CAPSTONE PROJECT NOTES 2

BUSINESS REPORT

ABSTRACT

The dataset contains information about the matches team India has played. The BCCI wants to make predictions regarding the winning prospects to make team India win.

Yashveer Kothari. A

Post Graduate Programme in Data Science & Business Analytics

LIST OF CONTENTS		
#	CONTENTS	PAGE #
CRICKET PREDICTIONS	PROBLEM STATEMENT	4
	DATA DICTIONARY	4
PROBLEMS	EXPLORATORY ANALYSIS	5
	DATA CLEANING AND PREPROCESSING	12
	MODEL BUILDING	19
	MODEL TUNNING	38
	BUSINESS RECOMMENDATION	50

LIST OF TABLES		
TABLE #	TABLE NAME	PAGE #
1	TOP 5 DATASET SAMPLE	5
2	LAST 5 DATASET SAMPLE	6
3	SHAPE OF THE DATASET	7
4	DATASET INFORMATION	7
5	DATASET DESCRIPTION	8
6	DATASET DUPLICATES	8
7	MISSING DATA	9
8	UNIQUE COUNT	10
9	DATA INFORMATION AFTER MODIFICATION	11
10	REPLACING THE MISSING VALUES	12
11	SKEWNESS	14
12	MATCH LIGHT	16
13	MATCH SELECTION	15
14	FIRST SELECTION	15
15	OPPONENT	16
16	SEASON	16
17	OFFSHORE	16
18	PLAYERS SCORED ZERO	17
19	PLAYER HIGHEST WICKET	17
20	ENCODING THE DATASET	19
21	DATASET AFTER ENCODING	20
22	SHAPE OF THE DATASET	20
23	FEATURE SELECTION	21
24	LOGISTIC REGRESSION – CLASSIFICATION REPORT – TRAIN DATA	23
25	LOGISTIC REGRESSION – CLASSIFICATION REPORT – TEST DATA	23

26	LINEAR DISCRIMINANT ANALYSIS – CLASSIFICATION REPORT -TRAIN DATA	25
27	VARIANCE	27
28	KNN – CLASSIFICATION REPORT TRAIN DATA	28
29	KNN – CLASSIFICATION REPORT TEST DATA	29
30	NAÏVE BAYES – CLASSIFICATION REPORT TRAIN DATA	31
31	NAÏVE BAYES – CLASSIFICATION REPORT TEST DATA	31
32	DESCISION TREE – CLASSIFICATION REPORT TRAIN DATA	33
33	DESCISION TREE – CLASSIFICATION REPORT TEST DATA	33
34	RF – CLASSIFICATION REPORT TRAIN DATA	36
35	RF – CLASSIFICATION REPORT TEST DATA	36
36	ANN – CLASSIFICATION REPORT TRAIN DATA	37
37	ANN – CLASSIFICATION REPORT TRAIN DATA	37
38	LR – CLASSIFICATION REPORT TRAIN DATA – TUNED	37
39	LR – CLASSIFICATION REPORT TEST DATA – TUNED	39
40	LDA – CLASSIFICATION REPORT TRAIN DATA – TUNED	40
41	LDA – CLASSIFICATION REPORT TEST DATA – TUNED	40
42	DT – CLASSIFICATION REPORT TRAIN DATA – TUNED	41
43	DT – CLASSIFICATION REPORT TEST DATA – TUNED	41
44	RF – CLASSIFICATION REPORT TRAIN DATA – TUNED	42
45	RF – CLASSIFICATION REPORT TEST DATA – TUNED	42
46	NAÏVE BAYES – CLASSIFICATION REPORT TRAIN DATA – TUNED	43
47	NAÏVE BAYES – CLASSIFICATION REPORT TEST DATA – TUNED	43
48	BAGGING – CLASSIFICATION REPORT TRAIN DATA – TUNED	44
49	BAGGING – CLASSIFICATION REPORT TEST DATA – TUNED	44

50	ADA BOOSTING - CLASSIFICATION REPORT TRAIN DATA	45
51	ADA BOOSTING - CLASSIFICATION REPORT TEST DATA	46
52	GRADIENT BOOSTING - CLASSIFICATION REPORT TRAIN DATA	47
53	GRADIENT BOOSTING - CLASSIFICATION REPORT TEST DATA	47
54	SMOTE - CLASSIFICATION REPORT TRAIN DATA	48
55	SMOTE - CLASSIFICATION REPORT TEST DATA	48
56	ALL MODEL COMPARISION	49

LIST OF FIGURES

FIGURE #	FIGURE NAME	PAGE #
1	BOXPLOT	13
2	HEATMAP	18
3	AUC/ ROC CURVE - TRAIN DATA – LOGISTIC REGRESSION	23
4	AUC/ ROC CURVE - TEST DATA – LOGISTIC REGRESSION	24
5	LDA- CONFUSION MATRIX – TRAIN DATA	25
6	LDA- CONFUSION MATRIX – TEST DATA	25
7	VARIANCE PLOT	28
8	KNN – CONFUSION MATRIX TRAIN DATA	29
9	KNN – CONFUSION MATRIX TEST DATA	30
10	NAÏVE BAYES – CONFUSION MATRIX TRAIN DATA	32
11	NAÏVE BAYES – CONFUSION MATRIX TRAIN DATA	32
12	DECISION TREE– CONFUSION MATRIX TRAIN DATA	34
13	DECISION TREE– CONFUSION MATRIX TEST DATA	34

PROBLEM STATEMENT

BCCI has hired an external analytics consulting firm for data analytics. The major objective of this tie up is to extract actionable insights from the historical match data and make strategic changes to make India win. Primary objective is to create Machine Learning models which correctly predicts a win for the Indian Cricket Team. Once a model is developed then you have to extract actionable insights and recommendation. Also, below are the details of the next 10 matches, India is going to play. You have to predict the result of the matches and if you are getting prediction as a Loss then suggest some changes and re-run your model again until you are getting Win as a prediction. You cannot use the same strategy in the entire series, because opponent will get to know your strategy and they can come with counter strategy. Hence for all the below 5 matches you have to suggest unique strategies to make India win.

DATA DICTIONARY

VARIABLES	DESCRIPTION
Game_number	Unique ID for each match
Result	Final result of the match
Avg_team_Age	Average age of the playing 11 players for that match
Match_light_type	type of match: Day, night or day & night
Match_format	Format of the match: T20, ODI or test
Bowlers_in_team	how many full-time bowlers has been player in the team
Wicket_keeper_in_team	how many full-time wicket keepers has been player in the team
All_rounder_in_team	how many full-time all-rounders has been player in the team
First_selection	First inning of team: batting or bowling
Opponent	Opponent team in the match
Season	What is the season of the city, where match has been played
Audience_number	Total number of audiences in the stadium
Offshore	Match played within country or outside of the country
Max_run_scored_1over	Maximum run scored in 1 over by team
Max_wicket_taken_1over	Maximum wicket taken in 1 over by team
Extra_bowls_bowled	Total number of extras bowled by team
Min_run_given_1over	Minimum run given by the bowler in one over
Min_run_scored_1over	Minimum run scored in 1 over by team

Max_run_given_1over	Maximum run given by the bowler in one over
extra_bowls_opponent	Total number of extras bowled by opponent
player_highest_run	Highest score in the match by one player
Players_scored_zero	Number of players out on zero run
player_highest_wicket	Highest wickets taken by single player in match

TABLE 1: TOP 5 DATASET SAMPLE

	Game_number	Result	Avg_team_Age	Match_light_type	Match_format	Bowlers_in_team	Wicket_keeper_in_team	All_rounder_in_team	First_selection
0	Game_1	Loss	18.0	Day	ODI	3.0	1	3.0	Bowling
1	Game_2	Win	24.0	Day	T20	3.0	1	4.0	Batting
2	Game_3	Loss	24.0	Day and Night	T20	3.0	1	2.0	Bowling
3	Game_4	Win	24.0	NaN	ODI	2.0	1	2.0	Bowling
4	Game_5	Loss	24.0	Night	ODI	1.0	1	3.0	Bowling

Opponent	Season	Audience_number	Offshore	Max_run_scored_1over	Max_wicket_taken_1over	Extra_bowls_bowled	Min_run_given_1over
Srilanka	Summer	9940.0	No	13.0	3	0.0	2
Zimbabwe	Summer	8400.0	No	12.0	1	0.0	0
Zimbabwe	NaN	13146.0	Yes	14.0	4	0.0	0
Kenya	Summer	7357.0	No	15.0	4	0.0	2
Srilanka	Summer	13328.0	No	12.0	4	0.0	0

Min_run_scored_1over	Max_run_given_1over	extra_bowls_opponent	player_highest_run	Players_scored_zero	player_highest_wicket
3.0	6.0	0	54.0	3	1
3.0	6.0	0	69.0	2	1
3.0	6.0	0	69.0	3	1
3.0	6.0	0	73.0	3	1
3.0	6.0	0	80.0	3	1

