Final Draft

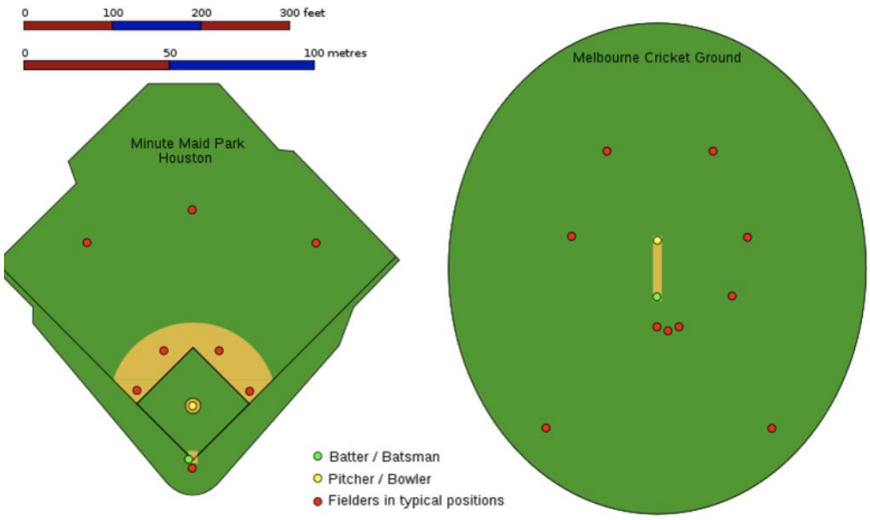
Pulkit Bhasin

Motivation

Cricket has always been an integral part of my childhood. It has united me with various people, and it helped me fit in after moving to India from the United States. I continued to explore the intricacies of the sport, and now I wish to harness the mechanisms of data science to find quantitative and objective answers to some subjective questions regarding the sport. These findings can potentially help a U-19 player tweak his game in order to maximize his chances of getting into the main team.

Overview

Cricket is a team sport in which players of a team have primarily two roles: batting and bowling (analogous to batsmen and pitchers in baseball). A batsman's task is to hit the ball and score runs, while the bowler's task is to prevent the batsman from scoring runs and picking up wickets (a wicket is analogous to three strikes in baseball). A boundary is scored by a batsman when a batsman hits the balls out of the ropes (analogous to a home run by a batter in baseball).



An important thing to note is that a player could both bat and bowl during his team's respective innings. The amount of runs scored and wickets taken can be considered as the fundamental statistics for a cricketer. However, there are more nuanced statistics as well, such as the following:

Runs: total number of runs scored by a player
Batting Average: runs scored per innings by a player

Balls Faced: the number of balls faced by a player
Batting Strike Rate: (number of runs scored per ball) * 100
100s: number of innings where a player scored more than 100 runs
50s: number of innings where a player scored more than 50 runs but less than 100 runs
4s: number of boundaries scored by a player
Wickets: number of wickets taken by a player
Bowling Average: runs conceded per wicket taken by a player
Bowling Strike Rate: (runs conceded per wicket taken by a player) * 100
Economy rate: runs conceded per over (an over constitutes of 6 deliveries) by a player
4W: number of innings where a player picked up 4 or more wickets
Captain: whether a player was the captain of their respective team or not

Just like every sport, a cricketer has to play on several lower platforms before they can play for their respective country. One such platform is the U-19 World Cup, where players below or equal to the age of 19 years represent their national side and compete with each other. Becoming a part of the U-19 team is a huge achievement, and several U-19 players go on to represent their countries on the highest level. A U-19 World Cup happens after every two years, and a player can participate in the tournament once or twice throughout his career. This project develops a machine learning model that determines the probability of an Indian U-19 batsman getting into the main team based on performance in the U-19 World Cup. This project analyzes which features described above play a more important role in determining whether an U-19 Indian batsman gets into the main team or not and whether a particular feature contributes positively or negatively to the chances.

