PREDICTING PLAYERS' PERFORMANCE IN ONE DAY INTERNATIONAL CRICKET MATCHES USING MACHINE LEARNING

Kalpdrum Passi and Niravkumar Pandey

Department of Mathematics and Computer Science,
Laurentian University,
Sudbury, Canada

ABSTRACT

Player selection is one the most important tasks for any sport and cricket is no exception. The performance of the players depends on various factors such as the opposition team, the venue, his current form etc. The team management, the coach and the captain select 11 players for each match from a squad of 15 to 20 players. They analyze different characteristics and the statistics of the players to select the best playing 11 for each match. Each batsman contributes by scoring maximum runs possible and each bowler contributes by taking maximum wickets and conceding minimum runs. This paper attempts to predict the performance of players as how many runs will each batsman score and how many wickets will each bowler take for both the teams. Both the problems are targeted as classification problems where number of runs and number of wickets are classified in different ranges. We used naïve bayes, random forest, multiclass SVM and decision tree classifiers to generate the prediction models for both the problems. Random Forest classifier was found to be the most accurate for both the problems.

KEYWORDS

Naïve Bayes, Random Forest, Multiclass SVM, Decision Trees, Cricket

1. INTRODUCTION

Cricket is a sport played by two teams with each side having eleven players. Each team is a right blend of batsmen, bowlers and all rounders. The batsmen's role is to score maximum runs possible and the bowlers have to take maximum wickets and restrict the other team from scoring runs at the same time. All rounders are the players who can both bat and bowl and they contribute by scoring runs and taking wickets. Each player contributes towards the overall performance of the team by giving his best performance in each match. Each player's performance varies with factors like the team he is playing against and the ground at which the match is being played. It is important to select the right players that can perform the best in each match. The performance of a player also depends on several factors like his current form, his performance against a particular team, his performance at a particular venue etc. The team management, the coach and the captain analyze each player's characteristics, abilities and past stats to select the best playing XI for a given match. In other words, they try to predict the players' performance for each match.

In this paper, we predict the players' performance in One Day International (ODI) matches by analyzing their characteristics and stats using supervised machine learning techniques. For this, we predict batsmen's and bowlers' performance separately as how many runs will a batsman score and how many wickets will a bowler take in a particular match.

2. LITERATURE REVIEW

An extensive online search produced very few articles related to players' performance prediction in the game of cricket. A very small number of researchers have studied the performance of cricket players. Muthuswamy and Lam [1] predicted the performance of Indian bowlers against seven international teams against which the Indian cricket team plays most frequently. They used back propagation network and radial basis network function to predict how many runs a bowler is likely to concede and how many wickets a bowler is likely to take in a given ODI match. Wikramasinghe [2] predicted the performance of batsmen in a test series using a hierarchical linear model. Barr and Kantor [3] defined a criterion for comparing and selecting batsmen in limited overs cricket. They defined a new measure P(out) i.e. probability of getting out and used a two-dimensional graphical representation with Strike Rate on one axis and P(out) on another. Then they define a selection criterion based on P(out), strike rate and batting average of the batsmen. Iyer and Sharda [4] used neural networks to predict the performance of players where they classify batsmen and bowlers separately in three categories – performer, moderate and failure. Based on the number of times a player has received different ratings, they recommend if the player should be included in the team to play World Cup 2007. Jhanwar and Paudi [5] predict the outcome of a cricket match by comparing the strengths of the two teams. For this, they measured the performances of individual players of each team. They developed algorithms to model the performances of batsmen and bowlers where they determine the potential of a player by examining his career performance and then his recent performances. Lemmer [6] defined a new measure called Combined Bowling Rate to measure the performance of bowlers. The combined bowling rate is a combination of three traditional bowling measures: bowling average, strike rate and economy. Bhattacharjee and Pahinkar. [7] used this combined bowling rate to analyze the performance of bowlers in Indian Premier League (IPL). They also determined other factors that affect the performance of bowlers and applied multiple regression model to identify the factors that are empirically responsible for the performance of bowlers. Mukharjee. [8] applied Social Network Analysis to rate batsmen and bowlers in a team performance. He generated a directed and weighted network of batsmen-bowlers using player-vs-player information available for test and ODI cricket. He also generated a network of batsmen and bowlers using the dismissal record of batsmen in the history of cricket. Shah [9] also defined new measures to measure players' performance. The new measure for batsmen takes into account the quality of each bowler he is facing and the new measure for bowlers considers the quality of each batsman he is bowling to. The aggregate of individual performance of a batsman against each bowler is the total performance index of the batsman. Similarly, the aggregate of individual performance of a bowler against each batsman is the total performance index of the bowler. Parker, Burns and Natarajan. [10] defined a model for valuation of players for IPL auction. Their model considered factors like previous bidding price of the player, experience of the player, strike rate etc. Prakash, Patvardhan. and Lakshmi [11] defined batting index and bowling index to rank players' performance for their models to predict outcomes of IPL matches. Ovens and Bukiet [12] applied a mathematical approach to suggest optimal batting orders for ODI matches.