TABLE 2: LAST 5 DATASET SAMPLE

	Game_number	Result	Avg_team_Age	Match_light_type	Match_format	Bowlers_in_team	Wicket_keeper_in_team	All_rounder_in_team	First_selection
2925	Game_2926	Win	30.0	Day	T20	3.0	1	4.0	Batting
2926	Game_2927	Win	30.0	Day	ODI	4.0	1	3.0	Bowling
2927	Game_2928	Win	30.0	Day and Night	ODI	4.0	1	3.0	Bowling
2928	Game_2929	Win	30.0	Day	ODI	4.0	1	3.0	Batting
2929	Game_2930	Win	30.0	Day	ODI	4.0	1	3.0	Batting

Opponent	Season	Audience_number	Offshore	Max_run_scored_1over	Max_wicket_taken_1over	Extra_bowls_bowled	Min_run_given_1over
South Africa	Summer	33950.0	No	15.0	3	8.0	0
Kenya	Summer	19663.0	No	14.0	4	8.0	2
Pakistan	Rainy	39823.0	Yes	14.0	4	10.0	2
Kenya	Rainy	14007.0	No	14.0	2	20.0	2
Kenya	Rainy	20839.0	No	12.0	4	4.0	5

Min_run_scored_1over	Max_run_given_1over	extra_bowls_opponent	player_highest_run	Players_scored_zero	player_highest_wicket
3.0	6.0	3	50.0	3	2
3.0	6.0	2	52.0	2	1
4.0	10.0	2	80.0	3	2
3.0	6.0	3	98.0	3	1
3.0	6.0	3	62.0	1	1

TABLE 3: SHAPE OF THE DATASET

```
The number of rows and columns in the dataset is (2930, 23) respectively
```

TABLE 4: DATASET INFORMATION

```
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 23 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Game_number             2930 non-null   object
 1   Result                  2930 non-null   object
 2   Avg_team_Age            2833 non-null   float64
 3   Match_light_type        2878 non-null   object
 4   Match_format            2860 non-null   object
 5   Bowlers_in_team         2848 non-null   float64
 6   Wicket_keeper_in_team   2930 non-null   int64
 7   All_rounder_in_team     2890 non-null   float64
 8   First_selection         2871 non-null   object
 9   Opponent                2894 non-null   object
 10  Season                  2868 non-null   object
 11  Audience_number         2849 non-null   float64
 12  Offshore                2866 non-null   object
 13  Max_run_scored_1over    2902 non-null   float64
 14  Max_wicket_taken_1over  2930 non-null   int64
 15  Extra_bowls_bowled      2901 non-null   float64
 16  Min_run_given_1over     2930 non-null   int64
 17  Min_run_scored_1over    2903 non-null   float64
 18  Max_run_given_1over     2896 non-null   float64
 19  extra_bowls_opponent    2930 non-null   int64
 20  player_highest_run      2902 non-null   float64
 21  Players_scored_zero     2930 non-null   object
 22  player_highest_wicket   2930 non-null   object
dtypes: float64(9), int64(4), object(10)
```

1. There are 2930 rows and 13 columns in the dataset.
2. There are 10 variables with object data type.
3. There are 4 datatypes with integer data type.
4. There are 9 datatypes with float data type.

TABLE 5: DATASET DESCRIPTION

	count	mean	std	min	25%	50%	75%	max
Avg_team_Age	2833.0	29.242852	2.264230	12.0	30.0	30.0	30.00	70.0
Bowlers_in_team	2848.0	2.913624	1.023907	1.0	2.0	3.0	4.00	5.0
Wicket_keeper_in_team	2930.0	1.000000	0.000000	1.0	1.0	1.0	1.00	1.0
All_rounder_in_team	2890.0	2.722491	1.092699	1.0	2.0	3.0	4.00	4.0
Audience_number	2849.0	46267.960688	48599.581459	7063.0	20363.0	34349.0	57876.00	1399930.0
Max_run_scored_1over	2902.0	15.199862	3.661010	11.0	12.0	14.0	18.00	25.0
Max_wicket_taken_1over	2930.0	2.713993	1.080623	1.0	2.0	3.0	4.00	4.0
Extra_bowls_bowled	2901.0	11.252671	7.780829	0.0	6.0	10.0	15.00	40.0
Min_run_given_1over	2930.0	1.952560	1.678332	0.0	0.0	2.0	3.00	6.0
Min_run_scored_1over	2903.0	2.762659	0.705759	1.0	2.0	3.0	3.00	4.0
Max_run_given_1over	2896.0	8.669199	5.003525	6.0	6.0	6.0	9.25	40.0
extra_bowls_opponent	2930.0	4.229693	3.626108	0.0	2.0	3.0	7.00	18.0
player_highest_run	2902.0	65.889387	20.331614	30.0	48.0	66.0	84.00	100.0

TABLE 6: DATASET DUPLICATES

```
The dataset contains 0 duplicate entries
```

- There are no duplicates in the dataset.

TABLE 7: MISSING DATA

```
There are 789 missing values in the dataset
```

Game_number	0
Result	0
Avg_team_Age	97
Match_light_type	52
Match_format	70
Bowlers_in_team	82
Wicket_keeper_in_team	0
All_rounder_in_team	40
First_selection	59
Opponent	36
Season	62
Audience_number	81
Offshore	64
Max_run_scored_1over	28
Max_wicket_taken_1over	0
Extra_bowls_bowled	29
Min_run_given_1over	0
Min_run_scored_1over	27
Max_run_given_1over	34
extra_bowls_opponent	0
player_highest_run	28
Players_scored_zero	0
player_highest_wicket	0

- We replace the missing value in the dataset with median and mode for Numerical variables and categorical variables respectively. (Refer Table 10)

TABLE 8: UNIQUE COUNT

```
unique count of Game_number
['Game_1' 'Game_2' 'Game_3' ... 'Game_2928' 'Game_2929' 'Game_2930']

unique count of Result
['Loss' 'Win']

unique count of Match_light_type
['Day' 'Day and Night' nan 'Night']

unique count of Match_format
['ODI' 'T20' 'Test' '20-20' nan]

unique count of First_selection
['Bowling' 'Batting' 'Bat' nan]

unique count of Opponent
['Srilanka' 'Zimbabwe' 'Kenya' 'Australia' 'England' 'South Africa'
 'Pakistan' 'West Indies' 'Bangladesh' nan]

unique count of Season
['Summer' nan 'Winter' 'Rainy']

unique count of Offshore
['No' 'Yes' nan]

unique count of Players_scored_zero
[3 2 1 4 'Three']

unique count of player_highest_wicket
[1 2 3 4 'Three' 5]
```

- **Modifying few unique data names:**

1. 20-20: ‘T20’
2. Bat: ‘Batting’
3. Three: ‘3’ in both 'Players_scored_zero' and 'player_highest_wicket' columns

- **Modified names in the dataset:**

```
['ODI' 'T20' 'Test' nan]
['Bowling' 'Batting' nan]
[3 2 1 4]
[1 2 3 4 5]
```

- **Modifying few data types:**

1. 'Players_scored_zero' and 'player_highest_wicket' to integer type ('int64').

TABLE 9: DATA INFORMATION AFTER MODIFICATION

```
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 23 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Game_number             2930 non-null   object
 1   Result                  2930 non-null   object
 2   Avg_team_Age            2833 non-null   float64
 3   Match_light_type        2878 non-null   object
 4   Match_format            2860 non-null   object
 5   Bowlers_in_team         2848 non-null   float64
 6   Wicket_keeper_in_team   2930 non-null   int64
 7   All_rounder_in_team     2890 non-null   float64
 8   First_selection         2871 non-null   object
 9   Opponent                2894 non-null   object
 10  Season                  2868 non-null   object
 11  Audience_number         2849 non-null   float64
 12  Offshore                2866 non-null   object
 13  Max_run_scored_1over    2902 non-null   float64
 14  Max_wicket_taken_1over  2930 non-null   int64
 15  Extra_bowls_bowled      2901 non-null   float64
 16  Min_run_given_1over     2930 non-null   int64
 17  Min_run_scored_1over    2903 non-null   float64
 18  Max_run_given_1over     2896 non-null   float64
 19  extra_bowls_opponent    2930 non-null   int64
 20  player_highest_run      2902 non-null   float64
 21  Players_scored_zero     2930 non-null   int64
 22  player_highest_wicket   2930 non-null   int64
dtypes: float64(9), int64(6), object(8)
```

TABLE 10: REPLACING THE MISSING VALUES

```
Result                   0
Avg_team_Age             0
Match_light_type         0
Match_format             0
Bowlers_in_team          0
All_rounder_in_team      0
First_selection          0
Opponent                 0
Season                   0
Audience_number          0
Offshore                 0
Max_run_scored_1over     0
Max_wicket_taken_1over   0
Extra_bowls_bowled       0
Min_run_given_1over      0
Min_run_scored_1over     0
Max_run_given_1over      0
extra_bowls_opponent     0
player_highest_run       0
Players_scored_zero      0
player_highest_wicket    0
dtype: int64
```

- The missing data in the dataset has been replaced as per the datatype using the median and mode. Hence There are no missing values in the dataset.

