

# Story

*Chirag Pandya*

*June 14, 2017*

## Introduction

The Indian Premier League (IPL) is a professional cricket league in India, which is held during April and May each year. Each team plays all other teams twice in a round-robin format. At the conclusion of the first stage, the top four teams will qualify for the Playoffs. The top two teams from the league phase will play against each other and the winner goes straight to the IPL final. The losing team plays the winner of the second match between the third and fourth team.

## The Question Phase and some insights

The question which I will answer by the end of this project is “Given the previous data of IPL matches, who will win the next match?”. There are various features which would contribute to predicting this outcome. This report shows the importance of some features which may prove significant for prediction and also identifies other features which will be calculated from the data sets. The importance of all these features will be confirmed with help of t-values and Beta constants determined by the prediction model. The model will predict a winner- team1 or team2. The team number will then be mapped to the name of the team for meaningful inference.

## The Model

There are many machine learning models which can predict a categorical output. The most frequently used are logistic regression, Decision Trees and SVMs. I will be using decision trees for modelling my training set and applying the model to predict values on a test data set. The same procedure will be done for logistic regression. A comparison of the two models’ accuracy will be ascertained. The plan also involves working with SVM’s as they handle prediction well when the training dataset has limited number of observations.

## Data Sanity / Tidying

The data is procured from a trusted source yet there are some sanity checks which are required to locate errors if any. I will start with reading the two csv files “matches.csv” and “deliveries.csv”.

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##   filter, lag

## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

## 'data.frame':   577 obs. of  18 variables:
##  $ id          : int   1 2 3 4 5 6 7 8 9 10 ...
##  $ season      : int  2008 2008 2008 2008 2008 2008 2008 2008 2008 2008 ...
##  $ city        : Factor w/ 31 levels "", "Abu Dhabi",...: 4 8 11 23 22 17 15 9 15 8 ...
```

```

## $ date          : Factor w/ 407 levels "2008-04-18","2008-04-19",...: 1 2 2 3 3 4 5 6 7 8 ...
## $ team1         : Factor w/ 13 levels "Chennai Super Kings",...: 7 1 10 8 2 5 2 1 2 5 ...
## $ team2         : Factor w/ 13 levels "Chennai Super Kings",...: 12 5 3 12 7 10 3 8 10 8 ...
## $ toss_winner   : Factor w/ 13 levels "Chennai Super Kings",...: 12 1 10 8 2 5 2 8 10 8 ...
## $ toss_decision : Factor w/ 2 levels "bat","field": 2 1 1 1 1 1 1 2 2 2 ...
## $ result        : Factor w/ 3 levels "no result","normal",...: 2 2 2 2 2 2 2 2 2 2 ...
## $ dl_applied    : int  0 0 0 0 0 0 0 0 0 0 ...
## $ winner        : Factor w/ 14 levels "", "Chennai Super Kings",...: 8 2 4 13 8 11 4 2 11 6 ...
## $ win_by_runs   : int  140 33 0 0 0 0 0 6 0 66 ...
## $ win_by_wickets : int  0 0 9 5 5 6 9 0 3 0 ...
## $ player_of_match: Factor w/ 188 levels "", "A Chandila",...: 26 100 101 117 42 171 181 107 185 76 ..
## $ venue         : Factor w/ 35 levels "Barabati Stadium",...: 15 23 9 35 8 27 24 16 24 23 ...
## $ umpire1       : Factor w/ 43 levels "A Nand Kishore",...: 6 26 4 38 9 4 19 16 6 4 ...
## $ umpire2       : Factor w/ 45 levels "A Nand Kishore",...: 29 40 14 13 19 28 5 14 23 5 ...
## $ umpire3       : logi  NA NA NA NA NA NA ...