Our work is probably the first generalized approach to predict how many runs will a batsman score and how many wickets will a player take on a particular match day. Muthuswamyand Lam [1] carried out a similar study predicting how many wickets will a bowler take using neural networks but their work was limited to eight Indian bowlers and is difficult to generalize for all the bowlers in the world. We used some supervised machine learning algorithms to build prediction models that can be used to predict the performance of any player in a given match.

3. DATA AND TOOLS

We obtained all our data from www.cricinfo.com using scraping tools, parsehub [13] and import.io [14]. For batting, we considered matches played from January 14, 2005 to July 10, 2017. The senior most player during this span was SR Tendulkar, so we collected innings by innings list of the performance of all the batsmen from December 18, 1989 when he played his first ODI match. For bowling, we considered matches played from January 2, 2000 to July 10, 2017. The senior most player during this span was PA de Silva, so we collected innings by innings list of the performance of all the batsmen from March 31, 1984 when he played his first ODI match. Since the past stats of the players such as average, strike rate etc. are not available directly online for each match they played, we calculated from the innings by innings list for each match. We imported all the data in MySQL tables and used php to manipulate them.

For predictive analytics, we used Weka and Dataiku. Both these tools are a collection of machine learning algorithms for data mining and also provide some preprocessing functionalities. All the results in this study have been obtained from Weka 3-9-1-oracle-jvm and Dataiku Data Science Studio on Mac OS 10.11.6 and Windows 10.

4. DATA PREPROCESSING

4.1 Calculating the Traditional Attributes

As mentioned in the previous section, the stats of the players such as average, strike rate etc. are not available directly for each game, we calculated these attributes from the innings by innings list using aggregate functions and mathematical formulae. These attributes are generally used to measure a player's performance. These attributes are as follows:

4.1.1 Batting Attributes

No. of Innings: The number of innings in which the batsman has batted till the day of the match. This attribute signifies the experience of the batsman. The more innings the batsman has played, the more experienced the player is.

Batting Average: Batting average commonly referred to as average is the average number of runs scored per innings. This attribute indicates the run scoring capability of the player.

Average = Runs Scored / Number of times dismissed

Strike Rate (SR): Strike rate is the average number of runs scored per 100 balls faced. This attribute indicates how quickly the batsman can score runs.

Strike Rate: (Runs Scored / Balls Faced) * 100

Centuries: Number of innings in which the batsman scored more than 100 runs.

Fifties: Number of innings in which the batsman scored more than 50 (and less than 100) runs.

Zeros: Number of innings in which the batsman was dismissed without scoring a single run.

Highest Score: The highest runs scored by a batsman in any (single) innings throughout his career.

4.1.2 Bowling Attributes

No. of Innings: The number of innings in which the bowler has bowled till the day of the match.

Overs: The number of overs bowled by a bowler.

Bowling Average: Bowling average is the number of runs conceded by a bowler per wicket taken.

Average: Number of runs conceded / Number of wickets taken

Bowling Strike Rate: Bowling strike rate is the number of balls bowled per wicket taken.