Dataset

In order to perform my analysis, I needed a dataset containing the World Cup statistics of every U-19 Indian batsman over the last 20 years and whether they were selected for the main Indian cricket team or not. Generating a dataset was an extremely long task, as I was unable to find a readymade dataset online. The data I needed was available on a website called cricinfo.com, which is a subsidiary of the ESPN network. This website contained the statistics of each Indian batsman for each World Cup in separate tables. In order to obtain this information from the website for each year in a format that was useful to me, I built a web scraping function that takes an espncricinfo website url as an input, creates a new Pandas dataframe (Pandas is a data science library in Python and a dataframe is a structure that it uses to store data tables), extracts the data from the website link, and inputs this data into the newly created dataframe. In order to do this, I used a python web scraping library called BeautifulSoup.

Using this function, I obtained 10 different dataframes: one for each World Cup in the years 2000 to 2020. I then concatenated these dataframes into one, and I had a dataset. However, there was one problem. The dataframe I generated had all the features I needed except for two: Captain (whether the player captained the Indian side in that particular year) and Played for India (whether the player went on to play for the Indian team or not). In order to add these two

features to the dataset, I manually researched the features for each player and inputted them into the dataframe. This task was a bit tedious and long, but after completing it, I finally had a complete dataset containing World Cup statistics of 74 batsmen over the last 20 years.

	Name	Runs	Average	Balls Faced	Strike Rate	100s	50s	4s	Wickets Taken	Bowling Average	Economy	Bowling Strike Rate	4W	Played for India	Captain
0	A Ratra	25.0	25.00	38.0	65.78	0.0	0.0	1.0	0.0	41.00	3.91	84.0	0.0	False	0
2	M Kaif	170.0	34.00	253.0	67.19	0.0	1.0	12.0	8.0	19.37	2.88	40.2	0.0	True	1
4	Manish Sharma	257.0	42.83	391.0	65.72	0.0	3.0	27.0	0.0	41.00	3.91	84.0	0.0	False	0
5	NK Patel	133.0	66.50	136.0	97.79	0.0	0.0	8.0	0.0	41.00	3.91	84.0	0.0	False	0
6	RS Ricky	340.0	42.50	552.0	61.59	1.0	2.0	36.0	0.0	41.00	3.91	84.0	0.0	False	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
113	MK Lomror	133.0	33.25	132.0	100.75	0.0	0.0	7.0	7.0	18.85	3.75	30.1	0.0	False	0
114	RK Bhui	47.0	15.66	69.0	68.11	0.0	0.0	6.0	0.0	56.66	5.16	89.0	0.0	False	0
115	RR Batham	40.0	13.33	48.0	83.33	0.0	0.0	4.0	5.0	15.20	3.18	28.6	0.0	False	0
116	RR Pant	267.0	44.50	256.0	104.29	1.0	2.0	33.0	0.0	56.66	5.16	89.0	0.0	True	0
118	SN Khan	355.0	71.00	409.0	86.79	0.0	5.0	35.0	0.0	56.66	5.06	89.0	0.0	False	0

74 rows × 15 columns

Feature Engineering

Although I had a dataset, I had to tackle two issues with it before I could implement my model and analyze the statistics. The first issue was that the material was not in the appropriate format for it to be processed. Secondly, I had too many features, and I wanted to cut down on that.

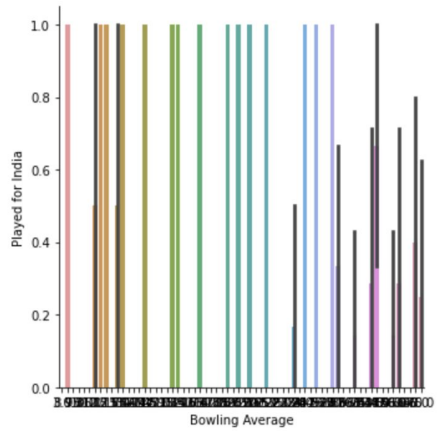
To tackle the first issue, I first had to carry out the following tasks: -