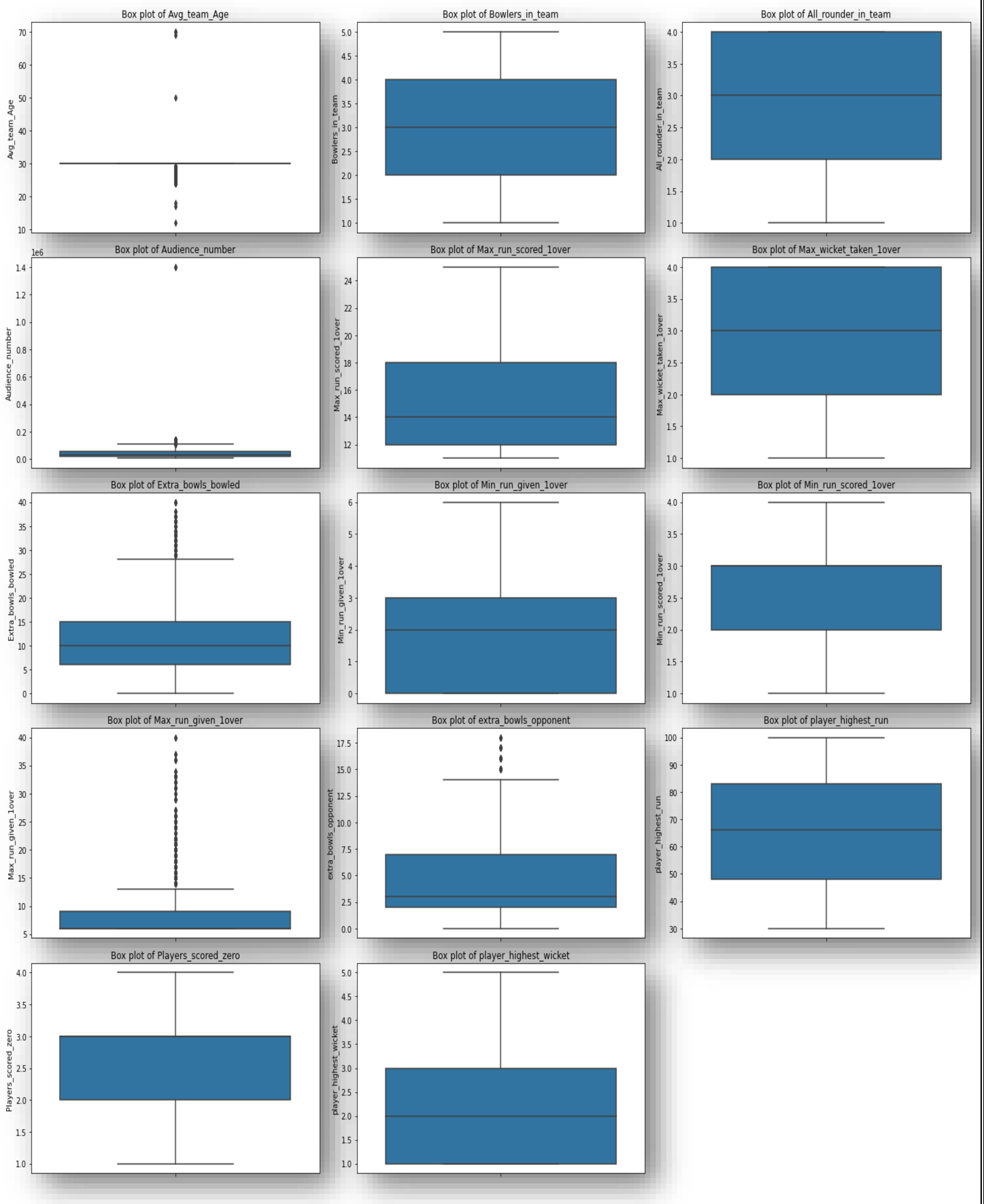
Numerical data

```
Avg_team_Age             0
Bowlers_in_team          0
All_rounder_in_team      0
Audience_number          0
Max_run_scored_1over     0
Max_wicket_taken_1over   0
Extra_bowls_bowled       0
Min_run_given_1over      0
Min_run_scored_1over     0
Max_run_given_1over      0
extra_bowls_opponent     0
player_highest_run       0
Players_scored_zero      0
player_highest_wicket    0
```

Categorical data

```
Match_light_type    0
Match_format        0
First_selection     0
Opponent            0
Season              0
Offshore            0
Result              0
```

FIGURE 1: BOXPLOT



1. Variable ‘avg_team_age’ is close to normal distribution as mean and median are similar though it has outliers.
2. Variable ‘Bowlers_in_team’ is normally distributed as mean and median are almost same, no outliers are present.
3. Variable ‘All_rounder_in_team’ is slightly left skewed as there is difference in mean and median and also 75 percentile and the maximum value are equal, no outliers are present.
4. Variable ‘audience_number’ is right skewed as mean is affected due to outliers present.
5. Variable ‘Max_run_scored_1over’ is slightly right skewed, no outliers are present.
6. Variable ‘Max_wicket_taken_1over’ is slightly left skewed as mean is less than median and also 75 percentile and the maximum value are same, no outliers are present.
7. Variable ‘Extra_bowls_bowled’ is right skewed as mean is higher than median, outliers are present.
8. Variable ‘Min_run_given_1over’ is close to normal and minimum value and 25 percentiles are equal, no outliers are present.
9. Variable ‘Min_run_scored_1over’ is slightly left skewed, 50 and 75 percentiles are equal, no outliers are present.
10. Variable ‘Max_run_given_1over’ is right skewed, minimum value,25,50 percentile has same values, outliers are present.
11. Variable ‘extra_bowls_opponent’ is right skewed, outliers are present.
12. Variable ‘player_highest_run’ is normally distributed, no outliers are present.
13. Variable ‘Players_scored_zero’ is slightly left skewed, 50 and 75 percentiles have the same value, no outliers are present.
14. Variable ‘player_highest_wicket’ is normally distributed, minimum value and 25 percentiles are equal.

TABLE 11: SKEWNESS

Avg_team_Age	5.068403
Bowlers_in_team	-0.296492
All_rounder_in_team	-0.335012
Audience_number	15.782867
Max_run_scored_1over	0.838907
Max_wicket_taken_1over	-0.305597
Extra_bowls_bowled	1.132432
Min_run_given_1over	0.433859
Min_run_scored_1over	-0.568821
Max_run_given_1over	2.692147
extra_bowls_opponent	0.916295
player_highest_run	-0.031472
Players_scored_zero	-0.505491
player_highest_wicket	1.026090

TABLE 12: MATCH LIGHT

Result	Loss	Win
Match_light_type		
Day	314	1779
Day and Night	135	406
Night	24	272

TABLE 13: MATCH FORMAT

Result	Loss	Win
Match_format		
ODI	269	1666
T20	180	690
Test	24	101

TABLE 14: FIRST SELECTION

Result	Loss	Win
First_selection		
Batting	172	977
Bowling	301	1480

TABLE 15: OPPONENT

Result	Loss	Win
Opponent		
Australia	24	80
Bangladesh	10	194
England	18	265
Kenya	93	483
Pakistan	17	236
South Africa	117	559
Srilanka	124	389
West Indies	4	154
Zimbabwe	66	97