Strike Rate: Number of balls bowled / Number of wickets taken

Four/Five Wicket Haul: Number of innings in which the bowler has taken more than four wickets.

4.2 Rating The Attributes

Each traditional attribute is given a rating from 1 to 5 based on its value with 1 being the minimum and 5 being the maximum.

These ratings are as follows:

No. of Innings:

For Consistency:
1 – 49: 1
50 – 99: 2
100 – 124: 3
125 – 149: 4
>=150: 5

For Form:
1 – 4: 1
5 – 9: 2
10 – 11: 3
12 – 14: 4
>=15: 5

For Opposition:
1 – 2: 1
3 – 4: 2
5 – 6: 3
7 – 9: 4
>=10: 5

For Venue:
1: 1
2: 2
3: 3
4: 4
>=5: 5

Batting Average (for all derived attributes):
        0.0   - 9.99: 1
        10.00 - 19.99: 2
        20.00 - 29.99: 3
        30.00 - 39.99: 4
        >=40: 5

Batting Strike Rate (for all derived attributes):
        0.0   - 49.99: 1
        50.00 - 59.99: 2
        60.00 - 79.99: 3
        80.00 - 100.00: 4
        >=100.00: 5

Centuries:

        For Consistency:
                1 – 4: 1
                5 – 9: 2
                10 – 14: 3
                15 – 19: 4
                >=20: 5

        For Form:
                1: 1
                2: 2
                3: 3
                4: 4
                >=5: 5

        For Opposition:
                1: 3
                2: 4
                >=3: 5

        For Venue:
                1: 4
                >=2: 5

Fifties:

        For Consistency:
                1 – 9: 1
                10 – 19: 2
                20 – 29: 3
                30 – 39: 4
                >=40: 5

        For Form & Opposition:
                1 – 2: 1
                3 – 4: 2
                5 – 6: 3
                7 – 9: 4

>=10: 5

For Venue:
    1: 4
    >=2:– 5

Zeros:

For Consistency:
    1 – 4: 1
    5 – 9: 2
    10 – 14: 3
    15 – 19: 4
    >=20: 5

For Form & Opposition:
    1: 1
    2: 2
    3: 3
    4: 4
    >=5: 5

Highest Score (For Venue Only):
    1 – 24: – 1
    25 – 49: 2
    50 – 99: 3
    100 – 150: 4
    >=150: 5

Overs:

For Consistency:
    1 – 99: 1
    100 – 249: 2
    250 – 499: 3
    500 – 1000: 4
    >=1000: 5

For Form & Opposition:
    1 – 9: 1
    10 – 24: 2
    25 – 49: 3
    50 – 100: 4
    >=100: 5

For Venue:
    1 – 9: 1
    10 – 19: 2
    20 – 29: 3
    30 – 39: 4
    >=40: 5

Bowling Average (for all derived attributes):
        0.00 - 24.99: 5
        25.00 - 29.99: 4
        30.00 - 34.99: 3
        35.00 - 49.99: 2
        >=50.00: 1

Bowling Strike Rate (for all derived attributes):
        0.00 - 29.99: 5
        30.00 -39.99: 4
        40.00 -49.99: 3
        50.00 -59.99: 2
        >=60.00: 1

Four/Five Wicket Haul:

    For Consistency:
        1 – 2: 3
        3 – 4: 4
        >=5: 5

    For Form, Opposition & Venue:
        1 – 2: 4
        >=3: 5

4.3 Calculating The Derived Attributes

We have derived four attributes for batsmen and bowlers to reflect their abilities and characteristics. These four attributes have been derived by taking a weighted average of the traditional attributes described above. The weights were determined using analytic hierarchy process developed by Thomas L. Saaty [15] [16]. These attributes are described below:

4.3.1 Consistency

This attribute describes how experienced the player is and how consistent he has been throughout his career. All the traditional attributes used in this formula are calculated over the entire career of the player.