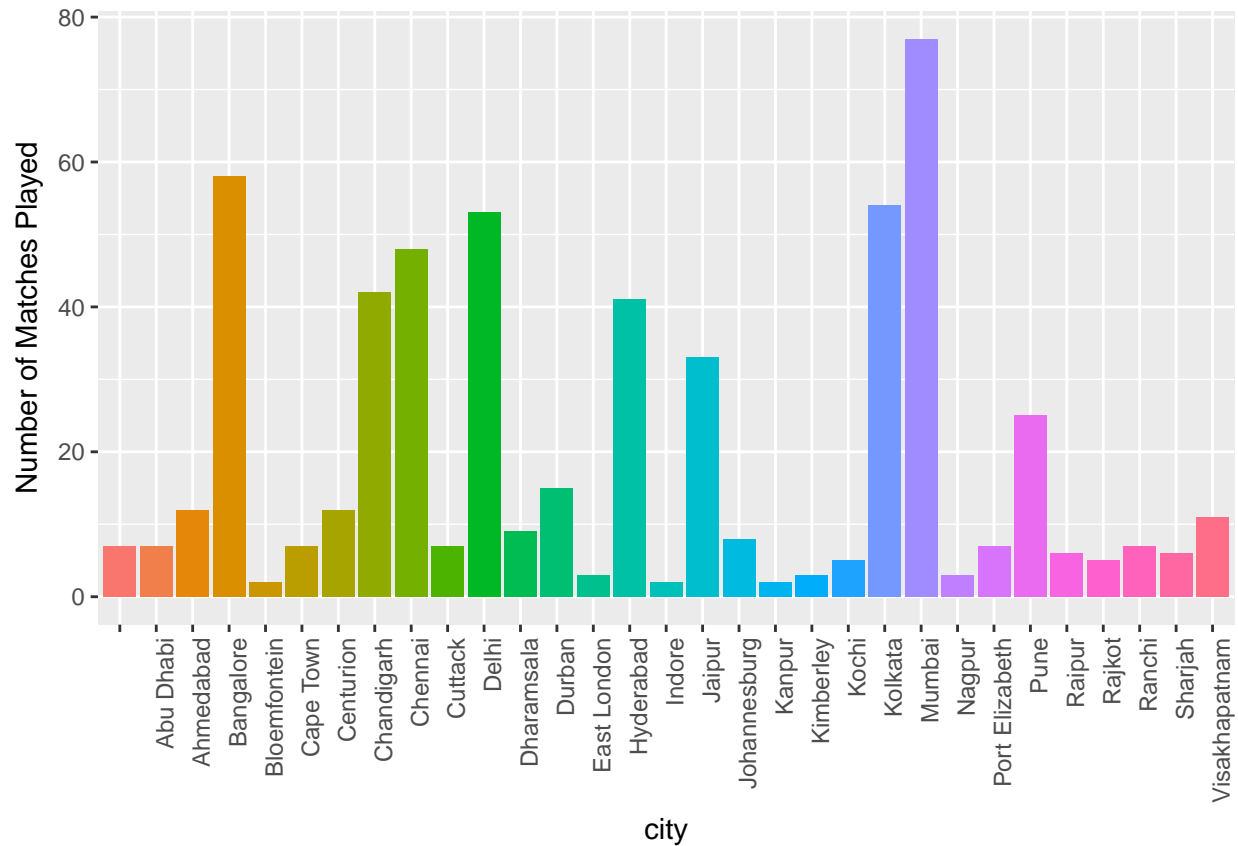
## 'data.frame': 136598 obs. of 21 variables:
## $ match_id      : int  1 1 1 1 1 1 1 1 1 1 ...
## $ inning        : int  1 1 1 1 1 1 1 1 1 1 ...
## $ batting_team   : Factor w/ 13 levels "Chennai Super Kings",...: 7 7 7 7 7 7 7 7 7 7 ...
## $ bowling_team   : Factor w/ 13 levels "Chennai Super Kings",...: 12 12 12 12 12 12 12 12 12 12 ...
## $ over          : int  1 1 1 1 1 1 1 2 2 2 ...
## $ ball          : int  1 2 3 4 5 6 7 1 2 3 ...
## $ batsman       : Factor w/ 436 levels "A Ashish Reddy",...: 354 61 61 61 61 61 61 61 61 61 ...
## $ non_striker    : Factor w/ 431 levels "A Ashish Reddy",...: 61 352 352 352 352 352 352 352 352 352 ...
## $ bowler        : Factor w/ 334 levels "A Ashish Reddy",...: 204 204 204 204 204 204 204 334 334 334 ...
## $ is_super_over  : int  0 0 0 0 0 0 0 0 0 0 ...
## $ wide_runs     : int  0 0 1 0 0 0 0 0 0 0 ...
## $ bye_runs      : int  0 0 0 0 0 0 0 0 0 0 ...
## $ legbye_runs   : int  1 0 0 0 0 0 1 0 0 0 ...
## $ noball_runs   : int  0 0 0 0 0 0 0 0 0 0 ...
## $ penalty_runs  : int  0 0 0 0 0 0 0 0 0 0 ...
## $ batsman_runs  : int  0 0 0 0 0 0 0 0 4 4 ...
## $ extra_runs    : int  1 0 1 0 0 0 1 0 0 0 ...
## $ total_runs    : int  1 0 1 0 0 0 1 0 4 4 ...
## $ player_dismissed: Factor w/ 413 levels "", "A Ashish Reddy",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ dismissal_kind : Factor w/ 10 levels "", "bowled", "caught",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ fielder       : Factor w/ 422 levels "", "A Ashish Reddy",...: 1 1 1 1 1 1 1 1 1 1 ...

```

In the “season” variable, the dataset contains data for 9 seasons. There should be 9 distinct values for season in the data set and values should range from 2008 to 2016. The following snippet is the output of the unique command.

