IPL DATA ANALYSIS

19BCE1322, AYUSH SINGHAL, Vellore Institute of Technology, Chennai, India

ABSTRACT

The Indian Premier Tournament (IPL) is a professional Twenty20 cricket league in India, played amongst eight teams representing eight different cities in March, April, and May of each year. Winning a cricket match is determined by a number of factors, including the team's home advantage, previous performances on that ground, records at the same venue, the players' overall experience, record against a specific opposition, and the team's overall current form, as well as the individual player's current form. The purpose of this research is to undertake a comprehensive examination of IPL data. We intend to do detailed powerplay, batsman, and bowler analyses, as well as use machine learning models to anticipate victories, runs scored, and wickets taken by a bowler, among other things. IPL match predictor is a machine learning (ML) based prediction approach in which data sets and previous stats are trained in all dimensions, including venue, runs left, balls left, wickets left, current run rate, required run rate, and so on, with each factor having a different strength, using a Logistic regression model to predict the probability of winning the match. Then, using tableau software, we'll create a dashboard that shows Team, Player, and Location data in a dynamic way.

KEYWORD

Predictive analysis, Logistic regression, Machine Learning, Tableau Dashboard, Batsmen Analysis, Forecasting

1. INTRODUCTION

Besides soccer, cricket is the most recognized and followed sport in the world, but it is the most beloved in India. Many research papers were published and a lot of work has been done in the last few years to predict the outcome of a cricket match using factors that affect the match outcome and supervised machine learning algorithms such as linear regression, support vector machines, logistic regression, decision tree, Bayes network, and random forest.

People will utilise the predictions provided by the machine learning model as technology advances and applications like as fantasy 11 and betting sites become more popular. Because the data from cricket matches is adequately labelled, unsupervised learning models are useless for our purpose. As a result, we'll employ supervised learning models.

Sports Analytics is a crucial part of the team's strategy; utilising this tool, the analyst educates the players, instructors, and other team members on how to make decisions on the field, such as whether to bat or bowl based on prior matches. Two of the most essential components of

Sports Analytics are as follows: - Analysis both on and off the field. For players to make essential judgments on the field, on-field analytics becomes critical. In analytics, data is crucial. Through this study, we aim to do complete powerplay, batsmen, bowler, and venue analysis as well as build machine learning models to predict the probability of winning by a particular team. Then using Tableau, we plan to create several dashboards providing detailed insights into IPL, player performances as well use it to forecast data.

2. LITERATURE SURVEY

A thorough search of the internet turned up relatively few publications about predicting player performance in cricket. Only a few academics have looked at the performance of cricket players. Lam and Muthuswamy [1] forecasted Indian bowlers' performance against seven foreign teams that the Indian cricket team plays the most. They utilized a backpropagation network and a radial basis network function to estimate how many runs a bowler would yield and how many wickets he would take in an ODI match.

Iyer and Sharda [2] employed neural networks to forecast players' performance, categorizing batsmen and bowlers into three groups: performer, middling, and failure. They propose whether a player should be included in the World Cup 2007 roster based on the number of times he has gotten various ratings.

Shah [3] established new metrics for evaluating players' performance. The new batsman's measure evaluates the quality of each bowler he faces, whereas the new bowler's measure considers the quality of each batter he bowls to. The overall performance index of a batter is the sum of each batsman's individual performance against each bowler. Similarly, a bowler's total performance index is the sum of his or her individual performances against each batsmen. Parker, Burns, and Natarajan are the members of the team.

Stylianos Kampakis [4] used machine learning models to forecast the results of the English Twenty Over County Cricket Cup from 2009 to 2014. He analysed the data in a multi-step process that yielded over 500 characteristics. He started with only team data and subsequently added team matched with player data. For feature selection, he employed Pearson correlation, mutual information, the chi-square test, and recursive feature reduction. Four alternative categorization techniques were used to input the selected features: naive Bayes, logistic regression, random forests, and gradient boosted decision trees. Principle component analysis was also looked at as a technique to improve the models' performance.

3. FEASIBILITY STUDY

The analysis parameters for an win prediction system and for forecasting rins, wickets and boundaries could be of any form. Here, we specifically study and build our model over our own dataset.

The most feasible solution to go through the proper analysis is to perform a exploratory data analysis for batsmen, bowler, teams with use of proper visualisation methods. For the model prediction, essential features have to identified, since our dataset didn't contain these features they will be generated based on given data. For example, to predict wins, complete match summary is generated from the given data.

4. ABOUT THE DATASET

The DATASET was gathered from www.kaggle.com. It contains details about all the matches and balls that have been played between 2008-2020. The dataset contains 2 csv files: matches.csv and delivery.csv.

matches.csv includes information about the match, such as the location, opposing teams, umpires, and outcomes.

deliveries.csv is a file that contains ball-by-ball data from all IPL matches, including information about the batting side, batsmen, bowlers, non-strikers, runs scored, and so on.

4.1 Feature components for analysis

Various attributes were used for both batsmen and bowlers in order to perform a detailed analysis. The ball_by_ball.csv was used to perform the above analysis, it contained the following parameters:

Inning
Id
over
ball
batsman
non_striker
bowler
batsman_runs
extra_runs
total_runs
non_boundary
is_wicket
dismissal_kind
player_dismissed

fielder
extras_type
batting_team
bowling_team

To do some extra batsmen analysis, I added attribute run_region which shows the the region in which runs were scored on that particular ball.

To build the win prediction model, the matches and ball_by_ball dataset were merged on match_id and new attributes were synthetically created. The resulting dataframe contained following attributes that were used to build our logistiuc regression model:

batting_team
bowling_team
city
runs_left
balls_left
wickets
total_runs_x
crr
rrr
result

1. **Design and flow of models**

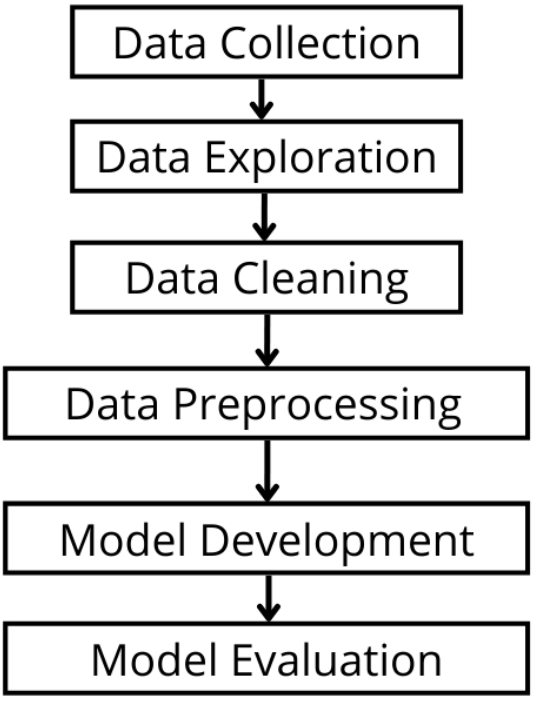

Fig.1 design and flow of model

5. MODULES

5.1. module 1 : data collection

In this module we extracted dataset from kaggle. To do batsman, bowler and team analysis much preprocessing was not required. For logistic regression model we build dataset from the above datasets.