Formula for batting:

$$\text{Consistency} = 0.4262*\text{average} + 0.2566*\text{no. of innings} + 0.1510*\text{SR} + 0.0787*\text{Centuries} + 0.0556*\text{Fifties} – 0.0328*\text{Zeros}$$

Formula for bowling:

$$\text{Consistency} = 0.4174*\text{no. of overs} + 0.2634*\text{no. of innings} + 0.1602*\text{SR} + 0.0975*\text{average} + 0.0615*\text{FF}$$

4.3.2 Form

Form of a player describes his performance over last one year. All the traditional attributes used in this formula are calculated over the matches played by the player in last 12 months from the day of the match.

Formula for batting:

Form = 0.4262*average + 0.2566*no. of innings + 0.1510*SR + 0.0787*Centuries + 0.0556*Fifties – 0.0328*Zeros

Formula for bowling:

Form = 0.3269*no. of overs + 0.2846*no. of innings + 0.1877*SR + 0.1210*average + 0.0798*FF

4.3.3 Opposition

Opposition describes a player's performance against a particular team. All the traditional attributes used in this formula are calculated over all the matches played by the player against the opposition team in his entire career till the day of the match.

Formula for batting:

Opposition = 0.4262*average + 0.2566*no. of innings + 0.1510*SR + 0.0787*Centuries + 0.0556*Fifties – 0.0328*Zeros

Formula for bowling:

Opposition = 0.3177*no. of overs + 0.3177*no. of innings + 0.1933*SR + 0.1465*average + 0.0943*FF

4.3.4 Venue

Venue describes a player's performance at a particular venue. All the traditional attributes used in this formula are calculated over all the matches played by the player at the venue in his entire career till the day of the match.

Formula for batting:

Venue = 0.4262*average + 0.2566*no. of innings + 0.1510*SR + 0.0787*Centuries + 0.0556*Fifties + 0.0328*HS

Formula for bowling:

Venue = 0.3018*no. of overs + 0.2783*no. of innings + 0.1836*SR + 0.1391*average + 0.0972*FF

4.3.5 Other Input Attributes

Batting Hand: The dominant hand of the batsman while batting. It has two possible values: Left or Right.

Bowling Hand: The dominant hand of the batsman while bowling. It has two possible values: Left or Right.

Batting Position: The number at which the batsman bats in the batting order.

Match Type: The type of the match. This attribute has four possible values: Normal, quarter-final, semi-final or final.

Match Time: The time at which the match is played. There are two possible values: Day or Day-night.

Strength of opposition: This is the batting/bowling strength of the opposition team. It is the average of the consistency measure of the batsmen/bowlers of the opposition team.

Ven: The relative venue for the teams. It has three possible values: Home, Away or Neutral.

Oppo: The opposition team.

The following attributes are used only for predicting runs.

Role: The playing role of the player. It can take following values:

Opening Batsman (OBT) – The two batsmen who usually bat at position one or two are called opening batsmen.

Top Order Batsman (TOB) – The batsmen who usually bat at position three or five are called top order batsmen.

Middle Order Batsman (MOB) - The batsmen who ususally bat at position five to eight are called middle order batsmen.

Batsman – The batsmen who usually bat at different positions are categorised simply as batsmen here.

Allrounder – The players who are equally skilled at both batting and bowling are called allrounders.

Batting Allrounder – The players who can both bat and bowl but are more skilled at batting than bowling, are called batting allrounders.

Bowling Allrounder – The players who can both bat and bowl but are more skilled at bowling than batting, are called bowling allrounders.

Bowler – The players who are expert bowlers but not so skilled at batting, are categorised as bowlers.

Captain: This is a binary attribute indicating whether a player is captain of the team.

WK: This is a binary attribute indicating whether a player is a wicketkeeper.

Innings: This attributes indicates if it is the first or the second innings of the match.

Tournament: The type of tournament in which the match is being played. It can take the following values:

Two Team Tournament (TT)

Three-Four Team Tournament (TFT)

Five Team Tournament (FT)

Toss: Indicates whether the player's team won or lost the toss.

