Cricket Players (Cricketers) Data Analysis

CS 439 - Introduction to Data Science
Course Project

Professor: Gerard de Melo

Name: Smitkumar Patel
NetID: Shp109
Group: 06
Recitation: 02
Email: shp109@rutgers.edu

Contents

DATA SETS:

For this project, I am using the following two data sets from 'https://data.world/':

Cricket Players Data from ESPN:

The data set contains information about the cricket players all over the world that played Tests, ODI, List A, First-class, T20I, and T20. There are a total of 90308 observations (rows) and 176 variables(columns).
URL: https://data.world/raghav333/cricket-players-espn
Below is the description of variables:

Attribute	Type	Description	Example
ID	Numerical or categorical	Unique ID of the cricket players	8772,532565,16856
NAME	Object	Name of the players- first name and last name	Henry Arkell, Richard Nyren, Sydney Maartensz
COUNTRY	Categorical	Name of the player's country	India, England, Australia
Full name	Object	Cricket Player's full name	Henry John Denham Arkell, Brian Richard Lander
Birthdate	Object or categorical (Date)	Cricket player's birthdate	06-26-1898,04-25-1734
Birthplace	categorical or object	Player's county/state/city	Ahmedabad, Gujarat, India, Melbourne, Victoria, Australia
Died	Categorical (Binary)	Player is alive/dead	Alive, Dead
Date_of_death	Object (Date) or categorical	Date when the player died	12/3/1982,1797-04-25
Age	Numerical	Player's age	32,51,20
Major teams	Categorical	Player's team Name	India Under-19s, United Arab Emirates
Batting style	Categorical (Binary)	Way of batting	Right-hand bat, Left-hand bat
Bowling style	Categorical	Way of blowing-fast/medium/leg break	Right-arm medium, Leg break, Leg break googly
Other	Categorical	Other's thing that players do mostly after retirement	Umpire, Coach
AWARDS	Object	Name of the Awards player achieved	Education: Master's Degree in History from Punjab University, ICC Regional Development Manager - Asia
BATTING_Tests_Mat	Numerical	Total number of Test matches player played	15,200,50
BATTING_Tests_Inns	Numerical	Number of times player did bat in Test innings	10, 40, 87
BATTING_Tests_NO	Numerical	Batting Test number	2,5,11
BATTING_Tests_Runs	Numerical	Total number of runs in Test cricket carrier	1000, 250, 88
BATTING_Tests_HS	Numerical	Total number of hours player did bat in Test matches	44, 5, 100

BATTING_Tests_Ave	Numerical	Total number of runs they have scored in Test/ number of times they have been out in Test	33.33,43.78
BATTING_Tests_BF	Numerical	The total number of balls faced by player in Test	244,2000,10
BATTING_Tests_SR	Numerical	The frequency of the player score runs in Test (Strike Rate) = (Runs Scored*100) / balls faced in Test	43.78,69.56
BATTING_Tests_100	Numerical	Number of times player scored 100 or more in Test	5,20,100
BATTING_Tests_50	Numerical	Number of times player scored between 50 to 99 in Test	5,20,100
BATTING_Tests_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in Test	15,5,20
BATTING_Tests_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in Test	2,5,50
BATTING_Tests_Ct	Numerical	Total number of catches taken in Test carrier	82,80,20
BATTING_Tests_St	Numerical	Total number of stumping made in Test carrier	10,100,50
BATTING_ODIs_Mat	Numerical	Total number of ODI matches player played	15,200,50
BATTING_ODIs_Inns	Numerical	Number of times player did bat in ODI innings	10, 40, 87
BATTING_ODIs_NO	Numerical	Batting ODI number	2,5,11
BATTING_ODIs_Runs	Numerical	Total number of runs in ODI cricket carrier	1000, 250, 88
BATTING_ODIs_HS	Numerical	Total number of hours player did bat in ODI matches	44, 5, 100
BATTING_ODIs_Ave	Numerical	Total number of runs they have scored in ODI / number of times they have been out in ODI	33.33,43.78
BATTING_ODIs_BF	Numerical	The total number of balls faced by player in ODI	244,2000,10
BATTING_ODIs_SR	Numerical	The frequency of the player score runs in ODI (Strike Rate) = (Runs Scored*100) / balls faced in ODI	43.78,69.56
BATTING_ODIs_100	Numerical	Number of times player scored 100 or more in ODI	5,20,100
BATTING_ODIs_50	Numerical	Number of times player scored between 50 to 99 in ODI	5,20,100
BATTING_ODIs_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in ODI	15,5,20
BATTING_ODIs_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in ODI	2,5,50

BATTING_ODIs_Ct	Numerical	Total number of catches taken in ODI carrier	82,80,20
BATTING_ODIs_St	Numerical	Total number of stumping made in ODI carrier	10,100,50
BATTING_T20Is_Mat	Numerical	Total number of international T20 matches player played	15,200,50
BATTING_T20Is_Inns	Numerical	Number of times player did bat in international T20 innings	10, 40, 87
BATTING_T20Is_NO	Numerical	Batting international T20 number	2,5,11
BATTING_T20Is_Runs	Numerical	Total number of runs in international T20 cricket carrier	1000, 250, 88
BATTING_T20Is_HS	Numerical	Total number of hours player did bat in international T20 matches	44, 5, 100
BATTING_T20Is_Ave	Numerical	Total number of runs they have scored in international T20 / number of times they have been out in international T20	33.33,43.78
BATTING_T20Is_BF	Numerical	The total number of balls faced by player in international T20	244,2000,10
BATTING_T20Is_SR	Numerical	The frequency of the player score runs in international T20(Strike Rate) = (Runs Scored*100) / balls faced in international T20	43.78,69.56
BATTING_T20Is_100	Numerical	Number of times player scored 100 or more in international T20	5,20,100
BATTING_T20Is_50	Numerical	Number of times player scored between 50 to 99 in international T20	5,20,100
BATTING_T20Is_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in international T20	15,5,20
BATTING_T20Is_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in international T20	2,5,50
BATTING_T20Is_Ct	Numerical	Total number of catches taken in international T20 carrier	82,80,20
BATTING_T20Is_St	Numerical	Total number of stumping made in international T20 carrier	10,100,50
BATTING_First-class_Mat	Numerical	Total number of First-class matches player played	15,200,50
BATTING_First-class_Inns	Numerical	Number of times player did bat in First-class innings	10, 40, 87
BATTING_First-class_NO	Numerical	Batting First-class number	2,5,11
BATTING_First-class_Runs	Numerical	Total number of runs in First-class cricket carrier	1000, 250, 88
BATTING_First-class_HS	Numerical	Total number of hours player did bat in First-class matches	44, 5, 100

BATTING_First-class_Ave	Numerical	Total number of runs they have scored in First-class / number of times they have been out in First-class	33.33,43.78
BATTING_First-class_BF	Numerical	The total number of balls faced by player in First-class	244,2000,10
BATTING_First-class_SR	Numerical	The frequency of the player score runs in First-class (Strike Rate) = (Runs Scored*100) / balls faced in First-class	43.78,69.56
BATTING_First-class_100	Numerical	Number of times player scored 100 or more in First-class	5,20,100
BATTING_First-class_50	Numerical	Number of times player scored between 50 to 99 in First-class	5,20,100
BATTING_First-class_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in First-class	15,5,20
BATTING_First-class_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in First-class	2,5,50
BATTING_First-class_Ct	Numerical	Total number of catches taken in First-class carrier	82,80,20
BATTING_First-class_St	Numerical	Total number of stumping made in First-class carrier	10,100,50
BATTING_List A_Mat	Numerical	Total number of List A match's player played	15,200,50
BATTING_List A_Inns	Numerical	Number of times player did bat in List A innings	10, 40, 87
BATTING_List A_NO	Numerical	Batting List A number	2,5,11
BATTING_List A_Runs	Numerical	Total number of runs in List A cricket carrier	1000, 250, 88
BATTING_List A_HS	Numerical	Total number of hours player did bat in List A matches	44, 5, 100
BATTING_List A_Ave	Numerical	Total number of runs they have scored in List A / number of times they have been out in List A	33.33,43.78
BATTING_List A_BF	Numerical	The total number of balls faced by player in List A	244,2000,10
BATTING_List A_SR	Numerical	The frequency of the player score runs in List A(Strike Rate) = (Runs Scored*100) / balls faced in List A	43.78,69.56
BATTING_List A_100	Numerical	Number of times player scored 100 or more in List A	5,20,100
BATTING_List A_50	Numerical	Number of times player scored between 50 to 99 in List A	5,20,100
BATTING_List A_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in List A	15,5,20

BATTING_List A_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in List A	2,5,50
BATTING_List A_Ct	Numerical	Total number of catches taken in List A carrier	82,80,20
BATTING_List A_St	Numerical	Total number of stumping made in List A carrier	10,100,50
BATTING_T20s_Mat	Numerical	Total number of T20 matches player played	15,200,50
BATTING_T20s_Inns	Numerical	Number of times player did bat in T20 innings	10, 40, 87
BATTING_T20s_NO	Numerical	Batting T20 number	2,5,11
BATTING_T20s_Runs	Numerical	Total number of runs in T20 cricket carrier	1000, 250, 88
BATTING_T20s_HS	Numerical	Total number of hours player did bat in T20 matches	44, 5, 100
BATTING_T20s_Ave	Numerical	Total number of runs they have scored in T20 / number of times they have been out in T20	33.33,43.78
BATTING_T20s_BF	Numerical	The total number of balls faced by player in T20	244,2000,10
BATTING_T20s_SR	Numerical	The frequency of the player score runs in T20(Strike Rate) = (Runs Scored*100) / balls faced in T20	43.78,69.56
BATTING_T20s_100	Numerical	Number of times player scored 100 or more in T20	5,20,100
BATTING_T20s_50	Numerical	Number of times player scored between 50 to 99 in T20	5,20,100
BATTING_T20s_4s	Numerical	Number of times player scored Four (ball bounces before going over or touch the perimeter) in T20	15,5,20
BATTING_T20s_6s	Numerical	Number of times player scored Six (ball go over the perimeter without bounce) in T20	2,5,50
BATTING_T20s_Ct	Numerical	Total number of catches taken in T20 carrier	82,80,20
BATTING_T20s_St	Numerical	Total number of stumping made in T20 carrier	10,100,50
BOWLING_Tests_Mat	Numerical	Total number of Test matches player played	15,200,50
BOWLING_Tests_Inns	Numerical	Number of times player did bowl in Test innings	10, 40, 87
BOWLING_Tests_Balls	Numerical	Total number of balls player throw in Test	2129, 336, 87
BOWLING_Tests_Runs	Numerical	Total number of runs given by player in Test	50, 1000, 20
BOWLING_Tests_Wkts	Numerical	Total number of wickets player took in Test carrier	70, 125, 50
BOWLING_Tests_BBI	Categorical or object	Best bowling in Test Innings(wickets/runs)	6/27,5/50

BOWLING_Tests_BBM	Categorical or Object	Best bowling in Test Match's(wickets/runs)	9/86,5/98
BOWLING_Tests_Ave	Numerical	Average number of run per wickets (Ave=total runs given in Test/total wickets in Test)	20.33,40.5
BOWLING_Tests_Econ	Numerical	Average number of runs per over (one over = 6 balls) in Test. Econ = total runs in Test/ total over bowled in Test	2.69,4.55
BOWLING_Tests_SR	Numerical	Average number of balls per wicket in Test. SR= Total balls in Test/ Total wickets in Test	45.1,32.5
BOWLING_Tests_4w	Numerical	Number of innings in which player took five wickets or more in Test	2,3,5
BOWLING_Tests_5w	Numerical	Number of innings in which player took four wickets in Test	8,5,1
BOWLING_Tests_10	Numerical	Number of matches in which player took Ten wickets or more in Test	5,8,7
BOWLING_ODIs_Mat	Numerical	Total number of ODI matches player played	15,200,50
BOWLING_ODIs_Inns	Numerical	Number of times player did bowl in ODI innings	10, 40, 87
BOWLING_ODIs_Balls	Numerical	Total number of balls player throw in ODI	2129, 336, 87
BOWLING_ODIs_Runs	Numerical	Total number of runs given by player in ODI	50, 1000, 20
BOWLING_ODIs_Wkts	Numerical	Total number of wickets player took in ODI carrier	70, 125, 50
BOWLING_ODIs_BBI	Categorical or object	Best bowling in ODI Innings(wickets/runs)	6/27,5/50
BOWLING_ODIs_BBM	Categorical or Object	Best bowling in ODI Match's(wickets/runs)	9/86,5/98
BOWLING_ODIs_Ave	Numerical	Average number of run per wickets (Ave=total runs given in ODI/total wickets in ODI)	20.33,40.5
BOWLING_ODIs_Econ	Numerical	Average number of runs per over (one over = 6 balls) in ODI. Econ = total runs in ODI/ total over bowled in ODI	2.69,4.55
BOWLING_ODIs_SR	Numerical	Average number of balls per wicket in ODI. SR= Total balls in ODI/ Total wickets in ODI	45.1,32.5
BOWLING_ODIs_4w	Numerical	Number of innings in which player took five wickets or more in ODI	2,3,5
BOWLING_ODIs_5w	Numerical	Number of innings in which player took four wickets in ODI	8,5,1
BOWLING_ODIs_10	Numerical	Number of matches in which player took Ten wickets or more in ODI	5,8,7
BOWLING_T20Is_Mat	Numerical	Total number of international T20 matches player played	15,200,50
BOWLING_T20Is_Inns	Numerical	Number of times player did bowl in international T20 innings	10, 40, 87

BOWLING_T20Is_Balls	Numerical	Total number of balls player throw in international T20	2129, 336, 87
BOWLING_T20Is_Runs	Numerical	Total number of runs given by player in international T20	50, 1000, 20
BOWLING_T20Is_Wkts	Numerical	Total number of wickets player took in international T20 carrier	70, 125, 50
BOWLING_T20Is_BBI	Categorical or object	Best bowling in international T20 Innings(wickets/runs)	6/27,5/50
BOWLING_T20Is_BBM	Categorical or Object	Best bowling in international T20 Match's(wickets/runs)	9/86,5/98
BOWLING_T20Is_Ave	Numerical	Average number of run per wickets (Ave=total runs given in international T20/total wickets in international T20)	20.33,40.5
BOWLING_T20Is_Econ	Numerical	Average number of runs per over (one over = 6 balls) in international T20. Econ = total runs in international T20/ total over bowled in international T20	2.69,4.55
BOWLING_T20Is_SR	Numerical	Average number of balls per wicket in international T20. SR= Total balls in international T20/ Total wickets in international T20	45.1,32.5
BOWLING_T20Is_4w	Numerical	Number of innings in which player took five wickets or more in international T20	2,3,5
BOWLING_T20Is_5w	Numerical	Number of innings in which player took four wickets in international T20	8,5,1
BOWLING_T20Is_10	Numerical	Number of matches in which player took Ten wickets or more in international T20	5,8,7
BOWLING_First-class_Mat	Numerical	Total number of First-class matches player played	15,200,50
BOWLING_First-class_Inns	Numerical	Number of times player did bowl in First-class innings	10, 40, 87
BOWLING_First-class_Balls	Numerical	Total number of balls player throw in First-class	2129, 336, 87
BOWLING_First-class_Runs	Numerical	Total number of runs given by player in First-class	50, 1000, 20
BOWLING_First-class_Wkts	Numerical	Total number of wickets player took in First-class carrier	70, 125, 50
BOWLING_First-class_BBI	Categorical or object	Best bowling in First-class Innings(wickets/runs)	6/27,5/50
BOWLING_First-class_BBM	Categorical or Object	Best bowling in First-class Match's(wickets/runs)	9/86,5/98
BOWLING_First-class_Ave	Numerical	Average number of run per wickets (Ave=total runs given in First-class/total wickets in First-class)	20.33,40.5

BOWLING_First-class_Econ	Numerical	Average number of runs per over (one over = 6 balls) in First-class. Econ = total runs in First-class/ total over bowled in First-class	2.69,4.55
BOWLING_First-class_SR	Numerical	Average number of balls per wicket in First-class. SR= Total balls in First-class/ Total wickets in First-class	45.1,32.5
BOWLING_First-class_4w	Numerical	Number of innings in which player took five wickets or more in First-class	2,3,5
BOWLING_First-class_5w	Numerical	Number of innings in which player took four wickets in First-class	8,5,1
BOWLING_First-class_10	Numerical	Number of matches in which player took Ten wickets or more in First-class	5,8,7
BOWLING_List A_Mat	Numerical	Total number of List A match's player played	15,200,50
BOWLING_List A_Inns	Numerical	Number of times player did bowl in List A innings	10, 40, 87
BOWLING_List A_Balls	Numerical	Total number of balls player throw in List A	2129, 336, 87
BOWLING_List A_Runs	Numerical	Total number of runs given by player in List A	50, 1000, 20
BOWLING_List A_Wkts	Numerical	Total number of wickets player took in List A carrier	70, 125, 50
BOWLING_List A_BBI	Categorical or object	Best bowling in List A Innings(wickets/runs)	6/27,5/50
BOWLING_List A_BBM	Categorical or Object	Best bowling in List A Match's(wickets/runs)	9/86,5/98
BOWLING_List A_Ave	Numerical	Average number of run per wickets (Ave=total runs given in List A/total wickets in List A)	20.33,40.5
BOWLING_List A_Econ	Numerical	Average number of runs per over (one over = 6 balls) in List A. Econ = total runs in List A/ total over bowled in List A	2.69,4.55
BOWLING_List A_SR	Numerical	Average number of balls per wicket in List A. SR= Total balls in List A/ Total wickets in List A	45.1,32.5
BOWLING_List A_4w	Numerical	Number of innings in which player took five wickets or more in List A	2,3,5
BOWLING_List A_5w	Numerical	Number of innings in which player took four wickets in List A	8,5,1
BOWLING_List A_10	Numerical	Number of matches in which player took Ten wickets or more in List A	5,8,7
BOWLING_T20s_Mat	Numerical	Total number of T20 matches player played	15,200,50
BOWLING_T20s_Inns	Numerical	Number of times player did bowl in T20 innings	10, 40, 87

