In [62]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
```

In [63]:

```
ball_by_ball = pd.read_csv('./Data/IPL_Ball_by_Ball_2008_2022.csv')
matches_result = pd.read_csv('./Data/IPL_Matches_Result_2008_2022.csv')
ipl_2023_teams = pd.read_csv('./Data/Ipl_2023 _cricketers - Team name.csv').rename(
    'Teams': 'team'
})
ipl_2023_venues = pd.read_csv('./Data/Ipl_2023 _cricketers - Venue.csv').rename(col
    'Venue': 'venue'
})
```

In [64]:

```
def log(*args):
    print('👉', *args)
```

In [65]:

```
def to_kebab_case(string):
    return '-'.join(
        string.replace(",", "").replace(".", "").split()
    ).lower()
```

Preparing training dataset

- Change column names, drop unnecessary columns [in ball_by_ball, matches_result]

In [66]:

```
ball_by_ball_orig = ball_by_ball

ball_by_ball = ball_by_ball.rename(columns={
    'ID': 'match_id',
    'ballnumber': 'ball_number',
    'non-striker': 'non_striker',
    'BattingTeam': 'batting_team',
}).loc[:, [
    'match_id',
    'innings',
    'batting_team',
    'overs',
    'ball_number',
    'batter',
    'bowler',
    'total_run',
]]
```

In [67]:

```
matches_result_orig = matches_result

matches_result = matches_result.rename(columns={
    'ID': 'match_id',
    'Team1': 'team_1',
    'Team2': 'team_2',
    'Venue': 'venue',
}).loc[:, [
    'match_id',
    'team_1',
    'team_2',
    'venue',
]]
```

In [68]:

```
print(ball_by_ball_orig.shape)
ball_by_ball_orig.head()
```

```
(225954, 17)
```

Out[68]:

	ID	innings	overs	ballnumber	batter	bowler	non-striker	extra_type	batsman
0	1312200	1	0	1	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	
1	1312200	1	0	2	YBK Jaiswal	Mohammed Shami	JC Buttler	legbyes	
2	1312200	1	0	3	JC Buttler	Mohammed Shami	YBK Jaiswal	NaN	
3	1312200	1	0	4	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	
4	1312200	1	0	5	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	

In [69]:

```
print(matches_result_orig.shape)
matches_result_orig.head()
```

```
(950, 20)
```

Out[69]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad
2	1312198	Kolkata	2022-05-25	2022	Eliminator	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata
3	1312197	Kolkata	2022-05-24	2022	Qualifier 1	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata
4	1304116	Mumbai	2022-05-22	2022	70	Sunrisers Hyderabad	Punjab Kings	Wankhede Stadium, Mumbai

In [70]:

```
print(ball_by_ball.shape)
ball_by_ball.head()
```

```
(225954, 8)
```

Out[70]:

	match_id	innings	batting_team	overs	ball_number	batter	bowler	total_run
0	1312200	1	Rajasthan Royals	0	1	YBK Jaiswal	Mohammed Shami	0
1	1312200	1	Rajasthan Royals	0	2	YBK Jaiswal	Mohammed Shami	1
2	1312200	1	Rajasthan Royals	0	3	JC Buttler	Mohammed Shami	1
3	1312200	1	Rajasthan Royals	0	4	YBK Jaiswal	Mohammed Shami	0
4	1312200	1	Rajasthan Royals	0	5	YBK Jaiswal	Mohammed Shami	0

In [71]:

```
print(matches_result.shape)
matches_result.head()
```

```
(950, 4)
```

Out[71]:

	match_id	team_1	team_2	venue
0	1312200	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad
1	1312199	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad
2	1312198	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata
3	1312197	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata
4	1304116	Sunrisers Hyderabad	Punjab Kings	Wankhede Stadium, Mumbai

- ## Some stats

In [72]:

```
log('ball_by_ball match_id.nunique:', ball_by_ball.match_id.nunique())
log('ball_by_ball batting_team.nunique:', ball_by_ball.batting_team.nunique())
log('ball_by_ball union1d(batter, bowler).shape:', np.union1d(
    ball_by_ball.batter.unique(), ball_by_ball.bowler.unique()
).shape)
log('ball_by_ball innings.unique:', ball_by_ball.innings.unique())
log('ball_by_ball overs.unique:', ball_by_ball.overs.unique())
```

