IPL

INDIAN PREMIER LEAGUE

PRESENTATION TEMPLATE

IPL Data Analysis

Presented By :-

Preeti Tikku

Shivam Kumar

Shashank

```
(1
  (150460, 21)
```

Overview of Objective

The primary goal of IPL (Indian Premier League) data analysis is to gain insights into various aspects of the league, such as team performance, player performance, match outcomes, trends over different seasons, and fan engagement.

- Dataset Description :-
- Two datasets (Delivery and match)
- numerical, categorical, text-based
- Provided by “GEEKSTER”
- Number of observations (150460) and variables (21) in deliveries DT
- Number of observations (636) and variables (18) in Matches DT

- **List of Libraries**

  Pandas
  NumPy
  Matplotlib
  Seaborn
  Statistics

- Purpose of Each:-

- Pandas: Data manipulation and analysis
- NumPy: Mathematical operations
- Matplotlib & Seaborn: Data visualization
- Statistics : stats purpose

Import Libraries and show top 5 rows

jupyter module4 Last Checkpoint: a day ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
# 1.	Import Libraries: Begin by importing the necessary libraries:
#       NumPy, Pandas, Statistics, Matplotlib, Seaborn, etc.

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statistics as stats
%matplotlib inline
```

In [2]:
```
# 2.	Load Data: Read the CSV file into a Pandas DataFrame.

delivery_df1=pd.read_csv("deliveries.csv")

match_df2=pd.read_csv("matches.csv")
```

In [10]:
```
print("Delivery Dataset Head")
delivery_df1.head()
```

Delivery Dataset Head

Out[10]:

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_super_over	...	bye_runs	legbye_runs	noball_runs	penalty_ru
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	0	...	0	0	0	

5 rows × 21 columns

Cleaning steps :- Matches Dataset

Finding null values
Removing null values or fill with”0” and “NA”

```
In [42]: #finding null values in this data by using null function
         match_df2.isnull().sum()

Out[42]: id                  0
         season              0
         city                0
         date                0
         team1               0
         team2               0
         toss_winner         0
         toss_decision       0
         result              0
         dl_applied          0
         winner              0
         win_by_runs         0
         win_by_wickets      0
         player_of_match     0
         venue               0
         umpire1             0
         umpire2             0
         umpire3           636
         dtype: int64
```

```
In [43]: # # Fill null values with 0 in a specific column

         match_df2['umpire3'] = match_df2['umpire3'].fillna(0)

In [45]: match_df2.info()

         <class 'pandas.core.frame.DataFrame'>
         RangeIndex: 636 entries, 0 to 635
         Data columns (total 18 columns):
          #   Column           Non-Null Count  Dtype
         ---  ------           --------------  -----
          0   id               636 non-null    int64
          1   season           636 non-null    int64
          2   city             636 non-null    object
          3   date             636 non-null    object
          4   team1            636 non-null    object
          5   team2            636 non-null    object
          6   toss_winner      636 non-null    object
          7   toss_decision    636 non-null    object
          8   result           636 non-null    object
          9   dl_applied       636 non-null    int64
          10  winner           636 non-null    object
          11  win_by_runs      636 non-null    int64
          12  win_by_wickets   636 non-null    int64
          13  player_of_match  636 non-null    object
          14  venue            636 non-null    object
          15  umpire1          636 non-null    object
          16  umpire2          636 non-null    object
          17  umpire3          636 non-null    float64
         dtypes: float64(1), int64(5), object(12)
```

Cleaning steps :- Deliveries Dataset

Finding null values
Removing null values or fill with”0” and “NA”

```
In [46]: #finding null values in this data by using null function
         delivery_df1.isnull().sum()

Out[46]: match_id                0
         inning                  0
         batting_team            0
         bowling_team            0
         over                    0
         ball                    0
         batsman                 0
         non_striker             0
         bowler                  0
         is_super_over           0
         wide_runs               0
         bye_runs                0
         legbye_runs             0
         noball_runs             0
         penalty_runs            0
         batsman_runs            0
         extra_runs              0
         total_runs              0
         player_dismissed    143022
         dismissal_kind      143022
         fielder             145091
         dtype: int64
```

```
delivery_df1.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150460 entries, 0 to 150459
Data columns (total 21 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   match_id          150460 non-null  int64
 1   inning            150460 non-null  int64
 2   batting_team      150460 non-null  object
 3   bowling_team      150460 non-null  object
 4   over              150460 non-null  int64
 5   ball              150460 non-null  int64
 6   batsman           150460 non-null  object
 7   non_striker       150460 non-null  object
 8   bowler            150460 non-null  object
 9   is_super_over     150460 non-null  int64
 10  wide_runs         150460 non-null  int64
 11  bye_runs          150460 non-null  int64
 12  legbye_runs       150460 non-null  int64
 13  noball_runs       150460 non-null  int64
 14  penalty_runs      150460 non-null  int64
 15  batsman_runs      150460 non-null  int64
 16  extra_runs        150460 non-null  int64
 17  total_runs        150460 non-null  int64
 18  player_dismissed  150460 non-null  object
 19  dismissal_kind    150460 non-null  object
 20  fielder           150460 non-null  object
dtypes: int64(13), object(8)
memory usage: 24.1+ MB
```

Statistics Status of all Numerical columns of Delivery Datasets

