A project report on

IPL MATCH PREDICTIONS

by

Rahul (rr3687)

for

Machine Learning (CSGY – 6923)

Submitted to Prof. Baris Coskun

ABSTRACT

IPL MATCH PREDICTIONS

Cricket is one of the most popular sports in the world. Indian Premier League (IPL), a sports league was contested during April and May every year by the teams representing the Indian cities. Cricket, especially the Twenty20 format, has maximum uncertainty, where a single over can completely change the momentum of the game. With millions of people following the Indian Premier League (IPL), analyzing the matches and developing a model for predicting the score of its matches is a real-world problem. A cricket match depends upon various factors, and in this work, the factors which significantly influence the score outcome of an IPL match were identified. Machine learning models were trained and used for predicting the total runs of the 1st inning of an IPL match.

Machine Learning tools predict future trends and behaviors. Data cleaning and analysis algorithms have been applied to the IPL dataset and the knowledge from each algorithm have been obtained and analyzed thoroughly as the results were obtained with good accuracy performance.

This project consists of analyzing the collected IPL data, making 1st innings score predictions, and displaying how close those predicted results were to the actual results.

KEYWORDS: Cricket, IPL, Machine learning, players, analysis, prediction

CHAPTER – I

1.1 INTRODUCTION

Indian Premier League (IPL) is a professional cricket league based on the Twenty20 format and is governed by the Board of Control for Cricket in India (BCCI). The league happens every year with participating teams' names representing various cities of India. There are many countries active in organizing Twenty20 cricket leagues, but IPL stands out as one of the most famous and renowned T20 cricket tournament in the whole world.

While most of the leagues are being hyped and team franchises are routinely losing money, IPL has stood out as an exception. As reported by *ESPN Cricinfo*, *Star Sports* spent $2.5 billion for exclusive broadcasting rights, the latest season of IPL (2018, 11th) saw a 29% increment in the number of viewers including both the digital streaming media and television. The 10th season had 130 million people streaming the league through their digital devices and 410 million people watching directly on the TV. The numbers prove that IPL is a successful Twenty20 format-based cricket league. So, it is really useful to associate analytics with this field and get conclusions that are vital in the decision-making.

Sports analytics is a promising research field that involves deriving valuable information about the game, based on past games played or even games in progress. The prediction of the runs scored by the team proves to be greatly beneficial to team members, team coaches, and also bettors. For example, game tactics can be developed by club managers based on the outcome of earlier matches or statistics related to certain players. IPL is a very dynamic league. The sports betting industry is growing at a fast rate. Bettors and bookies are incentivized to bet on the results during a match in progress.

In this project, I am using the sports analytics method of collecting and analyzing historical game information to derive essential knowledge from it, to promote successful decision making. The past IPL data (2008-2017) containing attributes such as batting team, bowling team, runs, wickets, overs, runs in the last 5 overs, wickets in the last 5 overs, etc. taken and analyzed to predict the final score of the batting team at the end of the first inning.

1.2 PROBLEM STATEMENT

The need to depend on data to draw better conclusions and decisions cannot be stressed enough in today's world. In IPL matches where massive amounts of money and time are invested, it becomes especially important to properly understand, analyze, and estimate things before concluding the results. There are questions about the application of Machine Learning in the field of cricket about predicting various outcomes of the matches. Some of these are - "Can machine learning technology derive accurate predictive models for cricket matches related to IPL and help in better decision-making? If so, what kind of analysis plays a role and which machine learning models can perform are the best-concerning accuracy measures?"

CHAPTER – II

2.1 DATASET

The required dataset was downloaded from Kaggle. It has a size of about 10 MB. The data includes ball by ball details of the first innings of the Indian Premier League (IPL) matches from the year 2008 to 2017. The dataset has 76014 rows and 15 columns.

The attributes present in the dataset are as follows:

- mid
- date
- venue
- batting_team
- bowling_team
- batsman
- bowler
- runs
- wickets
- overs
- runs_last_5
- wickets_last_5
- striker
- non-striker
- total

First five rows of the dataset

	mid	date	venue	batting_team	bowling_team	batsman	bowler	runs	wickets	overs	runs_last_5	wickets_last_5	striker	non-striker	total
0	1	2008-04-18	M Chinnaswamy Stadium	Kolkata Knight Riders	Royal Challengers Bangalore	SC Ganguly	P Kumar	1	0	0.1	1	0	0	0	222
1	1	2008-04-18	M Chinnaswamy Stadium	Kolkata Knight Riders	Royal Challengers Bangalore	BB McCullum	P Kumar	1	0	0.2	1	0	0	0	222
2	1	2008-04-18	M Chinnaswamy Stadium	Kolkata Knight Riders	Royal Challengers Bangalore	BB McCullum	P Kumar	2	0	0.2	2	0	0	0	222
3	1	2008-04-18	M Chinnaswamy Stadium	Kolkata Knight Riders	Royal Challengers Bangalore	BB McCullum	P Kumar	2	0	0.3	2	0	0	0	222
4	1	2008-04-18	M Chinnaswamy Stadium	Kolkata Knight Riders	Royal Challengers Bangalore	BB McCullum	P Kumar	2	0	0.4	2	0	0	0	222