BOWLING_T20s_Balls	Numerical	Total number of balls player throw in T20	2129, 336, 87
BOWLING_T20s_Runs	Numerical	Total number of runs given by player in T20	50, 1000, 20
BOWLING_T20s_Wkts	Numerical	Total number of wickets player took in T20 carrier	70, 125, 50
BOWLING_T20s_BBI	Categorical or object	Best bowling in T20 Innings(wickets/runs)	6/27,5/50
BOWLING_T20s_BBM	Categorical or Object	Best bowling in T20 Match's(wickets/runs)	9/86,5/98
BOWLING_T20s_Ave	Numerical	Average number of run per wickets (Ave=total runs given in T20/total wickets in T20)	20.33,40.5
BOWLING_T20s_Econ	Numerical	Average number of runs per over (one over = 6 balls) in T20. Econ = total runs in T20/ total over bowled in T20	2.69,4.55
BOWLING_T20s_SR	Numerical	Average number of balls per wicket in T20. SR= Total balls in T20/ Total wickets in T20	45.1,32.5
BOWLING_T20s_4w	Numerical	Number of innings in which player took five wickets or more in T20	2,3,5
BOWLING_T20s_5w	Numerical	Number of innings in which player took four wickets in T20	8,5,1
BOWLING_T20s_10	Numerical	Number of matches in which player took Ten wickets or more in T20	5,8,7

IPL Cricket Players Data:

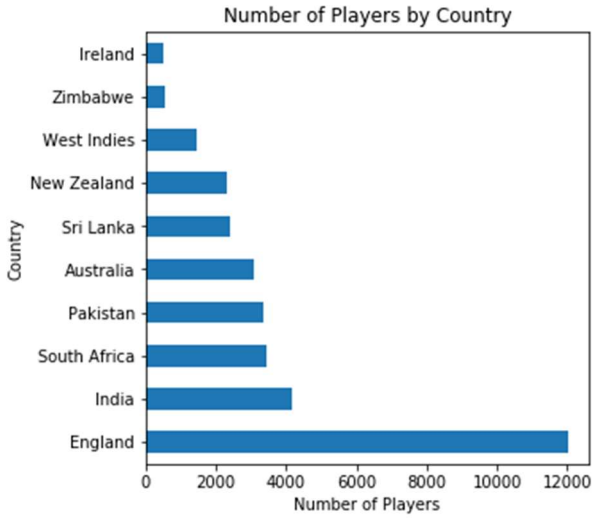
The data set contains information about the cricket players that played IPL (Indian Premier League). There are a total of 497 observations (rows) and 7 variables(columns).
URL: https://data.world/raghu543/ipl-data-till-2017/workspace/file?filename=Player.csv
Below is the description of variables:

Attribute	Type	Description	Example
PLAYER_SK	Numerical	index	1,2,3
Player_Id	Numerical or Categorical	Player Id	5,6,7
Player_Name	Categorical or Object	Player's Name	Sc Gangly, RT Ponting
DOB	Categorical (Date)	Player's Birthdate	1972-07-08, 1986-10-02
Batting_hand	Categorical (Binary)	Way of batting	Left-hand bat, Right-hand bat
Bowling_skill	Categorical	Way of bowling	Right-arm medium, Right-arm off break
Country_Name	Categorical	Player's country	India, South Africa

Note: I left join 'Cricket Players Data from ESPN' and 'IPL Cricket Players Data'. That means all records from 'Cricket Players Data from ESPN', and matched records from the 'IPL Cricket Players Data' on 'Country', 'Birthdate', and 'Last_Name'. In addition, I created new column that contains binary labels 'Play' (player played IPL) and 'Not Play' (player did not play IPL).

Analysis:

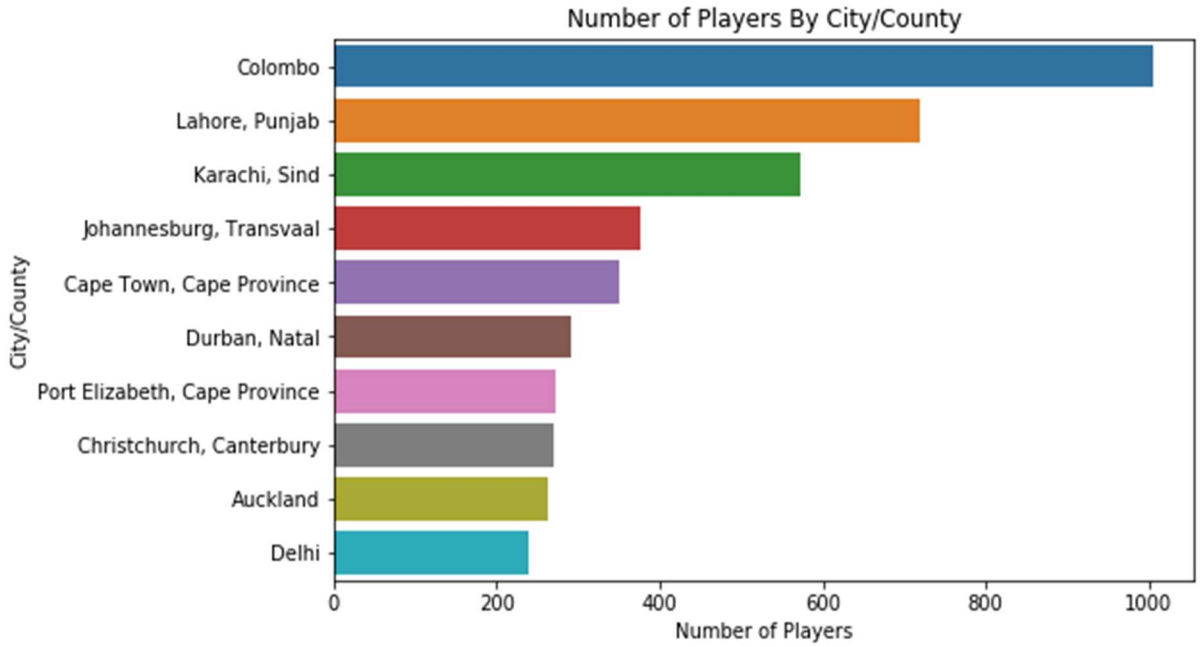
Top ten countries with the most cricket players:



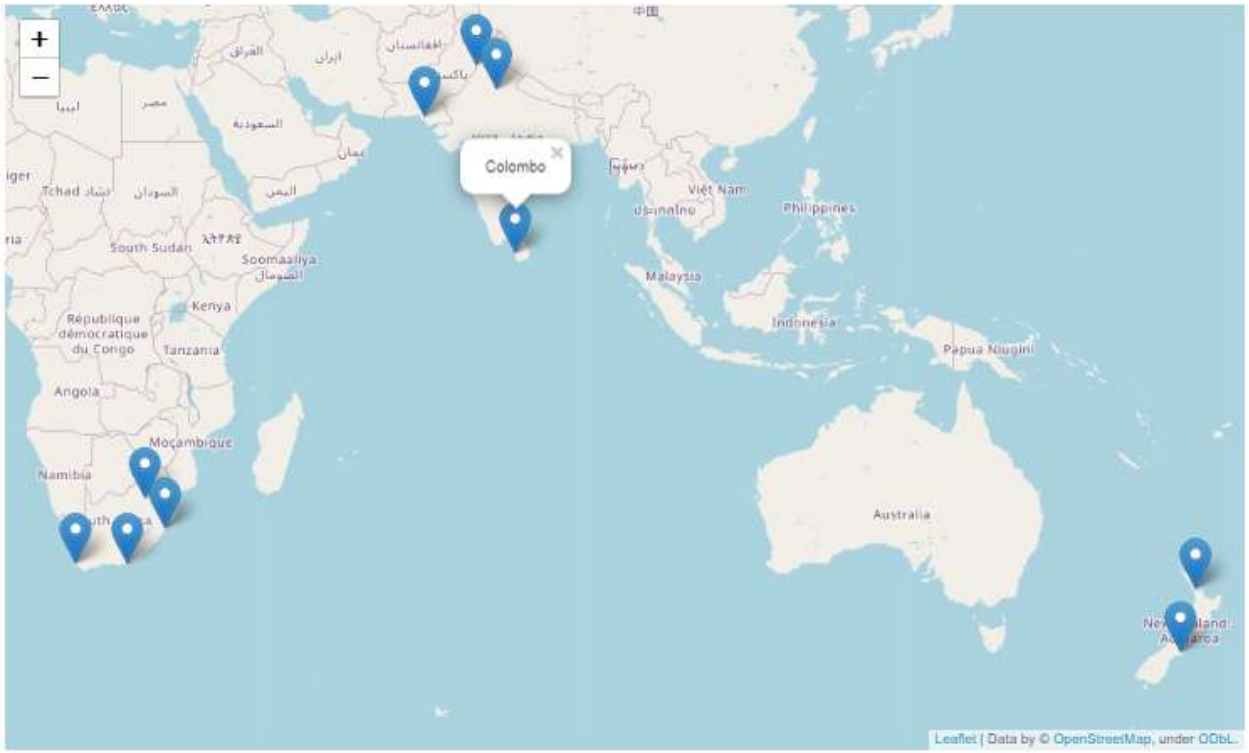
Number of players from country/city/county:

Following plot is an interactive. As you zoom in and out, the number of players will change based on the specific location.

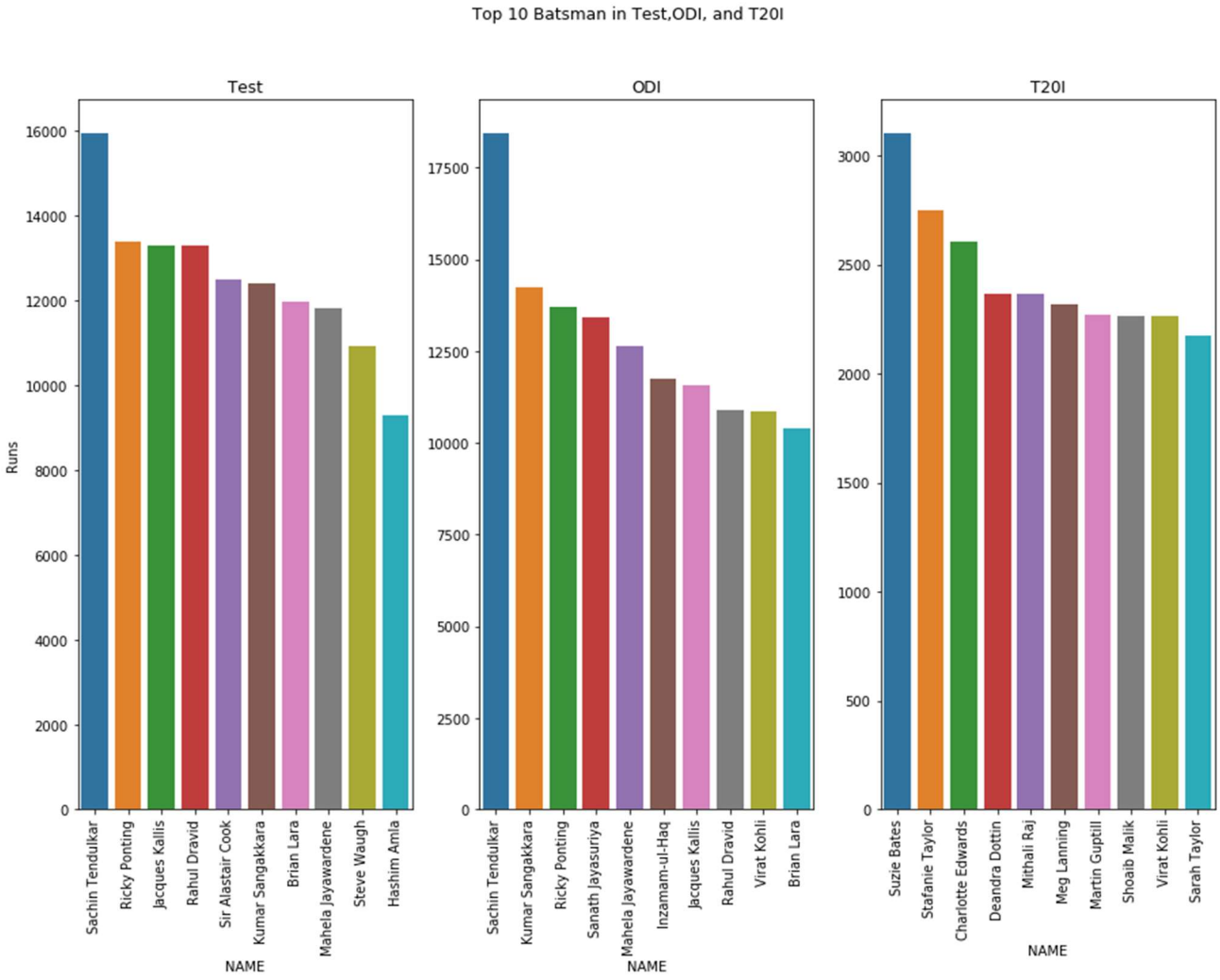
Top ten cities/counties where most players come from:



Largest group of players come from Colombo. The following plot shows where the top ten cities/counties are on the map. Clicking on marker shows the city/county name.

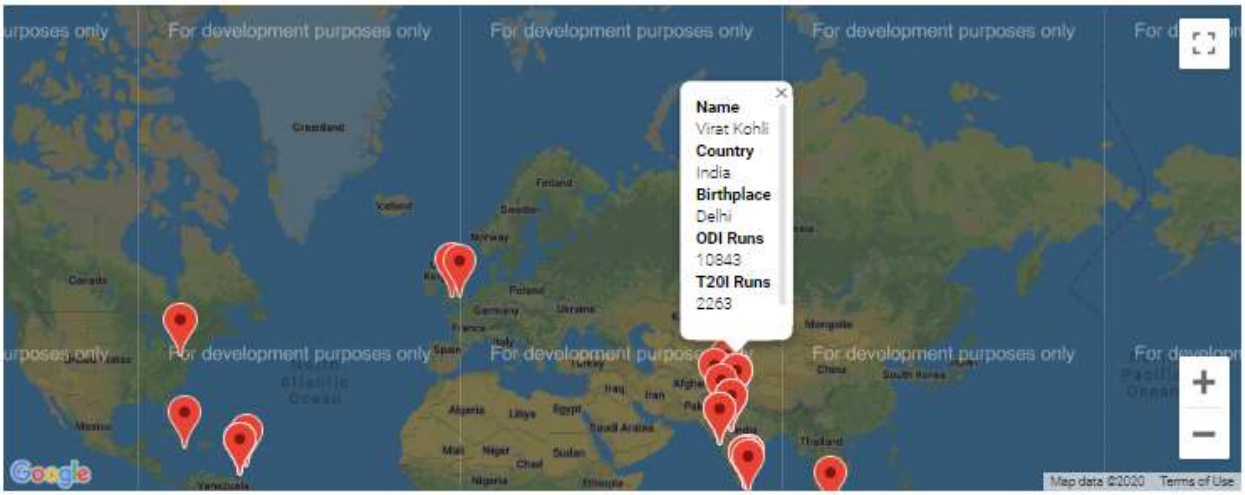


Top 10 batsmen in Test, ODI, and T20I:

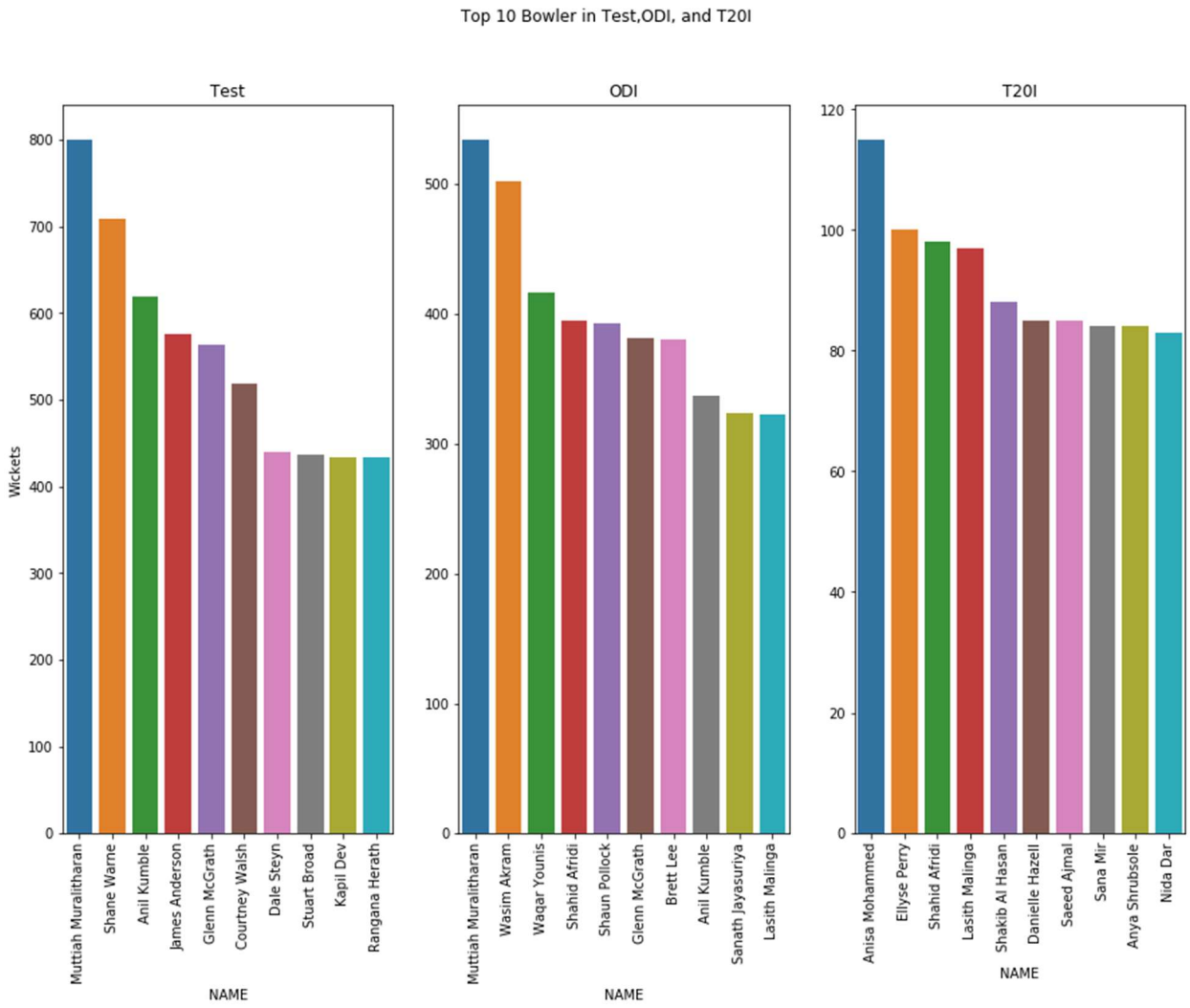


Note: Need google API key to get the following plot. Also, google map will show "for development purposes only" because google does not provide free API key now.