```
print("Delivery")
delivery_df1.describe()
```

Delivery

	match_id	inning	over	ball	is_super_over	wide_runs	bye_runs	legbye_runs	noball_runs	penalty_runs
count	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000	150460.000000
mean	318.281317	1.482188	10.142649	3.616483	0.000538	0.037498	0.004885	0.022232	0.004340	0.000066
std	182.955531	0.501768	5.674338	1.807698	0.023196	0.257398	0.114234	0.200104	0.072652	0.018229
min	1.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	161.000000	1.000000	5.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	319.000000	1.000000	10.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	476.000000	2.000000	15.000000	5.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	636.000000	4.000000	20.000000	9.000000	1.000000	5.000000	4.000000	5.000000	5.000000	5.000000

Statistic Status of all Numerical columns of Matches Datasets

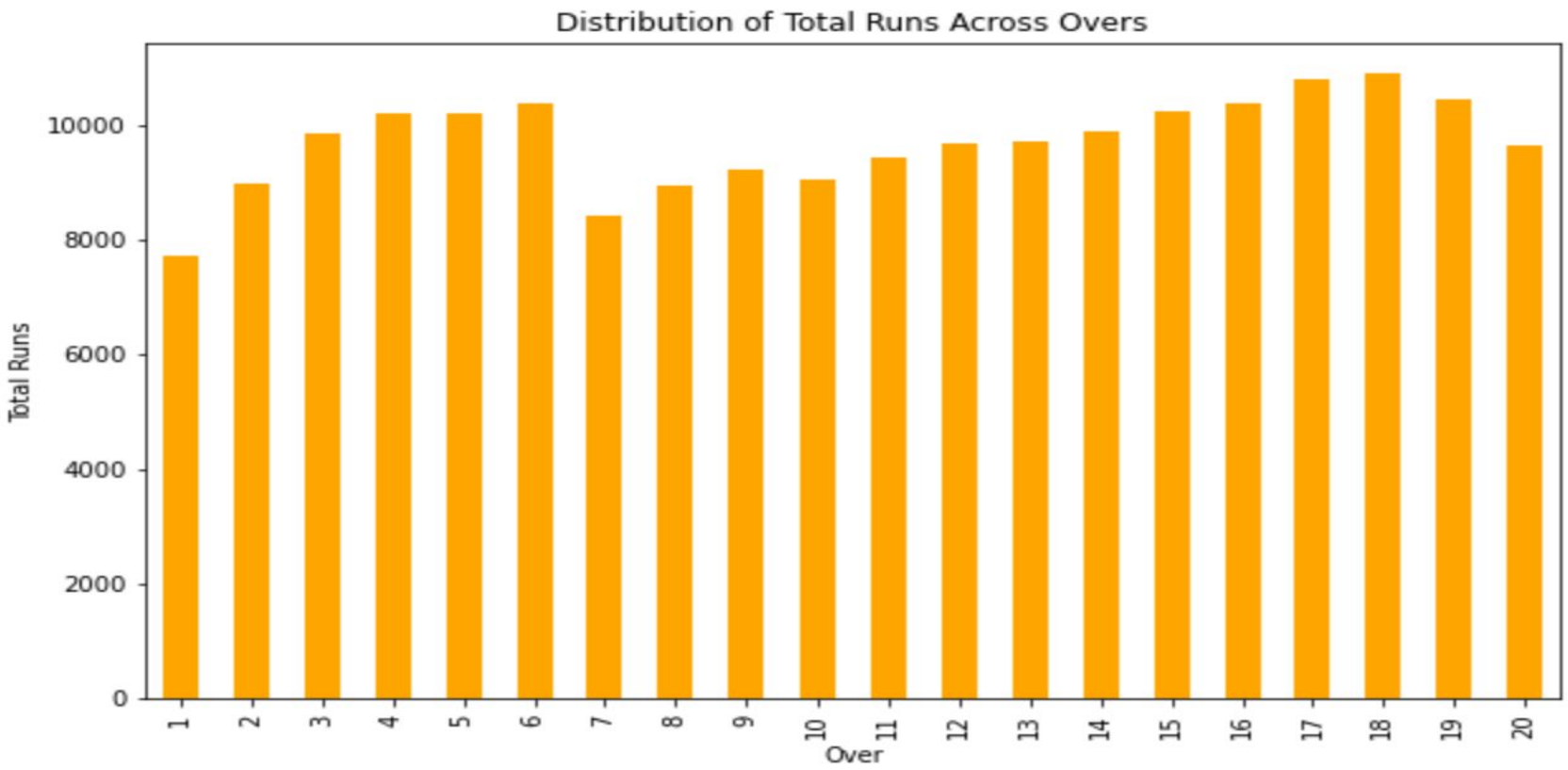
In [13]:
```
print("Matches")
match_df2.describe()
```