Fig. 2.1 First five rows of the original dataset

CHAPTER – III

3.1 DATA CLEANING:

I) Irrelevant features

The irrelevant features such as match id, date, venue, batsman, bowler, striker, and non-striker were removed from the dataset, so that unnecessary details like these do not affect our model training to the predict the 1st innings score by the batting team.

II) Rows with Null values

Rows having NaN values for the relevant column attributes were removed.

III) Inconsistent teams

Since its start in 2008, various teams such as Deccan Chargers, Kochi Tuskers, Pune Warriors, Gujarat Lions, Rising Pune Supergiant, have become defunct due to various reasons. These include financial issues, internal issues with BCCI, etc. As these teams do not exist today, there is no reason to make score predictions for these teams and therefore, these teams were removed from the dataset.

IV) Removing data of first 5 overs

Rows representing data of the first 5 overs were removed because it is very hard and inaccurate to predict the final score of the team when the team has just started playing the match. Also, the rules of the IPL say that the scores of the first 5 overs must be canceled in case the match gets canceled due to rain or some other reason. Therefore, removing rows of the first 5 overs was a wise choice.

3.2 CORRELATION MATRIX:

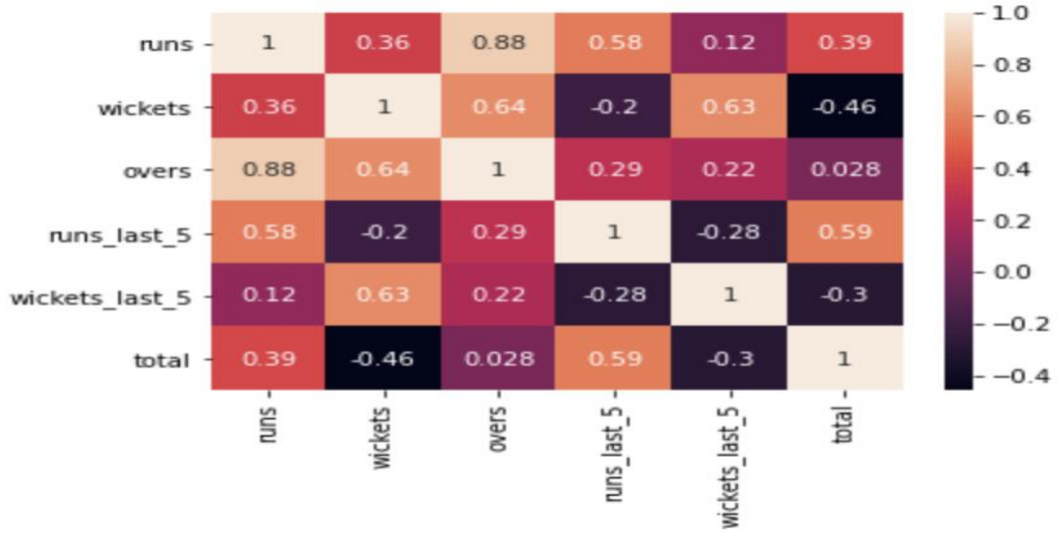

Fig. 3.1 Correlation Matrix

From the correlation matrix, we can see that *runs_last_5* (runs scored in the last 5 overs), *runs* (total runs until now) are significantly positively correlated with the *total* (total runs scored by the batting team at the end of 1st inning). While the *runs_last_5* and *runs* are positively correlated with the *total*, the *wickets* and *wickets_last_5* are negatively correlated with the *total* which is the expected behavior. As more players get out of the match i.e., more wickets, the chances of scoring greater total runs decrease.

