⭐ Cricket Hackathon 2023

- https://iitm-ipl.web.app/
- https://drive.google.com/drive/folders/1UA8LLt_D1W4dN-XrfUrbFg5PMrbuBzM4?sort=13&direction=a
- https://www.sportskeeda.com/cricket/yesterday-ipl-match-result
- https://www.iplt20.com/matches/schedule/men
- https://aiimsexams.org/ipl-schedule/
- https://www.icccricketschedule.com/ipl-2023-schedule-team-venue-time-table-pdf-point-table-ranking-winning-prediction/
- https://www.timesofsports.com/cricket/ipl/squad-2023/

In [1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

In [2]:
```
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 [3]:
```
np.random.seed(2)
```

In [4]:
```
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_venues = pd.read_csv('./Data/Ipl_2023 _cricketers - Venue.csv').rename(col
    'Venue': 'venue'
})
ipl_2023_teams = pd.read_csv('./Data/Ipl_2023 _cricketers - Team name.csv').rename(
    'Teams': 'team'
})
```

In [5]:
```
def log(*args):
    print('👉', *args)
```

In [6]:
```
def to_kebab_case(string):
    return '-'.join(
        string.replace(",", "").replace(".", "").split()
    ).lower()
```

💚 Cleaning `ball_by_ball`, `matches_result`

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

In [7]:
```
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 [8]:
```
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 [9]:
```
print(ball_by_ball_orig.shape)
ball_by_ball_orig.head()
```

```
(225954, 17)
```

Out[9]:

	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 [10]:
```
print(matches_result_orig.shape)
matches_result_orig.head()
```

```
(950, 20)
```

Out[10]:

	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 [11]:
```
print(ball_by_ball.shape)
ball_by_ball.head()
```

```
(225954, 8)
```

Out[11]:

	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 [12]:
```
print(matches_result.shape)
matches_result.head()
```

```
(950, 4)
```

Out[12]:

	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 [13]:
```
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 [14]:
```
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,)

- # Venues Mapping

In [15]:
```
matches_result_orig.groupby(['City', 'Venue'], dropna=False)['Venue'].describe()
```

Out[15]:

City	Venue	count	unique	top	freq
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

City	Venue	count	unique	top	freq
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

City	Venue	count	unique	top	freq
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 [16]:
```
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 [17]:

```
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 [18]:
```
venue_mapping_tags = {
  "delhi": "Arun Jaitley Stadium",
  "arun jaitley": "Arun Jaitley Stadium",
  "guwahati": "Barsapara Cricket Stadium",
  "barsapara": "Barsapara Cricket Stadium",
  "bhupen hazarika": "Barsapara Cricket Stadium",
  "assam cricket association stadium": "Barsapara Cricket Stadium",
  "lucknow": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
  "ekana": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
  "atal bihari": "Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium",
  "kolkata": "Eden Gardens",
  "eden gardens": "Eden Gardens",
  "dharamsala": "Himachal Pradesh Cricket Association Stadium",
  "himachal pradesh": "Himachal Pradesh Cricket Association Stadium",
  "bengaluru": "M Chinnaswamy Stadium",
  "bengalore": "M Chinnaswamy Stadium",
  "chinnaswamy": "M Chinnaswamy Stadium",
  "chennai": "MA Chidambaram Stadium",
  "chepauk": "MA Chidambaram Stadium",
  "chidambaram": "MA Chidambaram Stadium",
  "ahmedabad": "Narendra Modi Stadium",
  "narendra modi": "Narendra Modi Stadium",
  "mohali": "Punjab Cricket Association IS Bindra Stadium",
  "punjab cricket association": "Punjab Cricket Association IS Bindra Stadium",
  "is bindra": "Punjab Cricket Association IS Bindra Stadium",
  "hyderabad": "Rajiv Gandhi International Stadium",
  "rajiv gandhi": "Rajiv Gandhi International Stadium",
  "jaipur": "Sawai Mansingh Stadium",
  "sawai mansingh": "Sawai Mansingh Stadium",
  "mumbai": "Wankhede Stadium",
  "wankhede": "Wankhede Stadium"
}
```

In [19]:
```
# stadiums where ipl23 isn't happening (collectively considered as 'Other' venue af
np.setdiff1d(matches_result.venue.unique(), list(venue_mapping_normal.keys()))
```

Out[19]:
```
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)
```

- # Teams Mapping

In [20]:
```
set(matches_result['team_1'].unique()) == set(matches_result['team_2'].unique()) ==
```

Out[20]:
```
True
```

In [21]:
```
team_mapping = {
 '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 [22]:
```
# should be empty array before mapping, ['Delhi Daredevils', 'Kings XI Punjab'] aft
np.setdiff1d(
   list(team_mapping.keys()), matches_result['team_1'].unique()
)
```

Out[22]:
```
array([], dtype='<U27')
```

In [23]:
```
# old ipl teams (collectively considered as 'Other' team after mapping)
np.setdiff1d(
    matches_result['team_1'].unique(), list(team_mapping.keys())
)
```

Out[23]:
```
array(['Deccan Chargers', 'Gujarat Lions', 'Kochi Tuskers Kerala',
       'Pune Warriors', 'Rising Pune Supergiant',
       'Rising Pune Supergiants'], dtype=object)
```

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

In [24]:
```
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 [25]:
```
matches_result.venue[matches_result.venue == 'Other'].shape
```

Out[25]:
```
(359,)
```

In [26]:
```
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 [27]:
```
ball_by_ball.batting_team[ball_by_ball.batting_team == 'Other'].shape
```

Out[27]:
```
(23105,)
```

In [28]:
```
print(matches_result.shape)
print(ball_by_ball.shape)
```

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

In [29]:
```
all_ipl23_venues = ['Arun Jaitley Stadium', 'Barsapara Cricket Stadium',
       'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium',
       'Eden Gardens', 'Himachal Pradesh Cricket Association Stadium',
       'M Chinnaswamy Stadium', 'MA Chidambaram Stadium',
       'Narendra Modi Stadium',
       'Punjab Cricket Association IS Bindra Stadium',
       'Rajiv Gandhi International Stadium', 'Sawai Mansingh Stadium',
       'Wankhede Stadium']
```