```
Matches
```

Out[13]:

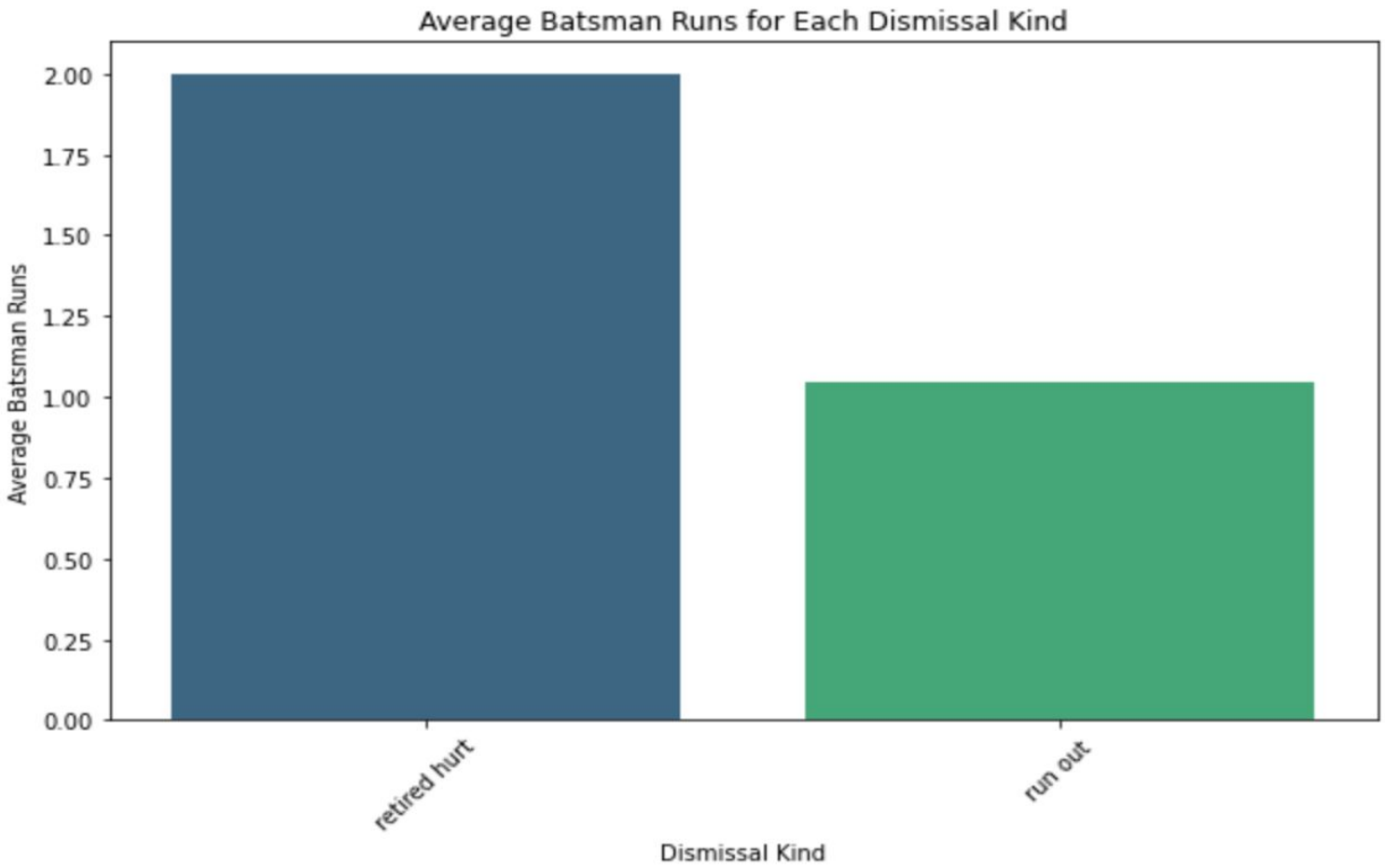
	id	season	dl_applied	win_by_runs	win_by_wickets	umpire3
count	636.000000	636.000000	636.000000	636.000000	636.000000	0.0
mean	318.500000	2012.490566	0.025157	13.682390	3.372642	NaN
std	183.741666	2.773026	0.156726	23.908877	3.420338	NaN
min	1.000000	2008.000000	0.000000	0.000000	0.000000	NaN