3.3 DATA PREPROCESSING AND ENCODING:

I) Label Encoding

There were 8 teams in our dataset. Teams were label encoded so that we can have numerical values for those team names to train our model

II) One Hot Encoding and Column Transformation

After Label Encoding, the teams were one-hot encoded. After this the columns *batting_team* and *bowling_team* were transformed to the columns with the name batting_team + (name of the team) / bowling_team + (name of the team). These columns were assigned value 1 (if that team played in the match), otherwise 0 was assigned . The snapshot of the data frame after one-hot encoding is shown below.

	batting_team_Chennai Super Kings	batting_team_Delhi Daredevils	batting_team_Kings XI Punjab	batting_team_Kolkata Knight Riders	batting_team_Mumbai Indians	batting_team_Rajasthan Royals	batting_team_Royal Challengers Bangalore	batt
0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	1.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	1.0	0.0	0.0	0.0	

5 rows × 22 columns

Fig. 3.2 Dataset after one-hot encoding and column transformation

CHAPTER – IV

4.1 MODEL PERFORMANCES AND TWO MOST IMPORTANT FEATURES

I) Linear Regression

- Mean Absolute Error (MAE): 13.065
- Mean Squared Error (MSE): 303.561
- Root Mean Squared Error (RMSE): 17.422

- **Two Most Important Features**
  - overs
  - wickets

II) Lasso Regression

- Mean Absolute Error (MAE): 13.13
- Mean Squared Error (MSE): 313.033
- Root Mean Squared Error (RMSE): 17.692

- **Two Most Important Features**
  - overs
  - wickets

III) Ridge Regression

- Mean Absolute Error (MAE): 13.065
- Mean Squared Error (MSE): 303.56
- Root Mean Squared Error (RMSE): 17.422

- **Two Most Important Features**
    - overs
    - wickets

IV) Kernel Ridge Regression

- Mean Absolute Error (MAE): 13.067
- Mean Squared Error (MSE): 303.603
- Root Mean Squared Error (RMSE): 17.424

V) Elastic Net Regression

- Mean Absolute Error (MAE): 13.179
- Mean Squared Error (MSE): 316.931
- Root Mean Squared Error (RMSE): 17.802

- **Two Most Important Features**
    - overs
    - wickets

VI) Support Vector Regression

- Mean Absolute Error (MAE): 12.869
- Mean Squared Error (MSE): 312.557
- Root Mean Squared Error (RMSE): 17.679

- **Two Most Important Features**
    - runs_last_5

- batting_team_Chennai Super Kings

One important thing to mention here is that in Support Vector Regression, *batting_team_Chennai Super Kings* came out as the 2nd most important feature after *runs_last_5*. This is quite odd because this is the one-hot encoded value of the batting team for Chennai Super Kings. None of the other trained models had one-hot encoded values of the batting or bowling teams to be in their top most important features.

VII) Decision Tree Regression

- Mean Absolute Error (MAE): 3.699
- Mean Squared Error (MSE): 117.436
- Root Mean Squared Error (RMSE): 10.817

- **Two Most Important Features**
  - runs_last_5
  - runs

VIII) Gradient Boost Regression

- Mean Absolute Error (MAE): 12.464
- Mean Squared Error (MSE): 279.393
- Root Mean Squared Error (RMSE): 16.715

- **Two Most Important Features**
  - runs_last_5
  - wickets

IX) Bagging Regression

- Mean Absolute Error (MAE): 4.629
- Mean Squared Error (MSE): 65.41
- Root Mean Squared Error (RMSE): 8.045

X) Random Forest Regression

- Mean Absolute Error (MAE): 4.611
- Mean Squared Error (MSE): 59.887
- Root Mean Squared Error (RMSE): 7.734

- **Two Most Important Features**
  - runs_last_5
  - runs

XI) Multi-Layer Perceptron Regression

- Mean Absolute Error (MAE): 10.581
- Mean Squared Error (MSE): 219.629
- Root Mean Squared Error (RMSE): 14.333