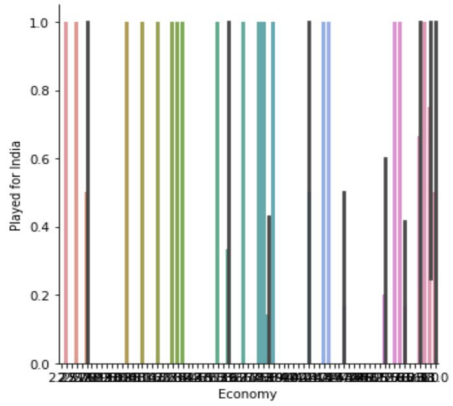
1) I had to drop the “Name” column

2) Some batsmen had never bowled a single over, and hence, their associated “Economy”, “Bowling Strike Rate”, “Bowling Average” values were blank. A machine learning model can not process blank values, and hence, I replaced these values with the maximum value in the “Economy” column, the maximum value in the “Bowling Strike Rate” column, and the maximum value in the “Bowling Average” column across all years, respectively. This is because the larger one’s economy, bowling strike rate, bowling average, the worse their bowling statistics are (as they conceded more runs), and a player who has never bowled a single over is inherently a bad bowler, and the statistics should recognize that.

3) Despite representing numeric/boolean values, each entry in the dataframe was represented a string (alphanumeric text), and hence, I converted each value in the dataframe in the Played for India into a boolean data type and all other values to a float data type and stored them as such.

Now, I had to find the optimal features that I required to create a model that was as efficient and accurate as possible. Hence, using the matplotlib and seaborn libraries, I created visualizations that helped me see whether there was a correlation between each feature and whether the player went on to play for India or not. Two such visualizations can be seen below, We can observe that there is no trend shown for **Played for India** with respect to **Economy** and **Bowling Average**, respectively, and hence, I did not include them in my model. Through my analysis, I deduced that the following features were most relevant to my project: **'Runs', 'Balls Faced', 'Strike Rate', '100s', '4s', 'Wickets Taken', 'Captain'.**





Model

To begin implementing my model, I first divided my dataset into two categories: training set (the dataset used to train the machine learning model) and test set (the dataset used to measure the accuracy of the machine learning model). Using a machine learning library in Python called Scikit, I randomly separated 70% of the data as the training set and 30% of the data as the test set.

I then used a machine learning technique called logistic regression to create a prediction model that can calculate the probability of any U-19 batsman based on his World Cup statistics. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. A binary variable is one that can have only two possible outcomes; in this project, the criterion feature - Played for India - is a binary variable, as it can have only two possible outputs: True or False. The logistic function used to model this the binary dependent variable can be observed as the following: -

$$p = 1/1 + e^{-(\beta 0 + \beta 1 X1 + \beta 2 X2 ..... \beta n Xn)}$$

where p is the probability of the predicted value being True, {X1, X2, ..., Xn} describes the set of predictor features used to predict the outcome and {β0, β1, β2, ..., βn) describes the set of coefficients assigned to each predictor feature by our machine learning model. This logistic function is also called the sigmoid function. Using the LogisticRegression module in the Scikit library, I implemented my logistic regression model. Another thing that this module helped me do was prevent overfitting. One risk of implementing machine learning models is that the developed algorithm could assign coefficients that are reflective of the training set and not the general data. Hence, I used a technique called regularization that prevents this from happening. Regularization can be thought of as a penalty against complexity. Increasing the regularization strength penalizes "large" weight coefficients. It uses the following equation: -

$$\hat{\boldsymbol{\theta}} = \operatorname{argmin}_{\theta} \sum_{i=1}^{n} (y_i - f_{\boldsymbol{\theta}}(x_i))^2 + \lambda R(\boldsymbol{\theta})$$

The first term is our minimized least squares term and we add another regularization term with parameter ƛ. Hence, using these techniques, I developed a logistic regression model.