```
## [1] 2008 2009 2010 2011 2012 2013 2014 2015 2016
```

Moving forward plotting the value of the “city” variable will show if there are any anomalies.



The plot shows that there are some blank observations in the “city” variable. On further exploration, it was found that the “city” variable was empty for matches in Dubai in the year 2014. This is rectified in the tidying phase of the project. There are less number of matches in venues located in South Africa. This is because IPL was held in South Africa instead of India in 2009.

Moving on, the “team” variables should have 13 unique values. Till date 13 teams have participated in IPL over 9 years. This is confirmed by the following unique command on “team1” and “team2” variables.

```
## [1] Kolkata Knight Riders      Chennai Super Kings
## [3] Rajasthan Royals          Mumbai Indians
## [5] Deccan Chargers           Kings XI Punjab
## [7] Royal Challengers Bangalore Delhi Daredevils
## [9] Kochi Tuskers Kerala      Pune Warriors
## [11] Sunrisers Hyderabad       Rising Pune Supergiants
## [13] Gujarat Lions
## 13 Levels: Chennai Super Kings Deccan Chargers ... Sunrisers Hyderabad

## [1] Royal Challengers Bangalore Kings XI Punjab
## [3] Delhi Daredevils           Kolkata Knight Riders
## [5] Rajasthan Royals           Mumbai Indians
## [7] Chennai Super Kings        Deccan Chargers
## [9] Pune Warriors              Kochi Tuskers Kerala
## [11] Sunrisers Hyderabad        Rising Pune Supergiants
## [13] Gujarat Lions
## 13 Levels: Chennai Super Kings Deccan Chargers ... Sunrisers Hyderabad
```

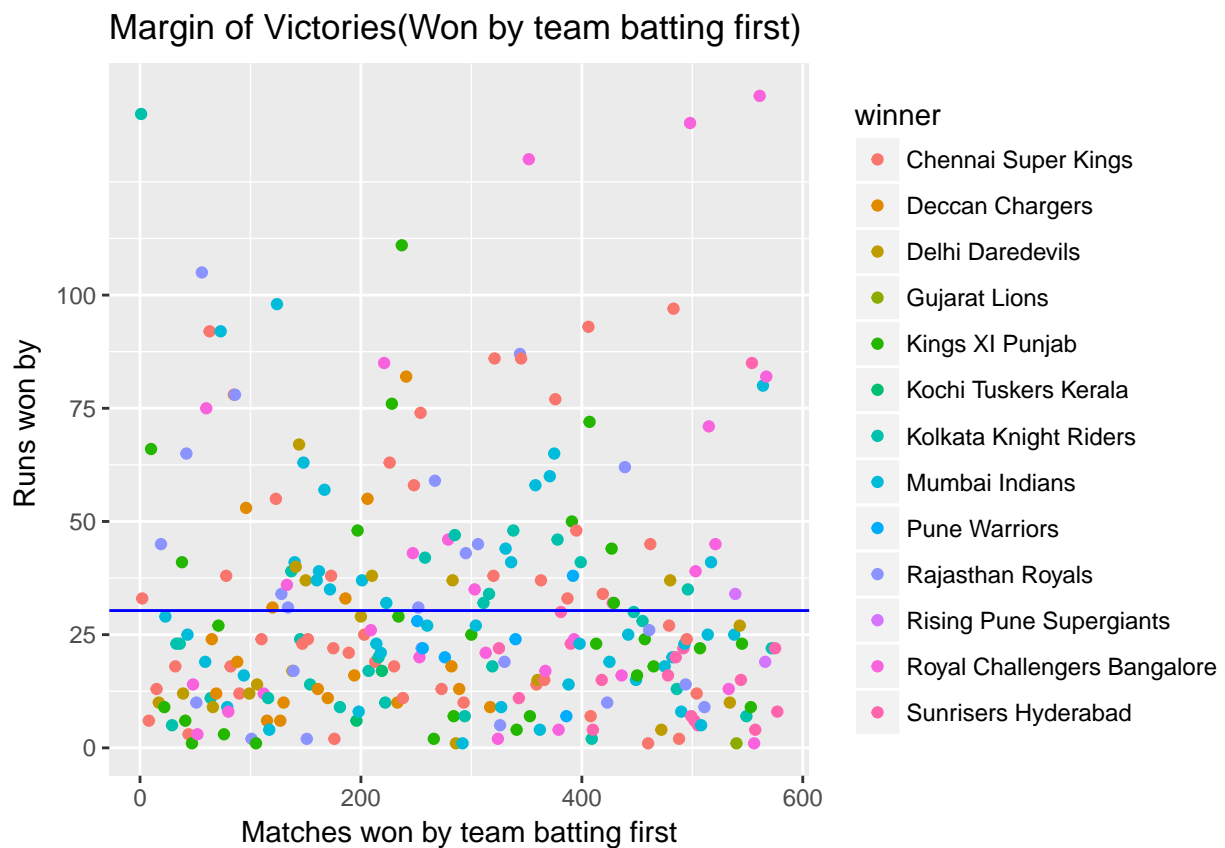
Similarly the “toss decision” variable should have a binary value of field or bat.

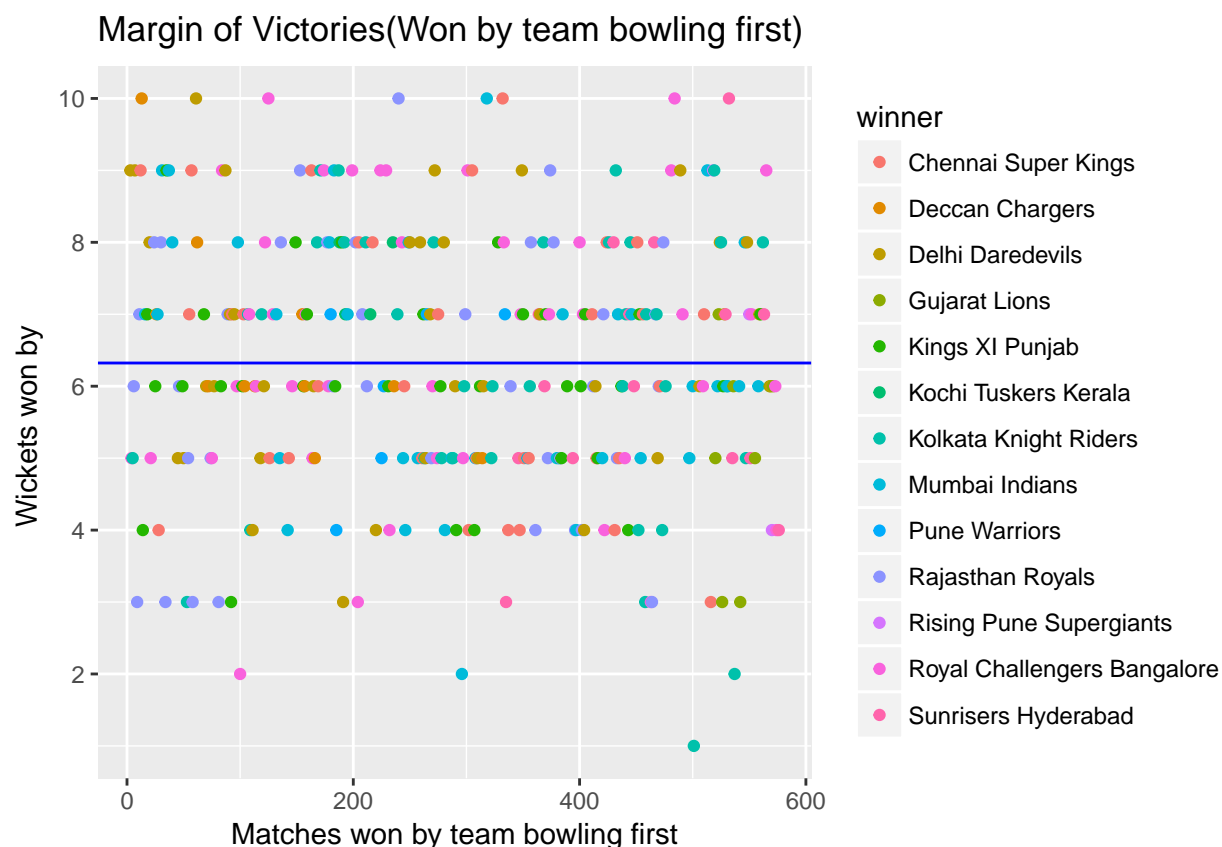
```
## [1] field bat
## Levels: bat field
```

“Winner” variable should have 14 distinct values i.e 13 for the team names and 1 value for when the match is tied.