Pressure: Indicates mental and psychological pressure on the player. It takes values from 1 to 5. Its value depends on the type of match being played and the teams that are playing the match. The values are defined as follows:

Normal matches: 1

Quarter Finals: 3

Semi Finals: 4

Finals: 5

Above values are incremented by 1 if the match is between India and Pakistan or Australia and England as these countries are strong rivals of each other.

Host: The country in which the match is being played.

Ground: The ground on which the match is being played.

4.3.6 Outputs

Both the problems are treated as classification problems.

Runs are predicted in five classes:

1 – 24: 1
25 – 49: 2
50 – 74: 3
75 – 99: 4
>=100: 5

Wickets are predicted in three classes:

0 - 1: 1
2 – 3: 2
>=4: 3

4.3.7 Data Cleaning

A large number of values of Opposition and Venue were zero. This is because a player has not played any match against a particular team or at a venue before the day of play. We treated such values as missing values and replaced them with the class average of corresponding attributes.

4.3.8 Class Imbalance

We observed that majority of the records fall within class 1 in both batting and bowling. This created a major imbalance in the distribution of values and affected the performance of the learning algorithms. To solve this problem, we applied an oversampling technique Supervised Minority Oversampling Technique (SMOTE) [17] on minority classes to make all the classes equally distributed. SMOTE over-samples minority classes by creating synthetic example tuples [17].

5. LEARNING ALGORITHMS

For generating the prediction models, we used supervised machine learning algorithms. In supervised learning algorithms, each training tuple is labeled with the class to which it belongs [18]. We used naïve bayes, decision trees, random forest and multiclass support vector machines for our experiments. These algorithms are explained in brief.

5.1 Naïve Bayes

Bayesian classifiers are statistical classifiers that predict the probability with which a given tuple belongs to a particular class [18]. Naïve Bayes classifier assumes that each attribute has its own individual effect on the class label, independent of the values of other attributes. This is called class-conditional independence. Bayesian classifiers are based on Bayes' theorem.

Bayes Theorem: Let X be a data tuple and C be a class label. Let X belongs to class C, then

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

where;

- P(C|X) is the posterior probability of class C given predictor X.
- P(C) is the prior probability of class.
- P(X|C) is the posterior probability of X given the class C.
- P(X) is the prior probability of predictor.

The classifier calculates P(C|X) for every class Ci for a given tuple X. It will then predict that X belongs to the class having the highest posterior probability, conditioned on X. That is X belongs to class Ci if and only if

$$P(Ci|X) > P(Cj|X) \quad \text{for } 1 \le j \le m, j \ne i.$$

5.2 Decision Trees

Decision tree induction is the process of creating decision trees for class-labeled training tuples [18]. A decision tree is basically a tree structure like a flowchart [18]. Each internal node of the tree represents a test on an attribute and each branch is the outcome of the test. Each leaf node is a class label. The first node at the top of the tree is the root node. To classify a given tuple X, the attributes of the tuple are tested against the decision tree starting from the root node to the leaf node which holds the class prediction of the tuple. Ross Quinlan introduced a decision tree algorithm called ID3 in his paper [19]. Later he introduced a successor of ID3 called C4.5 in [20] to overcome some shortcomings such as over-fitting. Unlike ID3, C4.5 can handle both continuous and discrete attributes, training data with missing values and attributes with differing costs. In a basic decision tree induction algorithm, all the training tuples are at the root node at start. The tuples are then partitioned recursively based on selected attributes. The attributes are selected based on an attribute selection method which specifies a heuristic procedure to determine the splitting criterion. The algorithm terminates if all the training tuples belong to the same class or there are no remaining attributes for further partitioning or all training tuples are used. ID3 uses the attribute selection measure called information gain, which is simply the difference of the information needed to classify a tuple and the information needed after the split. These two can be formularized as follows:

Expected information needed to classify a tuple in the training set D

$$Info(D) = -\sum_{i=1}^{m} p_i \, log_2(p_i)$$

where; pi is the nonzero probability that a tuple in D belongs to class Ci.

Information needed after the splitting (to arrive at the exact classification)

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

where A is the attribute on which the tuples are to be partitioned.