25%	159.750000	2010.000000	0.000000	0.000000	0.000000	NaN
50%	318.500000	2012.000000	0.000000	0.000000	4.000000	NaN
75%	477.250000	2015.000000	0.000000	20.000000	7.000000	NaN
max	636.000000	2017.000000	1.000000	146.000000	10.000000	NaN

❑ 1. Analyze the distribution of total runs across different overs.

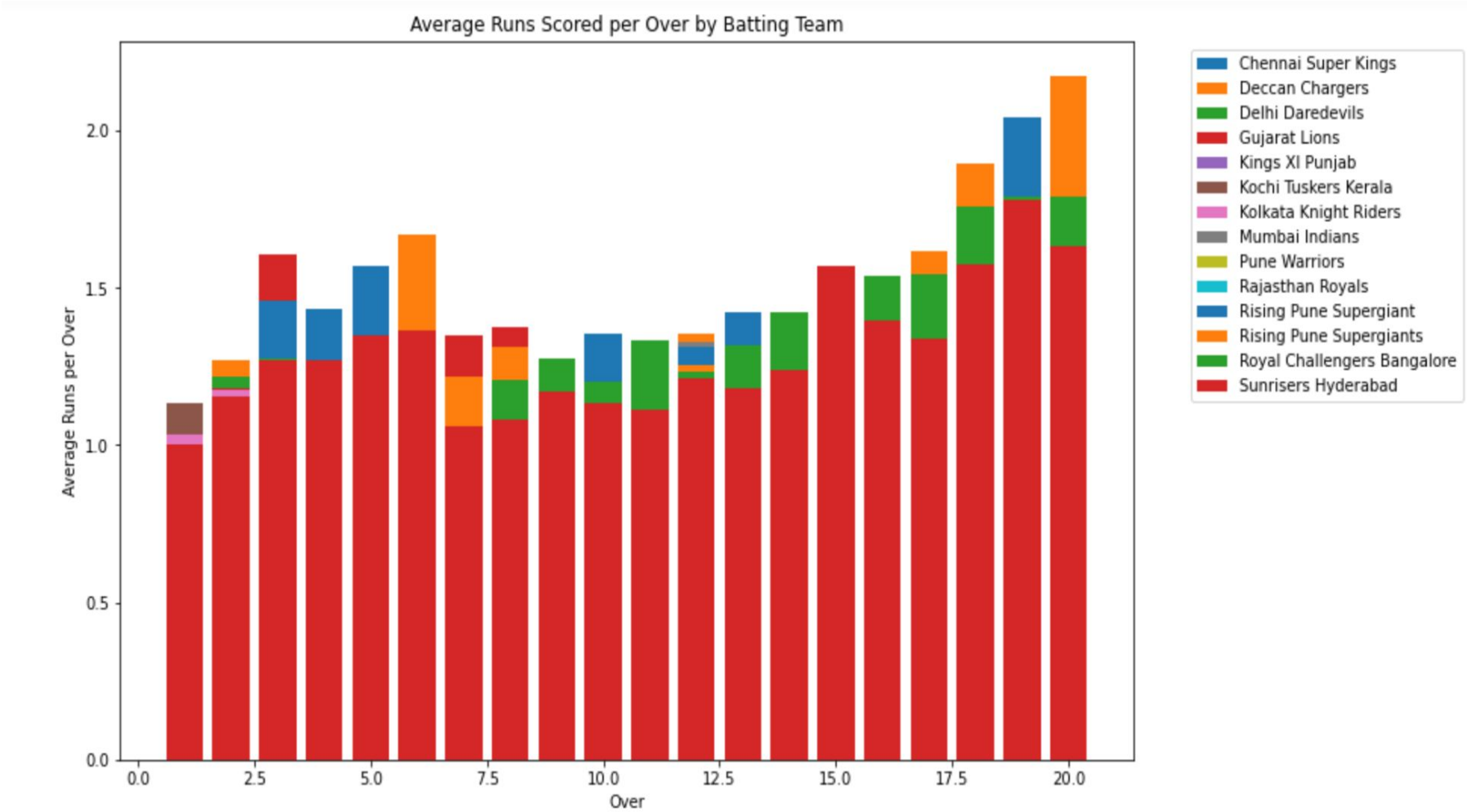