5.2. module 2 : data cleaning and dataset analysis

Data must be preprocessed and ready for analysis in order to gain greater insights from it. Noisy, missing, inconsistent, and skewed data are all possibilities. It's possible that there are duplicate records. Analyzing raw data might lead to incorrect conclusions. Data should be cleansed, altered, and reduced as needed to obtain better findings. We had to variety of preprocessing and cleaning.

In [21]:
```
matches.team1.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.team2.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.winner.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.venue.replace({'Feroz Shah Kotla Ground':'Feroz Shah Kotla',
                       'M Chinnaswamy Stadium':'M. Chinnaswamy Stadium',
                       'MA Chidambaram Stadium, Chepauk':'M.A. Chidambaram Stadium',
                        'M. A. Chidambaram Stadium':'M.A. Chidambaram Stadium',
                        'Punjab Cricket Association IS Bindra Stadium, Mohali':'Punjab Cricket Association Stadium',
                        'Punjab Cricket Association Stadium, Mohali':'Punjab Cricket Association Stadium',
                        'IS Bindra Stadium':'Punjab Cricket Association Stadium',
                       'Rajiv Gandhi International Stadium, Uppal':'Rajiv Gandhi International Stadium',
                       'Rajiv Gandhi Intl. Cricket Stadium':'Rajiv Gandhi International Stadium'},regex=True,inplace=True)
```

In [80]:
```
# crr = runs/overs
delivery_df['crr'] = (delivery_df['current_score']*6)/(120 - delivery_df['balls_left'])
```

In [81]:
```
delivery_df['rrr'] = (delivery_df['runs_left']*6)/delivery_df['balls_left']
```

In [82]:
```
def result(row):
    return 1 if row['batting_team'] == row['winner'] else 0
```

In [83]:
```
delivery_df['result'] = delivery_df.apply(result,axis=1)
```

In [84]:
```
final_df = delivery_df[['batting_team','bowling_team','city','runs_left','balls_left','wickets','total_runs_x','crr','rrr','r
```

In [85]:
```
final_df = final_df.sample(final_df.shape[0])
```

In [86]:
```
final_df.sample()
```

Out[86]:

	batting_team	bowling_team	city	runs_left	balls_left	wickets	total_runs_x	crr	rrr	result
17951	Kings XI Punjab	Deccan Chargers	Chandigarh	134	101	10	175	12.947368	7.960396	1

5.3. module 3 : analysis and visualization from dataset

The attributes from the obtained data set are compared with each other to find correlations and dependencies and then these are visualized using different types of graphs. The graphs obtained are then studied in detail to obtain further observations to see to what factor each attribute influences the prediction model.

5.4. module 4: model creation

We need to create machine learning models on this cleaned data now that we've processed and cleaned it.

5.5. module 5: dashboard creation

We will use tableau to craete multiple dashboards. One dashboard will focus on complete IPL analysis from 2008-2020 and other will focus on forecasting.

6. RISK ANALYSIS:

Due to vast amount of data available online and due to large magnitude of data in the existing dataset we have focused on predicting the win probability for 2nd innings of the cricket match. There are several other parameters that affect the win probability in 2nd innings of a cricket match like due factor, humidity factor, current form of team etc. These can affect the bias created. Since we didn't have the measures for them, we focused on attributes like city, runs_left, balls_left, wickets etc.

To forecast runs, wickets, boundaries by a player we have used ARIMA model. However some players may miss any season due to injury or personal reasons , which may affect the forecast.

7. IMPLEMENTATION

7.1 Importing dataset

In [2]:
```
matches=pd.read_csv("matches.csv")
deliveries=pd.read_csv("ballbyball.csv")
```

In [3]:
```
matches.head()
```

Out[3]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	resu
0	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	Kolkata Knight Riders	runs	
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings XI Punjab	Chennai Super Kings	Chennai Super Kings	bat	Chennai Super Kings	runs	
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils	Rajasthan Royals	Rajasthan Royals	bat	Delhi Daredevils	wickets	
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians	Royal Challengers Bangalore	Mumbai Indians	bat	Royal Challengers Bangalore	wickets	
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders	Deccan Chargers	Deccan Chargers	bat	Kolkata Knight Riders	wickets	

In [4]:
```
deliveries.head()
```

Out[4]:

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_dismis
0	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0	0	NaN	
1	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0	0	NaN	
2	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0	0	NaN	
3	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NaN	

7.2 Pre processing

In [12]:
```
run_region=['third man','point','cover','long on','long off','mid wicket','square leg','fine leg']
```

In [13]:
```
deliveries['runs_region']=np.random.choice(run_region,deliveries.shape[0])
```

In [14]:
```
deliveries.head()
```

Out[14]:

man_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_dismissed	fielder	extras_type	batting_team	bowling_team	runs_region
1	0	1	0	0	NaN	NaN	NaN	NaN	Kolkata Knight Riders	Royal Challengers Bangalore	third man
1	0	1	0	0	NaN	NaN	NaN	NaN	Kolkata Knight Riders	Royal Challengers Bangalore	fine leg
0	0	0	0	0	NaN	NaN	NaN	NaN	Kolkata Knight Riders	Royal Challengers Bangalore	fine leg
1	0	1	0	0	NaN	NaN	NaN	NaN	Kolkata Knight Riders	Royal Challengers Bangalore	cover
1	0	1	0	0	NaN	NaN	NaN	NaN	Kolkata Knight Riders	Royal Challengers Bangalore	mid wicket

Adding regions in which batsmen scored runs to delivery dataset.

In [21]:
```
matches.team1.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.team2.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.winner.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.venue.replace({'Feroz Shah Kotla Ground':'Feroz Shah Kotla',
                       'M Chinnaswamy Stadium':'M. Chinnaswamy Stadium',
                       'MA Chidambaram Stadium, Chepauk':'M.A. Chidambaram Stadium',
                        'M. A. Chidambaram Stadium':'M.A. Chidambaram Stadium',
                        'Punjab Cricket Association IS Bindra Stadium, Mohali':'Punjab Cricket Association Stadium',
                        'Punjab Cricket Association Stadium, Mohali':'Punjab Cricket Association Stadium',
                        'IS Bindra Stadium':'Punjab Cricket Association Stadium',
                       'Rajiv Gandhi International Stadium, Uppal':'Rajiv Gandhi International Stadium',
                       'Rajiv Gandhi Intl. Cricket Stadium':'Rajiv Gandhi International Stadium'},regex=True,inplace=True)
```

7.3 Visualisations

In [23]:
```
#plotting teams with maximum wins
mwins=pd.DataFrame({"winner":matches["winner"]})
count_wins=mwins["winner"].value_counts()
labels=[x for x in count_wins.keys()]
bar,ax=plt.subplots(figsize=(20,12))
ax=plt.pie(x=count_wins,autopct="%.1f%%",labels=labels)
plt.title("Most wins in IPL",fontsize=17)
plt.show()
```