```
☞ ball_by_ball match_id.nunique: 950
☞ ball_by_ball batting_team.nunique: 18
☞ ball_by_ball union1d(batter, bowler).shape: (652,)
☞ ball_by_ball innings.unique: [1 2 3 4 5 6]
☞ ball_by_ball overs.unique: [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
18 19]
```

In [73]:

```
log('matches_result match_id.nunique:', matches_result.match_id.nunique())
log('matches_result venue.nunique:', matches_result.venue.nunique())
log('matches_result union1d(team_1, team_2).shape:', np.union1d(
    matches_result.team_1.unique(), matches_result.team_2.unique()
).shape)
```

```
☞ matches_result match_id.nunique: 950
☞ matches_result venue.nunique: 49
☞ matches_result union1d(team_1, team_2).shape: (18,)
```

- ## Get Venues Mapping

In [74]:

```
matches_result_orig.groupby(['City', 'Venue'], dropna=False)['Venue'].describe()
```

Out[74]:

		count	unique	top	freq
City	**Venue**				
Abu Dhabi	**Sheikh Zayed Stadium**	29	1	Sheikh Zayed Stadium	29
	Zayed Cricket Stadium, Abu Dhabi	8	1	Zayed Cricket Stadium, Abu Dhabi	8
Ahmedabad	**Narendra Modi Stadium, Ahmedabad**	7	1	Narendra Modi Stadium, Ahmedabad	7
	Sardar Patel Stadium, Motera	12	1	Sardar Patel Stadium, Motera	12
Bangalore	**M Chinnaswamy Stadium**	65	1	M Chinnaswamy Stadium	65
Bengaluru	**M.Chinnaswamy Stadium**	15	1	M.Chinnaswamy Stadium	15
Bloemfontein	**OUTsurance Oval**	2	1	OUTsurance Oval	2
Cape Town	**Newlands**	7	1	Newlands	7
Centurion	**SuperSport Park**	12	1	SuperSport Park	12
Chandigarh	**Punjab Cricket Association IS Bindra Stadium**	10	1	Punjab Cricket Association IS Bindra Stadium	10
	Punjab Cricket Association IS Bindra Stadium, Mohali	11	1	Punjab Cricket Association IS Bindra Stadium, ...	11
	Punjab Cricket Association Stadium, Mohali	35	1	Punjab Cricket Association Stadium, Mohali	35
Chennai	**MA Chidambaram Stadium**	9	1	MA Chidambaram Stadium	9
	MA Chidambaram Stadium, Chepauk	48	1	MA Chidambaram Stadium, Chepauk	48
	MA Chidambaram Stadium, Chepauk, Chennai	10	1	MA Chidambaram Stadium, Chepauk, Chennai	10
Cuttack	**Barabati Stadium**	7	1	Barabati Stadium	7
Delhi	**Arun Jaitley Stadium**	14	1	Arun Jaitley Stadium	14
	Arun Jaitley Stadium, Delhi	4	1	Arun Jaitley Stadium, Delhi	4
	Feroz Shah Kotla	60	1	Feroz Shah Kotla	60

		count	unique	top	freq
City	**Venue**				
Dharamsala	**Himachal Pradesh Cricket Association Stadium**	9	1	Himachal Pradesh Cricket Association Stadium	9
Dubai	**Dubai International Cricket Stadium**	13	1	Dubai International Cricket Stadium	13
Durban	**Kingsmead**	15	1	Kingsmead	15
East London	**Buffalo Park**	3	1	Buffalo Park	3
Hyderabad	**Rajiv Gandhi International Stadium**	15	1	Rajiv Gandhi International Stadium	15
	Rajiv Gandhi International Stadium, Uppal	49	1	Rajiv Gandhi International Stadium, Uppal	49
Indore	**Holkar Cricket Stadium**	9	1	Holkar Cricket Stadium	9
Jaipur	**Sawai Mansingh Stadium**	47	1	Sawai Mansingh Stadium	47
Johannesburg	**New Wanderers Stadium**	8	1	New Wanderers Stadium	8
Kanpur	**Green Park**	4	1	Green Park	4
Kimberley	**De Beers Diamond Oval**	3	1	De Beers Diamond Oval	3
Kochi	**Nehru Stadium**	5	1	Nehru Stadium	5
Kolkata	**Eden Gardens**	77	1	Eden Gardens	77
	Eden Gardens, Kolkata	2	1	Eden Gardens, Kolkata	2
Mumbai	**Brabourne Stadium**	10	1	Brabourne Stadium	10
	Brabourne Stadium, Mumbai	17	1	Brabourne Stadium, Mumbai	17
	Dr DY Patil Sports Academy	17	1	Dr DY Patil Sports Academy	17
	Dr DY Patil Sports Academy, Mumbai	11	1	Dr DY Patil Sports Academy, Mumbai	11
	Wankhede Stadium	73	1	Wankhede Stadium	73
	Wankhede Stadium, Mumbai	31	1	Wankhede Stadium, Mumbai	31
Nagpur	**Vidarbha Cricket Association Stadium, Jamtha**	3	1	Vidarbha Cricket Association Stadium, Jamtha	3

		count	unique	top	freq
City	**Venue**				
Navi Mumbai	**Dr DY Patil Sports Academy, Mumbai**	9	1	Dr DY Patil Sports Academy, Mumbai	9
Port Elizabeth	**St George's Park**	7	1	St George's Park	7
Pune	**Maharashtra Cricket Association Stadium**	22	1	Maharashtra Cricket Association Stadium	22
	Maharashtra Cricket Association Stadium, Pune	13	1	Maharashtra Cricket Association Stadium, Pune	13
	Subrata Roy Sahara Stadium	16	1	Subrata Roy Sahara Stadium	16
Raipur	**Shaheed Veer Narayan Singh International Stadium**	6	1	Shaheed Veer Narayan Singh International Stadium	6
Rajkot	**Saurashtra Cricket Association Stadium**	10	1	Saurashtra Cricket Association Stadium	10
Ranchi	**JSCA International Stadium Complex**	7	1	JSCA International Stadium Complex	7
Sharjah	**Sharjah Cricket Stadium**	10	1	Sharjah Cricket Stadium	10
Visakhapatnam	**Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium**	13	1	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket St...	13
NaN	**Dubai International Cricket Stadium**	33	1	Dubai International Cricket Stadium	33
	Sharjah Cricket Stadium	18	1	Sharjah Cricket Stadium	18

👇: https://www.iplt20.com/matches/schedule/men