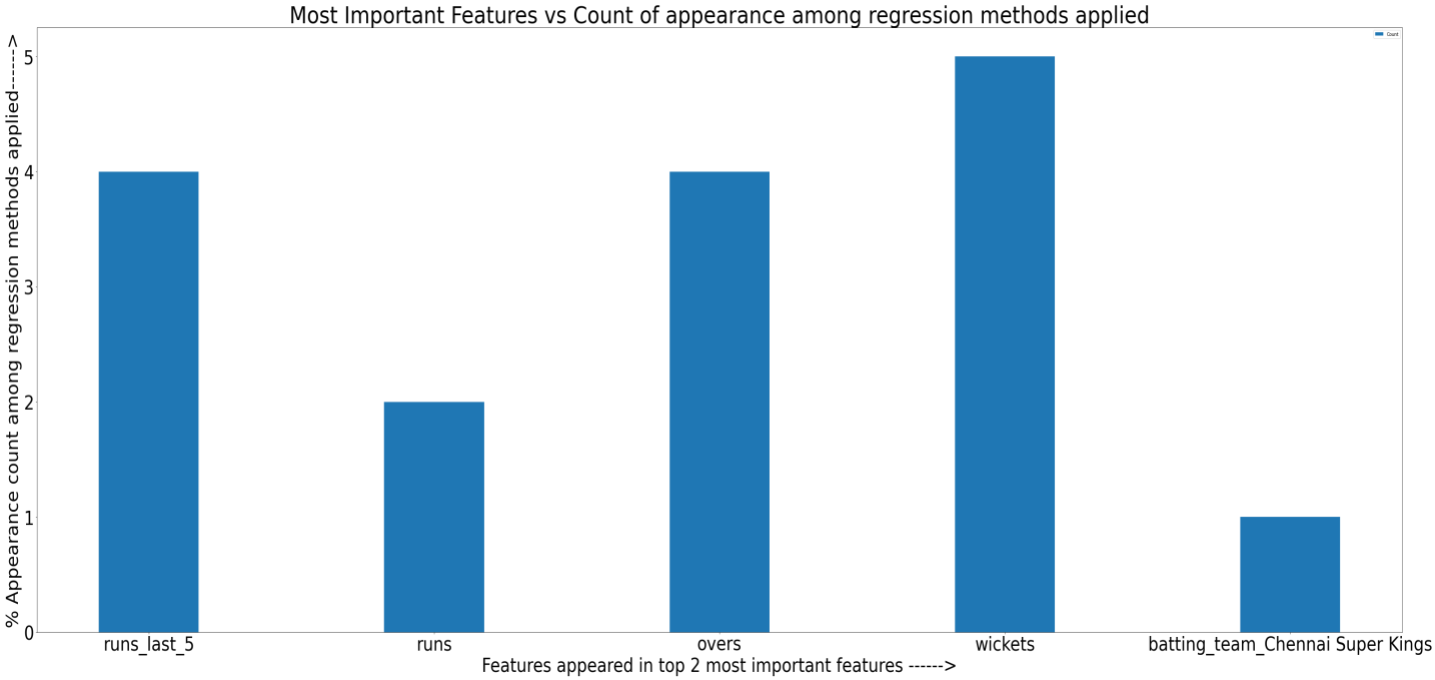

Fig. 4.1 Comparison for occurrence of the important features for multiple regression models

From the above bar graph, we can see that the *wickets* emerged as the most important feature for multiple regression algorithms applied in this project. It is closely followed by *overs* and *runs_last_5* (tie) as the second two most important features.

4.2 BEST MODEL SELECTION

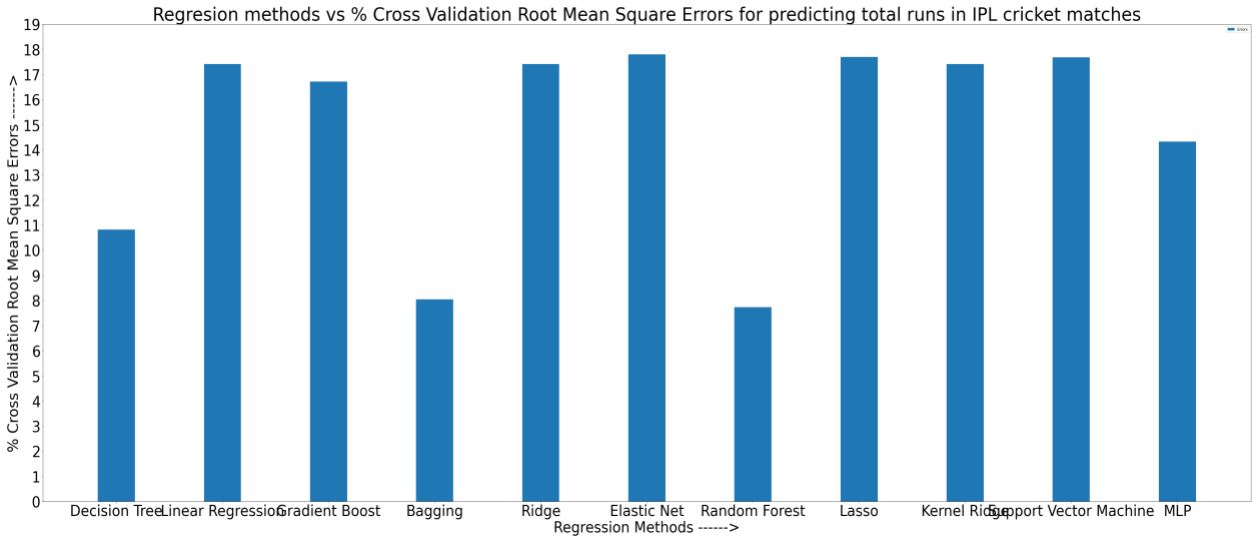

Fig. 4.2 Comparison of the RMSE values for the regression methods applied

Random Forest has the least RMSE, closely followed by Bagging, Decision Tree, and Multi-Layer Perceptron. Therefore, we chose Random Forest as our final model for making score predictions.