In [30]:
```
all_ipl23_teams = ['Chennai Super Kings', 'Delhi Capitals', 'Gujarat Titans',
       'Kolkata Knight Riders', 'Lucknow Super Giants', 'Mumbai Indians',
       'Punjab Kings', 'Rajasthan Royals', 'Royal Challengers Bangalore',
       'Sunrisers Hyderabad']
```

In [31]:
```
# True
np.array_equal(np.unique(
        list(venue_mapping_normal.values())
), all_ipl23_venues)
```

```
# True
np.array_equal(np.unique(
        list(team_mapping.values())
), all_ipl23_teams)
```

Out[31]: True

Out[31]: True

In [32]:
```
# ['Barsapara Cricket Stadium', 'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Crick
np.setdiff1d(
    list(venue_mapping_normal.values()), matches_result.venue.unique()
)
```

Out[32]:
```
array(['Barsapara Cricket Stadium',
       'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium'],
      dtype='<U60')
```

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

In [33]:
```
ball_by_ball.innings.unique()
```

Out[33]:
```
array([1, 2, 3, 4, 5, 6], dtype=int64)
```

In [34]:
```
ball_by_ball.overs.unique()
```

Out[34]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19], dtype=int64)
```

In [35]:
```
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[35]:
```
(70921, 8)
```

In [36]:
```
ball_by_ball.innings.unique()
```

Out[36]:
```
array([0, 1], dtype=int64)
```

In [37]:
```
ball_by_ball.overs.unique()
```

Out[37]:
```
array([0, 1, 2, 3, 4, 5], dtype=int64)
```

- ## Grouping (Integration of `ball_by_ball` and `matches_result`)

In [38]:
```
ball_by_ball_gb = ball_by_ball.groupby(['match_id', 'innings', 'batting_team'])
```

In [39]:
```
total_runs = ball_by_ball_gb['total_run'].sum()
batsmen = ball_by_ball_gb['batter'].unique()
bowlers = ball_by_ball_gb['bowler'].unique()
```

In [40]:
```
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 [41]:
```
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 [42]:
```
mask = data['batting_team'] == data['team_1']
data.loc[mask, 'bowling_team'] = data['team_2']
data.loc[~mask, 'bowling_team'] = data['team_1']
```

In [43]:
```
data.query('match_id == 829763')
```

Out[43]:

	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 [44]:
```
data.query('match_id == 829813')
```

Out[44]:

	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 [45]:
```
# 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 [46]:
```
# 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 [47]:
```
data = data[
    ['venue', 'innings', 'batting_team', 'bowling_team', 'count_batsmen', 'count_bo
]
```

Final training dataset: `data`

In [48]:
```
data
```

Out[48]:

	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 [49]:
```
data.groupby(['venue']).total_runs.describe()[['count', 'mean', '75%']].sort_values
```

Out[49]:

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 [50]:
```
data.groupby(['batting_team']).total_runs.describe()[['count', 'mean', '75%']].sort
```

Out[50]:

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 [51]:
```
data.groupby(['count_batsmen']).total_runs.describe()[['count', 'mean', '75%']].sor
```

Out[51]:

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 [52]:
```
data.groupby(['count_bowlers']).total_runs.describe()[['count', 'mean', '75%']].sor
```

Out[52]:

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 [53]:
```
df = data.groupby(['batting_team', 'venue']).total_runs.describe()[['count', 'mean'
for team in all_ipl23_teams:
    f' {team}'
    df.query(f'batting_team == "{team}"')
```

Out[53]:
```
' Chennai Super Kings'
```

Out[53]:

	batting_team	venue	count	mean	75%
3	Chennai Super Kings	Himachal Pradesh Cricket Association Stadium	2.0	37.000000	43.00
10	Chennai Super Kings	Eden Gardens	11.0	39.818182	47.50
15	Chennai Super Kings	Sawai Mansingh Stadium	6.0	40.833333	50.75
16	Chennai Super Kings	M Chinnaswamy Stadium	9.0	41.111111	55.00
27	Chennai Super Kings	Wankhede Stadium	23.0	43.652174	49.00
29	Chennai Super Kings	Punjab Cricket Association IS Bindra Stadium	6.0	44.000000	51.50
35	Chennai Super Kings	Rajiv Gandhi International Stadium	6.0	44.833333	52.25
50	Chennai Super Kings	MA Chidambaram Stadium	56.0	46.107143	53.25
54	Chennai Super Kings	Other	79.0	46.392405	53.50
79	Chennai Super Kings	Arun Jaitley Stadium	10.0	49.500000	54.50

Out[53]: `'🔏 Delhi Capitals'`

Out[53]:

	batting_team	venue	count	mean	75%
2	Delhi Capitals	Himachal Pradesh Cricket Association Stadium	3.0	35.666667	43.00
22	Delhi Capitals	Punjab Cricket Association IS Bindra Stadium	7.0	42.571429	47.00
40	Delhi Capitals	Wankhede Stadium	17.0	45.588235	55.00
42	Delhi Capitals	Eden Gardens	9.0	45.666667	53.00
49	Delhi Capitals	MA Chidambaram Stadium	10.0	46.000000	51.75
55	Delhi Capitals	Arun Jaitley Stadium	70.0	46.528571	53.75
57	Delhi Capitals	Other	81.0	46.728395	55.00
65	Delhi Capitals	Rajiv Gandhi International Stadium	8.0	48.000000	57.00
78	Delhi Capitals	M Chinnaswamy Stadium	9.0	49.444444	53.00
83	Delhi Capitals	Sawai Mansingh Stadium	6.0	49.833333	59.75
96	Delhi Capitals	Narendra Modi Stadium	3.0	57.666667	65.00

Out[53]: `'🔏 Gujarat Titans'`

Out[53]:

	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

Out[53]: `'Kolkata Knight Riders'`

Out[53]:

	batting_team	venue	count	mean	75%
13	Kolkata Knight Riders	Wankhede Stadium	15.0	40.733333	49.50
24	Kolkata Knight Riders	Other	76.0	43.473684	52.25
26	Kolkata Knight Riders	Narendra Modi Stadium	2.0	43.500000	44.25
33	Kolkata Knight Riders	Rajiv Gandhi International Stadium	8.0	44.250000	48.25
52	Kolkata Knight Riders	Arun Jaitley Stadium	10.0	46.300000	50.50
61	Kolkata Knight Riders	Eden Gardens	74.0	47.256757	51.00
75	Kolkata Knight Riders	MA Chidambaram Stadium	12.0	49.166667	58.00
81	Kolkata Knight Riders	Sawai Mansingh Stadium	6.0	49.666667	59.75
91	Kolkata Knight Riders	Punjab Cricket Association IS Bindra Stadium	7.0	52.428571	61.50
94	Kolkata Knight Riders	M Chinnaswamy Stadium	13.0	54.153846	61.00