```
In [75]: venue_mapping_normal = {
           "Arun Jaitley Stadium": "Arun Jaitley Stadium",
           "Arun Jaitley Stadium, Delhi": "Arun Jaitley Stadium",
           "Feroz Shah Kotla": "Arun Jaitley Stadium",
           "Barsapara Cricket Stadium": "Barsapara Cricket Stadium",
           "Barsapara Cricket Stadium, Guwahati": "Barsapara Cricket Stadium",
           "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium": "Bharat Ratna Shr
           "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium, Lucknow": "Bharat 
           "Eden Gardens": "Eden Gardens",
           "Eden Gardens, Kolkata": "Eden Gardens",
           "Himachal Pradesh Cricket Association Stadium": "Himachal Pradesh Cricket Associa
           "Himachal Pradesh Cricket Association Stadium, Dharamsala": "Himachal Pradesh Cri
           "M Chinnaswamy Stadium": "M Chinnaswamy Stadium",
           "M Chinnaswamy Stadium, Bengaluru": "M Chinnaswamy Stadium",
  "M Chinnaswamy Stadium, Bangalore": "M Chinnaswamy Stadium",
  "M.Chinnaswamy Stadium": "M Chinnaswamy Stadium",
  "M.Chinnaswamy Stadium, Bengaluru": "M Chinnaswamy Stadium",
  "M.Chinnaswamy Stadium, Bangalore": "M Chinnaswamy Stadium",
  "MA Chidambaram Stadium": "MA Chidambaram Stadium",
  "MA Chidambaram Stadium, Chennai": "MA Chidambaram Stadium",
  "MA Chidambaram Stadium, Chepauk": "MA Chidambaram Stadium",
  "MA Chidambaram Stadium, Chepauk, Chennai": "MA Chidambaram Stadium",
  "Narendra Modi Stadium": "Narendra Modi Stadium",
  "Narendra Modi Stadium, Ahmedabad": "Narendra Modi Stadium",
  "Punjab Cricket Association IS Bindra Stadium": "Punjab Cricket Association IS Bi
  "Punjab Cricket Association IS Bindra Stadium, Mohali": "Punjab Cricket Associati
  "Punjab Cricket Association Stadium, Mohali": "Punjab Cricket Association IS Bind
  "Rajiv Gandhi International Stadium": "Rajiv Gandhi International Stadium",
  "Rajiv Gandhi International Stadium, Hyderabad": "Rajiv Gandhi International Stad
  "Rajiv Gandhi International Stadium, Uppal": "Rajiv Gandhi International Stadium"
  "Sawai Mansingh Stadium": "Sawai Mansingh Stadium",
  "Sawai Mansingh Stadium, Jaipur": "Sawai Mansingh Stadium",
  "Wankhede Stadium": "Wankhede Stadium",
  "Wankhede Stadium, Mumbai": "Wankhede Stadium"
}
```

In [76]:

```
venue_mapping_kebab = {
  "arun-jaitley-stadium": "Arun Jaitley Stadium",
  "arun-jaitley-stadium-delhi": "Arun Jaitley Stadium",
  "feroz-shah-kotla": "Arun Jaitley Stadium",
  "barsapara-cricket-stadium": "Barsapara Cricket Stadium",
  "barsapara-cricket-stadium-guwahati": "Barsapara Cricket Stadium",
  "bharat-ratna-shri-atal-bihari-vajpayee-ekana-cricket-stadium": "Bharat Ratna Shr
  "bharat-ratna-shri-atal-bihari-vajpayee-ekana-cricket-stadium-lucknow": "Bharat R
  "eden-gardens": "Eden Gardens",
  "eden-gardens-kolkata": "Eden Gardens",
  "himachal-pradesh-cricket-association-stadium": "Himachal Pradesh Cricket Associa
  "himachal-pradesh-cricket-association-stadium-dharamsala": "Himachal Pradesh Cric
  "m-chinnaswamy-stadium": "M Chinnaswamy Stadium",
  "m-chinnaswamy-stadium-bengaluru": "M Chinnaswamy Stadium",
  "m-chinnaswamy-stadium-bangalore": "M Chinnaswamy Stadium",
  "mchinnaswamy-stadium": "M Chinnaswamy Stadium",
  "mchinnaswamy-stadium-bengaluru": "M Chinnaswamy Stadium",
  "mchinnaswamy-stadium-bangalore": "M Chinnaswamy Stadium",
  "ma-chidambaram-stadium": "MA Chidambaram Stadium",
  "ma-chidambaram-stadium-chennai": "MA Chidambaram Stadium",
  "ma-chidambaram-stadium-chepauk": "MA Chidambaram Stadium",
  "ma-chidambaram-stadium-chepauk-chennai": "MA Chidambaram Stadium",
  "narendra-modi-stadium": "Narendra Modi Stadium",
  "narendra-modi-stadium-ahmedabad": "Narendra Modi Stadium",
  "punjab-cricket-association-is-bindra-stadium": "Punjab Cricket Association IS Bi
  "punjab-cricket-association-is-bindra-stadium-mohali": "Punjab Cricket Associatio
  "punjab-cricket-association-stadium-mohali": "Punjab Cricket Association IS Bindr
  "rajiv-gandhi-international-stadium": "Rajiv Gandhi International Stadium",
  "rajiv-gandhi-international-stadium-hyderabad": "Rajiv Gandhi International Stadi
  "rajiv-gandhi-international-stadium-uppal": "Rajiv Gandhi International Stadium",
  "sawai-mansingh-stadium": "Sawai Mansingh Stadium",
  "sawai-mansingh-stadium-jaipur": "Sawai Mansingh Stadium",
  "wankhede-stadium": "Wankhede Stadium",
  "wankhede-stadium-mumbai": "Wankhede Stadium"
}
```

In [77]:
```
np.setdiff1d(matches_result.venue.unique(), list(venue_mapping_normal.keys()))
```

Out[77]:
```
array(['Barabati Stadium', 'Brabourne Stadium',
       'Brabourne Stadium, Mumbai', 'Buffalo Park',
       'De Beers Diamond Oval', 'Dr DY Patil Sports Academy',
       'Dr DY Patil Sports Academy, Mumbai',
       'Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium',
       'Dubai International Cricket Stadium', 'Green Park',
       'Holkar Cricket Stadium', 'JSCA International Stadium Complex',
       'Kingsmead', 'Maharashtra Cricket Association Stadium',
       'Maharashtra Cricket Association Stadium, Pune', 'Nehru Stadium',
       'New Wanderers Stadium', 'Newlands', 'OUTsurance Oval',
       'Sardar Patel Stadium, Motera',
       'Saurashtra Cricket Association Stadium',
       'Shaheed Veer Narayan Singh International Stadium',
       'Sharjah Cricket Stadium', 'Sheikh Zayed Stadium',
       "St George's Park", 'Subrata Roy Sahara Stadium',
       'SuperSport Park', 'Vidarbha Cricket Association Stadium, Jamtha',
       'Zayed Cricket Stadium, Abu Dhabi'], dtype=object)
```

- ## Get Teams Mapping

In [78]:
```
set(matches_result['team_1'].unique()) == set(matches_result['team_2'].unique()) ==
```

Out[78]:
```
True
```

In [79]:
```
# Rajasthan Royals
# Gujarat Titans
# Royal Challengers Bangalore
# Lucknow Super Giants
# Sunrisers Hyderabad
# Punjab Kings [Kings XI Punjab]
# Delhi Capitals [Delhi Daredevils]
# Mumbai Indians
# Chennai Super Kings
# Kolkata Knight Riders