```
Average runs scored in Powerplay: 9568.166666666666
Average runs scored in Middle Overs: 9408.444444444445
Average runs scored in Death Overs: 10445.8
```

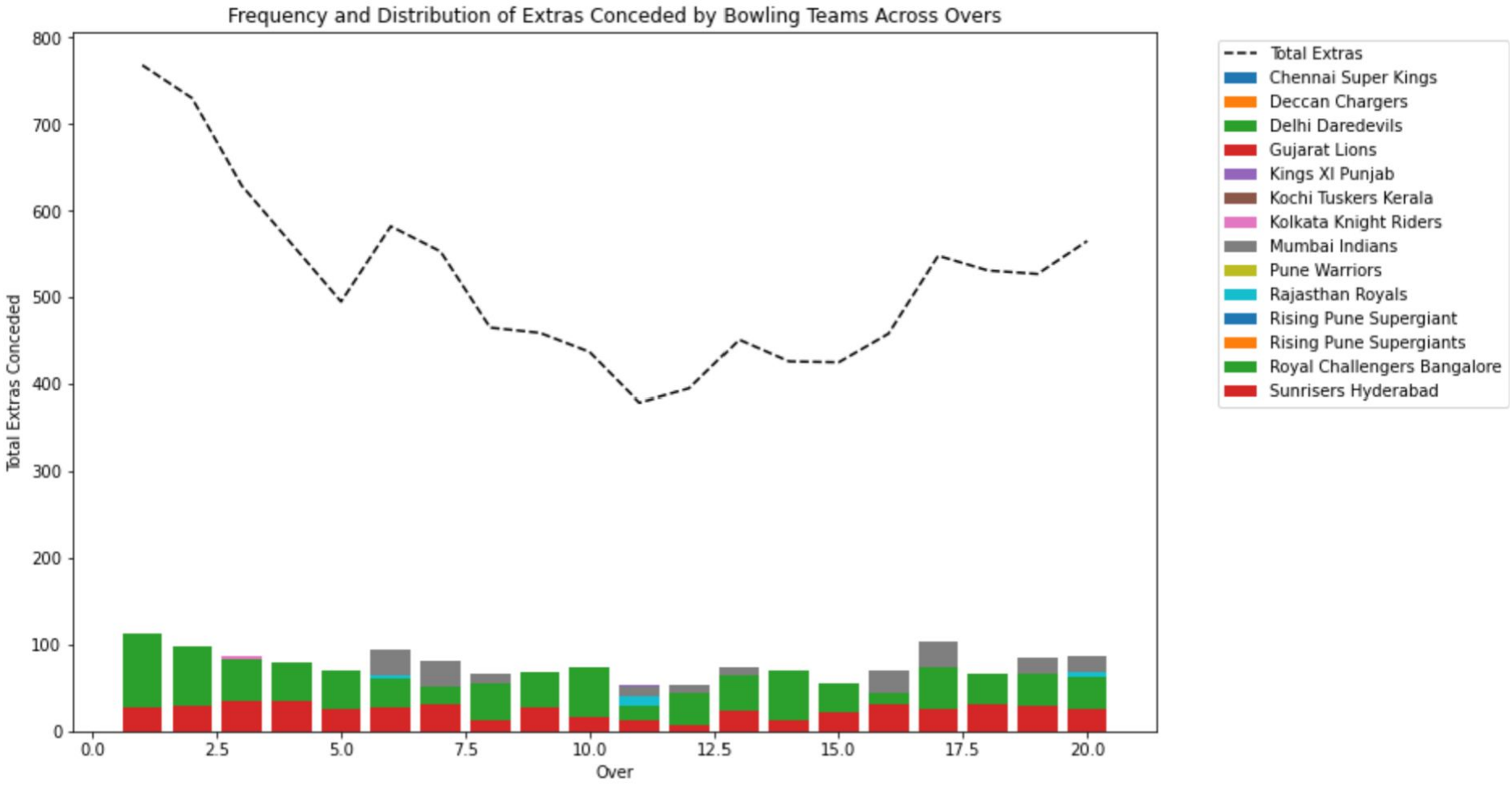
- **Filter out rows where batsman_runs are zero (to exclude non-scoring deliveries)**



❑ 3.Calculate average runs scored per over for each batting team