Out[53]: `'Lucknow Super Giants'`

Out[53]:

	batting_team	venue	count	mean	75%
4	Lucknow Super Giants	Wankhede Stadium	4.0	38.0	38.25
41	Lucknow Super Giants	Other	10.0	45.6	53.25
97	Lucknow Super Giants	Eden Gardens	1.0	62.0	62.00

Out[53]: `'Mumbai Indians'`

Out[53]:

	batting_team	venue	count	mean	75%
5	Mumbai Indians	Sawai Mansingh Stadium	7.0	38.000000	45.00
8	Mumbai Indians	M Chinnaswamy Stadium	12.0	39.416667	42.00
21	Mumbai Indians	Rajiv Gandhi International Stadium	11.0	42.545455	49.50
39	Mumbai Indians	Other	82.0	45.585366	53.00
43	Mumbai Indians	Eden Gardens	13.0	45.692308	51.00
48	Mumbai Indians	Wankhede Stadium	71.0	45.985915	53.00
56	Mumbai Indians	MA Chidambaram Stadium	13.0	46.615385	53.00
60	Mumbai Indians	Punjab Cricket Association IS Bindra Stadium	8.0	47.125000	53.75
88	Mumbai Indians	Himachal Pradesh Cricket Association Stadium	1.0	51.000000	51.00
89	Mumbai Indians	Arun Jaitley Stadium	13.0	51.384615	58.00

Out[53]: `'🔏 Punjab Kings'`

Out[53]:

	batting_team	venue	count	mean	75%
6	Punjab Kings	MA Chidambaram Stadium	8.0	39.250000	43.50
11	Punjab Kings	Himachal Pradesh Cricket Association Stadium	9.0	40.111111	45.00
18	Punjab Kings	Narendra Modi Stadium	3.0	41.666667	44.00
19	Punjab Kings	Sawai Mansingh Stadium	6.0	41.666667	50.25
53	Punjab Kings	Other	74.0	46.364865	51.75
68	Punjab Kings	Arun Jaitley Stadium	11.0	48.363636	53.50
69	Punjab Kings	M Chinnaswamy Stadium	12.0	48.416667	54.50
72	Punjab Kings	Eden Gardens	11.0	48.727273	54.00
76	Punjab Kings	Wankhede Stadium	18.0	49.222222	58.50
77	Punjab Kings	Punjab Cricket Association IS Bindra Stadium	56.0	49.321429	55.00
82	Punjab Kings	Rajiv Gandhi International Stadium	10.0	49.700000	51.00

Out[53]: `'🔏 Rajasthan Royals'`

Out[53]:

	batting_team	venue	count	mean	75%
17	Rajasthan Royals	M Chinnaswamy Stadium	7.0	41.142857	45.00
25	Rajasthan Royals	Rajiv Gandhi International Stadium	6.0	43.500000	46.25
30	Rajasthan Royals	Arun Jaitley Stadium	11.0	44.181818	44.50
31	Rajasthan Royals	Other	76.0	44.210526	52.00
38	Rajasthan Royals	Sawai Mansingh Stadium	47.0	45.361702	54.50
46	Rajasthan Royals	Wankhede Stadium	18.0	45.722222	54.25
63	Rajasthan Royals	Eden Gardens	10.0	47.400000	54.00
64	Rajasthan Royals	Punjab Cricket Association IS Bindra Stadium	7.0	48.000000	53.50
87	Rajasthan Royals	MA Chidambaram Stadium	7.0	51.000000	58.50
95	Rajasthan Royals	Narendra Modi Stadium	2.0	55.500000	61.25

Out[53]: `' Royal Challengers Bangalore'`

Out[53]:

	batting_team	venue	count	mean	75%
1	Royal Challengers Bangalore	Himachal Pradesh Cricket Association Stadium	1.0	34.000000	34.00
7	Royal Challengers Bangalore	Narendra Modi Stadium	3.0	39.333333	41.00
20	Royal Challengers Bangalore	Other	74.0	42.283784	49.00
23	Royal Challengers Bangalore	Eden Gardens	11.0	43.272727	52.00
28	Royal Challengers Bangalore	MA Chidambaram Stadium	12.0	43.750000	48.25
34	Royal Challengers Bangalore	Rajiv Gandhi International Stadium	10.0	44.300000	51.75
45	Royal Challengers Bangalore	M Chinnaswamy Stadium	74.0	45.716216	52.75
67	Royal Challengers Bangalore	Wankhede Stadium	16.0	48.250000	55.00
74	Royal Challengers Bangalore	Sawai Mansingh Stadium	7.0	49.000000	57.00
85	Royal Challengers Bangalore	Arun Jaitley Stadium	9.0	49.888889	58.00
92	Royal Challengers Bangalore	Punjab Cricket Association IS Bindra Stadium	7.0	53.571429	59.50

Out[53]: `' Sunrisers Hyderabad'`

Out[53]:

	batting_team	venue	count	mean	75%
9	Sunrisers Hyderabad	Sawai Mansingh Stadium	3.0	39.666667	45.00
36	Sunrisers Hyderabad	Eden Gardens	8.0	44.875000	48.25
44	Sunrisers Hyderabad	Other	54.0	45.703704	55.75
47	Sunrisers Hyderabad	Punjab Cricket Association IS Bindra Stadium	6.0	45.833333	55.75
62	Sunrisers Hyderabad	Arun Jaitley Stadium	10.0	47.300000	55.50
66	Sunrisers Hyderabad	Wankhede Stadium	11.0	48.090909	52.50
70	Sunrisers Hyderabad	MA Chidambaram Stadium	8.0	48.500000	54.50
73	Sunrisers Hyderabad	Rajiv Gandhi International Stadium	44.0	48.772727	59.00
86	Sunrisers Hyderabad	M Chinnaswamy Stadium	8.0	50.625000	56.75