team_mapping = { # 10 teams
 'Rajasthan Royals': 'Rajasthan Royals',
 'Gujarat Titans': 'Gujarat Titans',
 'Royal Challengers Bangalore': 'Royal Challengers Bangalore',
 'Lucknow Super Giants': 'Lucknow Super Giants',
 'Sunrisers Hyderabad': 'Sunrisers Hyderabad',
 'Mumbai Indians': 'Mumbai Indians',
 'Chennai Super Kings': 'Chennai Super Kings',
 'Kolkata Knight Riders': 'Kolkata Knight Riders',

 'Kings XI Punjab': 'Punjab Kings',
 'Punjab Kings': 'Punjab Kings',

 'Delhi Daredevils': 'Delhi Capitals',
 'Delhi Capitals': 'Delhi Capitals',
}
```

In [80]:

```
print(np.setdiff1d(
   list(team_mapping.keys()), matches_result['team_1'].unique()
))

print(np.setdiff1d(
    matches_result['team_1'].unique(), list(team_mapping.keys())
))
```

```
[]
['Deccan Chargers' 'Gujarat Lions' 'Kochi Tuskers Kerala' 'Pune Warriors'
 'Rising Pune Supergiant' 'Rising Pune Supergiants']
```

- ## Apply Venues/Teams Mapping [in matches_result, ball_by_ball]

In [81]:

```
matches_result.venue = matches_result.venue.map(venue_mapping_normal).fillna('Other

matches_result.team_1 = matches_result.team_1.map(team_mapping).fillna('Other')
matches_result.team_2 = matches_result.team_2.map(team_mapping).fillna('Other')

ball_by_ball.batting_team = ball_by_ball.batting_team.map(team_mapping).fillna('Oth
```

In [82]:

```
matches_result.venue[matches_result.venue == 'Other'].shape
```

Out[82]:

```
(359,)
```

In [83]:

```
print(matches_result.team_1[matches_result.team_1 == 'Other'].shape)
print(matches_result.team_2[matches_result.team_2 == 'Other'].shape)
```

```
(99,)
(96,)
```

In [84]:

```
ball_by_ball.batting_team[ball_by_ball.batting_team == 'Other'].shape
```

Out[84]:

```
(23105,)
```

In [85]:

```
print(matches_result.shape)
print(ball_by_ball.shape)
```

```
(950, 4)
(225954, 8)
```

- ## Remove NA Teams [in ball_by_ball] and Venues [in matches_result]

In [86]:

```
# matches_result = matches_result.dropna(subset=['team_1', 'team_2', 'venue'])
# print(matches_result.shape)
# ball_by_ball = ball_by_ball.dropna(subset=['batting_team'])
# print(ball_by_ball.shape)
```

- ## Select first 6 overs, Select innings 1 & 2, Map innings (1,2) to (0,1) [in ball_by_ball]

In [87]:
```
ball_by_ball.innings.unique()
```

Out[87]:
```
array([1, 2, 3, 4, 5, 6], dtype=int64)
```

In [88]:
```
ball_by_ball.overs.unique()
```

Out[88]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19], dtype=int64)
```

In [89]:
```
ball_by_ball = ball_by_ball.loc[(ball_by_ball.overs <= 5) & (ball_by_ball.innings <
ball_by_ball.innings = ball_by_ball.innings.replace({1: 0, 2: 1})
ball_by_ball.shape
```

Out[89]:
```
(70921, 8)
```

In [90]:
```
ball_by_ball.innings.unique()
```

Out[90]:
```
array([0, 1], dtype=int64)
```

In [91]:
```
ball_by_ball.overs.unique()
```

Out[91]:
```
array([0, 1, 2, 3, 4, 5], dtype=int64)
```

- ## Grouping

In [92]:
```
ball_by_ball_gb = ball_by_ball.groupby(['match_id', 'innings', 'batting_team'])
```