```
## [1] Kolkata Knight Riders      Chennai Super Kings
## [3] Delhi Daredevils           Royal Challengers Bangalore
## [5] Rajasthan Royals           Kings XI Punjab
## [7] Deccan Chargers            Mumbai Indians
## [9] Pune Warriors              Kochi Tuskers Kerala
## [11]                            Sunrisers Hyderabad
## [13] Rising Pune Supergiants     Gujarat Lions
## 14 Levels: Chennai Super Kings Deccan Chargers ... Sunrisers Hyderabad
```

To find the number of outliers for the “win\_by\_runs” and “win\_by\_wickets”, a plot is generated to map these variables against number of matches.



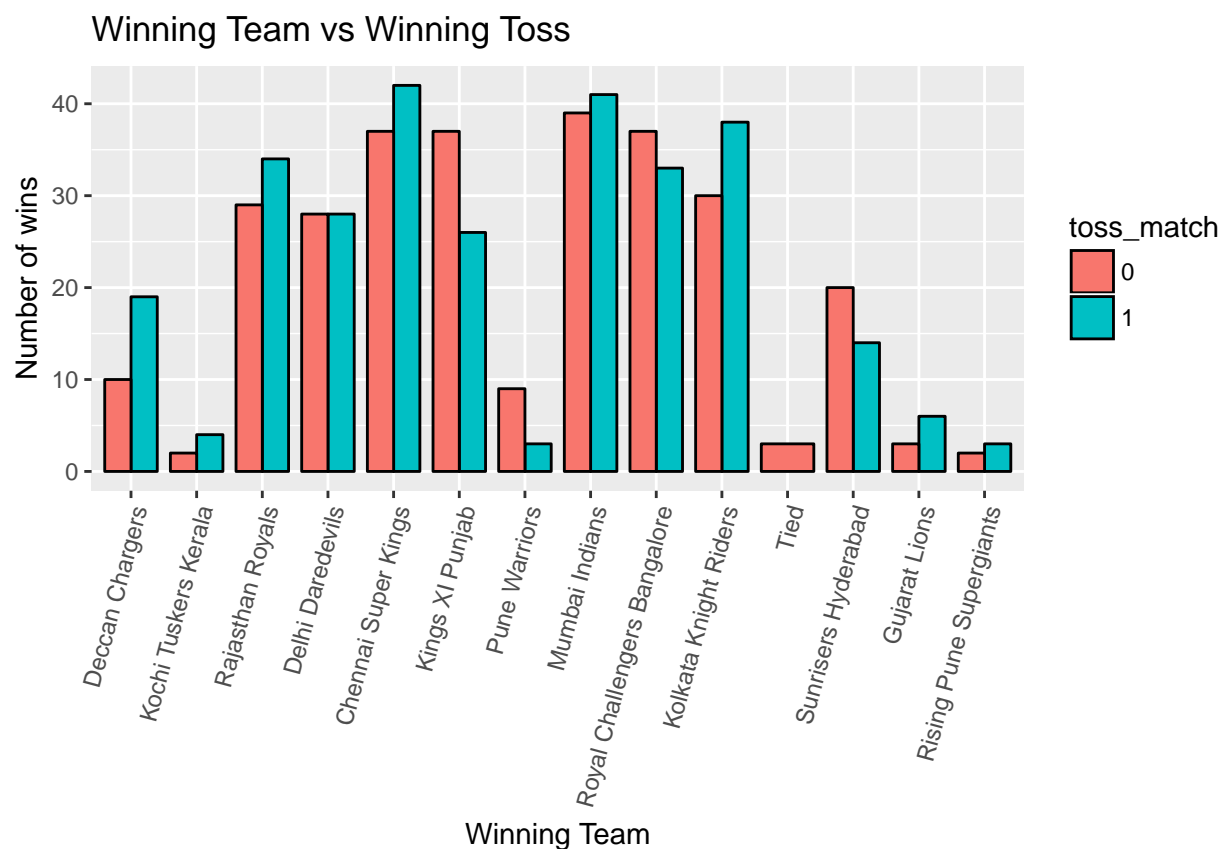


There are not more than 5 matches out of 577 which can be categorized as unusual. This is possible and hence no tidying or exploration is required.

## Exploration

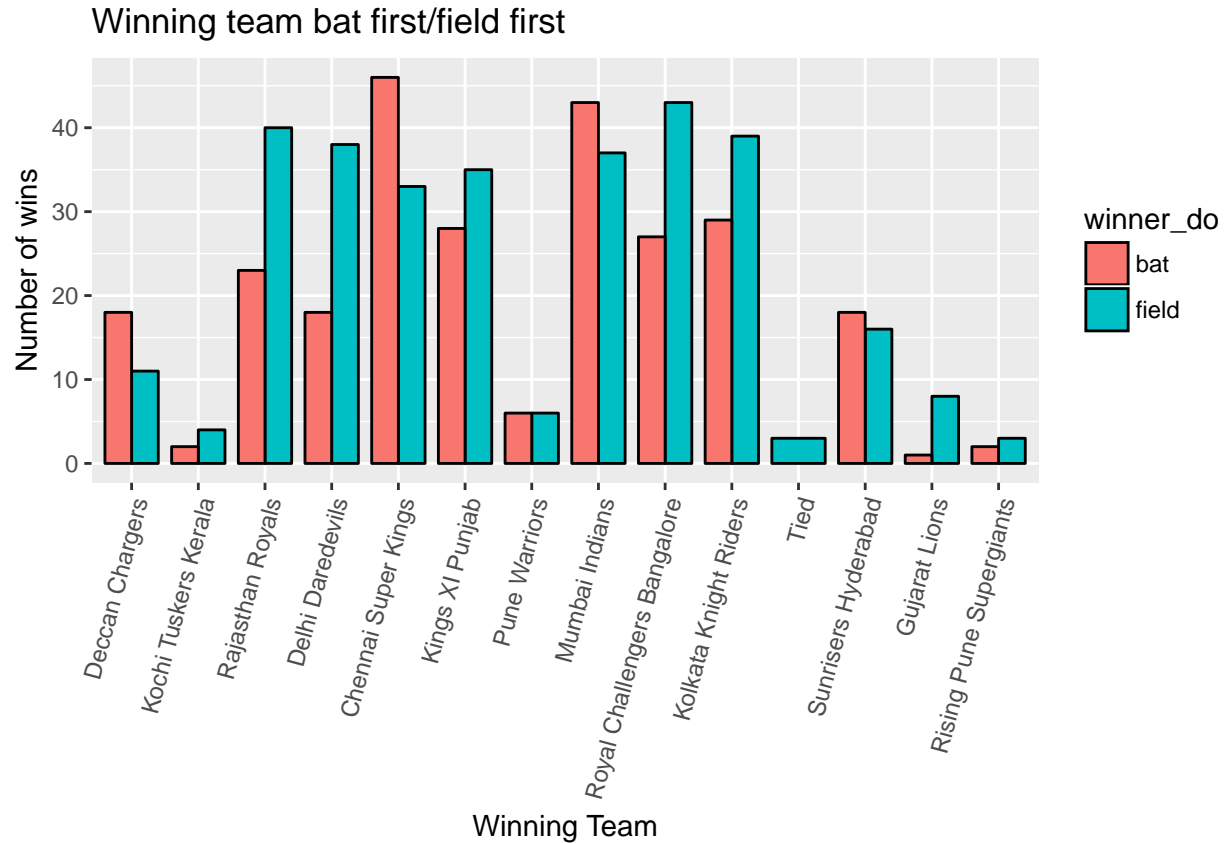
I will start exploring the most basic features which contributes to the sentiments of the winning team. Once these features are generated the machine learning model (Decision Trees) will make necessary inferences about the performance of each team.

The first feature which I explore is toss statistics. I have plotted a graph to depict the outcomes of each team on winning or losing a toss. For that we add two more variables namely “toss\_match” and “winner\_do”. “toss\_match” variable shows that the team who won the toss won the match, and “winner\_do” variable will indicate what the winning team opted for, i.e bat or field first.