Following plot shows where the top 10 batsmen in Test, ODI, and T20I come from. Clicking on the marker shows information about player.



Top 10 Bowlers in Test, ODI, and T20I:



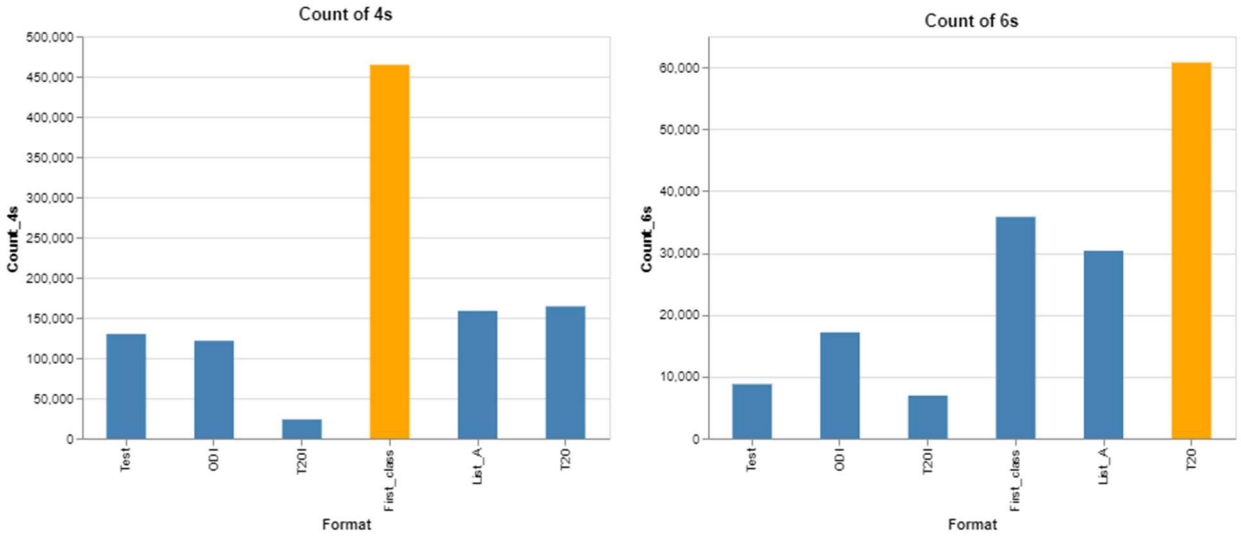
Top 10 batsmen and bowlers in Test, ODI, and T20:

The following plot shows where the top 10 bastmen and bowlers in Test,ODI, and T20I come from. Cilcking on the marker shows the player name. The mouse hover shows if the player is batsman or bowler.

Another view of 10 bastmen and bowlers in Test,ODI, and T20I:

The following output is acquire by HTML and JavaScript code. The output is interactive globe that shows top 10 batsmen and bowlers in Test, ODI, and T20 locations. Cliking on the marker shows the player name.

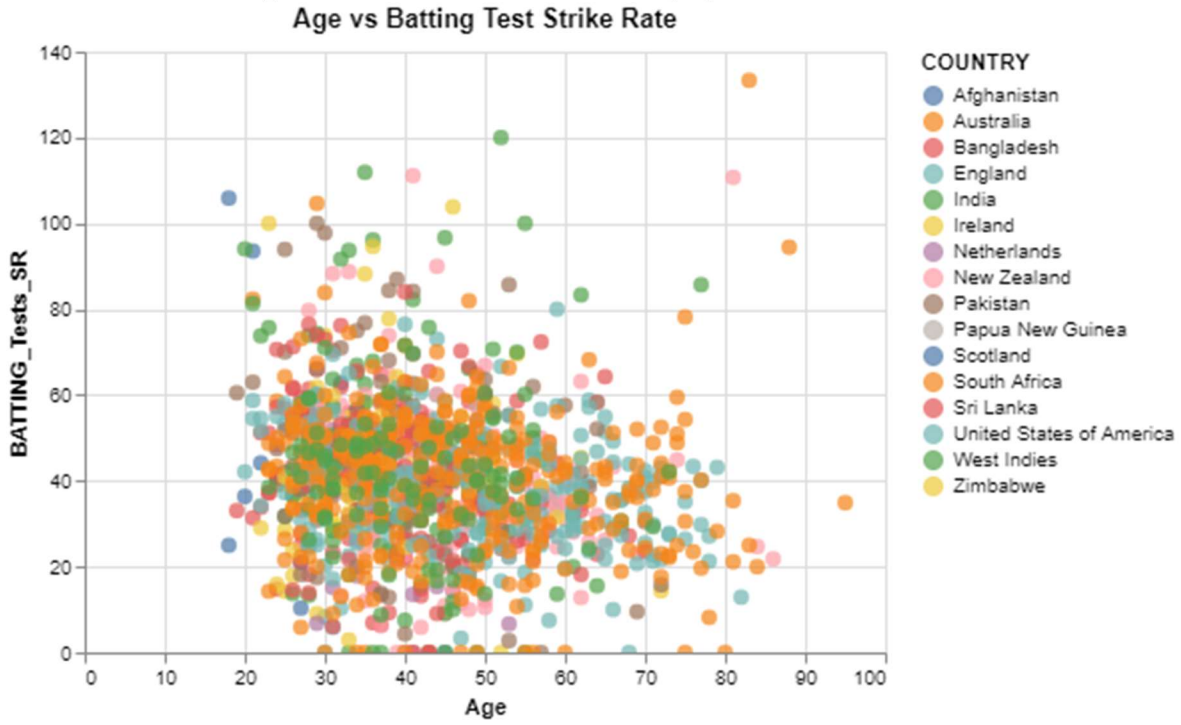
Count of 4s and 6s in each cricket format (Test, ODI, T20I, First_class, List_A, T20):



Highest number of 4s is in First_class and 6s in T20 cricket format.

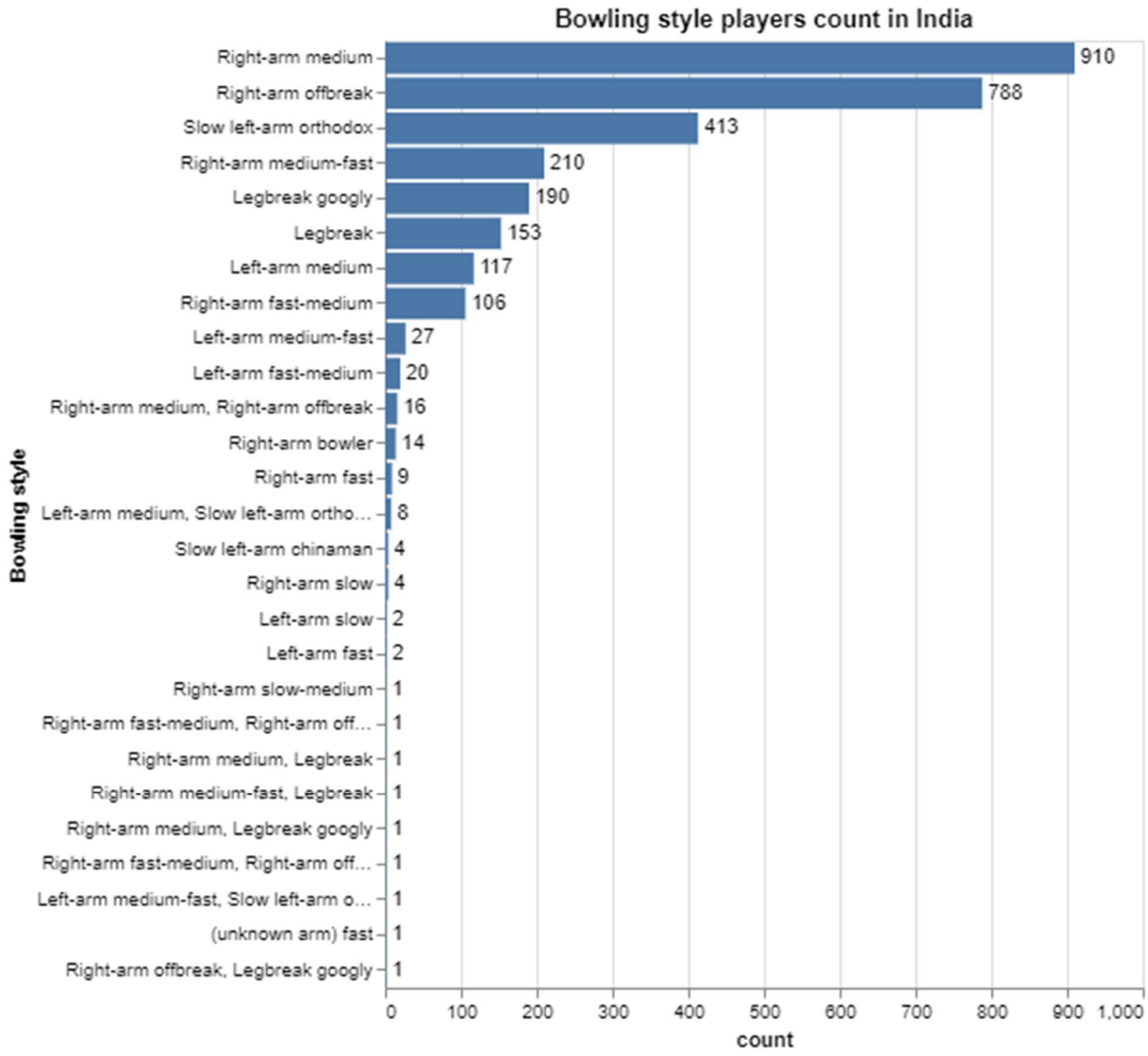
Age vs Batting Test Strike Rate (Test Run/Ball faced in Test):

The mouse hover on point shows information about players.

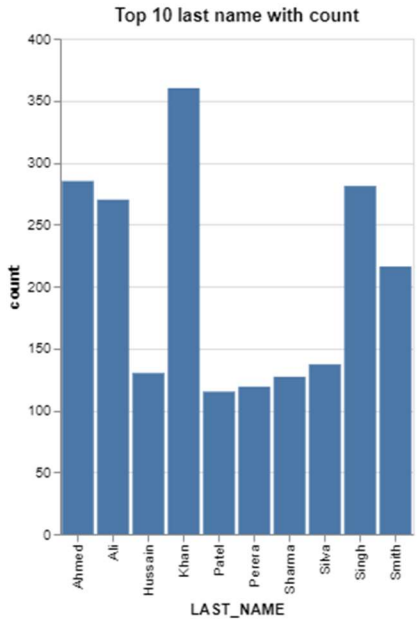


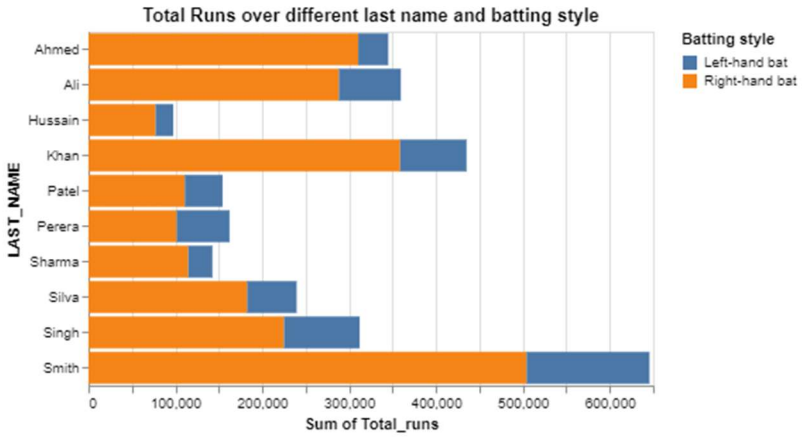
It seems like when age increases, the batting test strike rate decreases.

Bowling style players count in India:



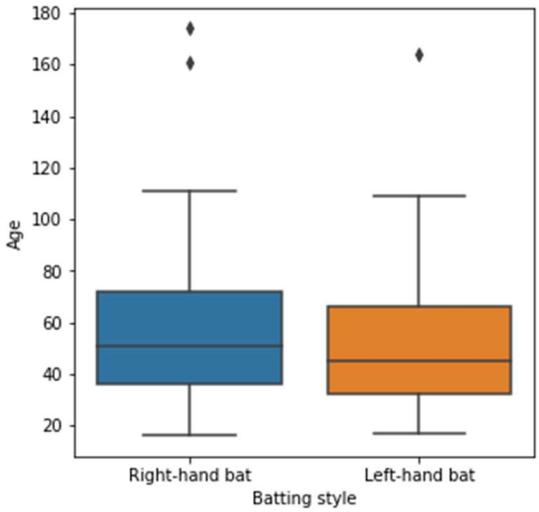
Total Runs of all format over top 10 players last name and batting style:





Highest number of last names is 'Khan'; however, players with the last name 'Smith' have more runs than 'Khan's'.

Age of players for various Batting style:



From above plot we can say that 50% of right-handed batsmen of age 50; however, 50% left-handed batsmen are younger than 50 years old.

Batting style vs Played_IPL status:

Given the type of batting style, probability of IPL status.