In [93]:
```
total_runs = ball_by_ball_gb['total_run'].sum()
batsmen = ball_by_ball_gb['batter'].unique()
bowlers = ball_by_ball_gb['bowler'].unique()
```

In [94]:
```
total_runs = total_runs.to_frame(name = 'total_runs').reset_index()
batsmen = batsmen.to_frame(name = 'batsmen').reset_index()
bowlers = bowlers.to_frame(name = 'bowlers').reset_index()
```

In [95]:
```
data = total_runs.merge(batsmen, how='right', on=['match_id','innings','batting_tea
data = data.merge(bowlers, how='right', on=['match_id','innings','batting_team'])
data = data.merge(matches_result, on=['match_id'])
```

In [96]:
```
mask = data['batting_team'] == data['team_1']
data.loc[mask, 'bowling_team'] = data['team_2']
data.loc[~mask, 'bowling_team'] = data['team_1']
```

In [97]:

```
data.query('match_id == 829763')
```

Out[97]:

	match_id	innings	batting_team	total_runs	batsmen	bowlers	team_1	team_
971	829763	0	Royal Challengers Bangalore	52	[CH Gayle, AB de Villiers, V Kohli, Mandeep Si...	[TG Southee, DS Kulkarni, JP Faulkner, SR Watson]	Royal Challengers Bangalore	Rajastha Royal

In [98]:

```
data.query('match_id == 829813')
```

Out[98]:

	match_id	innings	batting_team	total_runs	batsmen	bowlers	team_1	team_2
1020	829813	0	Delhi Capitals	54	[Q de Kock, SS Iyer]	[MA Starc, AB Dinda, HV Patel, D Wiese]	Royal Challengers Bangalore	Delhi Capitals
1021	829813	1	Royal Challengers Bangalore	2	[V Kohli, CH Gayle]	[J Yadav, Z Khan]	Royal Challengers Bangalore	Delhi Capitals

In [99]:

```
# match_id == 829763, data for one innings is missing
# match_id == 829813, total_runs for one innings is 2 (probably a mistake in data e
data = data.drop(data[(data['match_id'] == 829763) | (data['match_id'] == 829813)].
```

In [100…

```
# get count of batsmen & bowlers for each innings
data['count_batsmen'] = [len(x) for x in data['batsmen']]
data['count_bowlers'] = [len(x) for x in data['bowlers']]
```

In [101…

```
data = data[
    ['venue', 'innings', 'batting_team', 'bowling_team', 'count_batsmen', 'count_bo
]
```

Prepared training dataset

In [102…

```
data
```

Out[102]:

	venue	innings	batting_team	bowling_team	count_batsmen	count_bowlers
0	M Chinnaswamy Stadium	0	Kolkata Knight Riders	Royal Challengers Bangalore	3	3
1	M Chinnaswamy Stadium	1	Royal Challengers Bangalore	Kolkata Knight Riders	6	3
2	Punjab Cricket Association IS Bindra Stadium	0	Chennai Super Kings	Punjab Kings	3	3
3	Punjab Cricket Association IS Bindra Stadium	1	Punjab Kings	Chennai Super Kings	2	2
4	Arun Jaitley Stadium	0	Rajasthan Royals	Delhi Capitals	4	3
...	...	...	...	...	...	...
1893	Eden Gardens	1	Lucknow Super Giants	Royal Challengers Bangalore	4	3
1894	Narendra Modi Stadium	0	Royal Challengers Bangalore	Rajasthan Royals	3	2
1895	Narendra Modi Stadium	1	Rajasthan Royals	Royal Challengers Bangalore	3	4
1896	Narendra Modi Stadium	0	Rajasthan Royals	Gujarat Titans	3	4
1897	Narendra Modi Stadium	1	Gujarat Titans	Rajasthan Royals	4	3

1895 rows × 7 columns

In [103…

```
data.groupby(['venue']).total_runs.describe()[['count', 'mean', '75%']].sort_values
```

Out[103]:

venue	count	mean	75%
Himachal Pradesh Cricket Association Stadium	18.0	40.555556	48.00
Sawai Mansingh Stadium	94.0	45.042553	55.00
Other	718.0	45.362117	53.00
Wankhede Stadium	208.0	45.480769	53.25
Rajiv Gandhi International Stadium	128.0	45.585938	54.25
M Chinnaswamy Stadium	156.0	46.025641	54.25
Narendra Modi Stadium	14.0	46.071429	48.25
MA Chidambaram Stadium	134.0	46.425373	53.75
Eden Gardens	158.0	46.569620	52.00
Arun Jaitley Stadium	155.0	47.832258	55.00
Punjab Cricket Association IS Bindra Stadium	112.0	48.428571	55.00