In [54]:
```
df = data.groupby(['venue', 'batting_team']).total_runs.describe()[['count', 'mean'
for venue in all_ipl23_venues:
    f'🏟 {venue} 👇'
    df.query(f'venue == "{venue}"')
```

Out[54]: '🏟 Arun Jaitley Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
30	Arun Jaitley Stadium	Rajasthan Royals	11.0	44.181818	44.50
52	Arun Jaitley Stadium	Kolkata Knight Riders	10.0	46.300000	50.50
55	Arun Jaitley Stadium	Delhi Capitals	70.0	46.528571	53.75
62	Arun Jaitley Stadium	Sunrisers Hyderabad	10.0	47.300000	55.50
68	Arun Jaitley Stadium	Punjab Kings	11.0	48.363636	53.50
79	Arun Jaitley Stadium	Chennai Super Kings	10.0	49.500000	54.50
85	Arun Jaitley Stadium	Royal Challengers Bangalore	9.0	49.888889	58.00
89	Arun Jaitley Stadium	Mumbai Indians	13.0	51.384615	58.00
93	Arun Jaitley Stadium	Other	11.0	53.727273	62.50

Out[54]: '🏟 Barsapara Cricket Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%

Out[54]: '🏟 Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium 👇'

Out[54]:

venue	batting_team	count	mean	75%

Out[54]: '🏟 Eden Gardens 👇'

Out[54]:

	venue	batting_team	count	mean	75%
10	Eden Gardens	Chennai Super Kings	11.0	39.818182	47.50
23	Eden Gardens	Royal Challengers Bangalore	11.0	43.272727	52.00
36	Eden Gardens	Sunrisers Hyderabad	8.0	44.875000	48.25
42	Eden Gardens	Delhi Capitals	9.0	45.666667	53.00
43	Eden Gardens	Mumbai Indians	13.0	45.692308	51.00
61	Eden Gardens	Kolkata Knight Riders	74.0	47.256757	51.00
63	Eden Gardens	Rajasthan Royals	10.0	47.400000	54.00
72	Eden Gardens	Punjab Kings	11.0	48.727273	54.00
81	Eden Gardens	Other	9.0	49.666667	59.00
97	Eden Gardens	Lucknow Super Giants	1.0	62.000000	62.00
98	Eden Gardens	Gujarat Titans	1.0	64.000000	64.00

Out[54]: '🏟 Himachal Pradesh Cricket Association Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
1	Himachal Pradesh Cricket Association Stadium	Royal Challengers Bangalore	1.0	34.000000	34.00
2	Himachal Pradesh Cricket Association Stadium	Delhi Capitals	3.0	35.666667	43.00
3	Himachal Pradesh Cricket Association Stadium	Chennai Super Kings	2.0	37.000000	43.00
11	Himachal Pradesh Cricket Association Stadium	Punjab Kings	9.0	40.111111	45.00
88	Himachal Pradesh Cricket Association Stadium	Mumbai Indians	1.0	51.000000	51.00
90	Himachal Pradesh Cricket Association Stadium	Other	2.0	51.500000	54.75

Out[54]: '🏟 M Chinnaswamy Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
8	M Chinnaswamy Stadium	Mumbai Indians	12.0	39.416667	42.00
16	M Chinnaswamy Stadium	Chennai Super Kings	9.0	41.111111	55.00
17	M Chinnaswamy Stadium	Rajasthan Royals	7.0	41.142857	45.00
32	M Chinnaswamy Stadium	Other	12.0	44.250000	50.50
45	M Chinnaswamy Stadium	Royal Challengers Bangalore	74.0	45.716216	52.75
69	M Chinnaswamy Stadium	Punjab Kings	12.0	48.416667	54.50
78	M Chinnaswamy Stadium	Delhi Capitals	9.0	49.444444	53.00
86	M Chinnaswamy Stadium	Sunrisers Hyderabad	8.0	50.625000	56.75
94	M Chinnaswamy Stadium	Kolkata Knight Riders	13.0	54.153846	61.00

Out[54]: '🏟 MA Chidambaram Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
6	MA Chidambaram Stadium	Punjab Kings	8.0	39.250000	43.50
28	MA Chidambaram Stadium	Royal Challengers Bangalore	12.0	43.750000	48.25
49	MA Chidambaram Stadium	Delhi Capitals	10.0	46.000000	51.75
50	MA Chidambaram Stadium	Chennai Super Kings	56.0	46.107143	53.25
56	MA Chidambaram Stadium	Mumbai Indians	13.0	46.615385	53.00
71	MA Chidambaram Stadium	Sunrisers Hyderabad	8.0	48.500000	54.50
75	MA Chidambaram Stadium	Kolkata Knight Riders	12.0	49.166667	58.00
84	MA Chidambaram Stadium	Other	8.0	49.875000	54.00
87	MA Chidambaram Stadium	Rajasthan Royals	7.0	51.000000	58.50

Out[54]: '🏟 Narendra Modi Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
0	Narendra Modi Stadium	Gujarat Titans	1.0	31.000000	31.00
7	Narendra Modi Stadium	Royal Challengers Bangalore	3.0	39.333333	41.00
18	Narendra Modi Stadium	Punjab Kings	3.0	41.666667	44.00
26	Narendra Modi Stadium	Kolkata Knight Riders	2.0	43.500000	44.25
95	Narendra Modi Stadium	Rajasthan Royals	2.0	55.500000	61.25
96	Narendra Modi Stadium	Delhi Capitals	3.0	57.666667	65.00

Out[54]: '🏟 Punjab Cricket Association IS Bindra Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
22	Punjab Cricket Association IS Bindra Stadium	Delhi Capitals	7.0	42.571429	47.00
29	Punjab Cricket Association IS Bindra Stadium	Chennai Super Kings	6.0	44.000000	51.50
47	Punjab Cricket Association IS Bindra Stadium	Sunrisers Hyderabad	6.0	45.833333	55.75
51	Punjab Cricket Association IS Bindra Stadium	Other	8.0	46.250000	51.00
60	Punjab Cricket Association IS Bindra Stadium	Mumbai Indians	8.0	47.125000	53.75
64	Punjab Cricket Association IS Bindra Stadium	Rajasthan Royals	7.0	48.000000	53.50
77	Punjab Cricket Association IS Bindra Stadium	Punjab Kings	56.0	49.321429	55.00
91	Punjab Cricket Association IS Bindra Stadium	Kolkata Knight Riders	7.0	52.428571	61.50
92	Punjab Cricket Association IS Bindra Stadium	Royal Challengers Bangalore	7.0	53.571429	59.50