4.3 TEST RESULTS

Total no. of rows in testing dataframe = 8022

No. of test rows where predicted score is within a margin of 5 of the actual score

```
In [111]: len(results[results.prediction_score_with_margin_lte_5.notnull()])
Out[111]: 5137
```

No. of test rows where predicted score is within a margin between (5,10] of the actual score

```
In [112]: len(results[results.prediction_score_with_margin_gt_5_lte_10.notnull()])
Out[112]: 1617
```

No. of test rows where predicted score is within a margin between (10,15] of the actual score¶

```
In [113]: len(results[results.prediction_score_with_margin_gt_10_lte_15.notnull()])
Out[113]: 672
```

No. of test rows where predicted score has a difference greater than +/- 15 of the actual score¶

```
In [114]: len(results[results.prediction_score_with_margin_gt_15.notnull()])
Out[114]: 596
```

I) Out of a total of 8022 test rows, the score prediction for 5137 test rows was within a margin of 5 runs, the score prediction for 1617 rows was within a margin greater than 5 and less than equal to 10 runs, the score prediction for 672 rows was within a margin of greater than 10 and less than equal to 15 runs and finally, for 596 rows, the margin was greater than 15 runs.

II) It means that there are around 64% chances that the predicted score will be in a margin of 5 with the actual score. There are around 20% chances for the margin between (5, 10], and around 8% percent chances for the predicted score to be in a margin (10, 15]. Finally, there are around 7% chances for the predicted score to be in a margin greater than 15 of the actual score.

4.4 MAXIMUM AND MINIMUM DIFFERENCE BETWEEN ACTUAL AND PREDICTED SCORES

Finding the maximum difference between actual and predicted scores

```
In [115]: max_diff_row = (results['actual_score'] - results['prediction_score_with_margin_gt_15']).abs().idxmax()

In [116]: max_diff_row = results.loc[[max_diff_row]]

In [119]: max_diff_row
```

Out[119]:

_score	prediction_score_with_margin_lte_5	prediction_score_with_margin_gt_5_lte_10	prediction_score_with_margin_gt_10_lte_15	prediction_score_with_margin_gt_15
231	NaN	NaN	NaN	143.0

```
In [128]: abs(max_diff_row['actual_score'] - max_diff_row['prediction_score_with_margin_gt_15']).to_string(index=False)
```

Out[128]: ' 88.0'

Finding the minimum difference between actual and predicted scores

```
In [132]: min_diff_row = (results['actual_score'] - results['prediction_score_with_margin_lte_5']).abs().idxmin()

In [133]: min_diff_row = results.loc[[min_diff_row]]

In [134]: min_diff_row
```

Out[134]:

	batting_team	bowling_team	actual_score	prediction_score_with_margin_lte_5	prediction_score_with_margin_gt_5_lte_10	prediction_score_with_margin_gt_10_lt
12	Mumbai Indians	Delhi Daredevils	212	212.0	NaN	

```
In [135]: abs(min_diff_row['actual_score'] - min_diff_row['prediction_score_with_margin_lte_5']).to_string(index=False)
```

Out[135]: ' 0.0'

The maximum difference between actual and predicted scores noted was 88 while the minimum difference noted was 0.

4.5 EQUAL ACTUAL AND PREDICTED SCORES

Count of rows where predicted score is equal to the actual score

```
In [137]: len(results[results['actual_score'] - results['prediction_score_with_margin_lte_5'] == 0])

Out[137]: 953
```

As it can be seen above, for 953 test rows, the difference between actual and predicted scores was 0 which means that there are around 12% chances that the predicted score will be equal to the actual score.

REFERENCES

[1].Analyzing and predicting outcome of IPL cricket data, International Journal of Innovative Research in Science, Engineering and Technology - Vol. 8, Issue 4, April 2019

[2].Predictive Analysis of IPL Match Winner using ML, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-9 Issue-2S, December 2019

[3].https://www.dexlabanalytics.com/blog/how-stat-this-ipl-season-embrace-big-data-analysis-and-predict-it-right

[4].https://towardsdatascience.com/analysing-ipl-data-to-begin-data-analytics-with-python-5d2f610126a

[5].https://en.wikipedia.org/wiki/Strike_rate