TABLE 16: SEASON

Result	Loss	Win
Season		
Rainy	170	1201
Summer	238	680
Winter	65	576

TABLE 17: OFFSHORE

Result	Loss	Win
Offshore		
No	227	1894
Yes	246	563

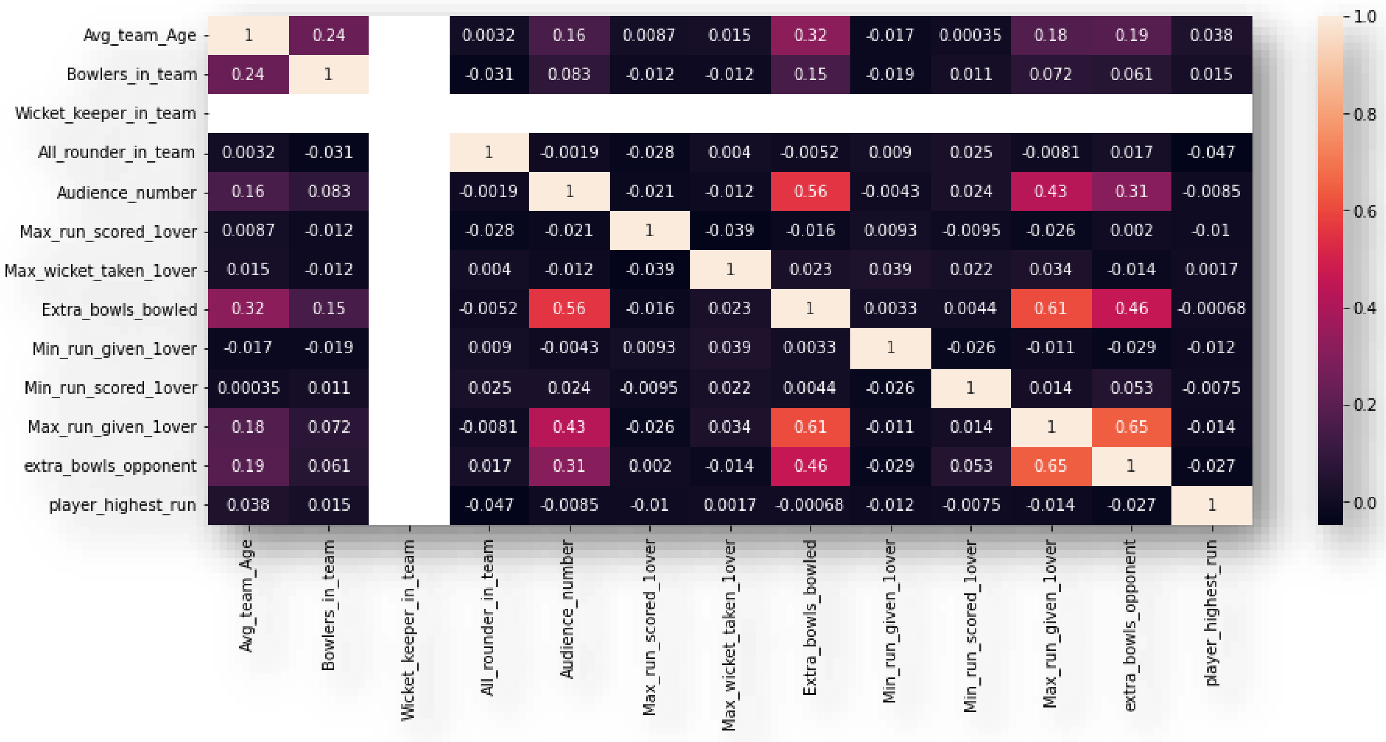
TABLE 18: PLAYERS SCORED ZERO

Result	Loss	Win
Players_scored_zero		
1	56	110
2	141	603
3	250	1485
4	26	259

TABLE 19: PLAYER HIGHEST WICKET

Result	Loss	Win
player_highest_wicket		
1	286	798
2	104	959
3	63	371
4	10	201
5	10	128

FIGURE 2: HEATMAP



INSIGHTS

- Around 72% of matches are played in India and only 28% are played out off India.
- Most of the matches are played in Rainy Season.
- Majority of the matches are played against South Africa (676).
- Around 71% ODI Matches are played in Day light.
- Team have win around 83% of the matches.
- 60% of the time team gets chance to bat first.
- Most of the matches are played against south Africa in which 531 times India secure to win.
- 1781-time team have opted to bowl first out of which 1480 have win the match.
- On average, 19% of the time when playing outside the country, and 65% when playing within the country, team manage to win.
- Extra_bowls_opponent has help team to win the matches.
- Inexperienced team (young player) has the higher chances to lose the match.
- We can see data is imbalance we have to do smote to resolve this issue.
- During the bowling contest, the team won 51% Matches while Batting 33%.
- Team has won 57% of ODI matches and 24% of T20 matches.

TABLE 20: ENCODING THE DATASET

```
1    2457
0     473
Name: Result, dtype: int64
```

```
1    1781
0    1149
Name: First_selection, dtype: int64
```

```
0    2121
1     809
Name: Offshore, dtype: int64
```

- As many machine learning models cannot work with string values we will encode the categorical variables and convert their datatypes to integer type.
- For variable like ‘Result’, ‘Offshore’ and ‘First_selection’ I have used simple categorical conversion technique This will convert the values into 0 and 1. As there is no level or order in the subcategory any encoding will give the same result.
- For remaining variable we have used dummies encoding technique.
- We can see the value count of this variable after encoding as below.

TABLE 21: DATASET AFTER ENCODING

	Game_number	Result	Avg_team_Age	Match_light_type	Match_format	Bowlers_in_team	Wicket_keeper_in_team	All_rounder_in_team	First_selection
0	Game_1	0	18.0	Day	ODI	3.0	1	3.0	1
1	Game_2	1	24.0	Day	T20	3.0	1	4.0	0
2	Game_3	0	24.0	Day and Night	T20	3.0	1	2.0	1
3	Game_4	1	24.0	Day	ODI	2.0	1	2.0	1
4	Game_5	0	24.0	Night	ODI	1.0	1	3.0	1

Opponent	...	Max_run_scored_1over	Max_wicket_taken_1over	Extra_bowls_bowled	Min_run_given_1over	Min_run_scored_1over	Max_run_given_1over
Srilanka	...	13.0	3	0.0	2	3.0	6.0
Zimbabwe	...	12.0	1	0.0	0	3.0	6.0
Zimbabwe	...	14.0	4	0.0	0	3.0	6.0
Kenya	...	15.0	4	0.0	2	3.0	6.0
Srilanka	...	12.0	4	0.0	0	3.0	6.0

Min_run_scored_1over	Max_run_given_1over	extra_bowls_opponent	player_highest_run	Players_scored_zero	player_highest_wicket
3.0	6.0	0	54.0	3	1
3.0	6.0	0	69.0	2	1
3.0	6.0	0	69.0	3	1
3.0	6.0	0	73.0	3	1
3.0	6.0	0	80.0	3	1

SPLITTING THE DATASET

- Our target variable is ‘Result. As we can see that there is a data imbalance in the variable.
- So, here I will be using the oversampling technique (i.e., SMOTE) and check whether it improves our model performance or not.
- We have stored all the predictor in x and target variable in y.

TABLE 22: SHAPE OF THE DATASET

```
Training set for independent variable is (2051, 30)
Test set for independent variable is (879, 30)
Training set for dependent variable is (2051,)
Test set for dependent variable is (879,)
```

- X train - denotes 70% training dataset (except the target column called "Result").
- X test- denotes 30% test dataset (except the target column called "Result").
- y train- denotes the 70% training dataset with only the target column called "Result".
- y test- denotes 30% test dataset with only the target column called "Result".

FEATURE SELECTION

- Chi-square test is used to determine the relationship between the predictor and target variable.
- In Feature selection, we aim to select the features which are highly dependent on the target variable.
- Higher the chi-square value indicate that the feature is more dependent on the target variable and can be select for model training.
- Chi-square score for Game number is null. So, we eliminate non-significant variable Game number.
- After second iteration we find Wicket keeper as non-significant variable as per chi-square test and same we can in the heat map. So, both the variable has been eliminated to train our model with remaining predictor.

TABLE 23: FEATURE SELECTION

	Feature	Scores
5	Audience_number	1.539290e+06
9	Extra_bowls_bowled	3.373231e+02
13	extra_bowls_opponent	1.676524e+02
6	Offshore	8.085816e+01
26	Opponent_Zimbabwe	5.582450e+01
...	...	...
2952	Game_number_Game_988	NaN
2954	Game_number_Game_99	NaN
2959	Game_number_Game_994	NaN
2960	Game_number_Game_995	NaN
2962	Game_number_Game_997	NaN

	Feature	Scores
5	Audience_number	1.539290e+06
9	Extra_bowls_bowled	3.373231e+02
13	extra_bowls_opponent	1.676524e+02
6	Offshore	8.085816e+01
26	Opponent_Zimbabwe	5.582450e+01
27	Season_Summer	4.525627e+01
12	Max_run_given_1over	3.913104e+01
10	Min_run_given_1over	3.077176e+01
15	Match_light_type_Day and Night	2.109785e+01
19	Opponent_Bangladesh	1.890085e+01
32	player_highest_wicket_2	1.749915e+01
25	Opponent_West Indies	1.614492e+01
24	Opponent_Srilanka	1.599199e+01
20	Opponent_England	1.482618e+01
34	player_highest_wicket_4	1.319061e+01
17	Match_format_T20	1.012452e+01
35	player_highest_wicket_5	1.009267e+01
28	Season_Winter	9.650935e+00
16	Match_light_type_Night	8.998566e+00
31	Players_scored_zero_4	8.890099e+00
3	All_rounder_in_team	7.763370e+00
22	Opponent_Pakistan	7.249808e+00
0	Avg_team_Age	6.308481e+00
18	Match_format_Test	2.171684e+00
8	Max_wicket_taken_1over	2.142275e+00
30	Players_scored_zero_3	1.500470e+00
11	Min_run_scored_1over	1.345619e+00
23	Opponent_South Africa	1.116496e+00
29	Players_scored_zero_2	1.017460e+00
4	First_selection	9.019972e-01
1	Bowlers_in_team	8.729197e-01
21	Opponent_Kenya	5.556134e-01
14	player_highest_run	4.414944e-01
33	player_highest_wicket_3	1.744586e-01
7	Max_run_scored_1over	7.549787e-02
2	Wicket_keeper_in_team	0.000000e+00

MODEL SELECTION

- For this classification the following models have been used:

    i. Logistic Regression
    ii. Linear Discriminant Analysis (LDA)
    iii. KNN
    iv. Decision Tree (CART)
    v. Random Forest
    vi. Naïve Bayes
    vii. ANN.