Played_IPL	Not Play	Play
Batting style		
Left-hand bat	0.987550	0.012450
Right-hand bat	0.993505	0.006495
All	0.992462	0.007538

Bowling style vs Played_IPL status:

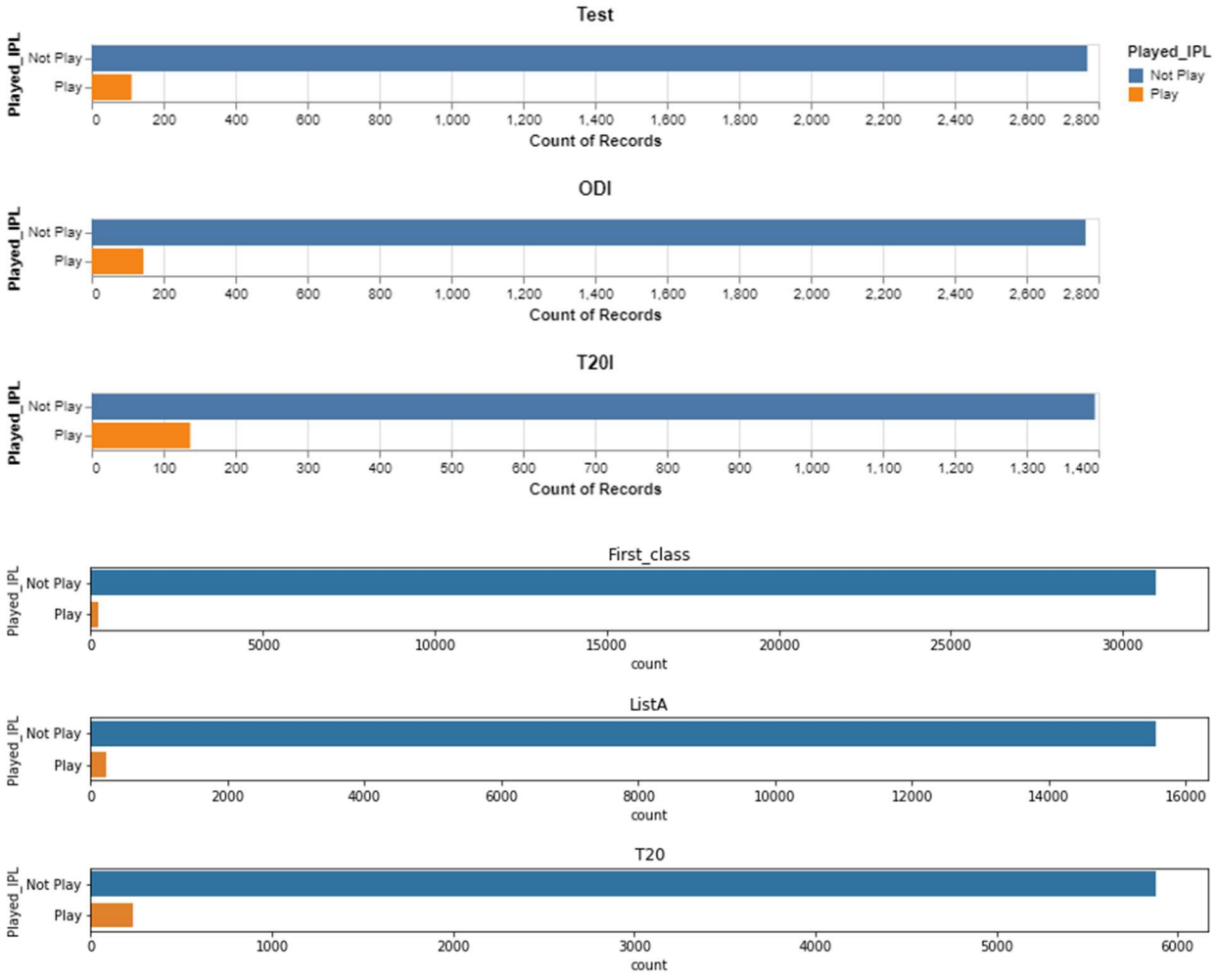
Played_IPL	Not Play	Play
Bowling style		
(roundarm)	1.000000	0.000000
(underarm)	1.000000	0.000000
(underarm), Right-arm fast	1.000000	0.000000
(underarm), Right-arm fast (roundarm)	1.000000	0.000000
(underarm), Right-arm fast-medium	1.000000	0.000000
...	...	...
Slow left-arm chinaman	0.953125	0.046875
Slow left-arm orthodox	0.991017	0.008983
Slow left-arm orthodox (roundarm)	1.000000	0.000000
Slow left-arm orthodox, Slow left-arm chinaman	1.000000	0.000000
All	0.991060	0.008940

119 rows × 2 columns

Please check out the jupyter notebook to see the entire table. The table contains 119 rows; therefore, cannot show it here. From the above table, we can say that “Right-arm off break, Leg break googly” blowing style players played more IPL than the rest.

Cricket format vs Played_IPL status:

	Format	IPL_Not_Played	IPL_Played
0	Test	2769	111
1	ODI	2764	144
2	T20I	1395	137
3	First_class	30945	231
4	List_A	15554	233
5	T20	5878	234

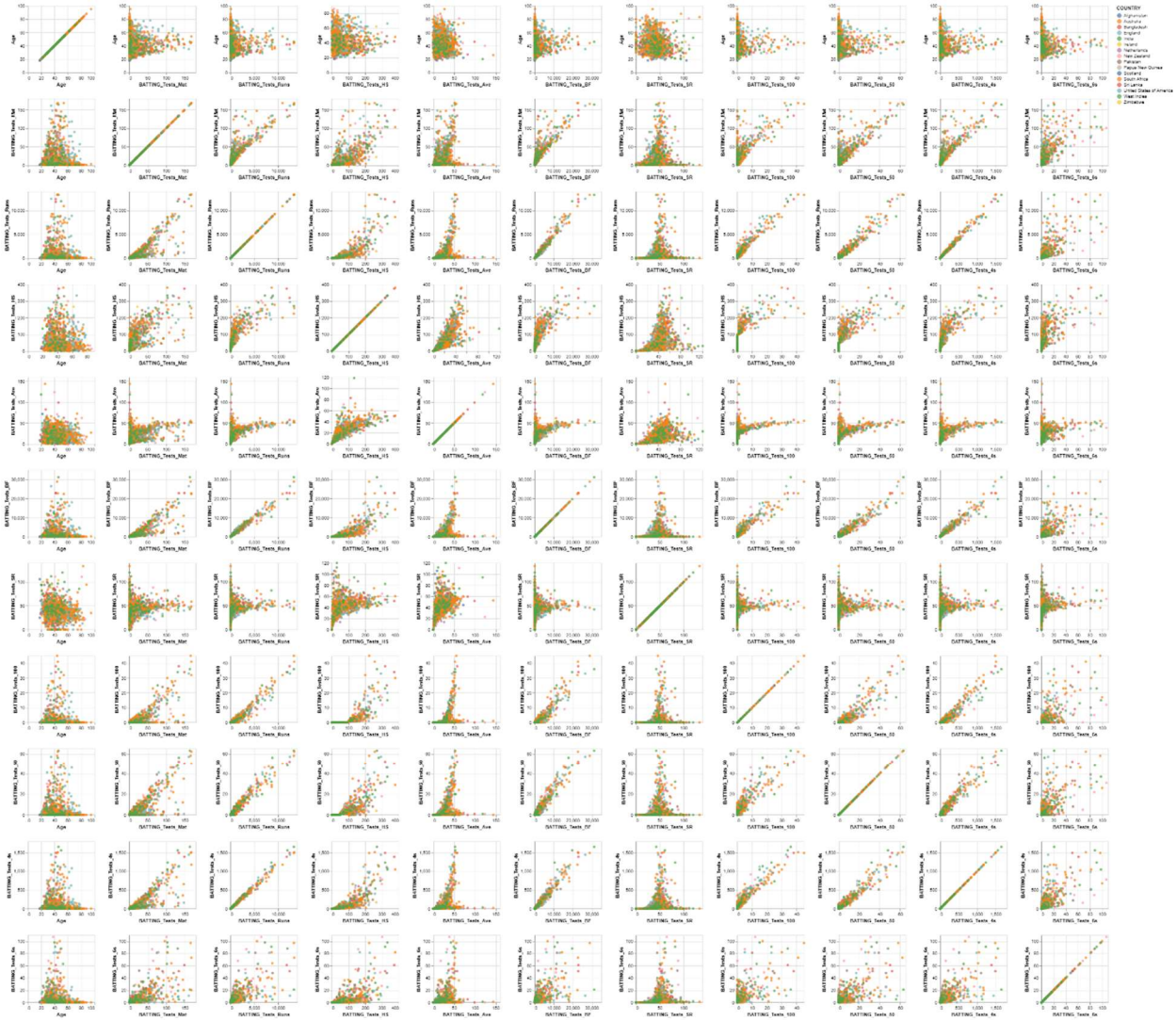


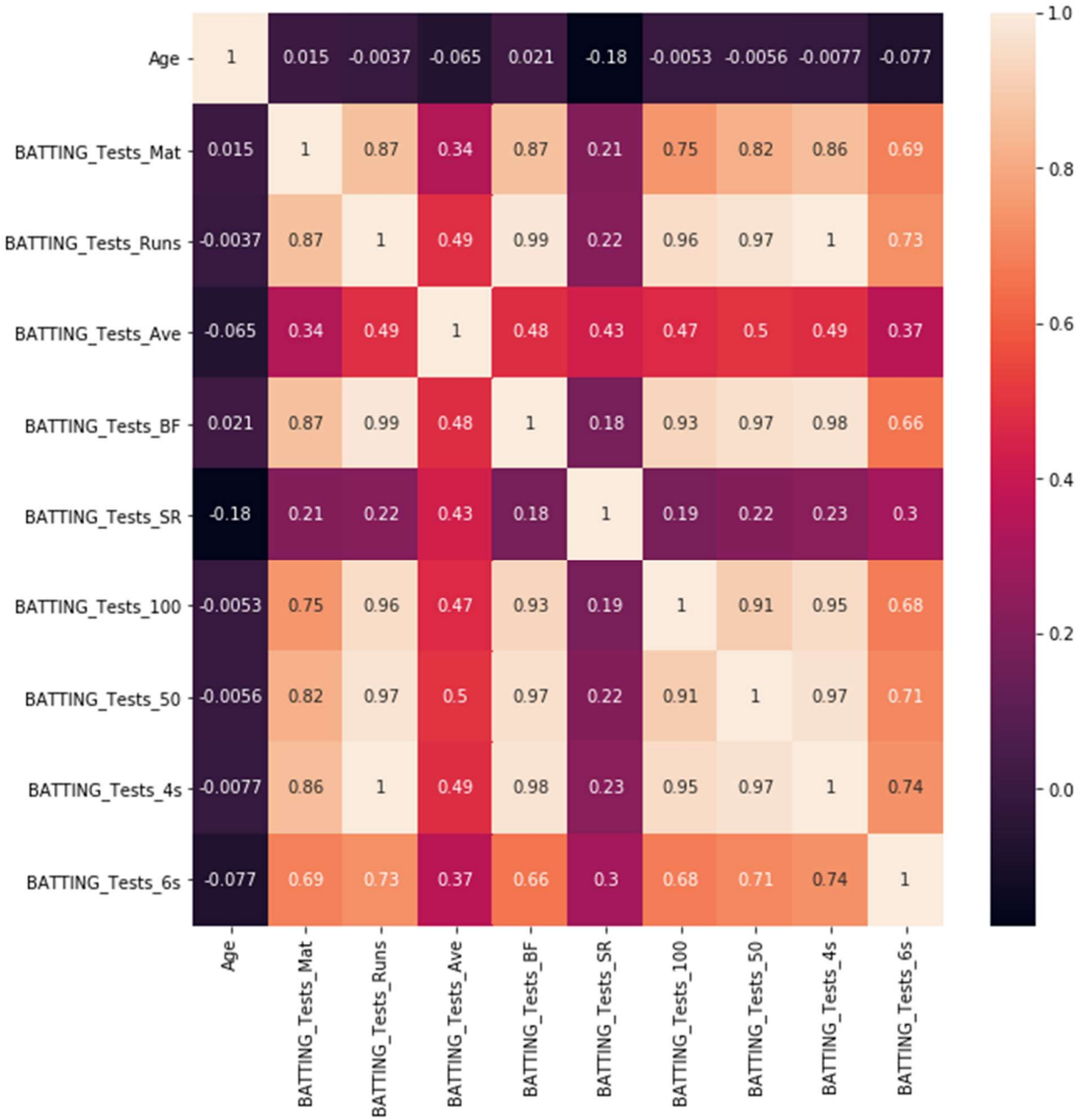
Country vs IPL status:

Second highest number of players played in IPL are from Australia. Highest number of players played in IPL are from India. Probability table is in jupyter notebook.

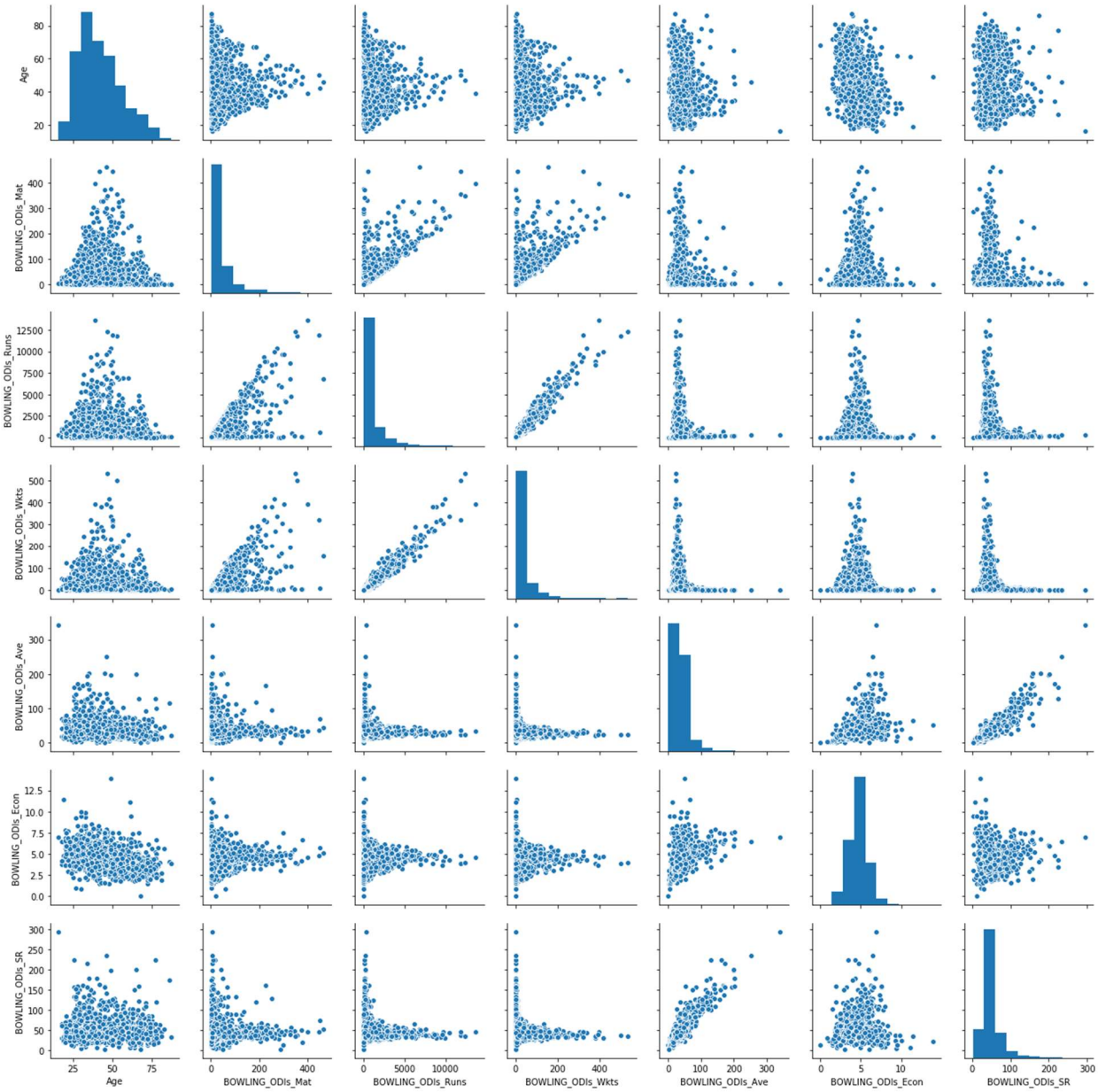
Test Batting Data Pair Plots and Correlations:

Following plot is too big to fit in here. However, the plot follow after this plot will help in interpret this plot.




Age 1 0.015 -0.0037 -0.065 0.021 -0.18 -0.0053 -0.0056 -0.0077 -0.077
BATTING_Tests_Mat 0.015 1 0.87 0.34 0.87 0.21 0.75 0.82 0.86 0.69
BATTING_Tests_Runs -0.0037 0.87 1 0.49 0.99 0.22 0.96 0.97 1 0.73
BATTING_Tests_Ave -0.065 0.34 0.49 1 0.48 0.43 0.47 0.5 0.49 0.37
BATTING_Tests_BF 0.021 0.87 0.99 0.48 1 0.18 0.93 0.97 0.98 0.66
BATTING_Tests_SR -0.18 0.21 0.22 0.43 0.18 1 0.19 0.22 0.23 0.3
BATTING_Tests_100 -0.0053 0.75 0.96 0.47 0.93 0.19 1 0.91 0.95 0.68
BATTING_Tests_50 -0.0056 0.82 0.97 0.5 0.97 0.22 0.91 1 0.97 0.71
BATTING_Tests_4s -0.0077 0.86 1 0.49 0.98 0.23 0.95 0.97 1 0.74
BATTING_Tests_6s -0.077 0.69 0.73 0.37 0.66 0.3 0.68 0.71 0.74 1
Age
BATTING_Tests_Mat
BATTING_Tests_Runs
BATTING_Tests_Ave
BATTING_Tests_BF
BATTING_Tests_SR
BATTING_Tests_100
BATTING_Tests_50
BATTING_Tests_4s
BATTING_Tests_6s
1.0
0.8
0.6
0.4
0.2
0.0

ODI bowling data Pair plots:



Model Description:

Build a model to classify IPL status to be able to know player will play IPL or not.