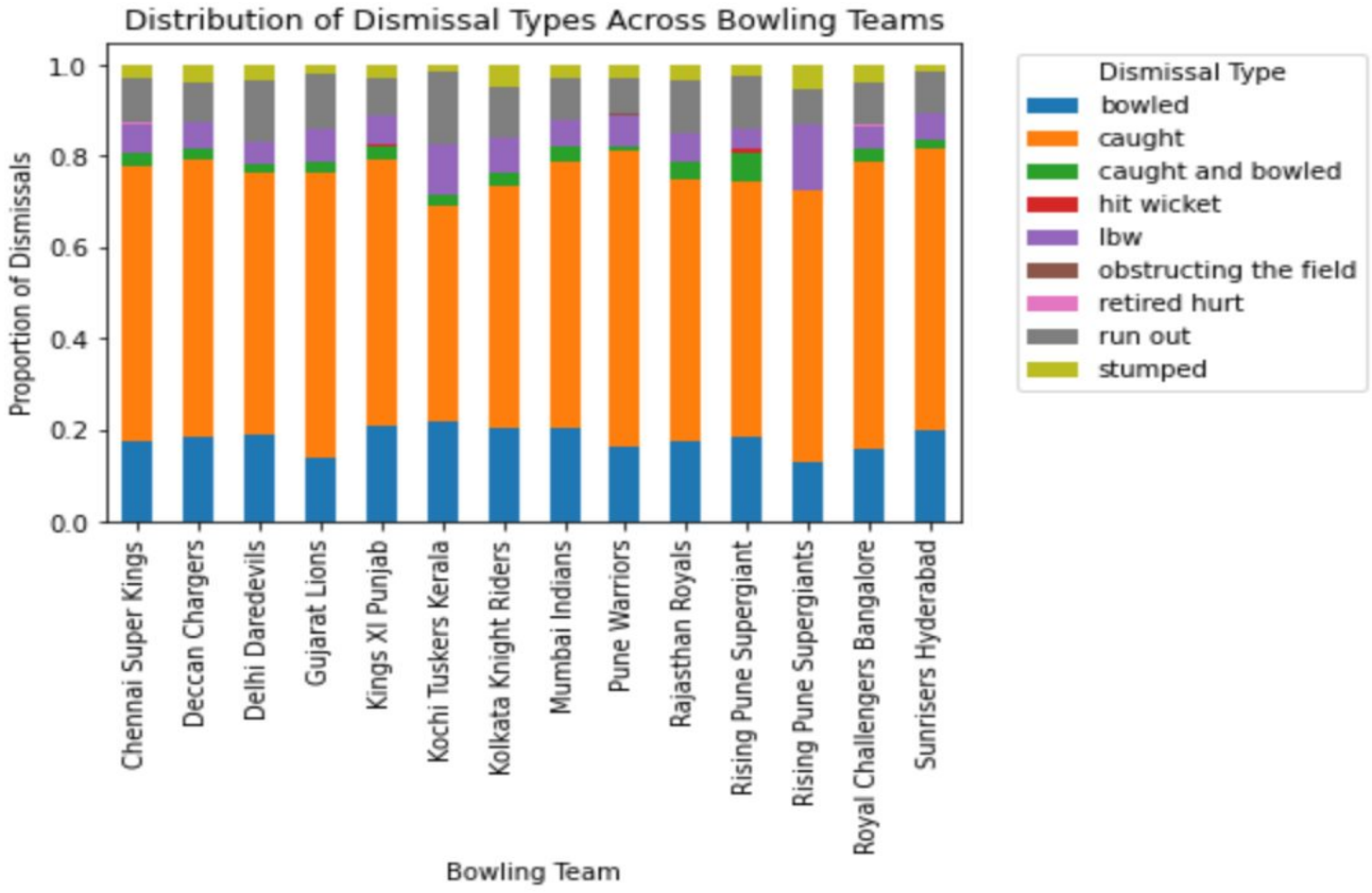
- **4. Analyze the frequency and distribution of extras (extra_runs) conceded by bowling teams across different overs. Are there specific phases of the game where teams tend to give away more extras?**



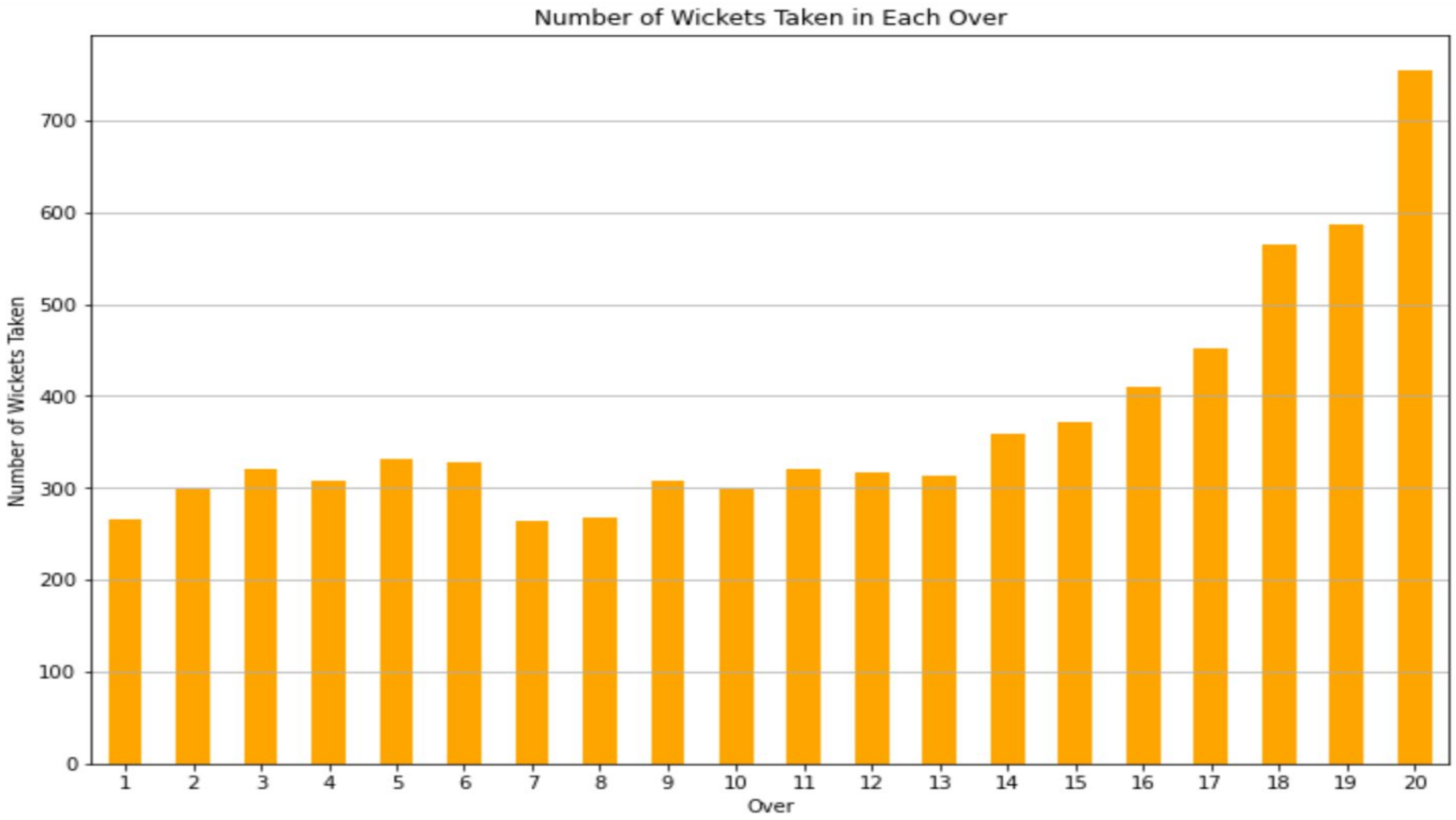