TABLE 24: LOGISTIC REGRESSION – CLASSIFICATION EPORT – TRAIN DATA

	precision	recall	f1-score	support
0	0.87	0.04	0.08	329
1	0.84	1.00	0.92	1722
accuracy			0.84	2051
macro avg	0.86	0.52	0.50	2051
weighted avg	0.85	0.84	0.78	2051

TABLE 25: LOGISTIC REGRESSION – CLASSIFICATION EPORT – TEST DATA

	precision	recall	f1-score	support
0	0.67	0.01	0.03	144
1	0.84	1.00	0.91	735
accuracy			0.84	879
macro avg	0.75	0.51	0.47	879
weighted avg	0.81	0.84	0.77	879

FIGURE 3: AUC/ ROC CURVE - TRAIN DATA – LOGISTIC REGRESSION

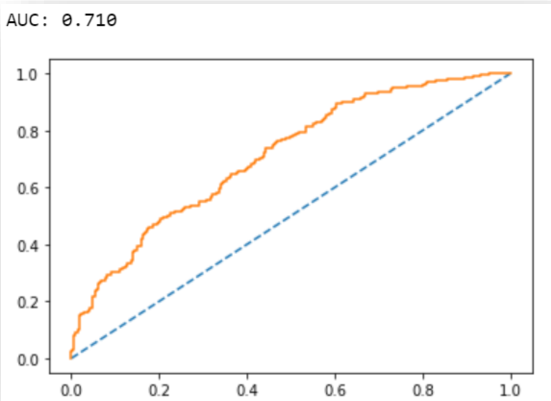
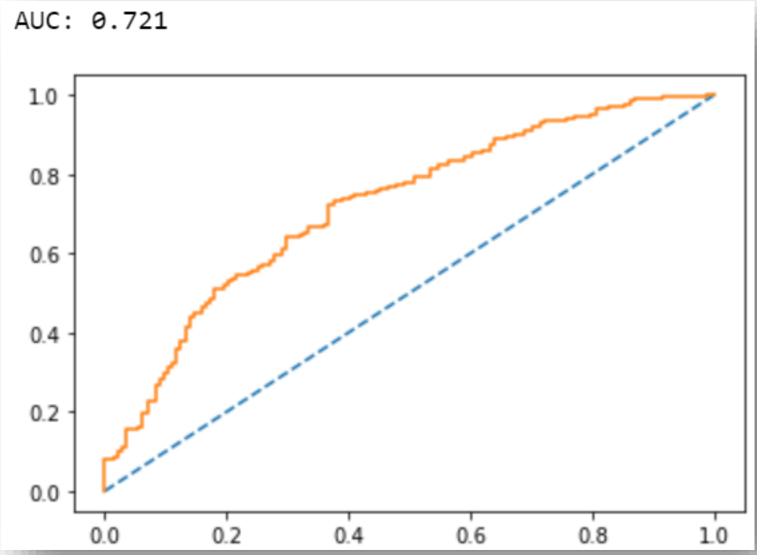


FIGURE 4: AUC/ ROC CURVE - TEST DATA – LOGISTIC REGRESSION



INSIGHTS

1. We can observe very low recall and F1 score for the zero class.
2. For the Training set we got Accuracy= 84 %, Precision= 78 %, recall= 51 %, f1-score= 48 %.
3. For the Testing set we got Accuracy= 84 %, Precision= 92 %, recall= 51 %, f1-score= 47 %.

TABLE 26: LINEAR DISCRIMINANT ANALYSIS - CLASSIFICATION REPORT - TRAIN DATA

	precision	recall	f1-score	support
0	0.69	0.39	0.50	329
1	0.89	0.97	0.93	1722
accuracy			0.87	2051
macro avg	0.79	0.68	0.71	2051
weighted avg	0.86	0.87	0.86	2051

```
array([[ 127,  202],
       [  57, 1665]], dtype=int64)
```

TABLE 27: LINEAR DISCRIMINANT ANALYSIS - CLASSIFICATION EPORT -TEST DATA

	precision	recall	f1-score	support
0	0.67	0.33	0.44	144
1	0.88	0.97	0.92	735
accuracy			0.86	879
macro avg	0.78	0.65	0.68	879
weighted avg	0.85	0.86	0.84	879

```
array([[ 47,  97],
       [ 23, 712]], dtype=int64)
```

FIGURE 5: LDA- CONFUSION MATRIX – TRAIN DATA

	Predict 1	Predict 0
1	127	202
0	57	1665

FIGURE 6: LDA- CONFUSION MATRIX – TEST DATA

	Predict 1	Predict 0
1	47	97
0	23	712

INSIGHTS

- LDA is performing well as compared to Logistic model.
- For the Training set we got Accuracy= 87 %, Precision= 80 %, recall= 66 %, f1-score= 70 %.
- For the Testing set we got Accuracy= 87 %, Precision= 79 %, recall= 67 %, f1-score= 71 %.

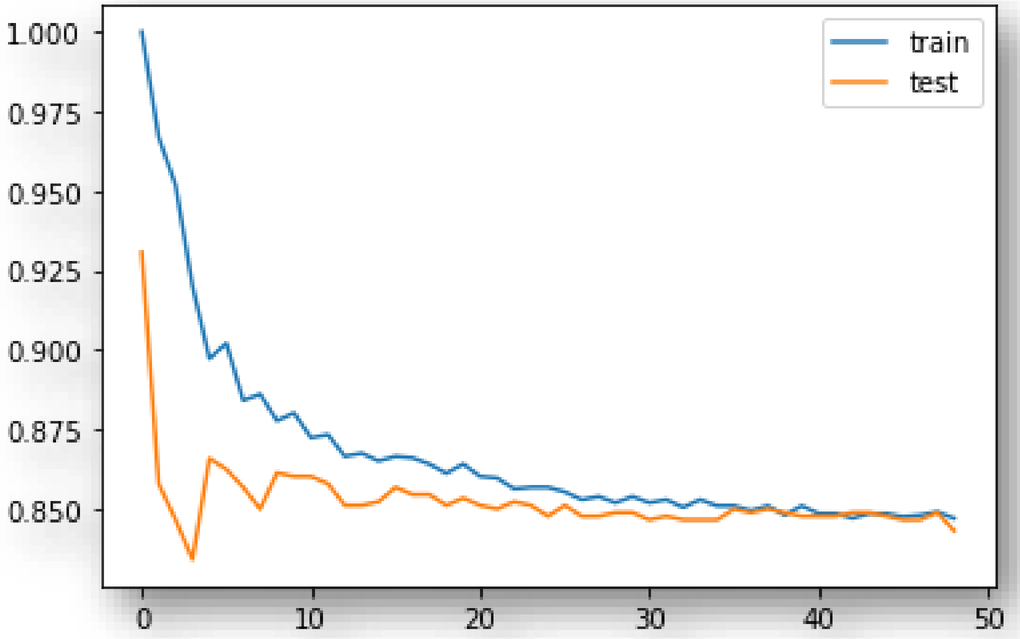
K- NEAREST NEIGHBOURS

- KNN is a distance based supervised machine learning algorithm that can be use to solve both classification and regression problems. It becomes very slow when we deal with large amount of data.
- For this classifier scaling data is necessary. KNN is a distance base algorithm, so we have scaled our data.
- Here, I have use z-score for scaling our data.

TABLE 27: VARIANCE

	Test	Train
0	0.930603	1.000000
1	0.857793	0.966845
2	0.846416	0.951731
3	0.833902	0.920039
4	0.865757	0.897123
5	0.862344	0.901999
6	0.856655	0.883959
7	0.849829	0.885909
8	0.861206	0.877621
9	0.860068	0.880059

FIGURE 7: VARIANCE PLOT



- From above table we can see that after 5th neighbors it's not changing that much. Variance is not changing that significantly for both Train and Test. So, our model will perform well if we build our model with 5 number of neighbors.