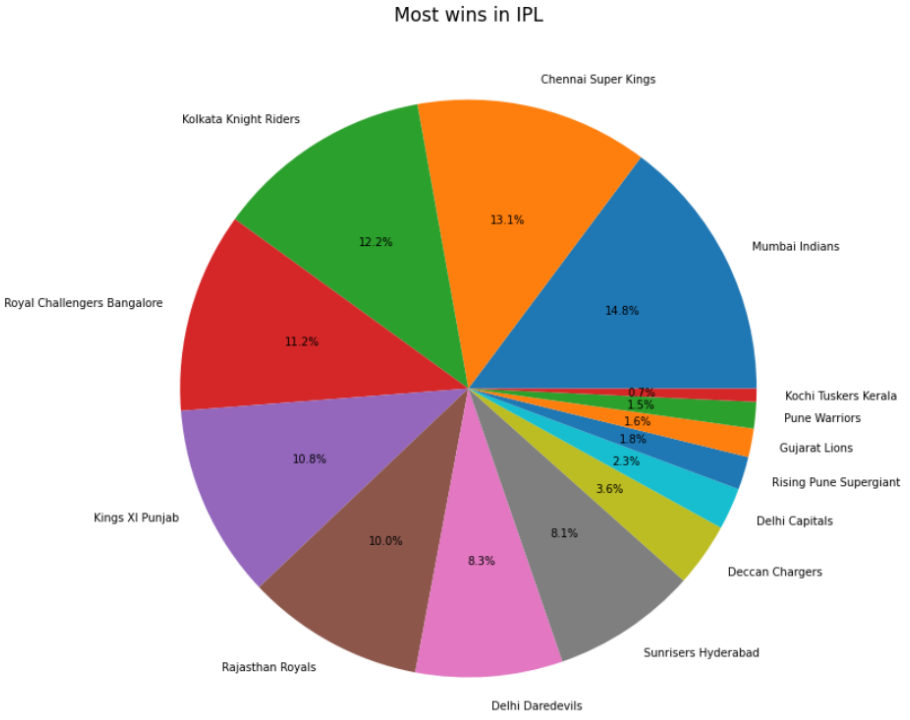

Figure 2 Most wins in IPL

In [26]:
```
colors = ['turquoise',] * 13
colors[5] = 'crimson'

fig=px.bar(data_frame=match_per_season,x=match_per_season.Season,y=match_per_season.matches,labels=dict(x="Season",y="Count")
fig.update_layout(title="Number of matches played in different seasons ",
                  titlefont={'size': 26},template='simple_white'
                  )
fig.update_traces(marker_line_color='black',
                  marker_line_width=2.5, opacity=1,marker_color=colors)
fig.show()
```

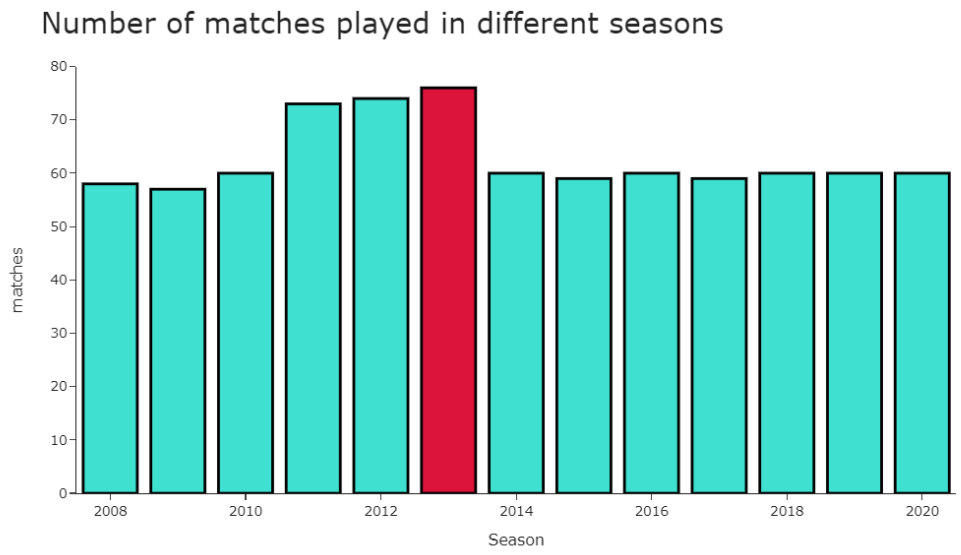

Figure 3. Matches played across seasons

We see that maximum matches were played during IPL 2013 as 10 teams participated during that season.

In [28]:
```
season=season_data.groupby(['Season'])['total_runs'].sum().reset_index()
p=season.set_index('Season')
fig = px.line(p, x=p.index, y="total_runs")
fig.update_layout(title="Total Runs Across the Seasons ",
                  titlefont={'size': 26},template='simple_white'
                  )
fig.show()
```

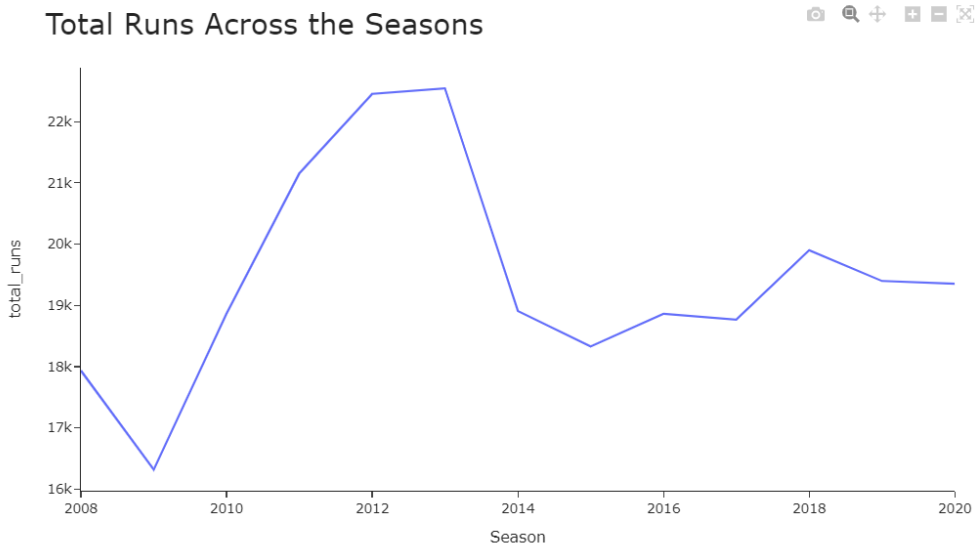

Figure 4. Runs scored accross Seasons

```
In [29]:
### decision made after winning toss
temp_series = matches.toss_decision.value_counts()
labels = (np.array(temp_series.index))
values = (np.array((temp_series / temp_series.sum())*100))
colors = ['turquoise', 'crimson']
fig = go.Figure(data=[go.Pie(labels=labels,
                             values=values,hole=.3)])
fig.update_traces(hoverinfo='label+percent', textinfo='label+percent', textfont_size=20,
                  marker=dict(colors=colors, line=dict(color='#000000', width=3)))
fig.update_layout(title="Toss decision percentage",
                  titlefont={'size': 30},
                  )
fig.show()
```