- **5.Explore the relationship between batsman-runs and the number of balls faced.**

	batsman	ball	batsman_runs	strike_rate
117	DL Chahar	6	14	233.333333
430	Umar Gul	19	39	205.263158
337	RS Sodhi	2	4	200.000000
69	BCJ Cutting	70	124	177.142857
35	AJ Tye	30	53	176.666667

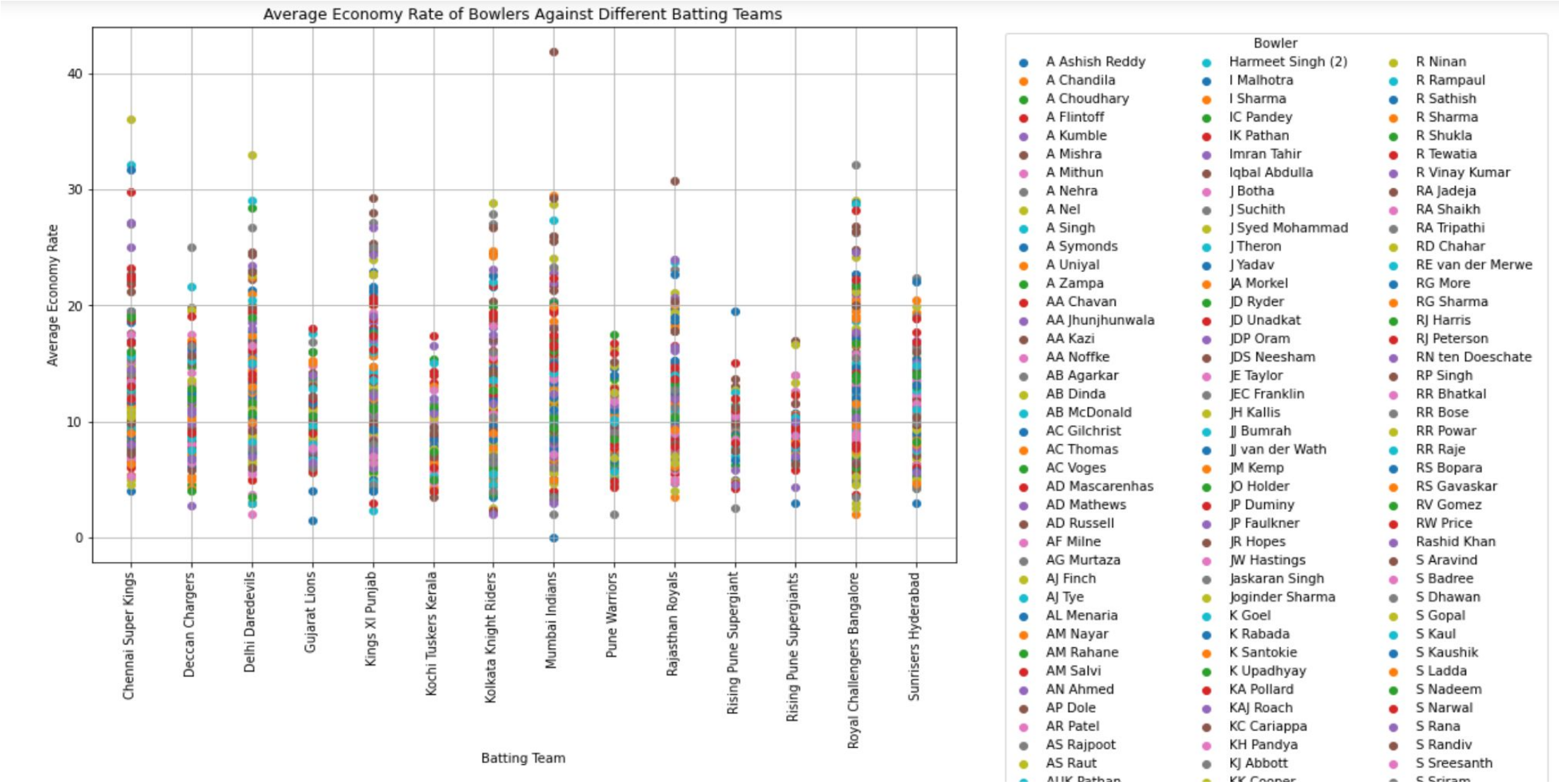
- **6.Group the data by 'bowling_team' and 'dismissal_kind', then calculate the count of each dismissal type**



