}_{i,r}$, not the probability estimates. We will fix them by setting the mean of the Dirichlet distribution to out prior estimate. Where our ${P^b}_{i,r}$ are random variables following a dirichlet distribution.

$$E[\,{\hat{P}^b}_{i,r}] = \frac{\boldsymbol{\alpha}_{i,r}}{\sum_k \boldsymbol{\alpha}_{k,r}} = {P_0}^b{}_{i,r}$$

For a dirichlet distribution the sum $\sum_k \boldsymbol{\alpha}_{k,r} = N_{p,r}$which is also called the hypercount, which, as we will see measures the weightage one gives to the prior. Hence:

$$\boldsymbol{\alpha}^0{}_{i,r} = {P_0}^b{}_{i,r}\, N_{p,r}$$

Once updated by incoming data $C_{i,b,r}$, the new updated hyper-parameters will be:

$$\boldsymbol{\alpha}_{i,b,r} = \boldsymbol{\alpha}^0{}_{i,r} + C_{i,b,r}$$

So now ${\hat{P}^b}_{i,r} \sim Dirichlet(\boldsymbol{\alpha}^0{}_{i,r} + C_{i,b,r})$ which means the updated probability estimates will be:

$${P^b}_{i,r} = E[\,{\hat{P}^b}_{i,r}] = \frac{\boldsymbol{\alpha}_{i,b,r}}{\sum_k \alpha_{k,b,r}} = \frac{\boldsymbol{\alpha}^0{}_{i,r} + C_{i,b,r}}{\sum_k \alpha^0{}_{k,r} + C_{k,b,r}} = \frac{{P_0}^b{}_{i,r}\, N_p + C_{i,b,r}}{\sum_k {P_0}^b{}_{k,r}\, N_p + C_{k,b,r}} = \frac{{P_0}^b{}_{i,r}\, N_p + C_{i,b,r}}{N_p + {N^b}_r}$$

Where ${N^b}_r$ is the number of deliveries faced by batsman b with required run rate r.

By now, everything has been fixed, and the formula contains things that can be extracted directly from the data, with one exception being N_p. This is abstractly the amount of weight (measured in in data points) that the prior constitutes. By looking at the data and being wary of giving too much weight to our prior we have set $N_p = 10$.

Hence, we have reduced the process of updating every batsman's probability distribution to the following formula:

$${P^b}_{i,r} = \frac{10\,{P_0}^b{}_{i,r} + C_{i,b,r}}{10 + {N^b}_r}$$

Results

Coming back to the model, we ran the model with the following line up of the Indian team: S Dhawan, RG Sharma, V Kohli, Yuvraj Singh, MS Dhoni, SK Raina, RA Jadeja, R Ashwin, JJ Bumrah, B Kumar, A Nehra. Dhawan, Sharma and Kohli were free parameters and the rest of the team was supposed to play at RRR of the match (free agents). We chose 155 as the target score which is the average first innings score. The model ran 36^3 combinations and figure 12 shows the results.

S Dhawan	RG Sharma	V Kohli	Win Probability
9	12	3	87.4
10	12	3	84.9
12	12	3	82.1
13	12	3	81.4
14	12	3	80.8
11	12	3	79.6
6	12	3	79.2
10	12	8	77.9
9	12	8	77.8
8	12	3	76.0
7	12	3	75.6
3	12	3	73.9

Figure 12

The result shows that when the target is 155, if Dhawan plays at a run-rate of 9, Sharma plays at 12 and Kohli plays at 3, India will have an 87.4% chance of winning a game. These results obviously don't factor in any of the other factors, such as pitch condition, quality of opposition, fall of wickets and so on. However, most of these factors can be added to the model. Moreover, we saw that these run-rates that the model is suggesting are independent of each other. Given Kohli is supposed to play at a run-rate of 7, Dhawan should play at 8 or 9 and Sharma at 12 no matter what (figure 13). This shows that the

results are independent of other player's run-rate which intuitively doesn't make sense because players on field adjust to each other' run-rates. Hence, no non-trivial correlations between assigned batting styles were discovered.