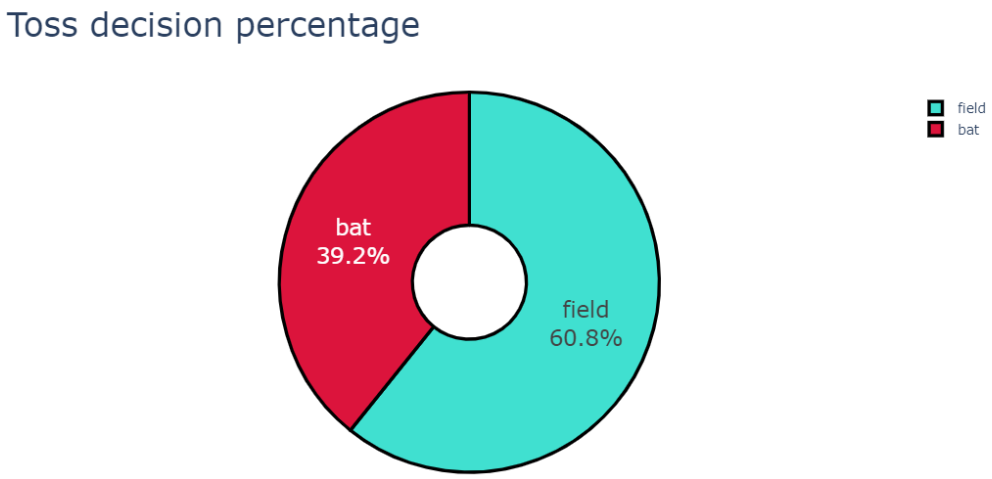

Figure 5. Decision made after winning toss

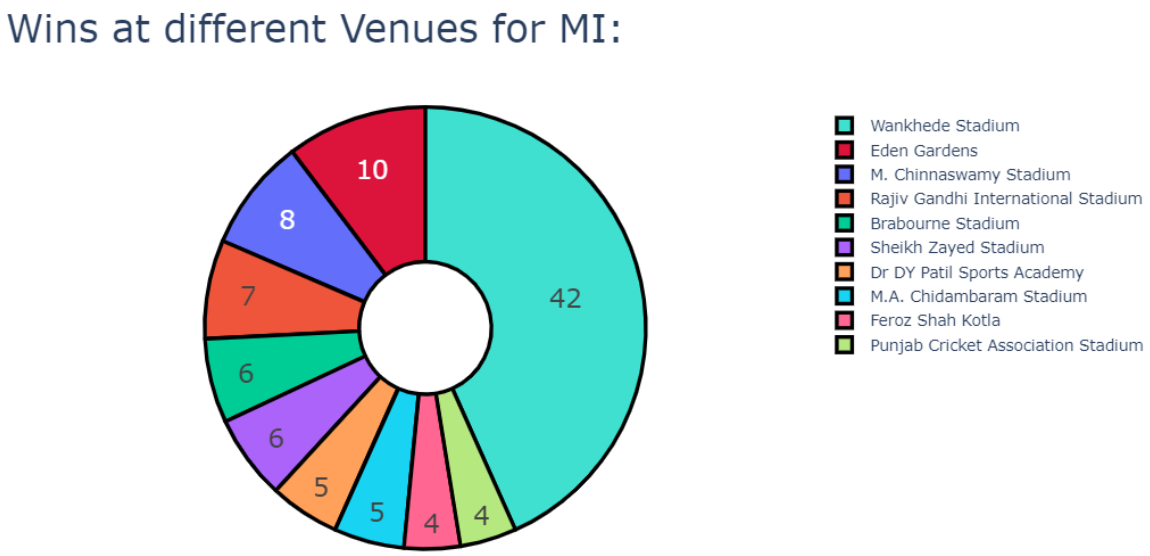

Figure 6. Lucky venue for MI

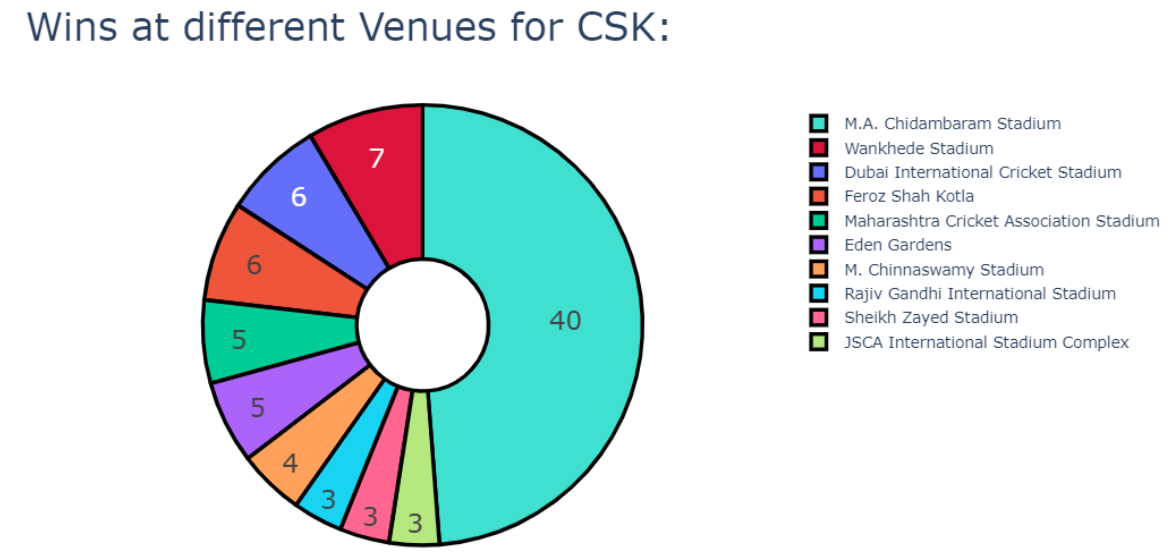

Figure 7. Lucky venue for CSk

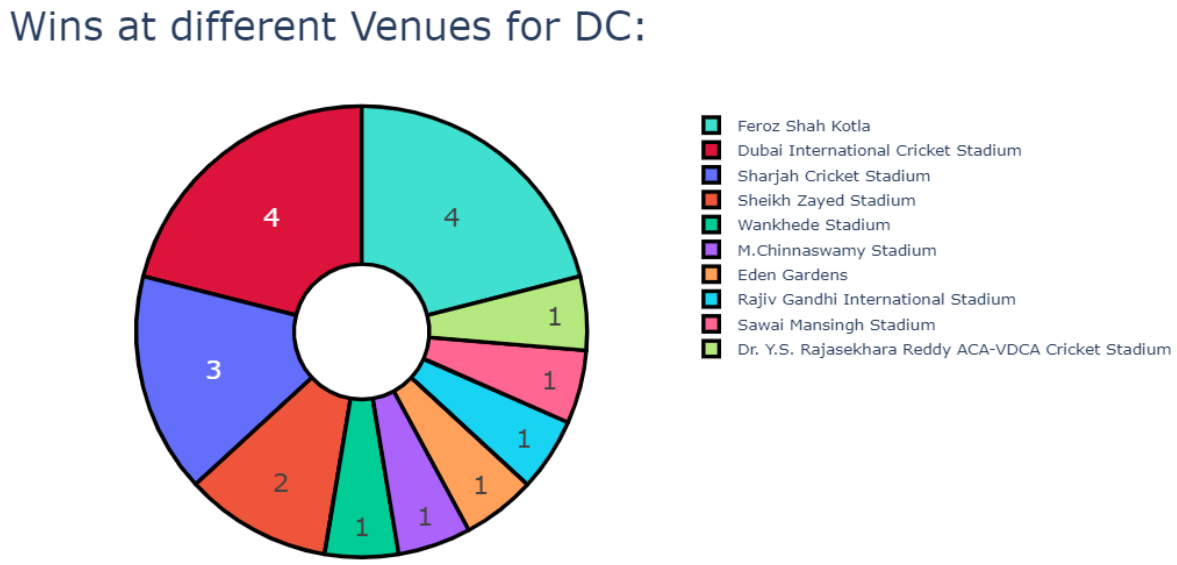

Figure 8. Lucky venue for DC

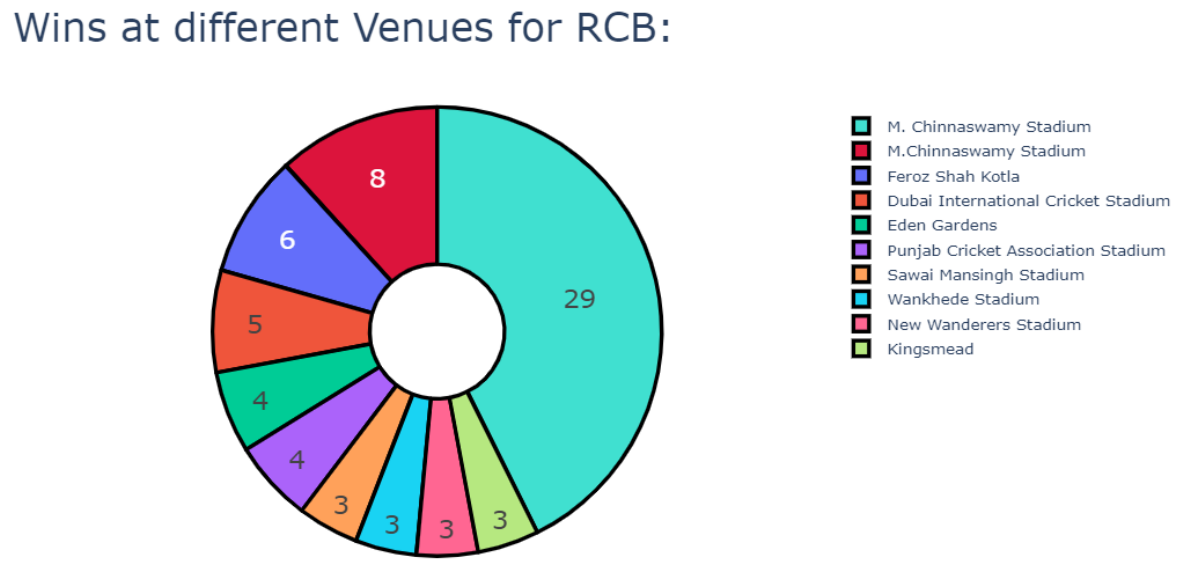

Figure 9. Lucky venue for RCB

7.3 BATSMEN ANALYSIS

VIRAT KOHLI

```
In [38]: filt=(deliveries['batsman']=='V Kohli')
         df_kohli=deliveries[filt]
         df_kohli.head()
```

Out[38]:

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_disr
211	335982	2	1	2	V Kohli	W Jaffer	I Sharma	0	0	0	0	0	NaN	
212	335982	2	1	3	V Kohli	W Jaffer	I Sharma	0	4	4	0	0	NaN	
213	335982	2	1	4	V Kohli	W Jaffer	I Sharma	1	0	1	0	0	NaN	
216	335982	2	2	1	V Kohli	W Jaffer	AB Dinda	0	0	0	0	0	NaN	
217	335982	2	2	2	V Kohli	W Jaffer	AB Dinda	0	0	0	0	1	bowled	