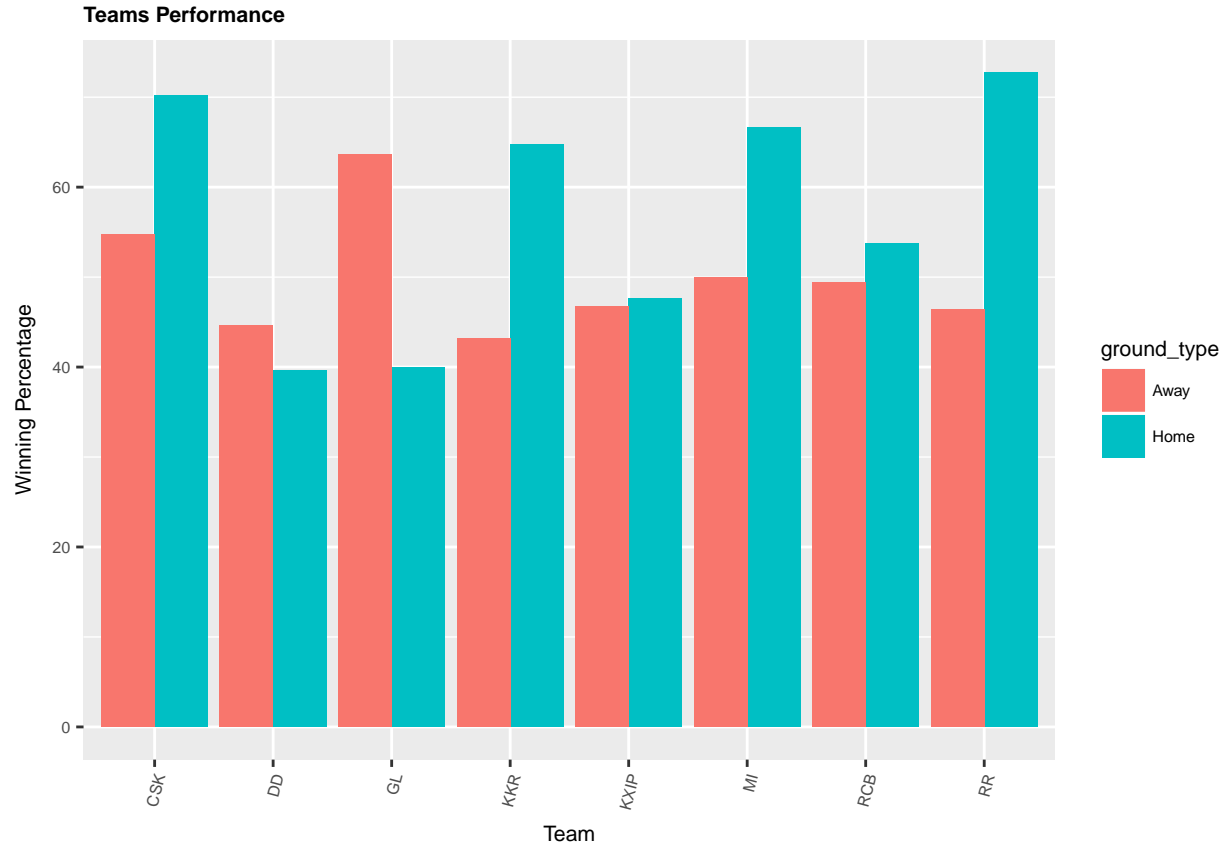
The plot shows that most teams have a higher winning percentage when they win the toss.

The next feature which I will explore is “what does a winning team elect for: Bat or Field first”. This is to determine if there is a significant relevance of winning teams opting to bat or field.



There are some very interesting inferences in this graph. The team Gujarat Lions performs better when it elects to field first. On the other hand, teams like Chennai Superkings and Deccan chargers do well when they elect to bat first. These inferences will help make deductions about the strength of each team.

Another important feature is the performance of teams on home and away grounds. It is a common notion that teams playing on their home grounds would perform better. The graph depicts the winning percentage of teams depending upon the location i.e “Home” or “Away”



Most teams perform well on their home ground, with the exception of Gujarat Lions and Delhi Daredevils. However, Gujarat Lions has played only 16 matches as it is a new team. This is the reason for the anomaly in the graph. The data is too small to make a generalized statement about the team's performance on home/away grounds. Eventually, this data will be normalized against a fixed number of matches to reduce the skew in the plot. The following table shows the total number of matches played by each team on their home ground and away ground and their winning percentages.