Then, information gain

$$Gain(A) = Info(D) - Info_A(D)$$

The attribute with highest information gain is selected as the splitting attribute.

C4.5 uses gain ratio as the attribute selection measure. Gain ratio is an extension to information gain in a sense because it normalizes information gain by using a split information value;

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2\left(\frac{|D_j|}{|D|}\right)$$

Then,

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$

The attribute with the highest gain ratio is selected as the splitting attribute.

5.3 Random Forest

Random Forests is an ensemble method for classification and regression [18]. Random forests are a set of decision trees where each tree is dependent on a random vector sampled independently and with the same distribution of all the trees in the forest [21]. The algorithm generates a number of decision trees creating a forest. Each decision tree is generated by selecting random attributes at each node to determine the split [18]. Tim Kam Ho introduced the first method for random forests using random subspace method in his paper [22]. Later, Breiman Leo extended the algorithm in his paper [21] and this method was official known as Random Forests. The general procedure to generate decision trees for random forests starts with a dataset D of d tuples. To generate k decision trees from the dataset, for each iteration k, a training set Di of d tuples is sampled with replacement from the dataset D. To construct a decision tree classifier, at each node, a small number of attributes from the available attributes are selected randomly as candidates for the split at the node. Then Classification And Regression Trees (CART) [23] method is used to grow the trees. The trees are then grown to maximum size and are not pruned. CART is a non-parametric decision tree induction technique that can generate classification and regression trees. CART recursively selects rules based on variables' values to get the best split. It stops splitting when it detects that no further gain can be made or some pre-determined stopping conditions are met.

5.4 Support Vector Machine

Vladimir Vapnik, Bernhard Boser and Isabell Guyon introduced the concept of support vector machine in their paper [24]. SVMs are highly accurate and less prone to overfitting. SVMs can be used for both numeric prediction and classification. SVM transforms the original data into a higher dimension using a nonlinear mapping. It then searches for a linear optimal hyperplane in this new dimension separating the tuples of one class from another. With an appropriate mapping to a sufficiently high dimension, tuples from two classes can always be separated by a hyperplane. The algorithm finds this hyperplane using support vectors and margins defined by the support vectors. The support vectors found by the algorithm provide a compact description of the learned prediction model. A separating hyperplane can be written as:

$$W \cdot X + b = 0$$

where W is a weight vector, W = {w1, w2, w3,..., wn}, n is the number of attributes and b is a scalar often referred to as a bias. If we input two attributes A1 and A2, training tuples are 2-D, (e.g., X = (x1, x2)), where x1 and x2 are the values of attributes A1 and A2, respectively. Thus, any points above the separating hyperplane belong to Class A1:

$$W \cdot X + b > 0$$

and any points below the separating hyperplane belong to Class A2:

$$W \cdot X + b < 0$$

SVM was originally used for binary classification. However, several multiclass SVM algorithms have also been developed. In Weka, we used LIBSVM package developed by Chih-Chung Chang and Chih-Jen Lin [25]. The package can be downloaded from http://www.csie.ntu.edu.tw/ cjlin/libsvm. LIBSVM is an easy to use package to apply multiclass SVM and has gained a wide popularity in machine learning.

6. RESULTS AND DISCUSSIONS

We used different sizes of training and test sets to find the best combination that gives the most accuracy. We used four machine learning algorithms: Naïve Bayes, Decision Trees, Random Forest and Support Vector Machine in our experiments. The results are tabulated below. Table 1 shows the accuracies of the algorithms for predicting runs and table 2 shows the accuracies of the algorithms for predicting wickets.