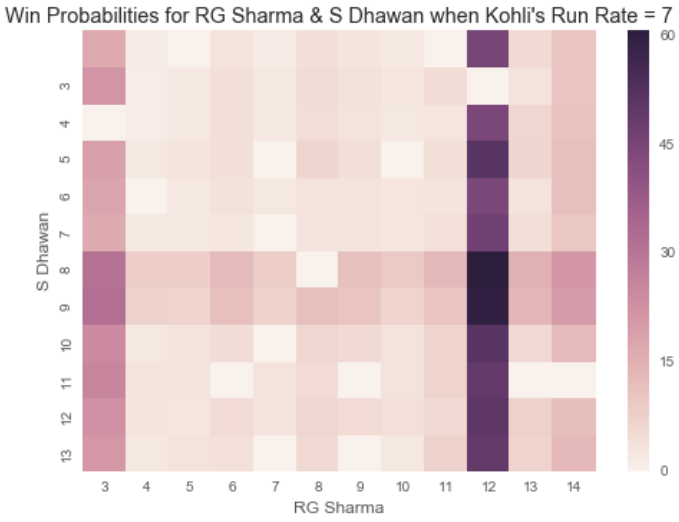

Figure 13

Train-Test

A randomized 80-20 train-test split was implemented on the Indian team. Only the training set was used to infer the probability distributions. Target scores were noted from the testing set which were used to check the accuracy of the model. The model gave a 70% accuracy which means that 70% of the 20-testing-set matches were predicted correctly by the simulation run using the 80-training-set.

Discussion

The model we have created is a good foray into the field of cricket analytics. The results are decently accurate. Practically, the model can not only be used to give assigned run-rates to the players but one can also perform player selection with a few adjustments. Additional free parameters could also be added for a more sophisticated model. Moreover,

we have run our simulations only on three players. Better computational machines would allow us to perform faster computations for all players.

On a side note, Bayesian inference applied to the data and delivery simulation was our own idea. Only later did we realize that these techniques are widely used in this field.

Further work on this model could improve the model to attain greater accuracy. Variables, such as quality of bowling and fielding are completely missing from the model. Although some of these variables are incorporated by Davis, Perera and Swartz (2015), they do not use run-rate as a proxy, hence, this is still an avenue for future work.

Works Cited

Bailey, M., & Clarke, S. (2006). Predicting the match outcome in one day international cricket matches while the match is in progress. *Journal of Science and Sports Medicine*, 480-487.

BBC Sport . (2007, January 1). Retrieved from BBC: http://news.bbc.co.uk/sport2/hi/cricket/6222943.stm

Bracewell, P. J., & Ruggiero, K. (2009). A parametric control chart for monitoring individual batting performances in cricket. *Journal of Quantitative Analysis in Sports*.

Clarke, S. R. (1988). Dynamic programming in one-day cricket-optimal scoring rates. *Journal of the Operational Research Society*, 331–337.

Clarke, S. R., & Norman, J. M. (1999). To run or not?: Some dynamic programming models in cricket. *Journal of the Operational Research Society*, 536–545.

Clarke, S. R., & Norman, J. M. (2003). Dynamic programming in cricket: Choosing a night. *Journal of the Operational Research Society*, 838–845.

Davis, J., H. Perera, & Swartz, T. B. (2015). A simulator for Twenty20 cricket. *Australian and New Zealand Journal of Statistics*, 55–71.

Dyte, D. (1998). Constructing a plausible test cricket simulation using available real world data. *Mathematics and Computers in Sport*, 153-159.

Elderton, i., & Wood, G. H. (1945). Cricket Scores and Some Skew Correlation Distributions: (An Arithmetical Study). *Journal of the Royal Statistical Society, 108*, 1-11. Retrieved from www.jstor.org/stable/2981192

Kimber, A. C., & Hansford, A. R. (1993). A statistical analysis of batting in cricket. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 443–455.

Norman, J. M., & Clarke, S. R. (2010). Optimal batting orders in cricket. *Journal of the Operational Research Society*, 980–986.

Preston, I., & Thomas, J. (2000). Batting strategy in limited overs cricket. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 95–106.

Preston, I., & Thomas, J. (2000). Batting strategy in limited overs cricket. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 95–106.

Scarf, P., Shi, X., & Akhtar, S. (2011). On the distribution of runs scored and batting strategy in test cricket. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 471-497.

Stevenson, O. G., & Brewer, B. J. (2017). Bayesian survival analysis of batsmen in Test cricket. *arXiv.org*.

Swartz, T. B., Gill, P. S., Beaudoin, D., & deSilva, B. M. (2006). Optimal batting orders in one-day cricket. *Computers and Operations Research*, 1939–1950.

Swartz, T., Gill, P., & Muthukumarana, S. (2009). Modelling and simulation for one-day cricket. *The Canadian Journal of Statistics*, 143-160.

Wikipedia. (n.d.). *WASP (cricket calculation tool)*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/WASP_(cricket_calculation_tool)