Input features:

COUNTRY, Died, Age, Batting style, Bowling style, BATTING_Tests_Mat, BATTING_Tests_Inns, BATTING_Tests_Runs, BATTING_Tests_HS, BATTING_Tests_Ave, BATTING_Tests_BF, BATTING_Tests_SR, BATTING_Tests_100, BATTING_Tests_50, BATTING_Tests_4s, BATTING_Tests_6s, BATTING_Tests_Ct, BATTING_Tests_St, BATTING_ODIs_Mat, BATTING_ODIs_Inns, BATTING_ODIs_Runs, BATTING_ODIs_HS, BATTING_ODIs_Ave, BATTING_ODIs_BF, BATTING_ODIs_SR, BATTING_ODIs_100, BATTING_ODIs_50, BATTING_ODIs_4s, BATTING_ODIs_6s, BATTING_ODIs_Ct, BATTING_ODIs_St, BATTING_T20Is_Mat, BATTING_T20Is_Inns, BATTING_T20Is_Runs, BATTING_T20Is_HS, BATTING_T20Is_Ave, BATTING_T20Is_BF, BATTING_T20Is_SR, BATTING_T20Is_100, BATTING_T20Is_50, BATTING_T20Is_4s, BATTING_T20Is_6s, BATTING_T20Is_Ct, BATTING_T20Is_St, BATTING_First-class_Mat, BATTING_First-class_Inns, BATTING_First-class_Runs, BATTING_First-class_HS, BATTING_First-class_Ave, BATTING_First-class_100, BATTING_First-class_50, BATTING_First-class_Ct, BATTING_First-class_St, BATTING_List A_Mat, BATTING_List A_Inns, BATTING_List A_Runs, BATTING_List A_HS, BATTING_List A_Ave, BATTING_List A_100, BATTING_List A_50, BATTING_List A_Ct, BATTING_List A_St, BATTING_T20s_Mat, BATTING_T20s_Inns, BATTING_T20s_Runs, BATTING_T20s_HS, BATTING_T20s_Ave, BATTING_T20s_BF, BATTING_T20s_SR, BATTING_T20s_100, BATTING_T20s_50, BATTING_T20s_4s, BATTING_T20s_6s, BATTING_T20s_Ct, BATTING_T20s_St, BOWLING_Tests_Mat, BOWLING_Tests_Inns, BOWLING_Tests_Balls, BOWLING_Tests_Runs, BOWLING_Tests_Wkts, BOWLING_Tests_Ave, BOWLING_Tests_Econ, BOWLING_Tests_SR, BOWLING_Tests_4w, BOWLING_Tests_5w, BOWLING_Tests_10, BOWLING_ODIs_Mat, BOWLING_ODIs_Inns, BOWLING_ODIs_Balls, BOWLING_ODIs_Runs, BOWLING_ODIs_Wkts, BOWLING_ODIs_Ave, BOWLING_ODIs_Econ, BOWLING_ODIs_SR, BOWLING_ODIs_4w, BOWLING_ODIs_5w, BOWLING_T20Is_Mat, BOWLING_T20Is_Inns, BOWLING_T20Is_Balls, BOWLING_T20Is_Runs, BOWLING_T20Is_Wkts, BOWLING_T20Is_Ave, BOWLING_T20Is_Econ, BOWLING_T20Is_SR, BOWLING_T20Is_4w, BOWLING_T20Is_5w, BOWLING_First-class_Mat, BOWLING_First-class_Balls, BOWLING_First-class_Runs, BOWLING_First-class_Wkts, BOWLING_First-class_Ave, BOWLING_First-class_Econ, BOWLING_First-class_5w, BOWLING_First-class_10, BOWLING_List A_Mat, BOWLING_List A_Balls, BOWLING_List A_Runs, BOWLING_List A_Wkts, BOWLING_List A_Ave, BOWLING_List A_Econ, BOWLING_List A_SR, BOWLING_List A_4w, BOWLING_List A_5w, BOWLING_T20s_Mat, BOWLING_T20s_Inns, BOWLING_T20s_Balls, BOWLING_T20s_Runs, BOWLING_T20s_Wkts, BOWLING_T20s_Ave, BOWLING_T20s_Econ, BOWLING_T20s_SR, BOWLING_T20s_4w, BOWLING_T20s_5w

Output variable:

Played_IPL

Data Splitting:

First, split the dataset into Train_intial and Test having ratio of 80-20. Then split Train_intial dataset into Train and Validation with ratio of 75-25.

Machine Learning Techniques for identification:

1. LogisticRegression
2. Support Vector Classification
3. K-nearest neighbors

4. Decision Tree
5. Random Forest
6. Stochastic Gradient Descent
7. Gaussian Processes classification
8. Naive Bayes
9. Adaboost
10. Gradient Boosting
11. Histogram-Based Gradient Boosting
12. Neural network models

Machine Learning:

I have tried the following machine learning algorithms to be able to determine which algorithm works best for cricket dataset.

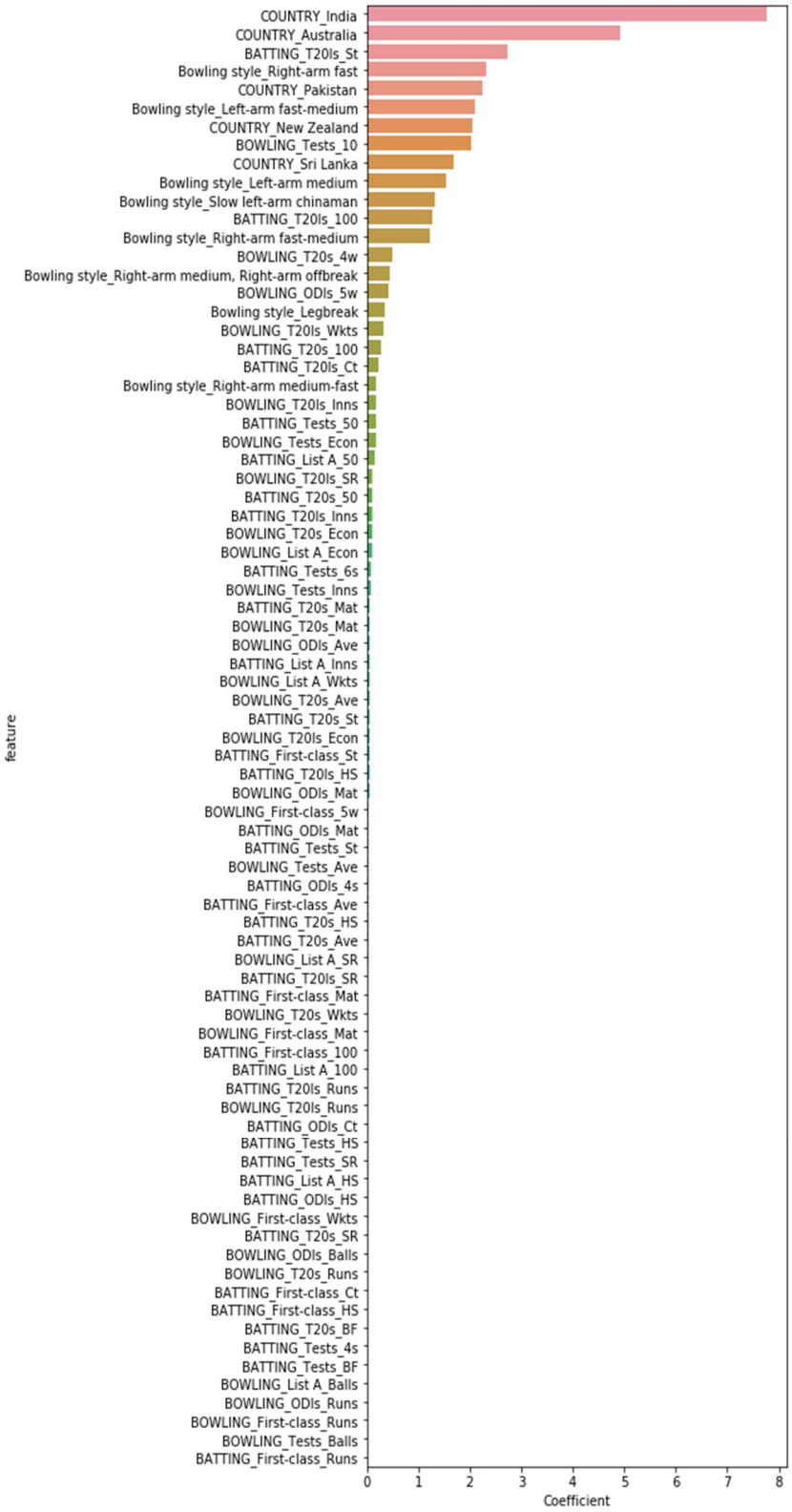
LogisticRegression

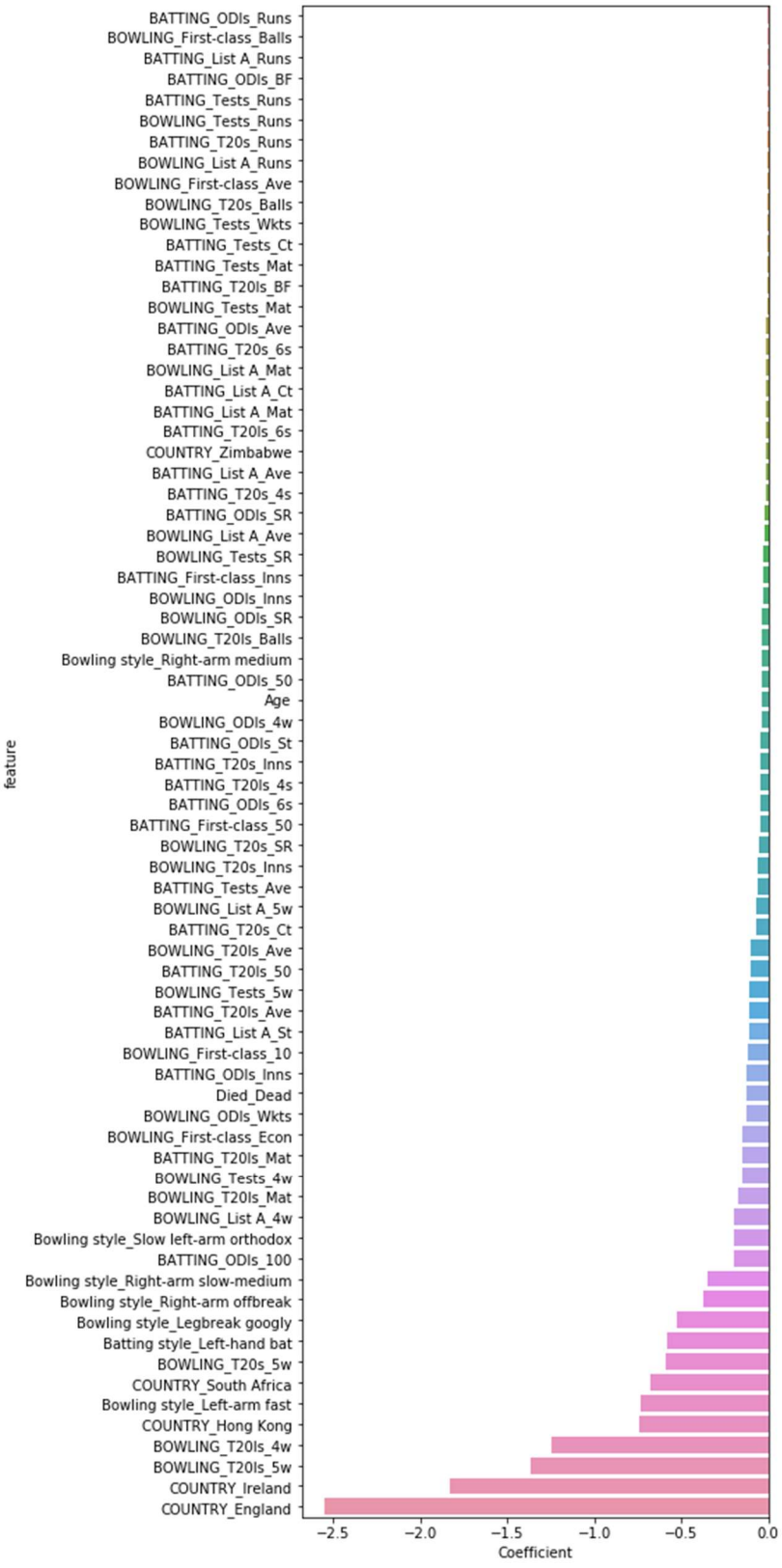
I used ‘l1’ regularization that shrink less important coefficient to zero. That would help us to identify the important features.

Parameters used:

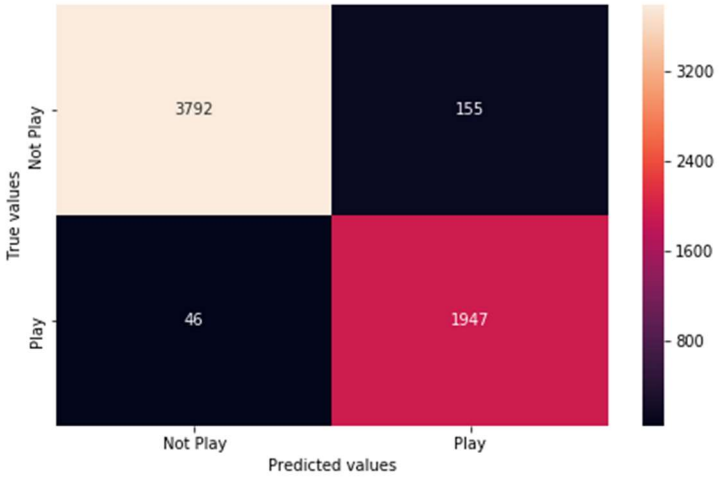
C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, l1_ratio=None, max_iter=100, multi_class='warn', n_jobs=None, penalty='l1', random_state=None, solver='liblinear', tol=0.0001, verbose=0, warm_start=False

Coefficients Visualization:


COUNTRY_India
COUNTRY_Australia
BATTING_T20Is_St
Bowling style_Right-arm fast
COUNTRY_Pakistan
Bowling style_Left-arm fast-medium
COUNTRY_New Zealand
BOWLING_Tests_10
COUNTRY_Sri Lanka
Bowling style_Left-arm medium
Bowling style_Slow left-arm chinaman
BATTING_T20Is_100
Bowling style_Right-arm fast-medium
BOWLING_T20s_4w
Bowling style_Right-arm medium, Right-arm offbreak
BOWLING_ODIs_5w
Bowling style_Legbreak
BOWLING_T20Is_Wkts
BATTING_T20s_100
BATTING_T20Is_Ct
Bowling style_Right-arm medium-fast
BOWLING_T20Is_Inns
BATTING_Tests_50
BOWLING_Tests_Econ
BATTING_List A_50
BOWLING_T20Is_SR
BATTING_T20s_50
BATTING_T20Is_Inns
BOWLING_T20s_Econ
BOWLING_List A_Econ
BATTING_Tests_6s
BOWLING_Tests_Inns
BATTING_T20s_Mat
BOWLING_T20s_Mat
BOWLING_ODIs_Ave
BATTING_List A_Inns
BOWLING_List A_Wkts
BOWLING_T20s_Ave
BATTING_T20s_St
BOWLING_T20Is_Econ
BATTING_First-class_St
BATTING_T20Is_HS
BOWLING_ODIs_Mat
BOWLING_First-class_5w
BATTING_ODIs_Mat
BATTING_Tests_St
BOWLING_Tests_Ave
BATTING_ODIs_4s
BATTING_First-class_Ave
BATTING_T20s_HS
BATTING_T20s_Ave
BOWLING_List A_SR
BATTING_T20Is_SR
BATTING_First-class_Mat
BOWLING_T20s_Wkts
BOWLING_First-class_Mat
BATTING_First-class_100
BATTING_List A_100
BATTING_T20Is_Runs
BOWLING_T20Is_Runs
BATTING_ODIs_Ct
BATTING_Tests_HS
BATTING_Tests_SR
BATTING_List A_HS
BATTING_ODIs_HS
BOWLING_First-class_Wkts
BATTING_T20s_SR
BOWLING_ODIs_Balls
BOWLING_T20s_Runs
BATTING_First-class_Ct
BATTING_First-class_HS
BATTING_T20s_BF
BATTING_Tests_4s
BATTING_Tests_BF
BOWLING_List A_Balls
BOWLING_ODIs_Runs
BOWLING_First-class_Runs
BOWLING_Tests_Balls
BATTING_First-class_Runs
feature
0
1
2
3
4
5
6
7
8
Coefficient


BATTING_ODIs_Runs
BOWLING_First-class_Balls
BATTING_List A_Runs
BATTING_ODIs_BF
BATTING_Tests_Runs
BOWLING_Tests_Runs
BATTING_T20s_Runs
BOWLING_List A_Runs
BOWLING_First-class_Ave
BOWLING_T20s_Balls
BOWLING_Tests_Wkts
BATTING_Tests_Ct
BATTING_Tests_Mat
BATTING_T20Is_BF
BOWLING_Tests_Mat
BATTING_ODIs_Ave
BATTING_T20s_6s
BOWLING_List A_Mat
BATTING_List A_Ct
BATTING_List A_Mat
BATTING_T20Is_6s
COUNTRY_Zimbabwe
BATTING_List A_Ave
BATTING_T20s_4s
BATTING_ODIs_SR
BOWLING_List A_Ave
BOWLING_Tests_SR
BATTING_First-class_Inns
BOWLING_ODIs_Inns
BOWLING_ODIs_SR
BOWLING_T20Is_Balls
Bowling style_Right-arm medium
BATTING_ODIs_50
Age
BOWLING_ODIs_4w
BATTING_ODIs_St
BATTING_T20s_Inns
BATTING_T20Is_4s
BATTING_ODIs_6s
BATTING_First-class_50
BOWLING_T20s_SR
BOWLING_T20s_Inns
BATTING_Tests_Ave
BOWLING_List A_5w
BATTING_T20s_Ct
BOWLING_T20Is_Ave
BATTING_T20Is_50
BOWLING_Tests_5w
BATTING_T20Is_Ave
BATTING_List A_St
BOWLING_First-class_10
BATTING_ODIs_Inns
Died_Dead
BOWLING_ODIs_Wkts
BOWLING_First-class_Econ
BATTING_T20Is_Mat
BOWLING_Tests_4w
BOWLING_T20Is_Mat
BOWLING_List A_4w
Bowling style_Slow left-arm orthodox
BATTING_ODIs_100
Bowling style_Right-arm slow-medium
Bowling style_Right-arm offbreak
Bowling style_Legbreak googly
Batting style_Left-hand bat
BOWLING_T20s_5w
COUNTRY_South Africa
Bowling style_Left-arm fast
COUNTRY_Hong Kong
BOWLING_T20Is_4w
BOWLING_T20Is_5w
COUNTRY_Ireland
COUNTRY_England
feature
−2.5
−2.0
−1.5
−1.0
−0.5
0.0
Coefficient