##	team	ground_type	total_match_played	total_win	winning_perc
## 1	CSK	Away	84	46	54.76190
## 2	CSK	Home	47	33	70.21277
## 3	DD	Away	83	37	44.57831
## 4	DD	Home	48	19	39.58333
## 5	KKR	Away	81	35	43.20988
## 6	KKR	Home	51	33	64.70588
## 7	MI	Away	80	40	50.00000
## 8	MI	Home	60	40	66.66667
## 9	KXIP	Away	92	43	46.73913
## 10	KXIP	Home	42	20	47.61905
## 11	RCB	Away	83	41	49.39759
## 12	RCB	Home	54	29	53.70370
## 13	RR	Away	84	39	46.42857
## 14	RR	Home	33	24	72.72727
## 15	GL	Away	11	7	63.63636
## 16	GL	Home	5	2	40.00000

The above performance metrics may help in predicting wins in the future. The next section is about identifying a feature vector to be provided to the model for prediction.



## Feature Engineering

The following are the suggested features which are known to directly contribute to the performance of a team. These features will be calculated and provided as input to the machine learning model which can then be used to predict future matches.

### **Season**

Depicts the year of the match Data type: int (to be converted to factor if needed)

### **city**

Data type: factor

### **team1 toss win**

Data type: binary int Possible outcomes : 1 if team1 won thte toss and 0 if it lost the toss