Out[54]: '🏟 Rajiv Gandhi International Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
12	Rajiv Gandhi International Stadium	Other	25.0	40.520000	48.00
21	Rajiv Gandhi International Stadium	Mumbai Indians	11.0	42.545455	49.50
25	Rajiv Gandhi International Stadium	Rajasthan Royals	6.0	43.500000	46.25
33	Rajiv Gandhi International Stadium	Kolkata Knight Riders	8.0	44.250000	48.25
34	Rajiv Gandhi International Stadium	Royal Challengers Bangalore	10.0	44.300000	51.75
35	Rajiv Gandhi International Stadium	Chennai Super Kings	6.0	44.833333	52.25
65	Rajiv Gandhi International Stadium	Delhi Capitals	8.0	48.000000	57.00
73	Rajiv Gandhi International Stadium	Sunrisers Hyderabad	44.0	48.772727	59.00
82	Rajiv Gandhi International Stadium	Punjab Kings	10.0	49.700000	51.00

Out[54]: '🏟 Sawai Mansingh Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
4	Sawai Mansingh Stadium	Mumbai Indians	7.0	38.000000	45.00
9	Sawai Mansingh Stadium	Sunrisers Hyderabad	3.0	39.666667	45.00
15	Sawai Mansingh Stadium	Chennai Super Kings	6.0	40.833333	50.75
19	Sawai Mansingh Stadium	Punjab Kings	6.0	41.666667	50.25
38	Sawai Mansingh Stadium	Rajasthan Royals	47.0	45.361702	54.50
59	Sawai Mansingh Stadium	Other	6.0	47.000000	55.25
74	Sawai Mansingh Stadium	Royal Challengers Bangalore	7.0	49.000000	57.00
80	Sawai Mansingh Stadium	Kolkata Knight Riders	6.0	49.666667	59.75
83	Sawai Mansingh Stadium	Delhi Capitals	6.0	49.833333	59.75

Out[54]: '🏟 Wankhede Stadium 👇'

Out[54]:

	venue	batting_team	count	mean	75%
5	Wankhede Stadium	Lucknow Super Giants	4.0	38.000000	38.25
13	Wankhede Stadium	Kolkata Knight Riders	15.0	40.733333	49.50
14	Wankhede Stadium	Other	11.0	40.818182	47.00
27	Wankhede Stadium	Chennai Super Kings	23.0	43.652174	49.00
40	Wankhede Stadium	Delhi Capitals	17.0	45.588235	55.00
46	Wankhede Stadium	Rajasthan Royals	18.0	45.722222	54.25
48	Wankhede Stadium	Mumbai Indians	71.0	45.985915	53.00
66	Wankhede Stadium	Sunrisers Hyderabad	11.0	48.090909	52.50
67	Wankhede Stadium	Royal Challengers Bangalore	16.0	48.250000	55.00
70	Wankhede Stadium	Gujarat Titans	4.0	48.500000	54.50
76	Wankhede Stadium	Punjab Kings	18.0	49.222222	58.50

💗 Data preprocessing

• Encoding of categorical inputs and feature scaling

In [55]:
```
data
```

Out[55]:

	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 [56]:
```
data.nunique()
```

Out[56]:
```
venue            11
innings           2
batting_team     11
bowling_team     11
count_batsmen     7
count_bowlers     5
total_runs       75
dtype: int64
```

In [57]:
```
pd.get_dummies(data)
```

Out[57]:

	innings	count_batsmen	count_bowlers	total_runs	venue_Arun Jaitley Stadium	venue_Eden Gardens	venue_Prade A
0	0	3	3	61	0	0	
1	1	6	3	26	0	0	
2	0	3	3	53	0	0	
3	1	2	2	63	0	0	
4	0	4	3	40	1	0	
...	...	...	...	...	...	...	
1893	1	4	3	62	0	1	
1894	0	3	2	46	0	0	
1895	1	3	4	67	0	0	
1896	0	3	4	44	0	0	
1897	1	4	3	31	0	0	

1895 rows × 37 columns

In [58]:
```
X = data.iloc[:, :-1]
y = data["total_runs"]
```

Normalization scales the data to a range of 0 to 1, while standardization scales the data to have a mean of 0 and a standard deviation of 1.

In [59]:
```
preprocessor = ColumnTransformer([
    ("onehot", OneHotEncoder(sparse_output=False), ["venue", "batting_team", "bowli
    ("scaler", StandardScaler(), ["count_batsmen", "count_bowlers"])
], remainder='passthrough')
```

In [60]:
```
X_preprocessed = preprocessor.fit_transform(X)
```

In [61]:
```
X_preprocessed.shape
```

Out[61]:
```
(1895, 36)
```

In [62]:
```
X_preprocessed[0]
```

Out[62]:
```
array([ 0.        ,  0.        ,  0.        ,  1.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  1.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  1.        ,  0.        , -0.31740491, -0.80500065,
        0.        ])
```

- ## Train-test split

In [63]:
```
X_train, X_test, y_train, y_test = train_test_split(X_preprocessed, y, test_size =
```

In [64]:
```
y_test.shape
```

Out[64]:
```
(379,)
```

- ## ❤ Evaluate

In [65]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

In [66]:
```
def evaluate(regressor, X_test, y_test):
    y_pred = np.round(
        regressor.predict(X_test)
    ).astype(int)

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

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

    # Calculate the R-squared score
    r2 = r2_score(y_test, y_pred)

    # Calculate total absolute error
    total_absolute_error = np.abs(y_test - y_pred).sum()

    results_df = pd.DataFrame()
    results_df.Actual = y_test
    results_df.Predicted = y_pred

    return results_df, mae, rmse, r2, total_absolute_error
```