Confusion Matrix:

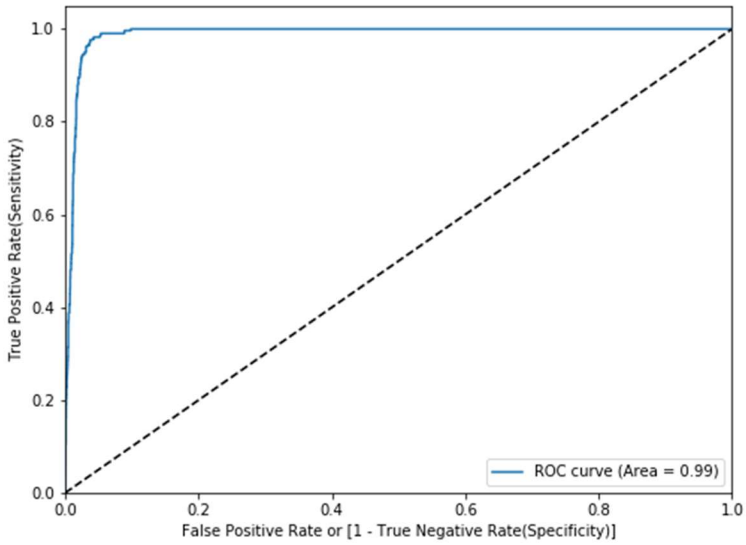



```
Misclassified samples: 201
Accuracy: 0.9661616161616161
```

Classification Report:

	precision	recall	f1-score	support
0	0.99	0.96	0.97	3947
1	0.93	0.98	0.95	1993
accuracy			0.97	5940
macro avg	0.96	0.97	0.96	5940
weighted avg	0.97	0.97	0.97	5940

Roc_Auc plot:



To be able to identify how well this algorithm works, I used 10-folds cross validation based on area under ROC curve. The following is the output:

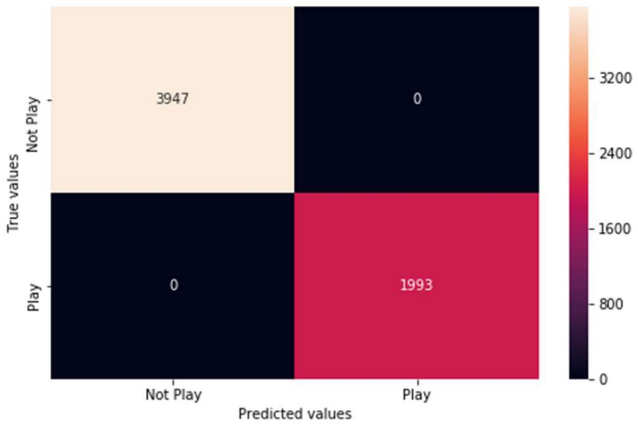
```
[0.99231221, 0.98811786, 0.99060101, 0.9866174, 0.99135716,
0.98788423, 0.99324516, 0.99362324, 0.9902317, 0.98809871]
```

Support Vector Classification

Parameters used:

C=1.0, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False

Confusion Matrix:




```
Misclassified samples: 0
Accuracy: 1.0
```

Classification Report:

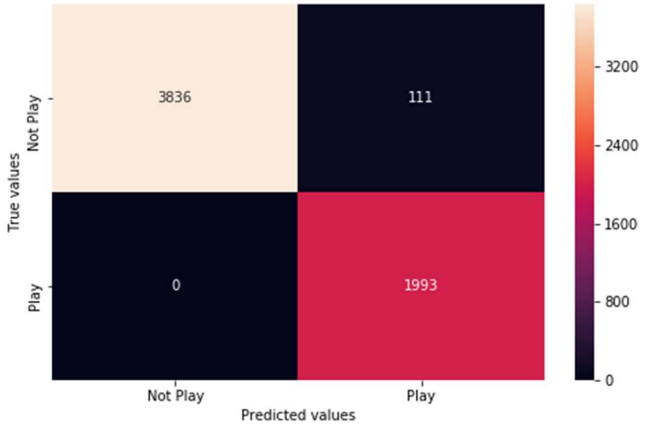
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      3947
           1       1.00      1.00      1.00      1993

    accuracy                           1.00      5940
   macro avg       1.00      1.00      1.00      5940
weighted avg       1.00      1.00      1.00      5940
```

K- -nearest neighbors

Parameters used:

> algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=None, n_neighbors=5, p=2, weights='uniform'

Confusion Matrix:

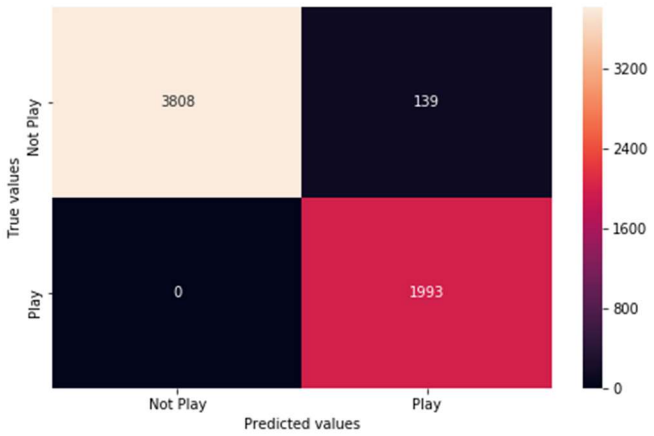


I used RandomizedSearchCV with 10-folds cross validation to be able to find most optimal parameters in K-nearest neighbor algorithm. I hyper tuned parameters base on area under ROC curve. The parameters I turned are n_neighbors and metric.
n_neighbors = 1 to 15
metric = ['canberra', 'euclidean', 'minkowski']

The following are the best parameters setting that acquire largest are under ROC curve:

> algorithm='auto', leaf_size=30, metric='canberra', metric_params=None, n_jobs=None, n_neighbors=5, p=2, weights='uniform'

Confusion matrix using optimal parameters:

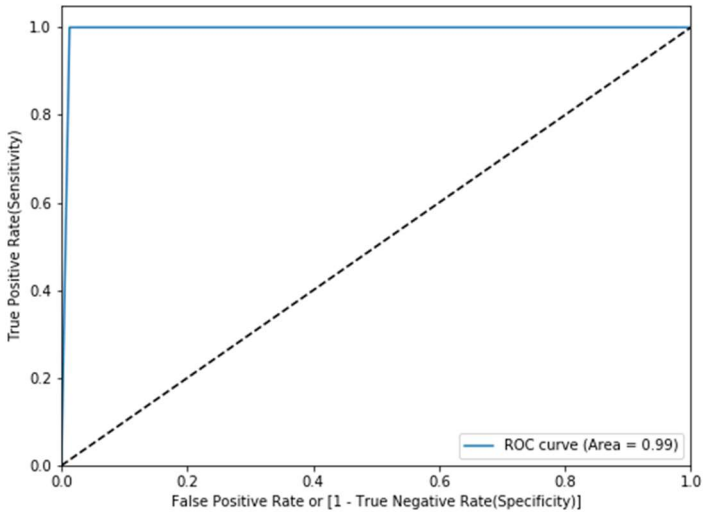



```
Misclassified samples: 139
Accuracy: 0.9765993265993266
```

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.96	0.98	3947
1	0.93	1.00	0.97	1993
accuracy			0.98	5940
macro avg	0.97	0.98	0.97	5940
weighted avg	0.98	0.98	0.98	5940

Roc_Auc plot:

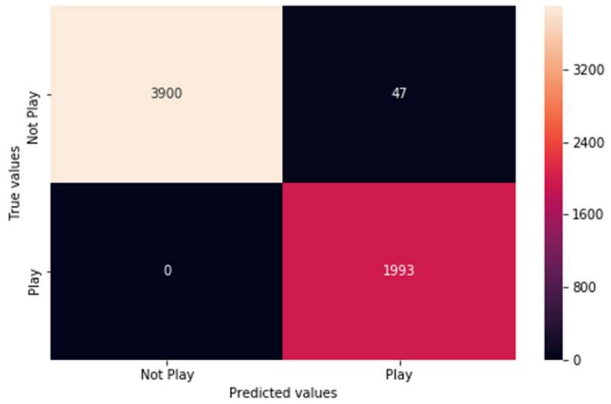


Decision Tree

Parameters used:

> class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'

Confusion Matrix:

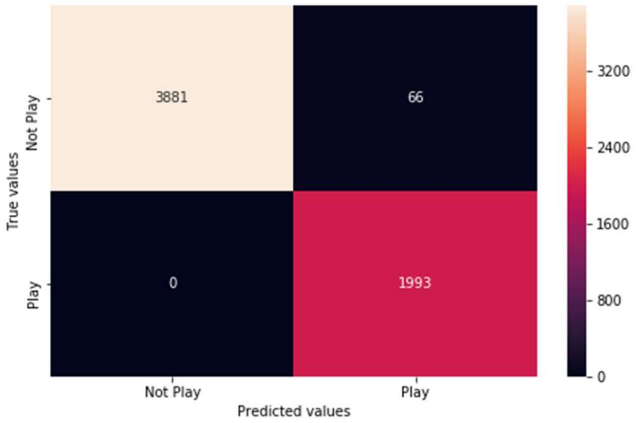


I used RandomizedSearchCV and GridSeachCv with 10-folds cross validation to be able to find most optimal parameters. The parameters I turned are max_depth and criterion base on area under ROC curve.
max_depth = 1 to 15
criterion= ['gini', 'entropy']

The following are the best parameters setting that acquire largest are under ROC curve using GridSearchCV:

> class_weight=None, criterion='entropy', max_depth=13, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'

Confusion matrix using optimal parameters:

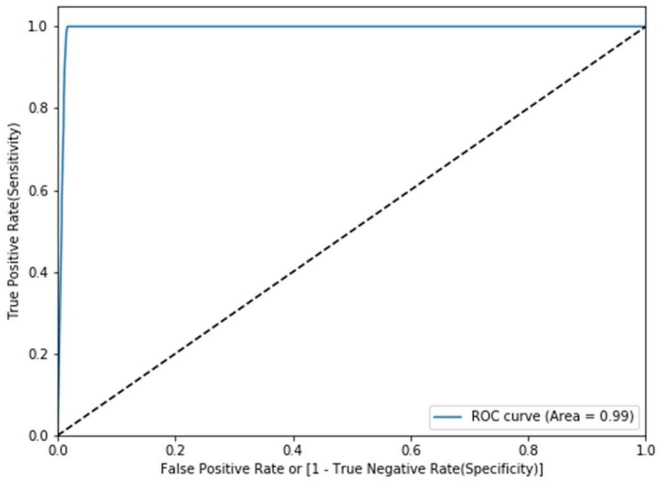



```
Misclassified samples: 66
Accuracy: 0.9888888888888889
```

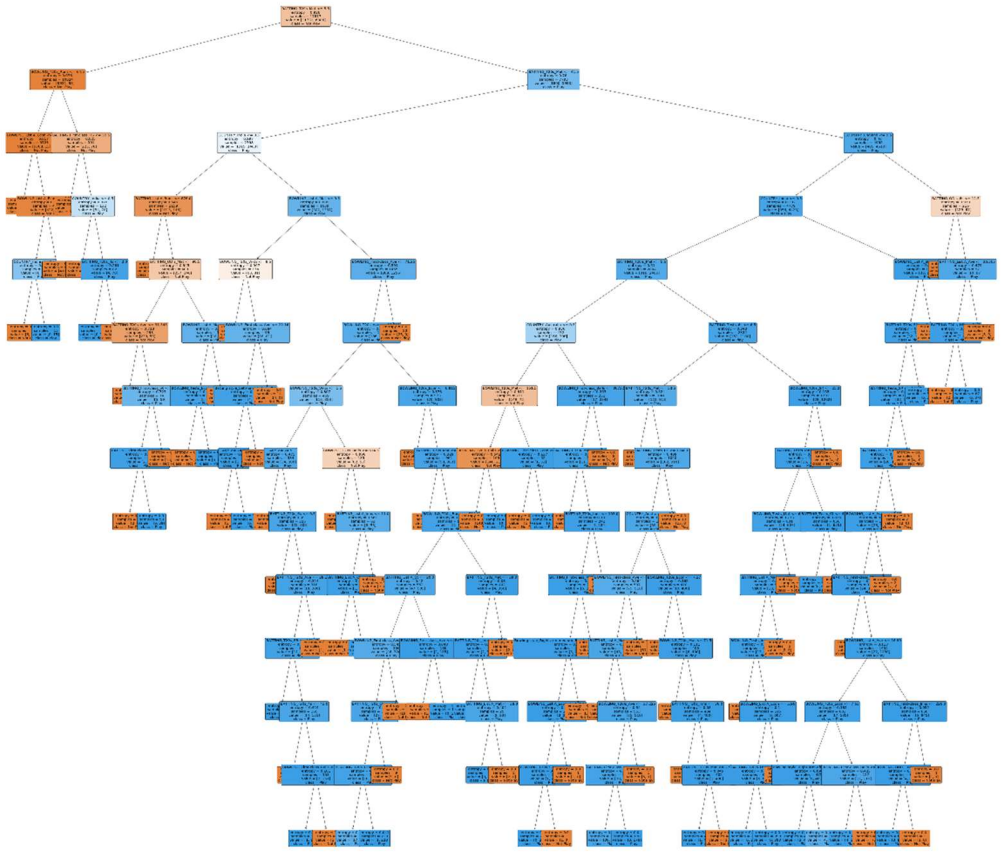
Classification Report:

	precision	recall	f1-score	support
0	1.00	0.98	0.99	3947
1	0.97	1.00	0.98	1993
accuracy			0.99	5940
macro avg	0.98	0.99	0.99	5940
weighted avg	0.99	0.99	0.99	5940

Roc_Auc plot:



Decision tree visualization (tree too big to fit in here):

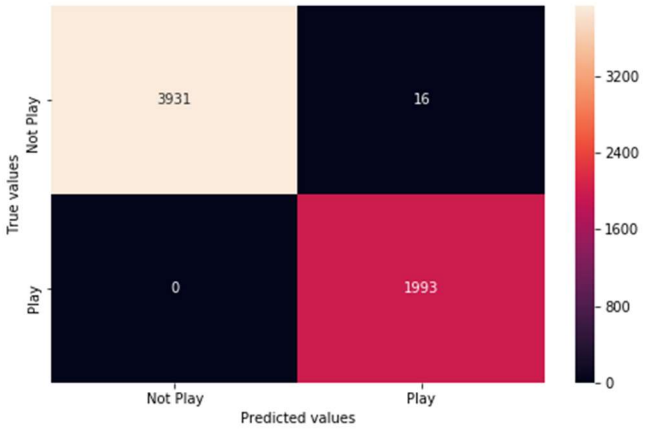


Random Forest

Parameters used:

bootstrap=True, class_weight=None, criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None, oob_score=False, random_state=None, verbose=0, warm_start=False

Confusion Matrix:

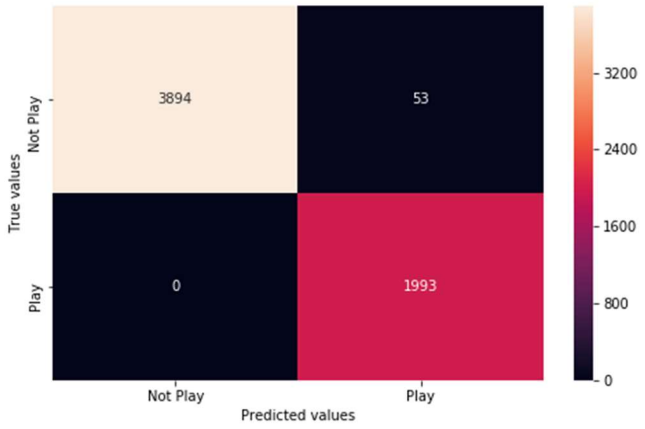


I used RandomizedSearchCV with 10-folds cross validation to be able to find most optimal parameters. The parameters I turned are max_depth and criterion base on area under ROC curve.
max_depth = 1 to 15
n_estimators= 10 to 20

The following are the best parameters setting that acquire largest are under ROC curve using RandomizedSearchCV:

bootstrap=True, class_weight=None, criterion='gini', max_depth=14, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=11, n_jobs=None, oob_score=False, random_state=None, verbose=0, warm_start=False

Confusion matrix using optimal parameters:

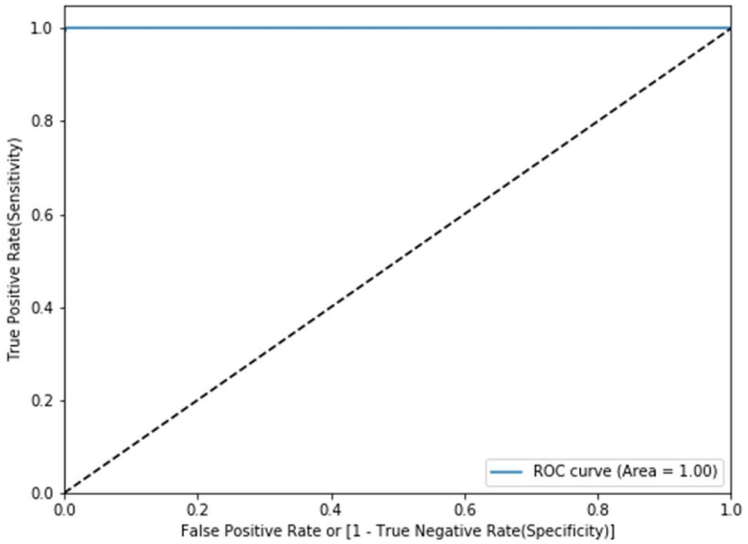