As we can see, Random Forest builds the most accurate prediction models for both batting and bowling in all the cases. Also, the accuracy of the models increases as we increase the size of the training dataset for all algorithms except in case of Naïve Bayes for batting where the accuracy decreases as we increase the size of the training set. Random Forest predicts runs with the highest accuracy of 90.74% when we use 90% of the dataset for training. Similarly, Random Forest predicts wickets with highest accuracy of 92.25% when we use 90% of the dataset for training. On the other hand, Naïve Bayes predicts runs with the least accuracy of 42.5% when we use 90% of the dataset for training. Naïve Bayes predicts wickets too with the least accuracy of 57.05% when we use 60% of the dataset for training. Decision Trees performs reasonably well with the maximum accuracy of 80.46% and the minimum accuracy of 77.93% for predicting runs. It predicts wickets with the maximum accuracy of 86.5% and the minimum accuracy of 84.4%, which is again reasonably well against the performance of Random Forest. The prediction models of SVM for predicting runs showed the maximum accuracy of 51.45% with 90% training data and the minimum accuracy of 50.54% with 60% training data. Also for wickets, SVM had the maximum accuracy of 68.78% with 90% training data and the minimum accuracy of 67.45% with

60% training data. Thus, surprisingly SVM was beaten by Random Forest and Decision Trees on both the datasets.

Table 1 Predicting runs

Classifier	Accuracy (%)			
	60% train 40% test	70% train 30% test	80% train 20% test	90% train 10% test
Naïve Bayes	43.08	42.95	42.47	42.50
Decision Trees	77.93	79.02	79.38	80.46
Random Forest	89.92	90.27	90.67	90.74
SVM	50.54	50.85	50.88	51.45

Table 2 Predicting wickets

Classifier	Accuracy (%)			
	60% train 40% test	70% train 30% test	80% train 20% test	90% train 10% test
Naïve Bayes	57.05	57.18	57.48	58.12
Decision Trees	84.40	85.12	85.99	86.50
Random Forest	90.68	91.26	91.80	92.25
SVM	67.45	67.53	68.35	68.78

Muthuswamy and Lam [1] achieved an accuracy of 87.10% with backpropagation network (BPN) and 91.43% with radial basis function network (RBFN) table 3. They predicted wickets in two classes: 0 or 1 and 2 or more. Their study was limited only to eight Indian bowlers who played ODI matches for India since year 2000 against seven countries. Our approach works for any player in the world and is also applicable to new players who will be playing for their country in future. Moreover, they considered only three input parameters: An ID assigned manually to a bowler, an ID assigned manually to the opposition team and the number of overs bowled by the bowler. We carried out a detailed study of the stats and characteristics of the players by considering a number of attributes that can potentially impact the performance of a player in any match.

Table 3 Predicting wickets using BPN and RBFN [1]

Model	Accuracy (%)
BPN	87.10
RBFN	91.43

7. CONCLUSION AND FUTURE WORK

Selection of the right players for each match plays a significant role in a team's victory. An accurate prediction of how many runs a batsman is likely to score and how many wickets a bowler is likely to take in a match will help the team management select best players for each match. In this paper, we modeled batting and bowling datasets based on players' stats and characteristics.

Some other features that affect players' performance such as weather or the nature of the wicket could not be included in this study due to unavailability of data. Four multiclass classification algorithms were used and compared. Random Forest turned out to be the most accurate classifier for both the datasets with an accuracy of 90.74% for predicting runs scored by a batsman and 92.25% for predicting wickets taken by a bowler. Results of SVM were surprising as it achieved an accuracy of just 51.45% for predicting runs and 68.78% for predicting wickets.

Similar studies can be carried out for other formats of the game i.e. test cricket and T20 matches. The models for these formats can be shaped to reflect required characteristics of the players; e.g. batsmen need to have patience and ability to play longer innings in test matches whereas score more runs in less overs in T20 matches. Similarly, bowlers need to have stronger wicket taking abilities in test matches and better economy rate i.e. conceding less runs in T20 matches. Moreover, attempts can be made to improve accuracies of the classifiers for ODI matches.

REFERENCES

[1]   S. Muthuswamy and S. S. Lam, "Bowler Performance Prediction for One-day International Cricket Using Neural Networks," in Industrial Engineering Research Conference, 2008.

[2]   I. P. Wickramasinghe, "Predicting the performance of batsmen in test cricket," Journal of Human Sport & Excercise, vol. 9, no. 4, pp. 744-751, May 2014.

[3]   G. D. I. Barr and B. S. Kantor, "A Criterion for Comparing and Selecting Batsmen in Limited Overs Cricket," Operational Research Society, vol. 55, no. 12, pp. 1266-1274, December 2004.