In [67]:
```
param_grids = {
    'Linear Regression': {
        'fit_intercept': [True, False]
    },
```

```
    'Ridge Regression': {
        'alpha': [0.1, 1.0, 10.0],
        'fit_intercept': [True, False],
        'solver': ['auto', 'svd', 'cholesky', 'lsqr', 'sparse_cg', 'sag', 'saga']
    },
    'Lasso Regression': {
        'alpha': [0.1, 1.0, 10.0],
        'fit_intercept': [True, False],
        'selection': ['cyclic', 'random']
    },
    'Elastic Net Regression': {
        'alpha': [0.1, 1.0, 10.0],
        'l1_ratio': [0.1, 0.5, 0.9],
        'fit_intercept': [True, False],
        'selection': ['cyclic', 'random']
    },
    'Support Vector Regression': {
        'kernel': ['linear', 'poly', 'rbf', 'sigmoid'],
        'C': [0.1, 1.0, 10.0],
        'epsilon': [0.01, 0.1, 1.0],
        'gamma': ['scale', 'auto']
    },
    'Decision Tree Regression': {
        'criterion': ['absolute_error', 'friedman_mse', 'poisson', 'squared_error']
        'splitter': ['best', 'random'],
        'max_depth': [None, 3, 5, 10],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 4]
    },
    'Random Forest Regression': {
        'n_estimators': [50, 100, 200],
        'criterion': ['poisson', 'absolute_error', 'squared_error', 'friedman_mse']
        'max_depth': [None, 3, 5, 10],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 4],
        'max_features': ['auto', 'sqrt', 'log2']
    },
    'Gradient Boosting Regression': {
        'n_estimators': [50, 100, 200],
        'learning_rate': [0.01, 0.1, 0.2],
        'max_depth': [3, 4, 5],
        'subsample': [0.8, 0.9, 1.0],
        'max_features': ['auto', 'sqrt', 'log2']
    },
    'Neural Network Regression': {
        'hidden_layer_sizes': [(50,), (100,), (50, 50), (100, 50)],
        'activation': ['identity', 'logistic', 'tanh', 'relu'],
        'solver': ['lbfgs', 'sgd', 'adam'],
        'alpha': [0.0001, 0.001, 0.01],
        'learning_rate': ['constant', 'invscaling', 'adaptive']
    }
}
```

In [68]:

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
from sklearn.svm import SVR
```

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.neural_network import MLPRegressor
```

In [69]:
```
# X = X_preprocessed

# # Define the regression models
# models = [
#     ('Linear Regression', LinearRegression(), param_grids['Linear Regression']),
#     ('Ridge Regression', Ridge(), param_grids['Ridge Regression']),
#     ('Lasso Regression', Lasso(), param_grids['Lasso Regression']),
#     ('Elastic Net Regression', ElasticNet(), param_grids['Elastic Net Regression'
#     ('Support Vector Regression', SVR(), param_grids['Support Vector Regression']
#     ('Decision Tree Regression', DecisionTreeRegressor(), param_grids['Decision T
#     ('Random Forest Regression', RandomForestRegressor(), param_grids['Random For
#     ('Gradient Boosting Regression', GradientBoostingRegressor(), param_grids['Gr
#     ('Neural Network Regression', MLPRegressor(), param_grids['Neural Network Reg
# ]

# # Perform grid search for each model and print the best parameters and mean squar
# best_model = None
# best_mse = np.inf
# for name, model, param_grid in models:
#     grid_search = GridSearchCV(estimator=model, param_grid=param_grid, scoring='n
#     grid_search.fit(X, y)
#     mse = -grid_search.best_score_
#     if mse < best_mse:
#         best_model = name
#         best_params = grid_search.best_params_
#         best_mse = mse
#     print(f"{name}: Best parameters = {grid_search.best_params_}, Best mean squar

# # Print the best model and its corresponding best parameters and mean squared err
# print(f"\nBest model: {best_model}")
# print(f"Best parameters: {best_params}")
# print(f"Best mean squared error: {best_mse:.2f}")
```

In [70]:
```
# Create a list of regression models
models0 = [
    ('Linear Regression', LinearRegression()),
    ('Ridge Regression', Ridge(alpha=1.0)),
    ('Lasso Regression', Lasso(alpha=0.1)),
    ('Elastic Net Regression', ElasticNet(alpha=0.1, l1_ratio=0.5)),
    ('Support Vector Regression', SVR(kernel='linear', C=1.0)),
    ('Decision Tree Regression', DecisionTreeRegressor(random_state=42)),
    ('Random Forest Regression', RandomForestRegressor(n_estimators=100, random_sta
    ('Gradient Boosting Regression', GradientBoostingRegressor(n_estimators=100, ra
    ('Neural Network Regression', MLPRegressor(hidden_layer_sizes=(50, 50), max_ite
]

# Train and evaluate each model
for name, model in models0:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
```

```
    # Calculate the mean absolute error (MAE)
    mae = mean_absolute_error(y_test, y_pred)

    print(f"{name}: mae = {mae:.2f}")
```

Out[70]: LinearRegression

LinearRegression()

Linear Regression: mae = 8.30

Out[70]: Ridge

Ridge()

Ridge Regression: mae = 8.22

Out[70]: Lasso

Lasso(alpha=0.1)

Lasso Regression: mae = 8.15

Out[70]: ElasticNet

ElasticNet(alpha=0.1)

Elastic Net Regression: mae = 8.16

Out[70]: SVR

SVR(kernel='linear')

Support Vector Regression: mae = 8.21

Out[70]: DecisionTreeRegressor

DecisionTreeRegressor(random_state=42)

Decision Tree Regression: mae = 11.52

Out[70]: RandomForestRegressor

RandomForestRegressor(random_state=42)

Random Forest Regression: mae = 8.66

Out[70]: GradientBoostingRegressor

GradientBoostingRegressor(random_state=42)

Gradient Boosting Regression: mae = 8.08

```
C:\Users\k26ra\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\neu
ral_network\_multilayer_perceptron.py:686: ConvergenceWarning: Stochastic Optimizer:
Maximum iterations (1000) reached and the optimization hasn't converged yet.
  warnings.warn(
```

Out[70]:
```
MLPRegressor(hidden_layer_sizes=(50, 50), max_iter=1000, random_sta
te=42)
```

```
Neural Network Regression: mae = 9.83
```