TABLE 28: KNN - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.89	0.05	0.10	329
1	0.85	1.00	0.92	1722
accuracy			0.85	2051
macro avg	0.87	0.53	0.51	2051
weighted avg	0.85	0.85	0.79	2051

0.8469039492930278

TABLE 29: KNN - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.88	0.05	0.09	144
1	0.84	1.00	0.91	735
accuracy			0.84	879
macro avg	0.86	0.52	0.50	879
weighted avg	0.85	0.84	0.78	879

FIGURE 8: KNN – CONFUSION MATRIX TRAIN DATA

	Predict 1	Predict 0
1	17	312
0	2	1720

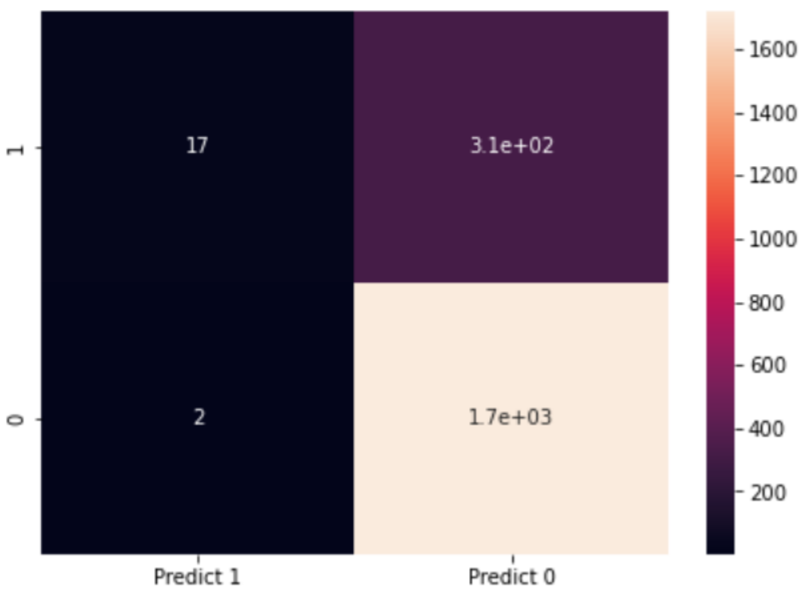
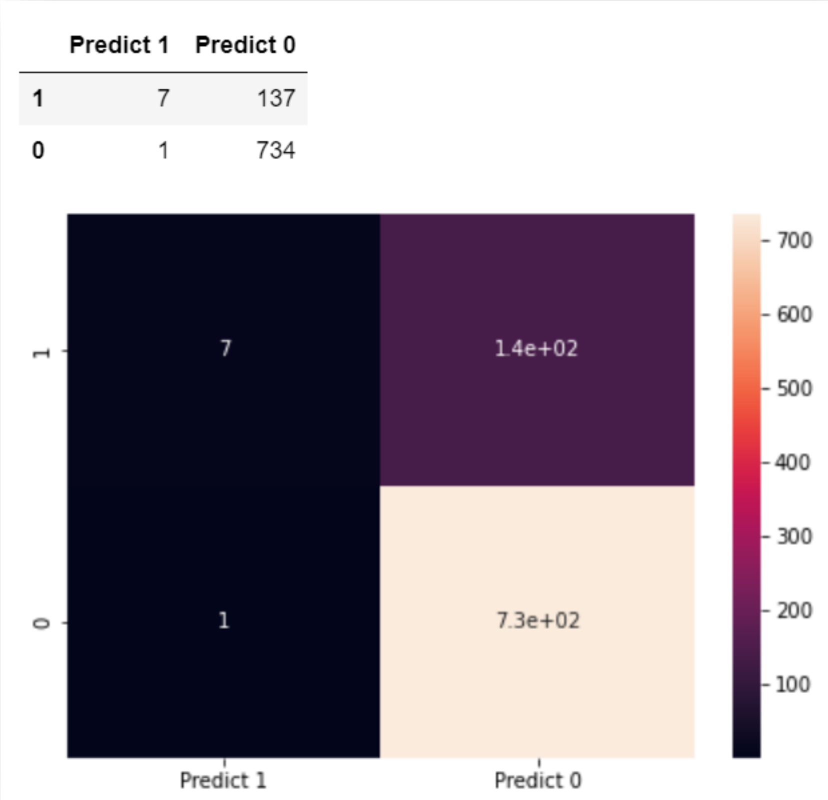


FIGURE 9: KNN – CONFUSION MATRIX TEST DATA



INSIGHT

- KNN model perform well.
- For the Training set we got Accuracy= 87 %, Precision= 80 %, recall= 66 %, f1-score= 70 %.
- For the Testing set we got Accuracy= 87 %, Precision= 79 %, recall= 67 %, f1-score= 71 %.

NAÏVE BAYES MODEL

- Naïve Bayes' classifiers is a model based on applying Bayes' theorem with strong independent assumption between the features.
- Here the method that we are going to use is the Gaussian NB method, also known as Bernoulli. A general assumption in this method is the data is following a normal distribution.

TABLE 30: NAÏVE BAYES - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.34	0.38	0.36	329
1	0.88	0.86	0.87	1722
accuracy			0.78	2051
macro avg	0.61	0.62	0.61	2051
weighted avg	0.79	0.78	0.79	2051

```
array([[ 126,  203],
       [ 245, 1477]], dtype=int64)
```

TABLE 31: NAÏVE BAYES - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.40	0.40	0.40	144
1	0.88	0.88	0.88	735
accuracy			0.80	879
macro avg	0.64	0.64	0.64	879
weighted avg	0.80	0.80	0.80	879

```
array([[ 58,  86],
       [ 86, 649]], dtype=int64)
```

FIGURE 10: NAÏVE BAYES – CONFUSION MATRIX TRAIN DATA

	Predict 1	Predict 0
1	17	312
0	2	1720

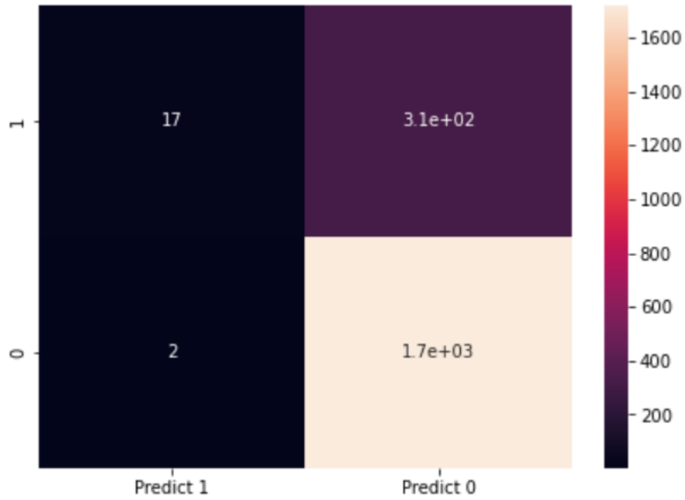
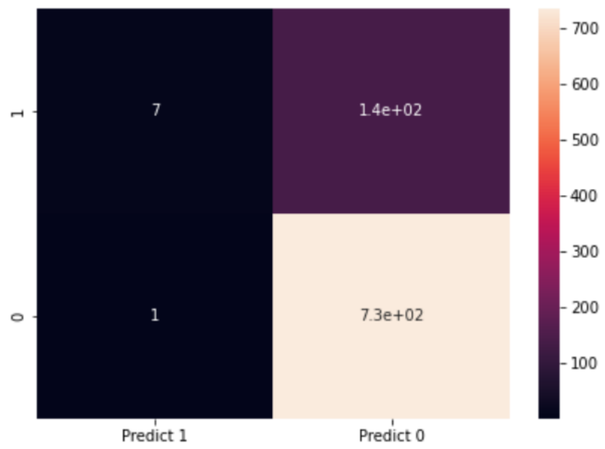


FIGURE 11: NAÏVE BAYES – CONFUSION MATRIX TEST DATA

	Predict 1	Predict 0
1	7	137
0	1	734



INSIGHTS

- Naïve Bayes' model performs well.
- For the Training set we got Accuracy= 76 %, Precision= 61 %, recall= 65 %, f1-score= 76 %.
- For the Testing set we got Accuracy= 78 %, Precision= 65 %, recall= 70 %, f1-score= 78 %.
- Surprisingly our recall and precision have increased in the test set.