[4]   S. R. Iyer and R. Sharda, "Prediction of athletes performance using neural networks: An application in cricket team selection," Expert Systems with Applications, vol. 36, pp. 5510-5522, April 2009.

[5]   M. G. Jhanwar and V. Pudi, "Predicting the Outcome of ODI Cricket Matches: A Team Composition Based Approach," in European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD 2016 2016), 2016.

[6]   H. H. Lemmer, "The combined bowling rate as a measure of bowling performance in cricket," South African Journal for Research in Sport, Physical Education and Recreation, vol. 24, no. 2, pp. 37-44, January 2002.

[7]   D. Bhattacharjee and D. G. Pahinkar, "Analysis of Performance of Bowlers using Combined Bowling Rate," International Journal of Sports Science and Engineering, vol. 6, no. 3, pp. 1750-9823, 2012.

[8]   S. Mukherjee, "Quantifying individual performance in Cricket - A network analysis of batsmen and bowlers," Physica A: Statistical Mechanics and its Applications, vol. 393, pp. 624-637, 2014.

[9]   P. Shah, "New performance measure in Cricket," ISOR Journal of Sports and Physical Education, vol. 4, no. 3, pp. 28-30, 2017.

[10] D. Parker, P. Burns and H. Natarajan, "Player valuations in the Indian Premier League," Frontier Economics, vol. 116, October 2008.

[11] C. D. Prakash, C. Patvardhan and C. V. Lakshmi, "Data Analytics based Deep Mayo Predictor for IPL-9," International Journal of Computer Applications, vol. 152, no. 6, pp. 6-10, October 2016.

[12] M. Ovens and B. Bukiet, "A Mathematical Modelling Approach to One-Day Cricket Batting Orders," Journal of Sports Science and Medicine, vol. 5, pp. 495-502, 15 December 2006.

[13] "Free web scraping - Download the most powerful web scraper | ParseHub," parsehub, [Online]. Available: https://www.parsehub.com.

[14] "Import.io | Extract data from the web," Import.io, [Online]. Available: https://www.import.io.

[15] T. L. Saaty, "A scaling method for priorities in a hierarchichal structure," Mathematical Psychology, vol. 15, pp. 234-281, 1977.

[16] T. L. Saaty, The Analytic Hierarchy Process, New York: McGraw-Hill, 1980.

[17] N. V. Chavla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," Artificial Intelligence Research, vol. 16, p. 321–357, June 2002.

[18] J. Han, M. Kamber and J. Pei, Data Mining: Concepts and Techniques, 3rd Edition ed., Waltham: Elsevier, 2012.

[19] J. R. Quinlan, "Induction of Decision Trees," Machine learning, vol. 1, no. 1, pp. 81-106, 1986.

[20] J. R. Quinlan, C4.5: Programs for Machine Learning, Elsevier, 2015.

[21] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.

[22] T. K. Ho, "The Random Subspace Method for Constructing Decision Forests," IEEE transactions on pattern analysis and machine intelligence, vol. 20, no. 8, pp. 832-844, August 1998.

[23] L. Breiman, J. Friedman, C. J. Stone and R. A. Olshen, Classification and regression trees, CRC Press, 1984.

[24] B. E. Boser, I. M. Guyon and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, 1992.

[25] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 3, April 2011.

AUTHORS

Kalpdrum Passi received his Ph.D. in Parallel Numerical Algorithms from Indian Institute of Technology, Delhi, India in 1993. He is an Associate Professor, Department of Mathematics & Computer Science, at Laurentian University, Ontario, Canada. He has published many papers on Parallel Numerical Algorithms in international journals and conferences. He has collaborative work with faculty in Canada and US and the work was tested on the CRAY XMP's and CRAY YMP's. He transitioned his research to web technology, and more recently has been involved in machine learning and data mining applications in bioinformatics, social media and other data science areas. He obtained funding from NSERC and Laurentian University for his research. He is a member of the ACM and IEEE Computer Society.