In [104…

```
data.groupby(['batting_team']).total_runs.describe()[['count', 'mean', '75%']].sort
```

Out[104]:

batting_team	count	mean	75%
Lucknow Super Giants	15.0	44.666667	56.00
Royal Challengers Bangalore	224.0	44.852679	52.25
Rajasthan Royals	191.0	45.172775	53.00
Chennai Super Kings	208.0	45.221154	53.00
Mumbai Indians	231.0	45.480519	53.00
Kolkata Knight Riders	223.0	46.076233	53.00
Other	194.0	46.226804	55.00
Gujarat Titans	16.0	46.250000	53.00
Delhi Capitals	223.0	46.609865	55.00
Sunrisers Hyderabad	152.0	47.118421	56.00
Punjab Kings	218.0	47.133028	53.00

In [105…

```
data.groupby(['count_batsmen']).total_runs.describe()[['count', 'mean', '75%']].sor
```

Out[105]:

count_batsmen	count	mean	75%
7	9.0	29.888889	32.00
6	59.0	34.847458	39.00
5	190.0	37.542105	44.75
4	499.0	42.679359	49.50
8	2.0	45.500000	53.75
3	684.0	47.545322	54.25
2	452.0	52.442478	59.00

In [106…

```
data.groupby(['count_bowlers']).total_runs.describe()[['count', 'mean', '75%']].sor
```

Out[106]:

count_bowlers	count	mean	75%
2	95.0	39.484211	47.0
3	767.0	43.615385	51.0
4	903.0	47.496124	55.0
5	124.0	53.451613	60.0
6	6.0	58.333333	60.0

In [108…

```
tmp = data.groupby(['batting_team', 'venue']).total_runs.describe()[['count', 'mean
tmp[tmp.batting_team == 'Gujarat Titans']
```

Out[108]:

	batting_team	venue	count	mean	75%
0	Gujarat Titans	Narendra Modi Stadium	1.0	31.0	31.0
37	Gujarat Titans	Other	10.0	45.1	50.0
71	Gujarat Titans	Wankhede Stadium	4.0	48.5	54.5
98	Gujarat Titans	Eden Gardens	1.0	64.0	64.0

- ## Encoding of categorical inputs and feature scaling

In [47]:

```
X = data.iloc[:, :-1]
y = data["total_runs"]
```

In [48]:

```
ct = ColumnTransformer(transformers = [
    ('ohe', OneHotEncoder(categories = "auto", drop='first', sparse_output=False),
], remainder = 'passthrough')

scaler = StandardScaler()

X_ohe = pd.DataFrame(ct.fit_transform(X))
X_std = scaler.fit_transform(X_ohe)
```

In [49]:

```
import numpy as np
from sklearn.preprocessing import OneHotEncoder

# Create some sample data
data = np.array([
    ['red', 'small'],
    ['green', 'large'],
    ['blue', 'medium'],
    ['green', 'medium'],
    ['red', 'large']
])

# Create an instance of OneHotEncoder
encoder = OneHotEncoder(categories='auto', drop='first', sparse=False)

# Fit and transform the data
encoded_data = encoder.fit_transform(data)

# Print the encoded data
print(encoded_data)
```

```
[[0. 1. 0. 1.]
 [1. 0. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 1. 0.]
 [0. 1. 0. 0.]]
```

```
C:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\pre
processing\_encoders.py:868: FutureWarning: `sparse` was renamed to `sparse_output`
in version 1.2 and will be removed in 1.4. `sparse_output` is ignored unless you lea
ve `sparse` to its default value.
  warnings.warn(
```

In [ ]:

```

```

In [50]:

```
X_std[0]
```

Out[50]:

```
array([-0.30159812, -0.09792738,  3.33877761, -0.27584983, -0.08627195,
       -0.78104128, -0.25063016, -0.26914524, -0.22845837, -0.351135  ,
       -0.36520297, -0.09227767,  2.7382034 , -0.0893237 , -0.3725884 ,
       -0.33771372, -0.36054686, -0.33479725, -0.3661304 , -0.29530656,
       -0.36427429, -0.09227767, -0.36520297, -0.0893237 , -0.3725884 ,
       -0.33868257, -0.36054686, -0.33479725,  2.73126737, -0.29530656,
       -1.72962611, -0.99947243, -0.31740491, -0.80500065])
```