TABLE 32: DESCISION TREE - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	1.00	1.00	1.00	329
1	1.00	1.00	1.00	1722
accuracy			1.00	2051
macro avg	1.00	1.00	1.00	2051
weighted avg	1.00	1.00	1.00	2051

```
[[ 329    0]
 [   0 1722]]
```

TABLE 33: DESCISION TREE - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.70	0.78	0.74	144
1	0.96	0.93	0.94	735
accuracy			0.91	879
macro avg	0.83	0.86	0.84	879
weighted avg	0.91	0.91	0.91	879

FIGURE 12: DECISION TREE– CONFUSION MATRIX TRAIN DATA

	Predict 1	Predict 0
1	329	0
0	0	1722

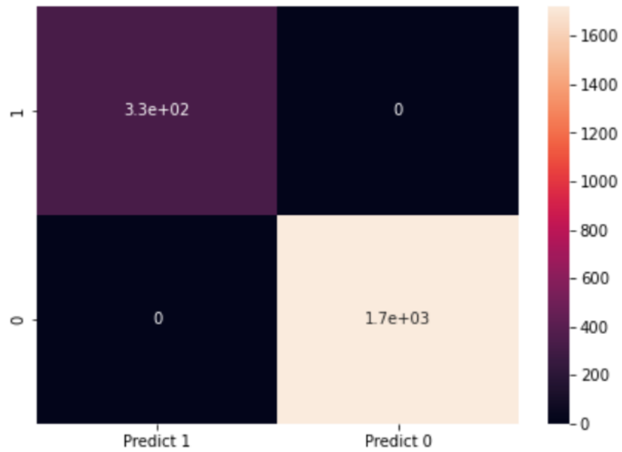


FIGURE 13: DECISION TREE– CONFUSION MATRIX TEST DATA

	Predict 1	Predict 0
1	113	31
0	49	686

- We can clearly see that our train set is over fitted. We can solve this issue by using pruning technique or by selecting important variable.
- For the Training set we got Accuracy= 100 %, Precision= 100 %, recall= 100 %, f1-score= 100 %.
- For the Testing set we got Accuracy= 89 %, Precision= 79 %, recall= 84 %, f1-score= 81 %.
- Important features are:

	imp
Audience_number	0.177754
player_highest_run	0.097720
Extra_bowls_bowled	0.093063
Season_Summer	0.071020
Max_run_scored_1over	0.059985
Offshore	0.054954
All_rounder_in_team	0.050431
extra_bowls_opponent	0.045109
Max_wicket_taken_1over	0.036960
Min_run_scored_1over	0.036248
Opponent_South Africa	0.033962
Bowlers_in_team	0.032577
Avg_team_Age	0.027182
Min_run_given_1over	0.023085
Max_run_given_1over	0.022397
Players_scored_zero_4	0.021311
Opponent_Kenya	0.019748
Match_light_type_Day and Night	0.018766

Opponent_Srilanka	0.014151
Match_format_Test	0.013810
Players_scored_zero_3	0.009585
Opponent_Zimbabwe	0.008340
Season_Winter	0.006366
First_selection	0.006239
Players_scored_zero_2	0.005913
Opponent_Pakistan	0.005672
Match_light_type_Night	0.003430
Opponent_Bangladesh	0.002413
Match_format_T20	0.001810
Opponent_England	0.000000
Opponent_West Indies	0.000000
player_highest_wicket_2	0.000000
player_highest_wicket_3	0.000000
player_highest_wicket_4	0.000000
player_highest_wicket_5	0.000000

RANDOM FOREST

- Random Forest an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.

TABLE 34: RF - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	1.00	1.00	1.00	329
1	1.00	1.00	1.00	1722
accuracy			1.00	2051
macro avg	1.00	1.00	1.00	2051
weighted avg	1.00	1.00	1.00	2051

```
[[ 329    0]
 [   0 1722]]
```

TABLE 35: RF - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.94	0.72	0.82	144
1	0.95	0.99	0.97	735
accuracy			0.95	879
macro avg	0.94	0.86	0.89	879
weighted avg	0.95	0.95	0.94	879

```
[[104  40]
 [  7 728]]
```

INSIGHTS

- We can clearly see that our train set is over fitted. We can solve this issue by using pruning technique or by selecting important variable that I will do in further model tunning part.
- For the Training set we got Accuracy= 100 %, Precision= 100 %, recall= 100 %, f1-score= 100 %.
- For the Testing set we got Accuracy= 95 %, Precision= 95 %, recall= 85 %, f1-score= 89 %.

TABLE 36: ANN - CLASSIFICATION REPORT TRAIN DATA

- Artificial neural networks (ANNs) use learning algorithms that can independently adjust or learn, in a sense as they have received new input.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	329
1	0.84	1.00	0.91	1722
accuracy			0.84	2051
macro avg	0.42	0.50	0.46	2051
weighted avg	0.70	0.84	0.77	2051

TABLE 37: ANN - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.00	0.00	0.00	144
1	0.84	1.00	0.91	735
accuracy			0.84	879
macro avg	0.42	0.50	0.46	879
weighted avg	0.70	0.84	0.76	879

- For the Training set we got Accuracy= 84 %, Precision= 42 %, recall= 50 %, f1-score= 46 %.
- For the Testing set we got Accuracy= 84 %, Precision= 42 %, recall= 50 %, f1-score= 46 %.

MODEL TUNNING AND ENSEMBLE METHOD

- Tuning is the process of maximizing a model's performance without overfitting or creating too high of a variance. In machine learning, this is accomplished by selecting appropriate "hyper-parameters"
- Grid Search is one of the most common methods of optimizing the parameters
- Models such as Bagging, Boosting, Gradient boosting, etc are prone to under or over fitting of data. Overfit means the model work well in train data but work relatively poor in test dataset. Underfit is vice-versa of overfitting model.

LOGISTIC REGRESSION WITH HYPER PARAMETERS

- Before fitting the model, it is important to know about the hyper parameters that is involved in model building.

- Penalty
- Solver
- Max_iter
- tol, etc.

- To find the best combination among these parameters we will use the "GridSearchCV" method. This method can perform multiple combinations of these parameters simultaneously and can provide us with the best optimum results.

TABLE 38: LR - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.76	0.34	0.47	329
1	0.89	0.98	0.93	1722
accuracy			0.88	2051
macro avg	0.82	0.66	0.70	2051
weighted avg	0.87	0.88	0.86	2051

```
: array([[ 111,  218],
        [  35, 1687]], dtype=int64)
```

TABLE 39: LR - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	0.69	0.33	0.45	144
1	0.88	0.97	0.92	735
accuracy			0.87	879
macro avg	0.78	0.65	0.69	879
weighted avg	0.85	0.87	0.85	879

```
: array([[ 48,  96],
        [ 22, 713]], dtype=int64)
```

INSIGHT

- The model performs well for both train and test set with hyper parameter as compared to without tunning.
- For the Training set we got Accuracy= 88 %, Precision= 83 %, recall= 66 %, f1-score= 70 %.
- For the Testing set we got Accuracy= 87 %, Precision= 79 %, recall= 66 %, f1-score= 46 %.

LDA WITH HYPER PARAMETERS

TABLE 40: LDA - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.72	0.37	0.49	329
1	0.89	0.97	0.93	1722
accuracy			0.88	2051
macro avg	0.80	0.67	0.71	2051
weighted avg	0.86	0.88	0.86	2051

0.8756704046806436

TABLE 41: LDA - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	0.71	0.35	0.47	144
1	0.88	0.97	0.93	735
accuracy			0.87	879
macro avg	0.80	0.66	0.70	879
weighted avg	0.86	0.87	0.85	879

0.8703071672354948

INSIGHT

- After turning our LDA model we can see that there is no change in the performance of the model.
- For the Training set we got Accuracy= 87 %, Precision= 80 %, recall= 66 %, f1-score= 70 %.
- For the Testing set we got Accuracy= 87 %, Precision= 78 %, recall= 66 %, f1-score= 70 %.

DESCISION TREE WITH HYPER PARAMETERS

TABLE 42: DT - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.75	0.17	0.27	329
1	0.86	0.99	0.92	1722
accuracy			0.86	2051
macro avg	0.81	0.58	0.60	2051
weighted avg	0.84	0.86	0.82	2051

```
[[  55  274]
 [  18 1704]]
```

TABLE 43: DT - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	0.75	0.12	0.21	144
1	0.85	0.99	0.92	735
accuracy			0.85	879
macro avg	0.80	0.56	0.57	879
weighted avg	0.84	0.85	0.80	879

```
[[ 18 126]
 [  6 729]]
```

INSIGHT

- After tunning the CART model, we succeed to come up with overfitting issue.
- For the Training set we got Accuracy= 86 %, Precision= 80 %, recall= 58 %, f1-score= 60 %.
- For the Testing set we got Accuracy= 85 %, Precision= 81 %, recall= 56 %, f1-score= 57 %.

RANDOM FOREST WITH HYPER PARAMETERS

TABLE 44: RF - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.75	0.17	0.27	329
1	0.86	0.99	0.92	1722
accuracy			0.86	2051
macro avg	0.81	0.58	0.60	2051
weighted avg	0.84	0.86	0.82	2051

```
[[  55  274]
 [  18 1704]]
```

TABLE 45: RF - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	0.75	0.12	0.21	144
1	0.85	0.99	0.92	735
accuracy			0.85	879
macro avg	0.80	0.56	0.57	879
weighted avg	0.84	0.85	0.80	879

```
[[ 18 126]
 [  6 729]]
```

INSIGHT

- After tunning the Random Forest model, we succeed to come up with overfitting issue.
- For the Training set we got Accuracy= 87 %, Precision= 91 %, recall= 60 %, f1-score= 63 %.
- For the Testing set we got Accuracy= 86 %, Precision= 91 %, recall= 58 %, f1-score= 59 %.