- # Models

In [71]:
```
models = {}
```

In [72]:
```
from sklearn.ensemble import AdaBoostRegressor
models['AdaBoostRegressor'] = regressor = AdaBoostRegressor(
    learning_rate=1, loss='exponential', n_estimators=100
)
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[72]:
```
AdaBoostRegressor(learning_rate=1, loss='exponential', n_estimators
=100)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[72]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 9.469656992084433,
 11.806151967275635,
 -0.041224385997210344,
 3589)
```

In [73]:
```
from sklearn.linear_model import LinearRegression
models['LinearRegression'] = regressor = LinearRegression()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[73]:
```
LinearRegression()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[73]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.313984168865435,
 10.285758784508173,
 0.20968492995380883,
 3151)
```

In [74]:
```
from sklearn.tree import DecisionTreeRegressor
models['DecisionTreeRegressor'] = regressor = DecisionTreeRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[74]:
```
▾ DecisionTreeRegressor

DecisionTreeRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[74]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 11.522427440633246,
 14.781879372516919,
 -0.6322508391085426,
 4367)
```

In [75]:
```
from sklearn.ensemble import RandomForestRegressor
models['RandomForestRegressor'] = regressor = RandomForestRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[75]:
```
▾ RandomForestRegressor

RandomForestRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[75]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.633245382585752,
 11.007073762958935,
 0.09495255539364522,
 3272)
```

In [76]:
```
from sklearn.neighbors import KNeighborsRegressor
models['KNeighborsRegressor'] = regressor = KNeighborsRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[76]:
```
▾ KNeighborsRegressor

KNeighborsRegressor()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[76]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.831134564643799,
 10.969975815522103,
 0.10104297005420015,
 3347)
```

In [77]:
```
from sklearn.svm import SVR
models['SVR'] = regressor = SVR()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[77]:
```
▾ SVR

SVR()
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[77]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 8.131926121372032,
 10.141999719549673,
 0.2316222487796913,
 3082)
```

In [78]:
```
import xgboost as xgb
models['XGBRegressor'] = regressor = xgb.XGBRegressor()
regressor.fit(X_train, y_train)
evaluate(regressor, X_test, y_test)
```

Out[78]:

▾ XGBRegressor

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_ty
pes=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_
type=None,
             interaction_constraints=None, learning_rate=None, max_
bin=None,
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[78]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 9.100263852242744,
 11.651831251093443,
 -0.014182156501153953,
 3449)
```

- # Evaluation [using IPL-2023 dataset]

In [79]:
```
import os
files = os.listdir('./FilesUsed')
all_X = []
all_y = []
for file in files:
    if 'test_file_matchid' in file:
        match_no = file[-6:-4]

        if int(match_no) < 20: continue

        X_file_name = './FilesUsed/' + file
        y_file_name = './FilesUsed/' + 'test_file_labels_matchid_' + match_no + '.c

        X = pd.read_csv(X_file_name).drop(columns=['Unnamed: 0'])
        y = pd.read_csv(y_file_name)['actual_runs']

        all_X += [X]
        all_y += [y]

        print(match_no, X_file_name, y_file_name)

X_IPL23 = pd.concat(all_X, axis=0, ignore_index=True)
y_IPL23 = pd.concat(all_y, axis=0, ignore_index=True)
```

```
20 ./FilesUsed/test_file_matchid_20.csv ./FilesUsed/test_file_labels_matchid_20.csv
21 ./FilesUsed/test_file_matchid_21.csv ./FilesUsed/test_file_labels_matchid_21.csv
22 ./FilesUsed/test_file_matchid_22.csv ./FilesUsed/test_file_labels_matchid_22.csv
23 ./FilesUsed/test_file_matchid_23.csv ./FilesUsed/test_file_labels_matchid_23.csv
24 ./FilesUsed/test_file_matchid_24.csv ./FilesUsed/test_file_labels_matchid_24.csv
25 ./FilesUsed/test_file_matchid_25.csv ./FilesUsed/test_file_labels_matchid_25.csv
26 ./FilesUsed/test_file_matchid_26.csv ./FilesUsed/test_file_labels_matchid_26.csv
27 ./FilesUsed/test_file_matchid_27.csv ./FilesUsed/test_file_labels_matchid_27.csv
28 ./FilesUsed/test_file_matchid_28.csv ./FilesUsed/test_file_labels_matchid_28.csv
29 ./FilesUsed/test_file_matchid_29.csv ./FilesUsed/test_file_labels_matchid_29.csv
30 ./FilesUsed/test_file_matchid_30.csv ./FilesUsed/test_file_labels_matchid_30.csv
31 ./FilesUsed/test_file_matchid_31.csv ./FilesUsed/test_file_labels_matchid_31.csv
32 ./FilesUsed/test_file_matchid_32.csv ./FilesUsed/test_file_labels_matchid_32.csv
33 ./FilesUsed/test_file_matchid_33.csv ./FilesUsed/test_file_labels_matchid_33.csv
```

In [80]:
```
len(all_X)
```

Out[80]: 14

In [81]:
```
X_IPL23.innings = X_IPL23.innings.replace({1: 0, 2: 1})

# get count of batsmen & bowlers for each innings
X_IPL23['count_batsmen'] = [len(x.split(",")) for x in X_IPL23['batsmen']]
X_IPL23['count_bowlers'] = [len(x.split(",")) for x in X_IPL23['bowlers']]
X_IPL23 = X_IPL23.drop(columns=['batsmen', 'bowlers'])[
    ['venue', 'innings', 'batting_team', 'bowling_team', 'count_batsmen', 'count_bo
]
```

In [82]:
```
ambiguous_venues = np.setdiff1d(X_IPL23.venue.unique(), list(venue_mapping_normal.k
ambiguous_venues_mapping = {}
for venue in ambiguous_venues:
    venue_kebab_case = to_kebab_case(venue)
```