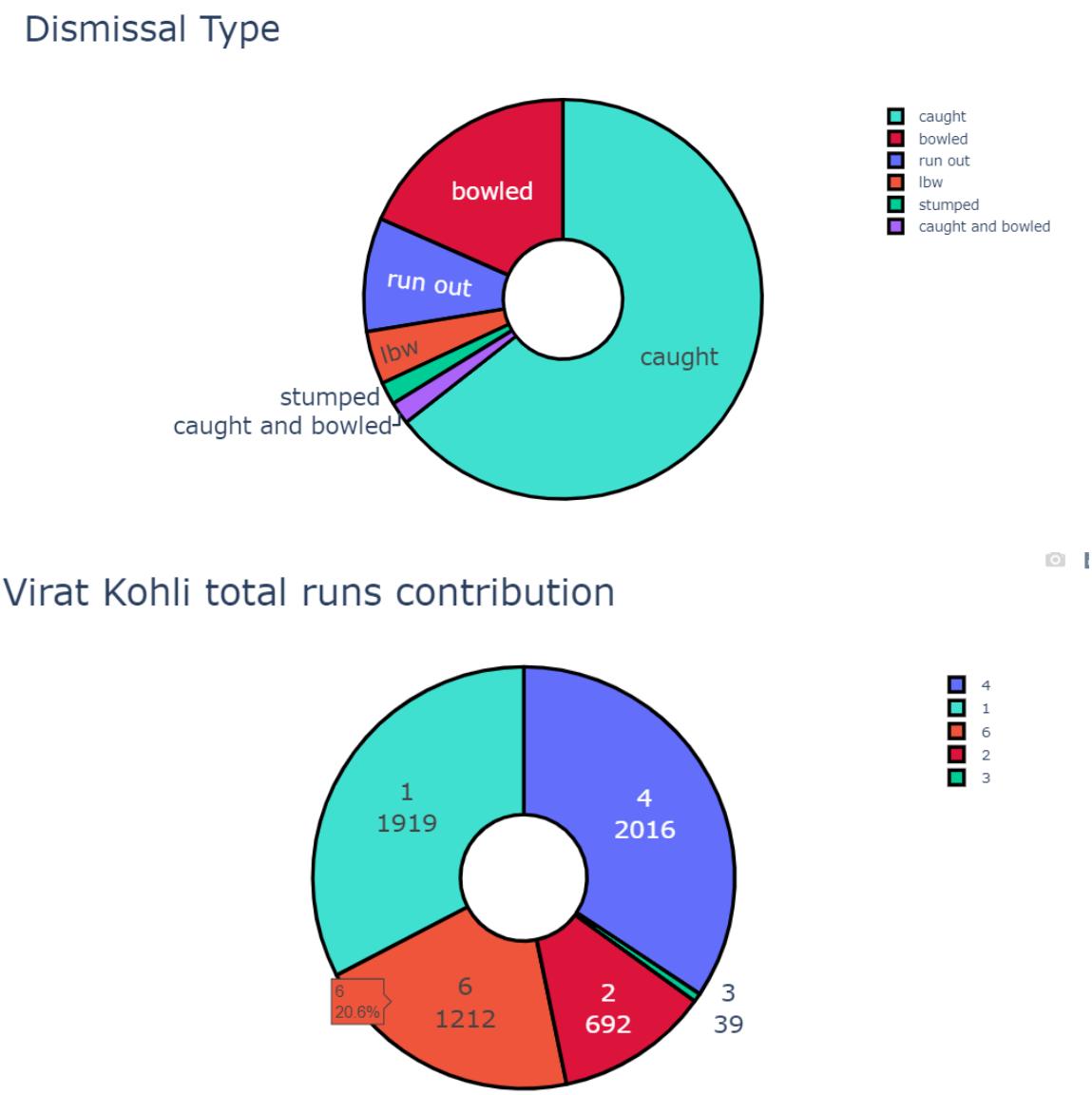

Figure 10. Virat Kohli Dismissal type

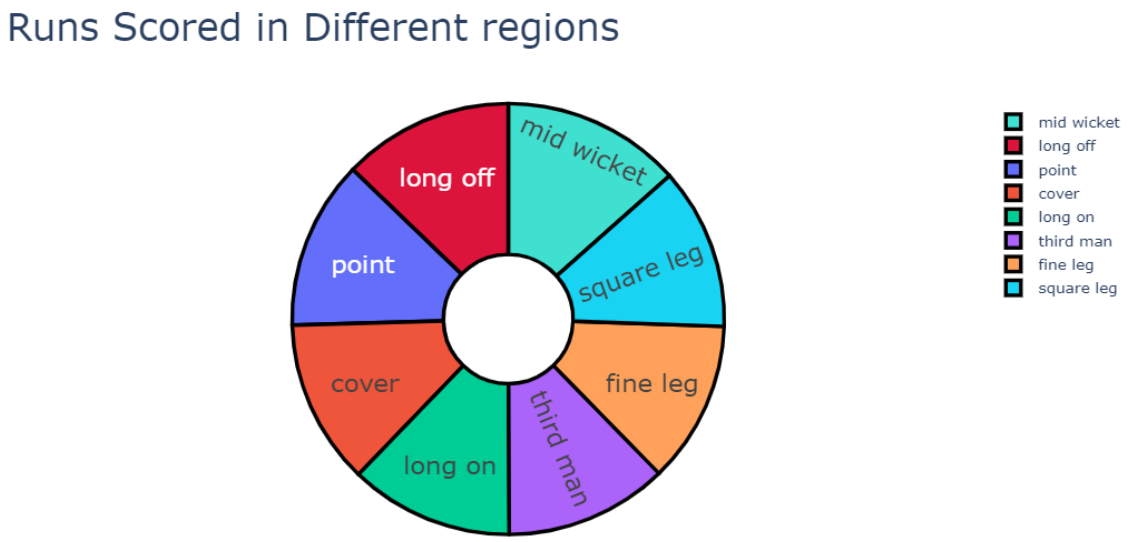

Figure 11. Runs scored by VK in different regions

7.4 BOWLER ANALYSIS

```
In [57]:
filt=(deliveries['bowler']=='JJ Bumrah')
df_bumrah=deliveries[filt]
df_bumrah.head()
```

Out[57]:

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_d
76613	597999	1	4	1	V Kohli	CH Gayle	JJ Bumrah	4	0	4	0	0	NaN	
76614	597999	1	4	2	V Kohli	CH Gayle	JJ Bumrah	4	0	4	0	0	NaN	
76615	597999	1	4	3	V Kohli	CH Gayle	JJ Bumrah	0	0	0	0	0	NaN	
76616	597999	1	4	4	V Kohli	CH Gayle	JJ Bumrah	4	0	4	0	0	NaN	
76617	597999	1	4	5	V Kohli	CH Gayle	JJ Bumrah	0	0	0	0	1	lbw	

```
In [58]:
values = df_bumrah['dismissal_kind'].value_counts()
labels=df_bumrah['dismissal_kind'].value_counts().index
fig = go.Figure(data=[go.Pie(labels=labels,values=values,hole=.3)])
fig.update_traces(hoverinfo='label+percent', textinfo='label', textfont_size=20,
                  marker=dict(colors=colors, line=dict(color='#000000', width=3)))
fig.update_layout(title="Dismissal kind",
                  titlefont={'size': 30},
                  )
fig.show()
```

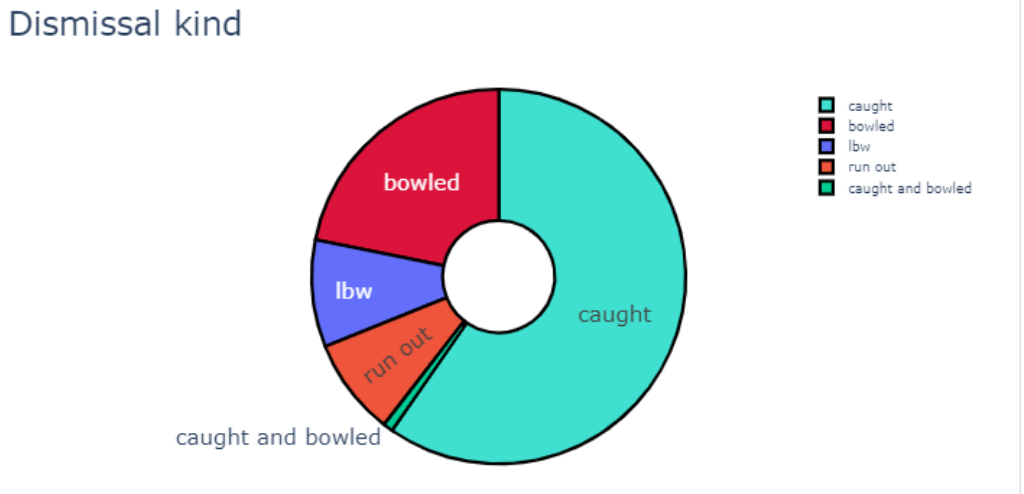

Figure 12. Bumrah Dismissal kind

7.5 WIN PREDICTION MODEL

WIN PROBABILITY PREDICTION

In [59]:
```
match = pd.read_csv('matches_prob.csv')
delivery = pd.read_csv('deliveries_prob.csv')
total_score_df = delivery.groupby(['match_id','inning']).sum()['total_runs'].reset_index()
total_score_df = total_score_df[total_score_df['inning'] == 1]
total_score_df
```