Testing the accuracy of this model was rather difficult, as my model predicted probabilities and not discrete outcomes. Therefore, I devised my own measure of testing accuracy. I modeled every probability value >= 0.5 as True and every probability value < 0.5 as False. I used my model to predict the probability for every player represented in the test set, and then calculated the model accuracy. My model obtained an accuracy of 82%, and upon further analysis, I discovered that it was more accurate in predicting which player gets into the team compared to predicting which player doesn't get into the team.

```
X = pd.DataFrame(data=os_data_X,columns=columns )
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
print('Accuracy of logistic regression classifier on test set: {:.2f}'.format(logreg.score(X_train, y_train)))
```

```
Accuracy of logistic regression classifier on test set: 0.82
```

Analysis

Using the statsmodels.api library, I analyzed the coefficients my model assigned to each feature. One issue was, however, that the data each feature represented varied in range. A batsman's runs will always be greater than equal to his batting average, and hence, it wouldn't be possible for me to analyze whether a larger coefficient assigned to the ***Runs*** feature is due to it being more significant or due to the bias in the data. Hence, I first designed a normalizing function, that normalized the data, so that all features represented data between 0 and 1. Using the function, I obtained a dataframe with normalized data, which I could use to analyze the coefficients. Using the statsmodels.api library, I obtained the following table:

```
Optimization terminated successfully.
         Current function value: 0.480838
         Iterations 6
                         Results: Logit
================================================================
Model:              Logit            Pseudo R-squared: 0.176
Dependent Variable: 0.0000           AIC:              85.1641
Date:               2020-05-02 23:47 BIC:              101.2925
No. Observations:   74               Log-Likelihood:   -35.582
Df Model:           6                LL-Null:          -43.181
Df Residuals:       67               LLR p-value:      0.018772
Converged:          1.0000           Scale:            1.0000
No. Iterations:     6.0000
----------------------------------------------------------------
       Coef.     Std.Err.      z      P>|z|     [0.025     0.975]
----------------------------------------------------------------
0      20.9020     7.7281    2.7047   0.0068     5.7553   36.0487
1     -13.4992     4.4435   -3.0380   0.0024   -22.2083   -4.7902
2      -4.6904     1.3789   -3.4014   0.0007    -7.3931   -1.9877
3      -2.3735     2.8490   -0.8331   0.4048    -7.9575    3.2104
4      -0.0182     4.9856   -0.0036   0.9971    -9.7898    9.7535
5      -0.4466     1.1888   -0.3757   0.7071    -2.7766    1.8834
6       1.2401     0.8180    1.5161   0.1295    -0.3631    2.8433
================================================================
```

The first row represents the **Runs** column; the second row represents the **Balls Faced** column; the third row represents the **Strike Rate** column; the fourth row represents the **100s** column; the fifth row represents the **4s** column; the sixth row represents the **Wickets Taken** column; the seventh row represents the **Captain** column.

Positive coefficients have been assigned to **Runs, Captain**.

Negative coefficients have been assigned to **Balls Faced, Strike Rate, 100s, 4s, Wickets Taken**.

The most significant features, with respect to magnitude, are **Runs, Balls Faced, Strike Rate, 100s, Wickets Taken, Captain**.

Findings

Through my analysis, I went on to make the following inferences: -

- The most important factor, unsurprisingly, is the number of runs scored by a player (represented by **Runs**). Scoring more runs increases the chances of a player significantly.

- A high negative coefficient assigned to **Balls Faced** seems counter-intuitive, as playing more deliveries should be an indication of a better batsman. However a negative coefficient assigned to the feature shows that perhaps, if a player has similar stats, then the number of balls faced is inversely proportional to the probability, as taking more balls to score the same number of runs would be inefficient.

- An interesting find was the importance of being the captain of the team (represented by the **Captain** feature) for that particular World Cup, as being captain of their team boosts the player's chances significantly. This further shows that leadership and authority plays a large role in a game like cricket, where captaincy is not only relevant on field but its importance makes you noticable off the field as well.

- A surprising factor was the negative coefficient assigned to the number of wickets taken (represented by the **Wickets Taken** feature), which shows that if a batsman picks up

more wickets, he is less likely to get into the team. This shows that the Indian team demanded players to be specialized in their roles and that a batsman who was a much better batsman was chosen ahead of a player who might not be as good a batsman but also bowled a bit.

Conclusion

Thus, by the end of this project, I had developed a model that was able to predict the probability of an U-19 Indian batsman getting into the main cricket team. I want to expand the availability of this model, and hence, I hope to develop a web application that allows a user to input their statistics and receive a number as an output that represents the probability. Furthermore, I would also look towards developing a similar model for bowlers that would allow me to generalize my model to all Indian U-19 cricketers.