### **team2 toss win**

Data type: binary int Possible outcomes : 1 if team2 won thte toss and 0 if it lost the toss

### **team1 bat/field first**

Data type: binary int Possible outcomes: 1 if team1 fields first and 0 if it bats first

### **team2 bat/field first**

Data type: binary int Possible outcomes: 1 if team2 fields first and 0 if it bats first

### **Percentage of matches won by team 1**

Data type: float

### **Number of matches won by team 1**

Data type: int

### **Winning record of team 1(Number of matches won - number of matches lost)**

Data type: ternary int

positive value = 1

equal value = 0

negative value = -1

**Percentage of matches won by team 2**

Data type: float

**Number of matches won by team 2**

Data type: int

**Winning record team 2 (Number of matches won - number of matches lost)**

Data type: ternary int

positive value = 1

equal value = 0

negative value = -1

**Team1 home/away**

Data type: Binary int

home = 1

away = 0

**Team2 home/away**

Data type: Binary int

home = 1

away = 0

**Average run rate of team1**

Data type: float

**Average run rate of team2**

Data type: float

**Difference of run rate team1 - team2**

Data type: ternary int positive = 1 zero = 0 negative = -1

**Average wicket rate per over team1**

Data type: int (rounded)

**Average wicket rate per over team2**

Data type: int (rounded)