NAÏVE BAYES WITH HYPER PARAMETERS

TABLE 46: NAÏVE BAYES - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.25	0.84	0.38	329
1	0.94	0.51	0.66	1722
accuracy			0.56	2051
macro avg	0.60	0.68	0.52	2051
weighted avg	0.83	0.56	0.62	2051

```
array([[277,  52],
       [842, 880]], dtype=int64)
```

TABLE 47: NAÏVE BAYES - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	0.25	0.83	0.39	144
1	0.94	0.52	0.67	735
accuracy			0.57	879
macro avg	0.60	0.68	0.53	879
weighted avg	0.83	0.57	0.62	879

```
array([[120,  24],
       [354, 381]], dtype=int64)
```

INSIGHT

- After scaling the data our NB model start performing poor.
- For the Training set we got Accuracy= 58 %, Precision= 59 %, recall= 67 %, f1-score= 53 %.
- For the Testing set we got Accuracy= 59 %, Precision= 60 %, recall= 69 %, f1-score= 55 %.

BAGGING USING RANDOM FOREST

- Bagging is an ensemble technique. Ensemble techniques are the machine learning techniques that combine several base models to get an optimal model.
- Bagging is designed to improve the performance of existing machine learning algorithms used in statistical classification or regression.
- Here, we will use random forest as the base classifier. We use Hyper-parameters that we have obtain from Grid Search.

TABLE 48: BAGGING - CLASSIFICATION REPORT TRAIN DATA - TUNED

	precision	recall	f1-score	support
0	0.94	0.09	0.17	329
1	0.85	1.00	0.92	1722
accuracy			0.85	2051
macro avg	0.89	0.55	0.54	2051
weighted avg	0.87	0.85	0.80	2051

```
[[  30  299]
 [   2 1720]]
```

TABLE 49: BAGGING - CLASSIFICATION REPORT TEST DATA - TUNED

	precision	recall	f1-score	support
0	1.00	0.08	0.14	144
1	0.85	1.00	0.92	735
accuracy			0.85	879
macro avg	0.92	0.54	0.53	879
weighted avg	0.87	0.85	0.79	879

```
[[ 11 133]
 [  0 735]]
```

INSIGHT

- Bagging performs good but not good as compared to simple random forest model.
- For the Training set we got Accuracy= 86 %, Precision= 87 %, recall= 56 %, f1-score= 56 %.
- For the Testing set we got Accuracy= 85 %, Precision= 89 %, recall= 55 %, f1-score= 55 %.

BOOSTING

- Boosting is also an ensemble technique. It converts weak learners to strong learners.
- Each time base learning algorithm is applied, it generates a new weak learner prediction rule. This is an iterative process, and the boosting algorithm combines these weak rules into a single strong prediction rule.
- There are many types of Boosting techniques. We are going to use following technique for this case study.
  1. ADA Boosting.
  2. Gradient Boosting.

ADA - BOOSTING

- This model is used to increase the efficiency of binary classifiers, but now used to improve multiclass classifiers as well.

TABLE 50: ADA BOOSTING - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.78	0.39	0.52	329
1	0.89	0.98	0.93	1722
accuracy			0.88	2051
macro avg	0.84	0.68	0.73	2051
weighted avg	0.87	0.88	0.87	2051

```
[[ 127  202]
 [  36 1686]]
```

TABLE 51: ADA BOOSTING - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.70	0.33	0.45	144
1	0.88	0.97	0.92	735
accuracy			0.87	879
macro avg	0.79	0.65	0.69	879
weighted avg	0.85	0.87	0.85	879

```
[[ 48  96]
 [ 21 714]]
```

INSIGHT

- ADA Boosting is performing good as compare to another model.
- For the Training set we got Accuracy= 88 %, Precision= 82 %, recall= 67 %, f1-score= 71 %.
- For the Testing set we got Accuracy= 87 %, Precision= 82 %, recall= 66 %, f1-score= 70 %.

GRADIENT BOOSTING

- This model is just like the ADA Boosting model. Gradient Boosting works by sequentially adding the misidentified predictors and under-fitted predictions to the ensemble, ensuring the errors identified previously are corrected.
- This method tries to fit the new predictor to the residual errors made by the previous one.

TABLE 52: GRADIENT BOOSTING - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.96	0.57	0.71	329
1	0.92	1.00	0.96	1722
accuracy			0.93	2051
macro avg	0.94	0.78	0.84	2051
weighted avg	0.93	0.93	0.92	2051

```
[[ 187  142]
 [   8 1714]]
```

TABLE 53: GRADIENT BOOSTING - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.89	0.44	0.59	144
1	0.90	0.99	0.94	735
accuracy			0.90	879
macro avg	0.89	0.71	0.76	879
weighted avg	0.90	0.90	0.88	879

```
[[ 63  81]
 [  8 727]]
```

INSIGHT

- Gradient boosting model is performing good for this classification problem.
- For the Training set we got Accuracy= 93 %, Precision= 94 %, recall= 81 %, f1-score= 86 %.
- For the Testing set we got Accuracy= 90 %, Precision= 89 %, recall= 70 %, f1-score= 75 %.

SMOTE

- SMOTE (Synthetic Minority over sampling Technique) is used when we encounter with data imbalance problem.
- We know that we were having data imbalance in our target variable, so we can also.

TABLE 54: SMOTE - CLASSIFICATION REPORT TRAIN DATA

	precision	recall	f1-score	support
0	0.96	0.89	0.92	1722
1	0.90	0.96	0.93	1722
accuracy			0.93	3444
macro avg	0.93	0.93	0.93	3444
weighted avg	0.93	0.93	0.93	3444

```
: array([[1530,  192],
        [  64, 1658]], dtype=int64)
```

TABLE 55: SMOTE - CLASSIFICATION REPORT TEST DATA

	precision	recall	f1-score	support
0	0.60	0.45	0.52	144
1	0.90	0.94	0.92	735
accuracy			0.86	879
macro avg	0.75	0.70	0.72	879
weighted avg	0.85	0.86	0.85	879

```
array([[ 65,  79],
       [ 43, 692]], dtype=int64)
```

INSIGHT

- From above table using SMOTE technique doesn’t increase the performance of the model.
- Smote technique performing good on training set but underfitting in the test set it can be eliminated by dropping non important variable.
- For the Training set we got Accuracy= 93 %, Precision= 91 %, recall= 93 %, f1-score= 93 %.
- For the Testing set we got Accuracy= 86 %, Precision= 75 %, recall= 70 %, f1-score= 72 %.

TABLE 56: ALL MODEL COMPARISION

	LR-TRAIN	LR-TEST	LR(Tune)-TRAIN	LR(Tune)-TEST	LDA-Train	LDA-Test	LDA(Tune)-Train	LDA(Tune)-Test	KNN-Train	KNN-Test	NB-Train	NB-Test
Accuracy	84	84	88	87	87	87	87	87	90	87	76	78
F1 Score	48	47	70	70	70	71	70	70	77	71	62	66
Recall	51	51	66	66	66	67	66	66	72	68	65	70
Precision	78	92	83	79	80	79	80	78	88	79	61	65
AUC	73	73	83	82	82	83	82	83	95	83	76	78

	CART-TR	CART-TEST	RF-TRAIN	RF-TEST	ANN-rain	ANN-Test	Bagging-Train	Bagging-Test	Ada-train	Ada-Test	GB-Train	GB-Test	Smote-rf	Smote-rf-test
Accuracy	86	85	87	86	84	84	88	87	88	87	93	90	91	83
F1 Score	60	57	63	59	46	46	70	70	71	70	85	76	91	70
Recall	58	56	60	58	50	50	66	66	67	66	80	70	91	71
Precision	80	81	91	91	42	42	83	79	82	82	94	89	91	70
AUC	83	82	92	89	76	75	90	87	87	85	95	89	96	83

- In this classification problem the most important measurement matrix we see is Recall, precision, accuracy, and F1-Score.
- In this case, precision is the total predicted win and loss. Recall is total Actually win and loss.
- F1- score is the harmonic mean of precision and recall.
- In this case our most important matrix is Recall because we must predict winning for the Indian team and must reduce the false positive rate.
- Comparing all models, going with ‘Gradient Boosting Model’ for this Case study.
- Gradient Boost Model have less False ‘+ve’ and False ‘-ve’ for both win and loss Classes. Compare to other model it has Higher Precision, Recall and Accuracy for both Train and Test.

BUSINESS RECOMMENDATION

1. Try to collect more some more predictor, like total score, bowling style etc. for better Model.
2. Try to add more than 3 all-rounders in the team that will improve the team performance.
3. If team opt for bowling first with an Avg team age of 30, with 4 bowlers in the team has higher chance to win against England in test match in Rainy season in England
4. If team opt for bowling first with an Avg team age of 30, minimum 3 bowlers in the team, scoring average 15 runs per over has higher chance to win against Australia in T20 match in Winter season in India.
5. If team opt for Batting first with an Avg team age of 30, with 3 bowlers in the team and at least one player should score century has higher chance to win against Sri Lanka in ODI match in Winter season in India.

THE END