```
    if venue_kebab_case in venue_mapping_kebab:
        ambiguous_venues_mapping[venue] = venue_mapping_kebab[venue_kebab_case]
    else:
        venue_lower = venue.lower()
        for tag in venue_mapping_tags:
            if tag in venue_lower: ambiguous_venues_mapping[venue] = venue_mapping_

venue_mapping_final = {**venue_mapping_normal, **ambiguous_venues_mapping}
np.setdiff1d(X_IPL23.venue.unique(), list(venue_mapping_final.keys()))
```

Out[82]:
```
array([], dtype=object)
```

In [83]:
```
X_IPL23.venue = X_IPL23.venue.map(venue_mapping_final).fillna('Other').replace({
    'Barsapara Cricket Stadium': 'Other',
    'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium': 'Other'
})
```

In [84]:
```
X_IPL23
```

Out[84]:

	venue	innings	batting_team	bowling_team	count_batsmen	count_bowlers
0	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Delhi Capitals	3	5
1	M Chinnaswamy Stadium	1	Delhi Capitals	Royal Challengers Bangalore	3	2
2	Other	0	Lucknow Super Giants	Punjab Kings	2	4
3	Other	1	Punjab Kings	Lucknow Super Giants	4	4
4	Wankhede Stadium	0	Kolkata Knight Riders	Mumbai Indians	4	4
5	Wankhede Stadium	1	Mumbai Indians	Kolkata Knight Riders	3	4
6	Narendra Modi Stadium	0	Gujarat Titans	Rajasthan Royals	4	4
7	Narendra Modi Stadium	1	Rajasthan Royals	Gujarat Titans	4	2
8	M Chinnaswamy Stadium	0	Chennai Super Kings	Royal Challengers Bangalore	3	3
9	M Chinnaswamy Stadium	1	Royal Challengers Bangalore	Chennai Super Kings	4	3
10	Rajiv Gandhi International Stadium	0	Mumbai Indians	Sunrisers Hyderabad	3	4
11	Rajiv Gandhi International Stadium	1	Sunrisers Hyderabad	Mumbai Indians	4	3
12	Sawai Mansingh Stadium	0	Lucknow Super Giants	Rajasthan Royals	2	3
13	Sawai Mansingh Stadium	1	Rajasthan Royals	Lucknow Super Giants	2	3
14	Punjab Cricket Association IS Bindra Stadium	0	Royal Challengers Bangalore	Punjab Kings	2	4

	venue	innings	batting_team	bowling_team	count_batsmen	count_bowlers
15	Punjab Cricket Association IS Bindra Stadium	1	Punjab Kings	Royal Challengers Bangalore	6	4
16	Arun Jaitley Stadium	0	Kolkata Knight Riders	Delhi Capitals	5	3
17	Arun Jaitley Stadium	1	Delhi Capitals	Kolkata Knight Riders	3	5
18	MA Chidambaram Stadium	0	Sunrisers Hyderabad	Chennai Super Kings	3	3
19	MA Chidambaram Stadium	1	Chennai Super Kings	Sunrisers Hyderabad	2	3
20	Other	0	Gujarat Titans	Lucknow Super Giants	3	4
21	Other	1	Lucknow Super Giants	Gujarat Titans	2	4
22	Wankhede Stadium	0	Punjab Kings	Mumbai Indians	3	5
23	Wankhede Stadium	1	Mumbai Indians	Punjab Kings	3	5
24	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Rajasthan Royals	4	3
25	M Chinnaswamy Stadium	1	Rajasthan Royals	Royal Challengers Bangalore	3	4
26	Eden Gardens	0	Chennai Super Kings	Kolkata Knight Riders	2	4
27	Eden Gardens	1	Kolkata Knight Riders	Chennai Super Kings	4	3

In [85]:
```
X_IPL23_preprocessed = preprocessor.transform(X_IPL23)
```

In [86]:
```
X_IPL23_preprocessed.shape
```

Out[86]:
```
(28, 36)
```

In [87]:
```
X_IPL23_preprocessed[0]
```

Out[87]:
```
array([ 0.        ,  0.        ,  0.        ,  1.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        1.        ,  0.        ,  0.        ,  1.        ,  0.        ,
        0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        ,  0.        ,  0.        , -0.31740491,  2.03573722,
        0.        ])
```

In [88]:
```
evaluate(models['LinearRegression'], X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[88]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 7.928571428571429,
 10.579630023236703,
 0.113857836751593,
 222.0)
```

In [89]:
```
# evaluate(models0['LinearRegression'], X_IPL23_preprocessed, y_IPL23)

# Train and evaluate each model
for name, model in models0:
    y_pred = np.round(
        model.predict(X_IPL23_preprocessed)
    ).astype(int)

    y_test = y_IPL23

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

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

    # Calculate the R-squared score
    r2 = r2_score(y_test, y_pred)

    # Calculate total absolute error
    total_absolute_error = np.abs(y_test - y_pred).sum()

    print(f"{name}: total_absolute_error = {total_absolute_error:.2f}", r2)
```

```
Linear Regression: total_absolute_error = 222.00 0.113857836751593
Ridge Regression: total_absolute_error = 213.00 0.14750522584749615
Lasso Regression: total_absolute_error = 225.00 0.09830652246357052
Elastic Net Regression: total_absolute_error = 218.00 0.1268643905197573
Support Vector Regression: total_absolute_error = 220.00 0.06579013804316003
Decision Tree Regression: total_absolute_error = 326.00 -0.6388257747886938
Random Forest Regression: total_absolute_error = 268.00 -0.12478414977733343
Gradient Boosting Regression: total_absolute_error = 239.00 0.012915669463883672
Neural Network Regression: total_absolute_error = 332.00 -0.6987690225897987
```

In [90]:
```python
class ConstantRegressor:
    def __init__(self, n):
        self.n = n

    def predict(self, X):
        return np.repeat(self.n, X.shape[0])
```

In [91]:
```python
evaluate(ConstantRegressor(40), X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[91]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 12.178571428571429,
 14.972594011345242,
 -0.7748290870166721,
 341.0)
```

In [92]:
```python
evaluate(ConstantRegressor(46), X_IPL23_preprocessed, y_IPL23)
```

```
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:19: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Actual = y_test
C:\Users\k26ra\AppData\Local\Temp\ipykernel_8592\2192238407.py:20: UserWarning: Pand
as doesn't allow columns to be created via a new attribute name - see https://panda
s.pydata.org/pandas-docs/stable/indexing.html#attribute-access
  results_df.Predicted = y_pred
```

Out[92]:
```
(Empty DataFrame
 Columns: []
 Index: [],
 9.464285714285714,
 11.893875975235563,
 -0.11997737990649004,
 265.0)
```