Out[59]:

	match_id	inning	total_runs
0	1	1	207
2	2	1	184
4	3	1	183
6	4	1	163
8	5	1	157
...	...	...	...
1518	11347	1	143
1520	11412	1	136
1522	11413	1	171
1524	11414	1	155
1526	11415	1	152

756 rows × 3 columns

In [71]:
```
delivery_df['current_score'] = delivery_df.groupby('match_id').cumsum()['total_runs_y']
```

In [72]:
```
delivery_df['runs_left'] = delivery_df['total_runs_x'] - delivery_df['current_score']
```

In [73]:
```
delivery_df['balls_left'] = 126 - (delivery_df['over']*6 + delivery_df['ball'])
```

In [74]:
```
delivery_df
```

Out[74]:

	match_id	city	winner	total_runs_x	inning	batting_team	bowling_team	over	ball	batsman	...	penalty_runs	batsman_runs	extra_ru
125	1	Hyderabad	Sunrisers Hyderabad	207	2	Royal Challengers Bangalore	Sunrisers Hyderabad	1	1	CH Gayle	...	0	1	
126	1	Hyderabad	Sunrisers Hyderabad	207	2	Royal Challengers Bangalore	Sunrisers Hyderabad	1	2	Mandeep Singh	...	0	0	
127	1	Hyderabad	Sunrisers Hyderabad	207	2	Royal Challengers Bangalore	Sunrisers Hyderabad	1	3	Mandeep Singh	...	0	0	
128	1	Hyderabad	Sunrisers Hyderabad	207	2	Royal Challengers Bangalore	Sunrisers Hyderabad	1	4	Mandeep Singh	...	0	2	
129	1	Hyderabad	Sunrisers Hyderabad	207	2	Royal Challengers Bangalore	Sunrisers Hyderabad	1	5	Mandeep Singh	...	0	4	

In [80]:
```
# crr = runs/overs
delivery_df['crr'] = (delivery_df['current_score']*6)/(120 - delivery_df['balls_left'])
```

In [81]:
```
delivery_df['rrr'] = (delivery_df['runs_left']*6)/delivery_df['balls_left']
```

In [82]:
```
def result(row):
    return 1 if row['batting_team'] == row['winner'] else 0
```

In [83]:
```
delivery_df['result'] = delivery_df.apply(result,axis=1)
```

In [84]:
```
final_df = delivery_df[['batting_team','bowling_team','city','runs_left','balls_left','wickets','total_runs_x','crr','rrr','r
```

In [85]:
```
final_df = final_df.sample(final_df.shape[0])
```

In [86]:
```
final_df.sample()
```

Out[86]:

	batting_team	bowling_team	city	runs_left	balls_left	wickets	total_runs_x	crr	rrr	result
17951	Kings XI Punjab	Deccan Chargers	Chandigarh	134	101	10	175	12.947368	7.960396	1

In [87]:
```
final_df.dropna(inplace=True)
```

In [88]:
```
final_df = final_df[final_df['balls_left'] != 0]
```

In [89]:
```
final_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 71342 entries, 45678 to 50638
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   batting_team  71342 non-null  object
 1   bowling_team  71342 non-null  object
 2   city          71342 non-null  object
 3   runs_left     71342 non-null  int64
 4   balls_left    71342 non-null  int64
 5   wickets       71342 non-null  int32
 6   total_runs_x  71342 non-null  int64
 7   crr           71342 non-null  float64
 8   rrr           71342 non-null  float64
 9   result        71342 non-null  int64
dtypes: float64(2), int32(1), int64(4), object(3)
memory usage: 5.7+ MB
```

In [108]:
```
temp_df,target = match_progression(delivery_df,75,pipe)
temp_df
```

```
Target- 137
```

Out[108]:

	end_of_over	runs_after_over	wickets_in_over	lose	win
10708	1	2	0	10.6	89.4
10716	2	14	0	5.8	94.2
10722	3	1	1	13.3	86.7
10729	4	8	1	19.3	80.7
10735	5	4	1	32.6	67.4
10741	6	9	0	27.3	72.7
10747	7	5	0	27.4	72.6
10753	8	4	0	28.8	71.2
10759	9	16	0	16.8	83.2
10766	10	6	0	16.0	84.0
10773	11	6	0	15.2	84.8
10779	12	15	0	8.7	91.3
10785	13	8	0	7.3	92.7
10791	14	4	0	7.7	92.3
10797	15	8	0	6.4	93.6
10803	16	11	0	4.4	95.6
10809	17	7	0	3.7	96.3
10816	18	4	0	3.9	96.1

\

```
In [109]: import matplotlib.pyplot as plt
          plt.figure(figsize=(18,8))
          plt.plot(temp_df['end_of_over'],temp_df['wickets_in_over'],color='yellow',linewidth=3)
          plt.plot(temp_df['end_of_over'],temp_df['win'],color='#00a65a',linewidth=4)
          plt.plot(temp_df['end_of_over'],temp_df['lose'],color='red',linewidth=4)
          plt.bar(temp_df['end_of_over'],temp_df['runs_after_over'])
          plt.title('Target-' + str(target))

Out[109]: Text(0.5, 1.0, 'Target-137')
```

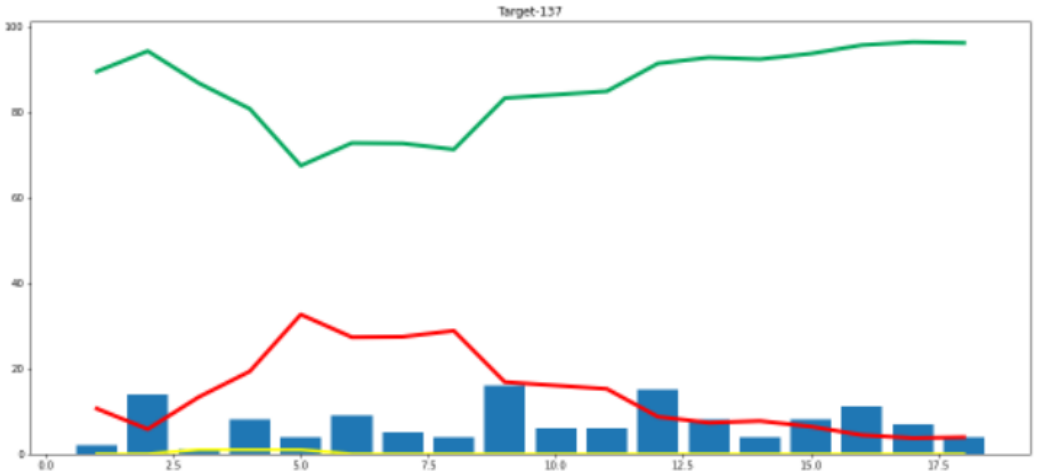

Figure 13. Win probabilty visualisation for a sample match.