```
Misclassified samples: 53
Accuracy: 0.9910774410774411
```

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.99	0.99	3947
1	0.97	1.00	0.99	1993
accuracy			0.99	5940
macro avg	0.99	0.99	0.99	5940
weighted avg	0.99	0.99	0.99	5940

Roc_Auc plot:

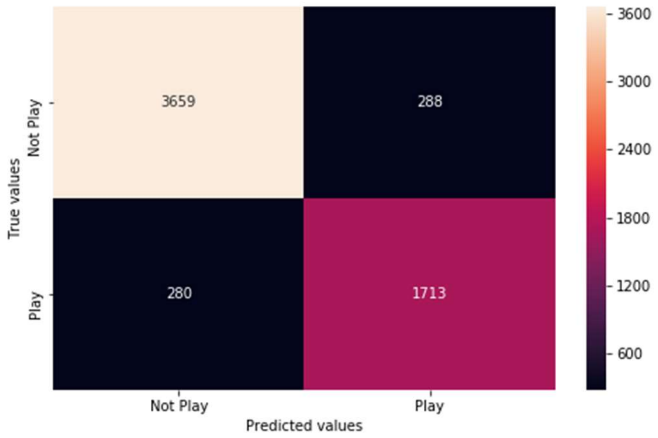


Stochastic Gradient Descent

Parameters used:

alpha=0.0001, average=False, class_weight=None, early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True, l1_ratio=0.15, learning_rate='optimal', loss='log', max_iter=1000, n_iter_no_change=5, n_jobs=None, penalty='l2', power_t=0.5, random_state=None, shuffle=True, tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False

Confusion Matrix:

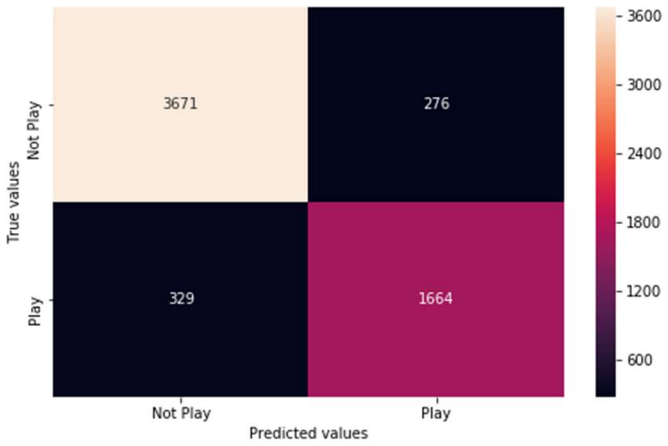


I used GridSearchCV with 10-folds cross validation to be able to find most optimal parameters. The parameters I turned are loss and penalty base on area under ROC curve.
Loss = ['hinge', 'log', 'modified_huber', 'squared_hinge']
Penalty = ['l2','l1']

The following are the best parameters setting that acquire largest are under ROC curve using GridSearchCV:

> alpha=0.0001, average=False, class_weight=None, early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True, l1_ratio=0.15, learning_rate='optimal', loss='modified_huber', max_iter=1000, n_iter_no_change=5, n_jobs=None, penalty='l1', power_t=0.5, random_state=None, shuffle=True, tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False

Confusion matrix using optimal parameters:

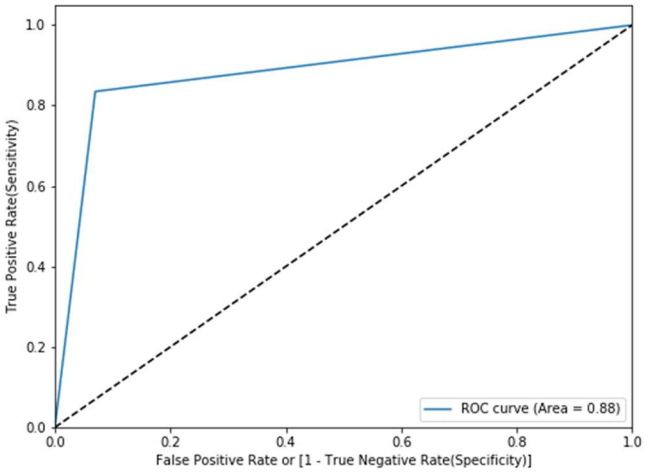


```
Misclassified samples: 605
Accuracy: 0.8981481481481481
```

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.93	0.92	3947
1	0.86	0.83	0.85	1993
accuracy			0.90	5940
macro avg	0.89	0.88	0.89	5940
weighted avg	0.90	0.90	0.90	5940

Roc_Auc plot:



Gaussian Processes classification

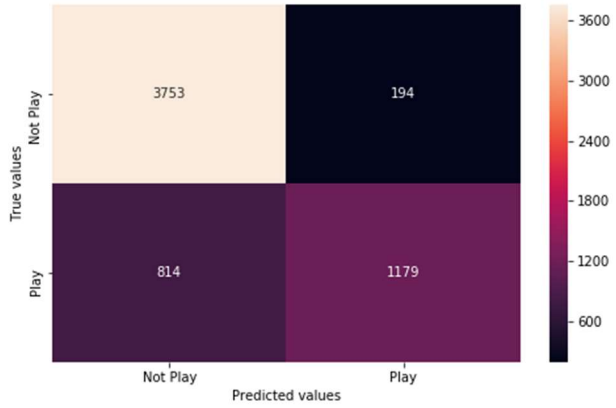
Gaussian processes classifier took to long to fit train data.

Naive Bayes

Parameters used:

priors=None, var_smoothing=1e-09

Confusion Matrix:

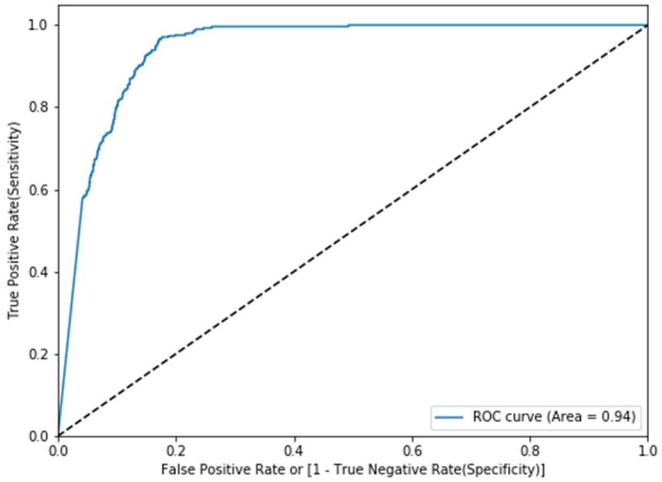



```
Misclassified samples: 1008
Accuracy: 0.8303030303030303
```

Classification Report:

	precision	recall	f1-score	support
0	0.82	0.95	0.88	3947
1	0.86	0.59	0.70	1993
accuracy			0.83	5940
macro avg	0.84	0.77	0.79	5940
weighted avg	0.83	0.83	0.82	5940

Roc_Auc plot:



To be able to identify how well this algorithm works, I used 10-folds cross validation based on area under ROC curve. The following is the output:

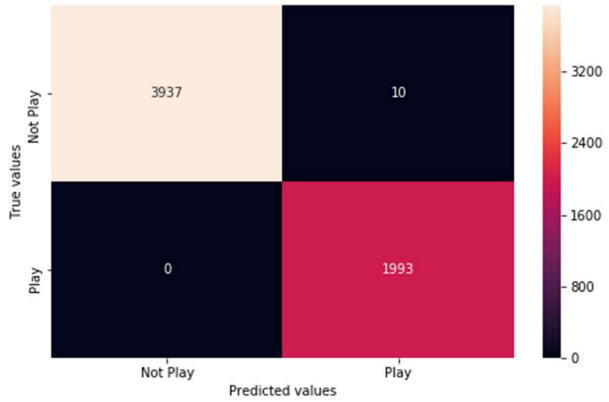
```
[0.94062524, 0.93974714, 0.9344533, 0.93524063, 0.93468851,
0.93834218, 0.94382783, 0.94014061, 0.93520917, 0.93616718]
```

Adaboost

Parameters used:

algorithm='SAMME.R', base_estimator=DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=12, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'),
learning_rate=1.0, n_estimators=50, random_state=None

Confusion Matrix:

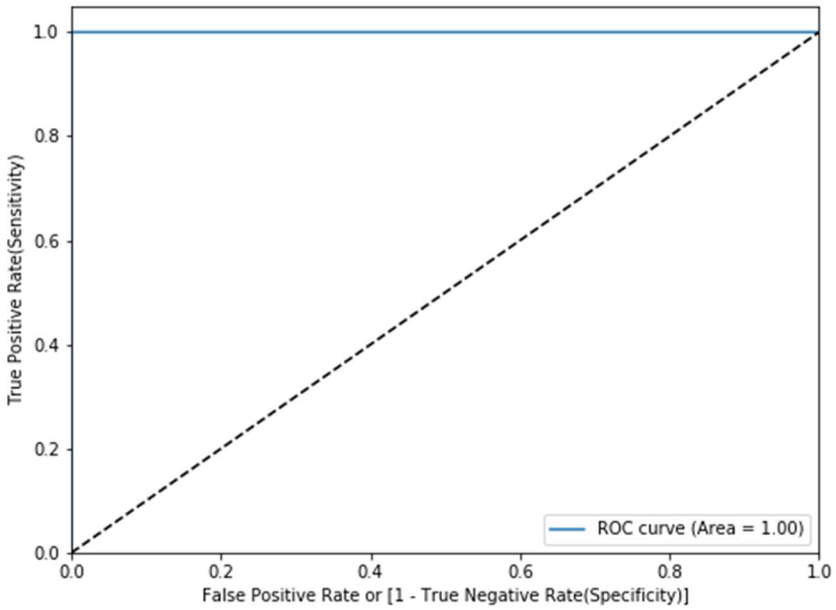



```
Misclassified samples: 10
Accuracy: 0.9983164983164983
```

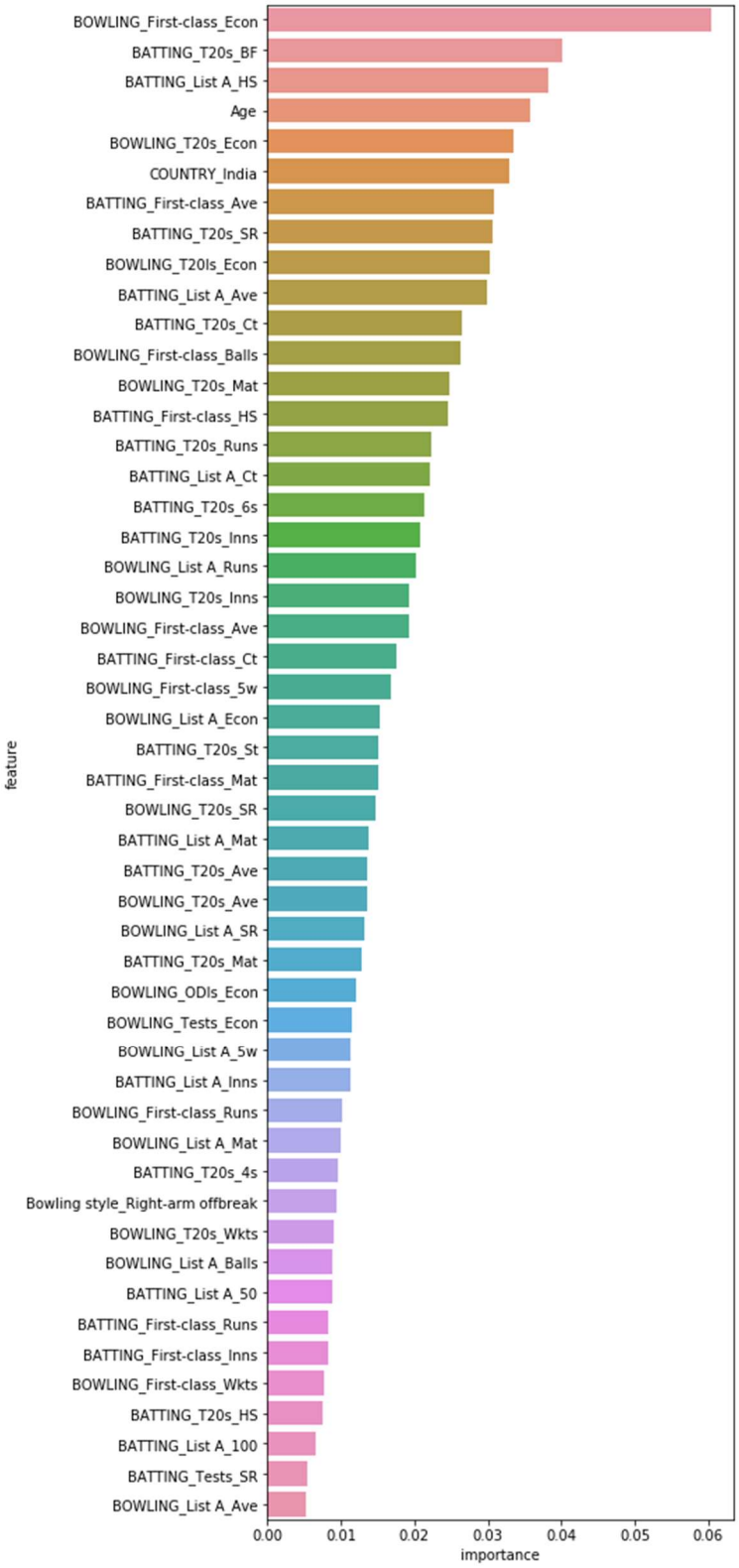
Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	3947
1	1.00	1.00	1.00	1993
accuracy			1.00	5940
macro avg	1.00	1.00	1.00	5940
weighted avg	1.00	1.00	1.00	5940

Roc_Auc plot:


1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate(Sensitivity)
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate or [1 - True Negative Rate(Specificity)]
ROC curve (Area = 1.00)

First 50 important features:



To be able to identify how well this algorithm works, I used 10-folds cross validation based on area under ROC curve. The following is the output:

```
[1., 1., 1., 1., 1., 1., 1., 1., 1., 1.]
```

Gradient Boosting

Parameters used:

criterion='friedman_mse', init=None, learning_rate=0.1, loss='deviance', max_depth=3, max_features=None, max_leaf_nodes=None,min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=100, n_iter_no_change=None, presort='auto', random_state=None, subsample=1.0, tol=0.0001, validation_fraction=0.1, verbose=0, warm_start=False

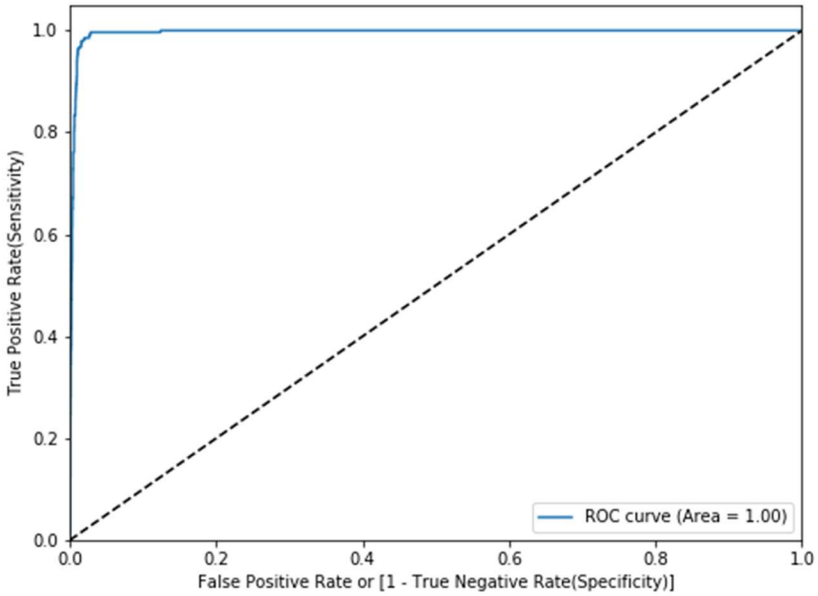
Confusion Matrix:


```
Misclassified samples: 139
Accuracy: 0.9765993265993266
```

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.97	0.98	3947
1	0.94	1.00	0.97	1993
accuracy			0.98	5940
macro avg	0.97	0.98	0.97	5940
weighted avg	0.98	0.98	0.98	5940

Roc_Auc plot:



To be able to identify how well this algorithm works, I used 10-folds cross validation based on area under ROC curve. The following is output:

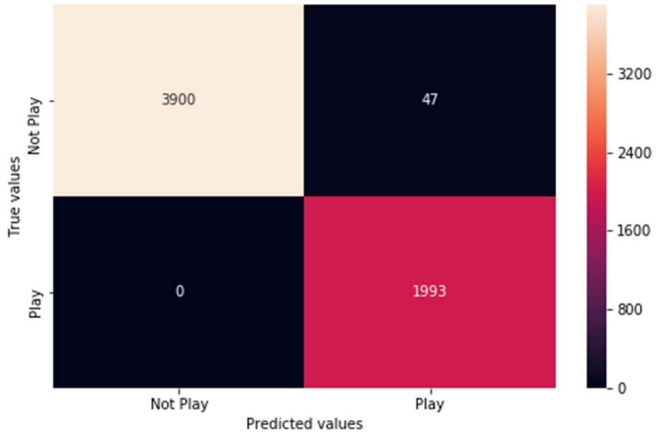
```
[0.99705906, 0.99647538, 0.99579776, 0.99563438, 0.99586327,
0.99505187, 0.99747424, 0.99781048, 0.99396194, 0.99439097]
```

Histogram-Based Gradient Boosting

Parameters used:

l2_regularization=0.0, learning_rate=0.1, loss='auto', max_bins=256, max_depth=None, max_iter=100, max_leaf_nodes=31, min_samples_leaf=20, n_iter_no_change=None, random_state=None, scoring=None, tol=1e-07, validation_fraction=0.1, verbose=0

Confusion Matrix:

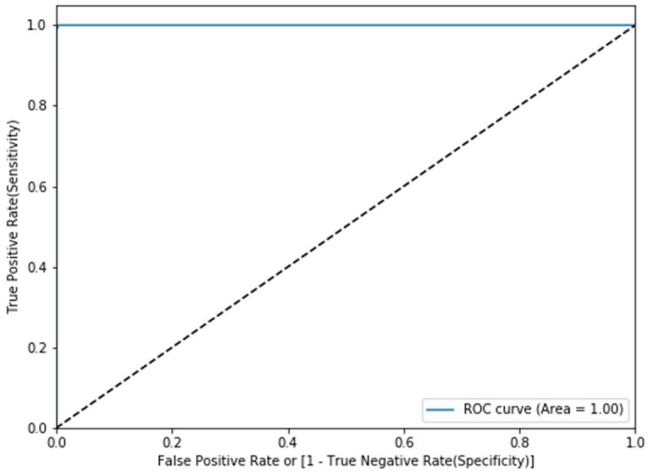



```
Misclassified samples: 47
Accuracy: 0.9920875420875421
```

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.99	0.99	3947
1	0.98	1.00	0.99	1993
accuracy			0.99	5940
macro avg	0.99	0.99	0.99	5940
weighted avg	0.99	0.99	0.99	5940

Roc_Auc plot:



To be able to identify how well this algorithm works, I used 10-fold cross validation based on area under ROC curve. The following is the output:

```
[1., 0.99927937, 0.99965508, 0.99998816, 0.99999526,
0.99997869, 0.99999842, 1., 0.99992892, 0.99966787]
```

Neural network models

Parameters used:

activation='relu', alpha=1e-05, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(289, 100, 50), learning_rate='constant', learning_rate_init=0.001, max_iter=200, momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5, random_state=1, shuffle=True, solver='lbfgs', tol=0.0001, validation_fraction=0.1, verbose=False, warm_start=False

Confusion Matrix:

I used GridSeachCv with 10-folds cross validation to be able to find most optimal parameters. The parameters I turned are hidden_layer_sizes and solver base on area under ROC curve.
hidden_layer_sizes= [(100,50,5), (100,50), (100,)]
solver=['lbfgs','sgd']

The following are the best parameters setting that acquire largest are under ROC curve using GridSearchCV:

activation='relu', alpha=1e-05, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(100,), learning_rate='constant', learning_rate_init=0.001, max_iter=200, momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,random_state=1, shuffle=True, solver='lbfgs', tol=0.0001, validation_fraction=0.1, verbose=False,

warm_start=False

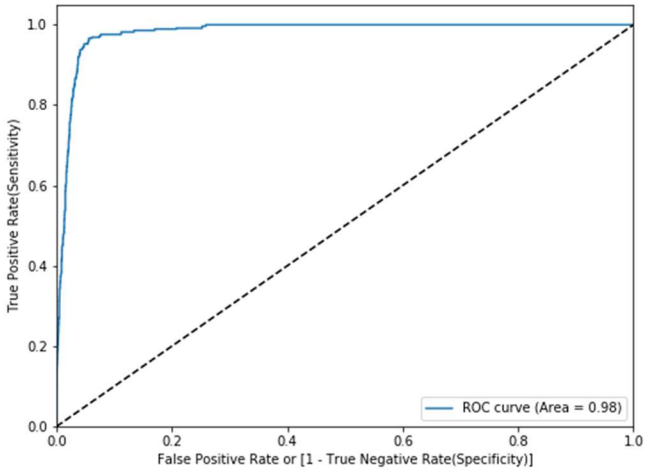
Confusion matrix using optimal parameters:


```
Misclassified samples: 290
Accuracy: 0.9511784511784511
```

Classification Report:

	precision	recall	f1-score	support
0	0.98	0.95	0.96	3947
1	0.91	0.95	0.93	1993
accuracy			0.95	5940
macro avg	0.94	0.95	0.95	5940
weighted avg	0.95	0.95	0.95	5940

Roc_Auc plot:



Summary of Accuracy and area under ROC curve:

Note: All the accuracy and area under roc curve are acquire by 10-fold cross validation on training dataset.

Model	Average Accuracy	Average Area under Roc Curve	Note
LogisticRegression	0.966788161	0.990200503	
Support Vector Classification	--	--	Took too much time to fit training data
K-nearest neighbors	0.981100088	0.994323267	
Decision Tree	0.989266064	0.994361751	
Random Forest	0.99040264	0.999867555	
Stochastic Gradient Descent	0.899986776	0.964317684	
Gaussian Processes classification	--	--	Took too much time to fit training data
Naive Bayes	0.821736747	0.937844178	
Adaboost	0.9989056	1	
Gradient Boosting	0.979584458	0.995913181	
Histogram-Based Gradient Boosting	0.99360147	0.999849177	
Neural network models	0.939638222	0.977137591	

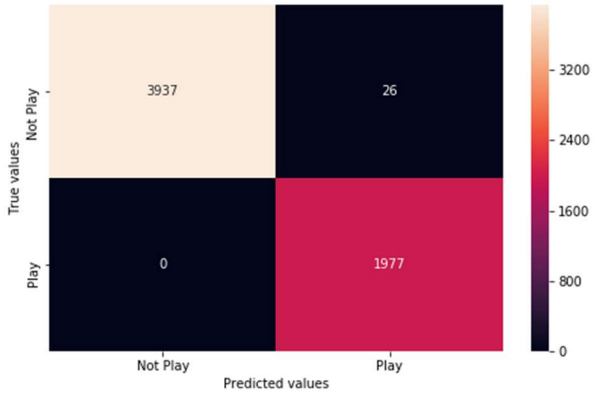
Voting Classifier on Validation Dataset:

I used a voting classifier that uses majority vote to predict the IPL status. As estimators, I used Random Forest, Adaboost, and Histogram-Based Gradient Boosting models because they have better accuracy and area under ROC curve.

Parameters used:

estimators=[('rf', RandomForestClassifier(bootstrap=True,class_weight=None, criterion='gini', max_depth=14, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=11, n_jobs=None, oob_score=False, random_state=None, verbose=0,...,splitter='best'), learning_rate=1.0, n_estimators=50, random_state=None)), ('hbgb', HistGradientBoostingClassifier(l2_regularization=0.0, learning_rate=0.1, loss='auto', max_bins=256, max_depth=None, max_iter=100, max_leaf_nodes=31, min_samples_leaf=20, n_iter_no_change=None, random_state=None,scoring=None, tol=1e-07, validation_fraction=0.1, verbose=0))], n_jobs=None, weights=None)

Confusion Matrix:




```
Misclassified samples: 26
Accuracy: 0.9956228956228956
```

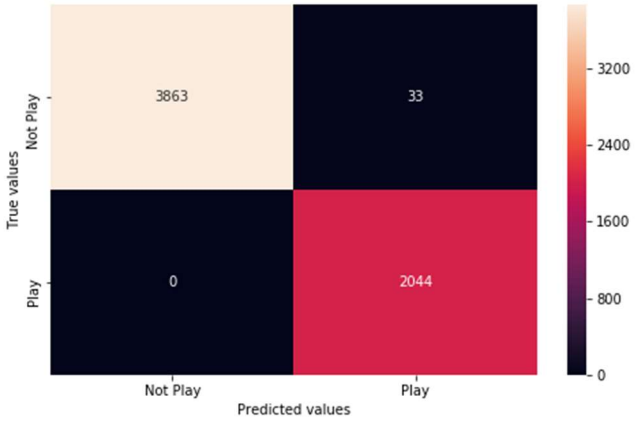
Classification Report:

	Precision	recall	f1-score	support
0	1.00	0.99	1.00	3963
1	0.99	1.00	0.99	1977
accuracy			1.00	5940
macro avg	0.99	1.00	1.00	5940
weighted avg	1.00	1.00	1.00	5940

10-fold cross validation average ROC_AUC score on Training data: **0.999206974326345**

Apply Voting Classifier model on Test Data:

Confusion Matrix:




```
Misclassified samples: 33
Accuracy: 0.9944444444444445
```

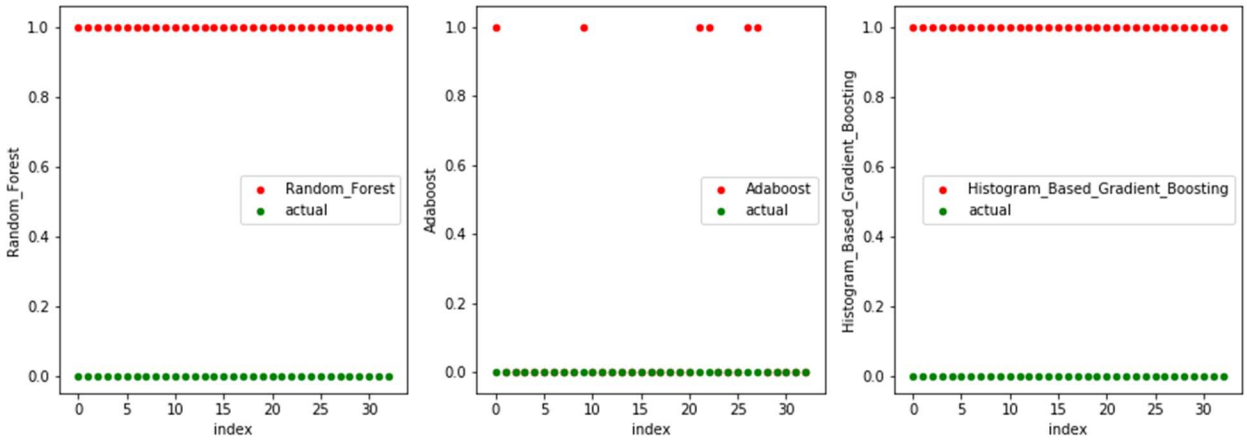
Classification Report:

	precision	recall	f1-score	support
0	1.00	0.99	1.00	3896
1	0.98	1.00	0.99	2044
accuracy			0.99	5940
macro avg	0.99	1.00	0.99	5940
weighted avg	0.99	0.99	0.99	5940

ROC_AUC Score: `0.9992299794661191`

Misclassified data labels Visualization:

If the estimator models predict the label correctly, then the actual marker will override on it. From the plot we can see that Adaboost only misclassified 6 out of 33 observations.



Unique work:

I predicted the IPL status using player's Test, ODI, T20I, First-Class, List A, and T20 cricket format. Mostly online, other people predicted the outcome of the IPL matches or the player's price tag. I plotted top 10 successful batsmen's and bowler's locations on globe using cesium.

Conclusion:

Analysis of datasets:

1) The majority of players come from England (Country).
2) The largest group of players come from Colombo (City/county).
3) The top 10 successful batsmen's and bowler's locations in all cricket format.
4) The highest numbers of 4s is in First_class and 6s in T20 cricket format.
5) When age increases, batting test strike rate decreases.
6) The highest number of bowling style player is Right-arm medium.
7) The highest numbers of last names are 'Khan'; however, players with the last name 'Smith' have more runs than 'Khan's'.
8) 50% of right-handed batsman of age 50; however, 50% left-hand batsman are younger than 50 years old.
9) Given the type of batting style, left-hand batsmen played more IPL than right-hand batsmen.
10) Right-arm off break, Leg break googly bowling style players played more IPL than the rest.
11) The highest number of players played in IPL are also played T20 cricket format.
12) Second highest number of players played in IPL are from Australia. The highest number of players played in IPL are from India.
13) Correlations of batting test data – Test runs and 4s have correlation of 1.
14) Correlations of ODI bowling data

Machine learning:

I performed 10-folds cross validation on each model.
Logistic Regression, K-nearest neighbors, Decision Tree, Random Forest, Adaboost, Gradient Boosting, and Histogram-Based Gradient Boosting models performed similar with accuracy of 95% plus and area under the curve of 99% plus. Neural network models also work pretty much the same with accuracy of 93% and area under the curve of 97%. Stochastic Gradient Descent performed well with accuracy of 89% and area under the curve 96%. Naive Bayes acquire the lowest accuracy and area under the curve compared to all other models. Support Vector Classification and Gaussian Processes classification do not work due to the size of the data set. It took long time to fit training data. In addition, ridge regularization (l2) does not converge for Logistic Regression model. K-nearest neighbors and Random Forest converge really fast.

Overall, Random Forest, Adaboost, and Histogram-Based Gradient Boosting have better accuracy and area under ROC curve. Therefore, I used voting classifier that uses majority vote to predict the IPL status. Voting classifier model works pretty well on test data and obtain 99% accuracy and area under ROC curve.

By doing some analysis on correctly classified and misclassified IPL status's observation, I believe that similar features can be the cause of misclassification. Some players might have similar record as other however for some reasons they did not play IPL. Among all missed classified observations Adaboost classifier predicted almost every observation's IPL status

correctly. Only few observations predicted incorrectly by Adaboost classifier. To find concrete reasons for misclassification, I have to do more analysis on misclassified observations.

If I have more time:
I will take test batting data, ODI batting data, and T20I batting data separately.
Then, predict BATTING_Tests_Ave of the players using linear regression for test batting data.
independent variables : Different combination of variables except BATTING_Tests_Ave
dependent variable : BATTING_Tests_Ave
Then, evaluate performance metrics to be able to know that the model is good enough to capture players BATTING_Tests_Ave or not. Same process for ODI batting data and T20I batting data to be able to predict BATTING_ODIs_Ave and BATTING_T20Is_Ave. Moreover, I would like to predict bowling economy rate for one of the cricket formats (either test, ODI, or T20I bowling data).

If I could start over (New ideas):

1. I would like to find out "who is the most bat sponsor in the world?" (For example, MRF (Madras Rubber Factory) sponsored Sachin Tendulkar-Indian former international cricketer.) In addition, I would like to know what business impact does it have? Then, create a machine learning model that can predict the prices that the companies have to pay each player based on their batting record for putting their sticker on the bat.

2. Which brand shoes used the most by cricket players in international cricket?

Acknowledgments:

https://geopy.readthedocs.io/en/stable/
https://nbviewer.jupyter.org/github/python-visualization/folium/blob/master/examples/MarkerCluster.ipynb
https://python-visualization.github.io/folium/quickstart.html
https://jupyter-gmaps.readthedocs.io/en/latest/tutorial.html
https://altair-viz.github.io/
https://sandcastle.cesium.com/
https://elitedatascience.com/imbalanced-classes
https://towardsdatascience.com/how-to-visualize-a-decision-tree-in-5-steps-19781b28ffe2
https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea
https://github.com/justmarkham/scikit-learn-videos/blob/master/09_classification_metrics.ipynb
https://github.com/justmarkham/scikit-learn-videos/blob/master/08_grid_search.ipynb

Libraries used:
Pandas: https://pandas.pydata.org/docs/
Re: https://docs.python.org/3/library/re.html
train_test_split: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
classification_report: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html
confusion_matrix: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
accuracy_score: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html
resample: https://scikit-learn.org/stable/modules/generated/sklearn.utils.resample.html
shuffle: https://scikit-learn.org/stable/modules/generated/sklearn.utils.shuffle.html
numpy: https://numpy.org/doc/
seaborn: https://seaborn.pydata.org/
matplotlib: https://matplotlib.org/3.2.1/contents.html
metrics: https://scikit-learn.org/stable/modules/model_evaluation.html
cross_val_score: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html
Tree: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
RandomizedSearchCV: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html
GridSearchCV: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
plot_tree: https://scikit-learn.org/stable/modules/generated/sklearn.tree.plot_tree.html
enable_hist_gradient_boosting: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingRegressor.html
RBF:https://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.RBF.html
LogisticRegression:https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
Support Vector Classification (SVC): https://scikit-

learn.org/stable/modules/svm.html#classification
K-nearest neighbor(KNeighborsClassifiers): https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
Decision Tree: https://scikit-learn.org/stable/modules/tree.html#classification
Random Forest(RandomForestClassifier): https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees
Stochastic Gradient Descent(SGDClassifier): https://scikit-learn.org/stable/modules/sgd.html#classification
Gaussian Processes classification(GaussianProcessClassifier): https://scikit-learn.org/stable/modules/gaussian_process.html#gaussian-process-classification-gpc
Naive Bayes(GaussianNB):https://scikit-learn.org/stable/modules/naive_bayes.html#categorical-naive-bayes
Adaboost(AdaBoostClassifier):https://scikit-learn.org/stable/modules/ensemble.html#adaboost
Gradient Boosting(GradientBoostingClassifier):https://scikit-learn.org/stable/modules/ensemble.html#gradient-tree-boosting
Histogram-Based Gradient Boosting(HistGradientBoostingClassifier):https://scikit-learn.org/stable/modules/ensemble.html#histogram-based-gradient-boosting
Neural network models(MLPClassifier):https://scikit-learn.org/stable/modules/neural_networks_supervised.html#classification

Tools used:
Jupyter Notebook, Visual Studio Code

Programing languages used:
Python, HTML, JavaScript