- ## Train-test split

In [51]:

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X_std, y, test_size = 0.2)
```

In [52]:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

def evaluate(regressor):
    regressor.fit(X_train, y_train)
    y_pred = regressor.predict(X_test)

    # Calculate the mean absolute error (MAE)
    mae = mean_absolute_error(y_test, y_pred)
    print('MAE:', mae)

    # Calculate the root mean squared error (RMSE)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    print('RMSE:', rmse)

    # Calculate the R-squared score
    r2 = r2_score(y_test, y_pred)
    print('R-squared:', r2)
```

- ## Models

In [53]:

```
from sklearn.ensemble import AdaBoostRegressor
regressor = AdaBoostRegressor(
    learning_rate=1, loss='exponential', n_estimators=100, random_state=42
)
evaluate(regressor)
```

```
MAE: 8.474651550052043
RMSE: 10.811789111331231
R-squared: 0.13567234625080005
```

In [54]:

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
evaluate(regressor)
```

```
MAE: 7.781309345474129
RMSE: 10.009973010420477
R-squared: 0.2591179220379092
```

In [55]:

```
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor()
evaluate(regressor)
```

```
MAE: 11.419525065963061
RMSE: 14.59587537067961
R-squared: -0.5752285303095366
```

In [56]:

```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor()
evaluate(regressor)
```

```
MAE: 8.022084432717678
RMSE: 10.356927945311773
R-squared: 0.20686852335306882
```

In [57]:

```python
from sklearn.neighbors import KNeighborsRegressor
regressor = KNeighborsRegressor()
evaluate(regressor)
```

```
MAE: 9.856992084432719
RMSE: 12.343051627650661
R-squared: -0.12649266266135295
```

In [58]:

```python
from sklearn.svm import SVR
regressor = SVR()
evaluate(regressor)
```

```
MAE: 8.058753580672196
RMSE: 10.361738289128354
R-squared: 0.20613160191841462
```

In [59]:

```python
import xgboost as xgb
regressor = xgb.XGBRegressor()
evaluate(regressor)
```

```
MAE: 8.519698880278655
RMSE: 10.971088654552387
R-squared: 0.11001492165704674
```

In [60]:

```python
import tensorflow as tf
from tensorflow.keras import layers, models

# Define the model architecture
model = models.Sequential([
    layers.Dense(256, activation='relu', input_shape=(X_train.shape[1],)),
    layers.Dense(128, activation='relu'),
    layers.Dense(1)
])

# Compile the model
model.compile(optimizer='adam', loss='mean_absolute_error', metrics=['mae'])

# Fit the model to the training data
history = model.fit(X_train, y_train, epochs=200, batch_size=128, verbose=False)

# Evaluate the model on the test set
test_loss = model.evaluate(X_test, y_test)

# Print the test loss
print('Test loss:', test_loss)
```

```
12/12 [==============================] - 0s 958us/step - loss: 9.6944 - mae: 9.6944
Test loss: [9.694378852844238, 9.694378852844238]
```

In [61]:

```python
# import tensorflow as tf
# from tensorflow.keras import layers, models

# # Define a matrix of hyperparameters to test
# params = {
#     'batch_size': [16, 32],
#     'epochs': [50, 100],
#     'learning_rate': [0.001, 0.01]
# }

# # Define the model architecture
# def build_model(learning_rate=0.001):
#     model = models.Sequential([
#         layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
#         layers.Dense(32, activation='relu'),
#         layers.Dense(1)
#     ])
#     optimizer = tf.keras.optimizers.Adam(learning_rate=learning_rate)
#     model.compile(optimizer=optimizer, loss='mse', metrics=['mae'])
#     return model

# # Loop through the hyperparameter matrix and fit the model for each combination
# for batch_size in params['batch_size']:
#     for epochs in params['epochs']:
#         for learning_rate in params['learning_rate']:
#             print(f"Fitting model with batch_size={batch_size}, epochs={epochs},
#             model = build_model(learning_rate=learning_rate)
#             history = model.fit(X_train, y_train, epochs=epochs, batch_size=batch
#             test_loss, test_mae = model.evaluate(X_test, y_test)
#             print(f"Test loss: {test_loss}, Test MAE: {test_mae}")
```