7.6 TABLEAU DASHBOARD

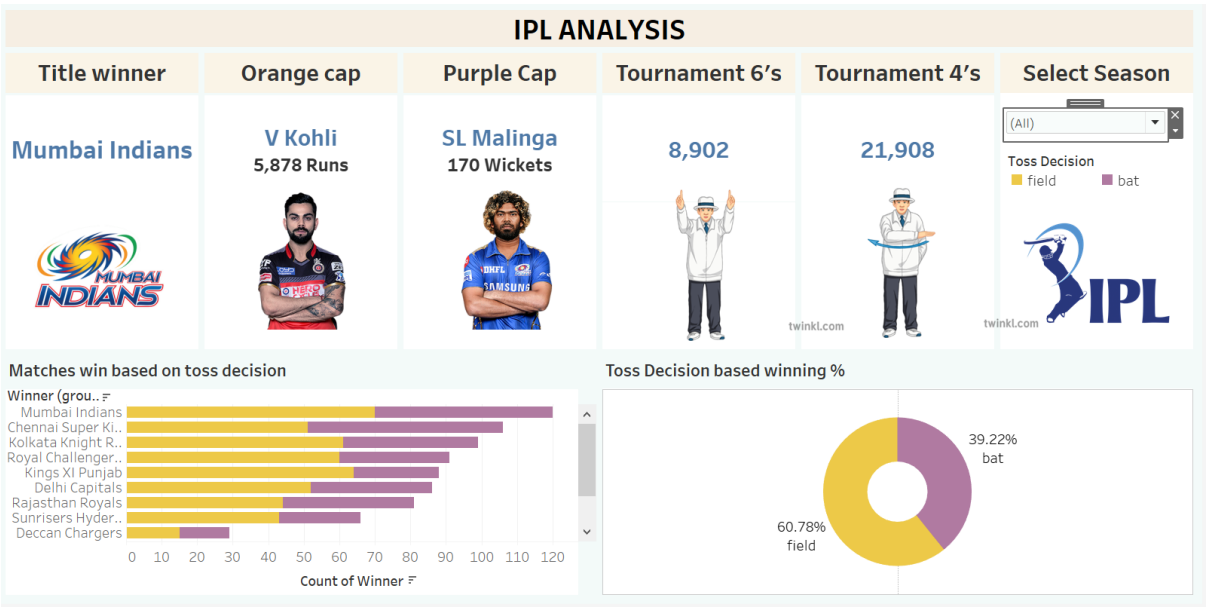

Figure 14. Ipl dashboard.

Each card on dashboard represents a seperate sheet. It shows title winner, orange cap holder, purple cap holder, tournaent four's , tournament sixe's and much more.

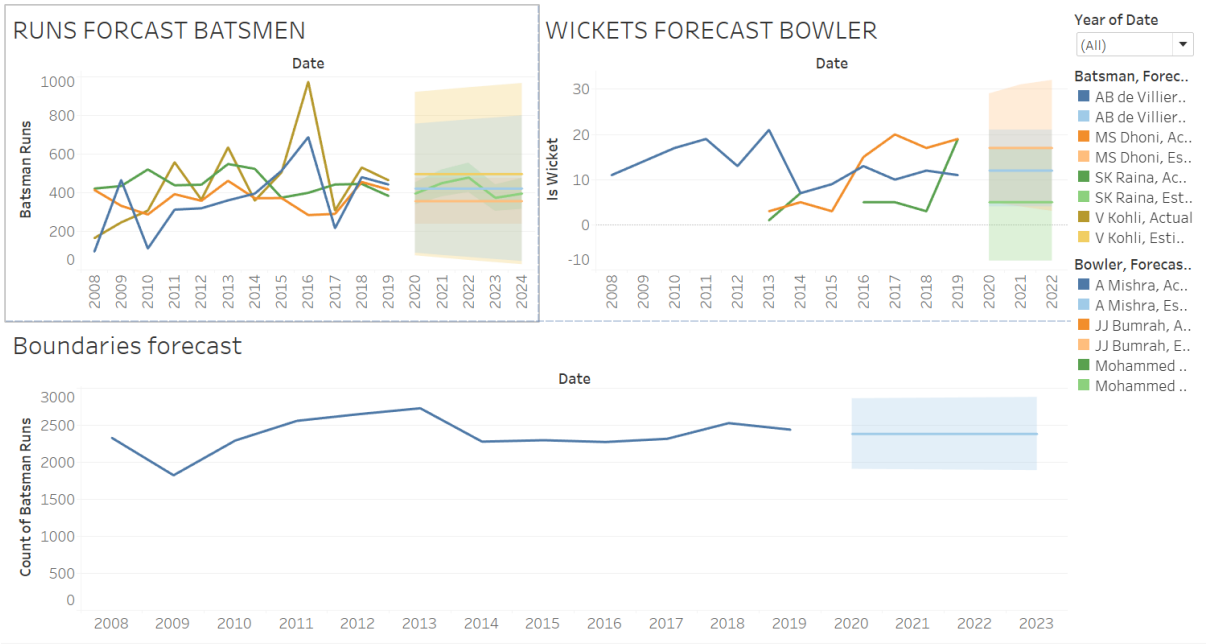

Figure 15. Forecast dashboard.

This dashboard focuses on forecasting runs, wickets, and boundaries.

CONCLUSION

This project seeks to comprehend the dataset of IPL data over the last ten years. It aids in the comprehension of the major machine learning algorithms and their implementation in Python. It generates the Model and Training datasets and uses the generated model to forecast WIN.

After doing a detailed analysis of batsmen, bowler and teams, interactive dashboards were created in Tableau. ARIMA model was used to predict runs, wickets and bounaries.

By utilising this, the Indian Premier League and its fans may make informed judgements about the team's performance and anticipate trophy winners who will lead to future success.

REFERENCES:

(1) S. Abhishek, Ketaki V. Patil, P. Yuktha and S. Meghana, Predictive Analysis of IPL Match Winner using Machine Learning Techniques", International Journal of Innovative Technology and Exploring Engineering, Vol. 9, No. 1, pp. 430-435, 2019

(2) Raza Ul Mustafa, M. Saqib Nawaz, M. Ikram Ullah Lali, Tehseen Zia and Waqar Mehmood, "Predicting the Cricket Match outcome using Crowd Opinions on Social Networks:
A Comparative Study of Machine Learning Methods", Malaysian Journal of Computer Science, Vol. 30, No. 1, pp. 63-76, 2017.

(3) C. Deep Prakash Dayalbagh, C. Patvardhan and C. Vasantha Lakshmi, "Data Analytics based Deep Mayo Predictor for IPL-9", International Journal of Computer Applications, Vol. 152, No. 6, pp. 6-11, 2016.

(4) S. Kampakis and W. Thomas, "Using machine learning to predict the outcome of English county twenty over cricket matches," arXiv preprint arXiv:1511.05837, 2015.

(5) Rakshit Patel, Mihir Brahmbatt, "Insights of IPL: 2008 to 2020 and why it is interesting", ISSN: 2454-132

(6) Anurag Sinha, "Application of Machine Learning in Cricket and Predictive Analytics of IPL 2020"

Github link:

https://github.com/singhalayush55/19BCE1322_DV_JCOMP