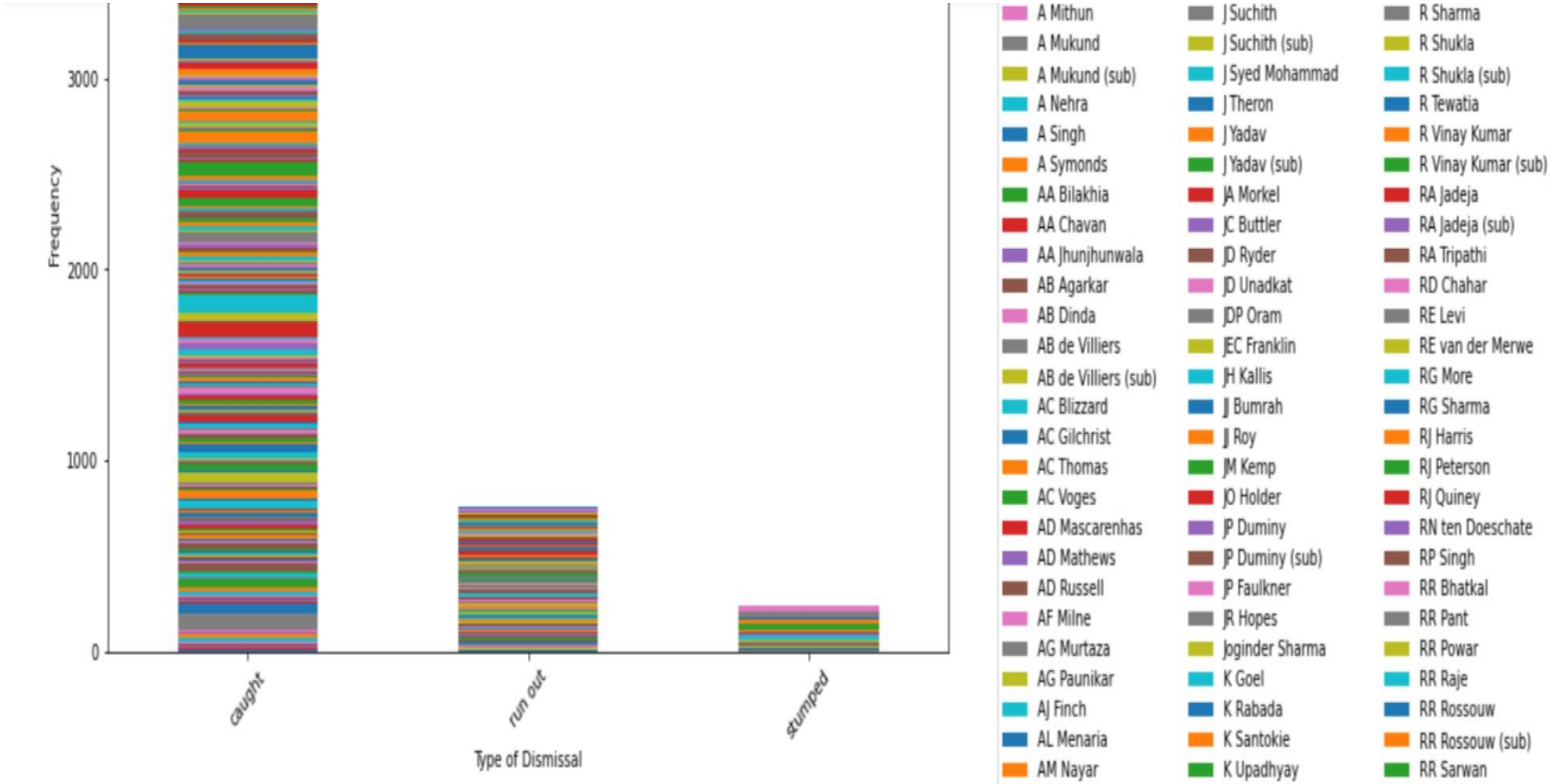
❑ 7.Group the data by 'over' and count the number of wickets taken in each over



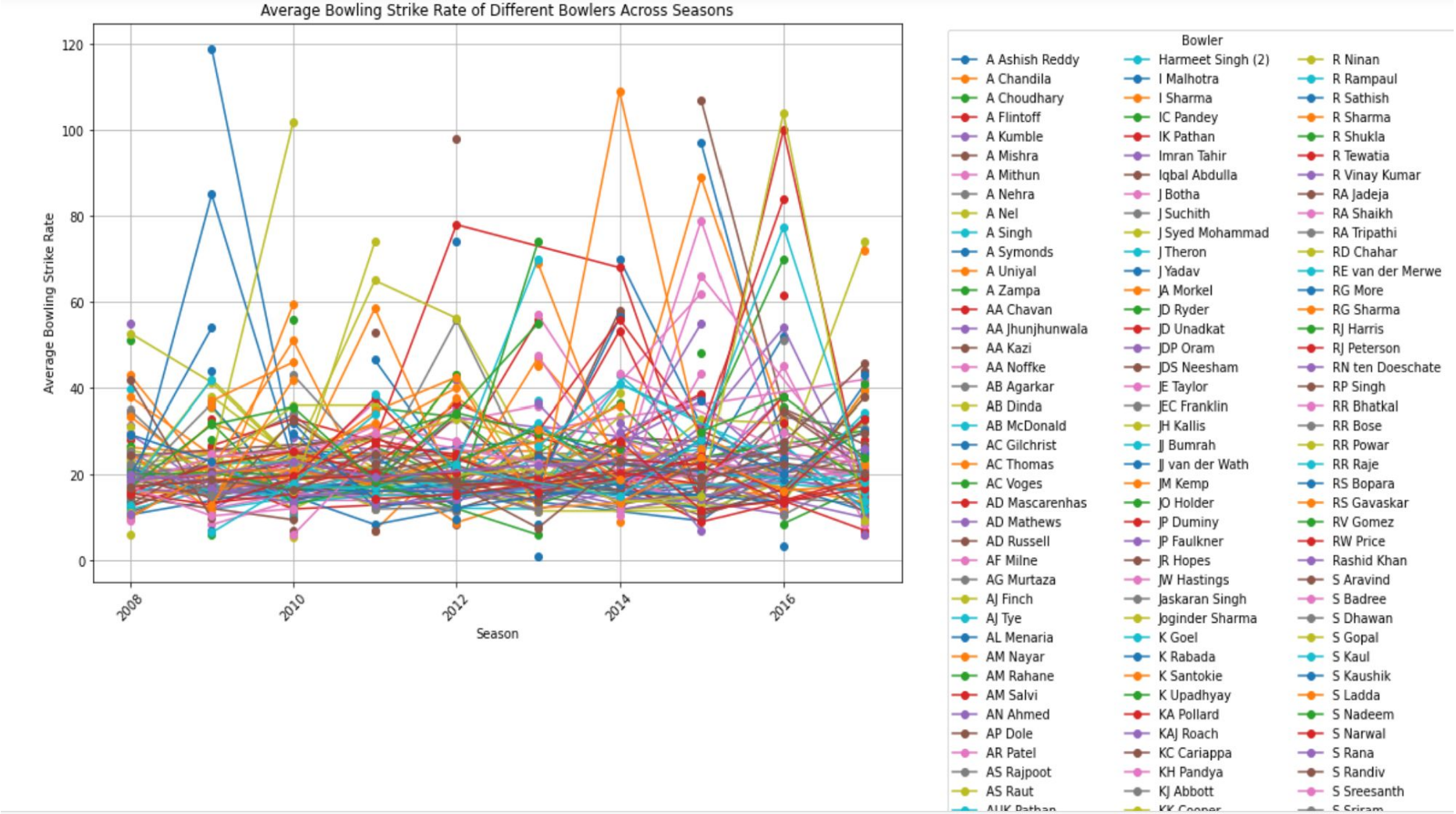
❑ 8.Group the data by 'bowler' and 'batting-team', then calculate the average runs conceded per over for each combination



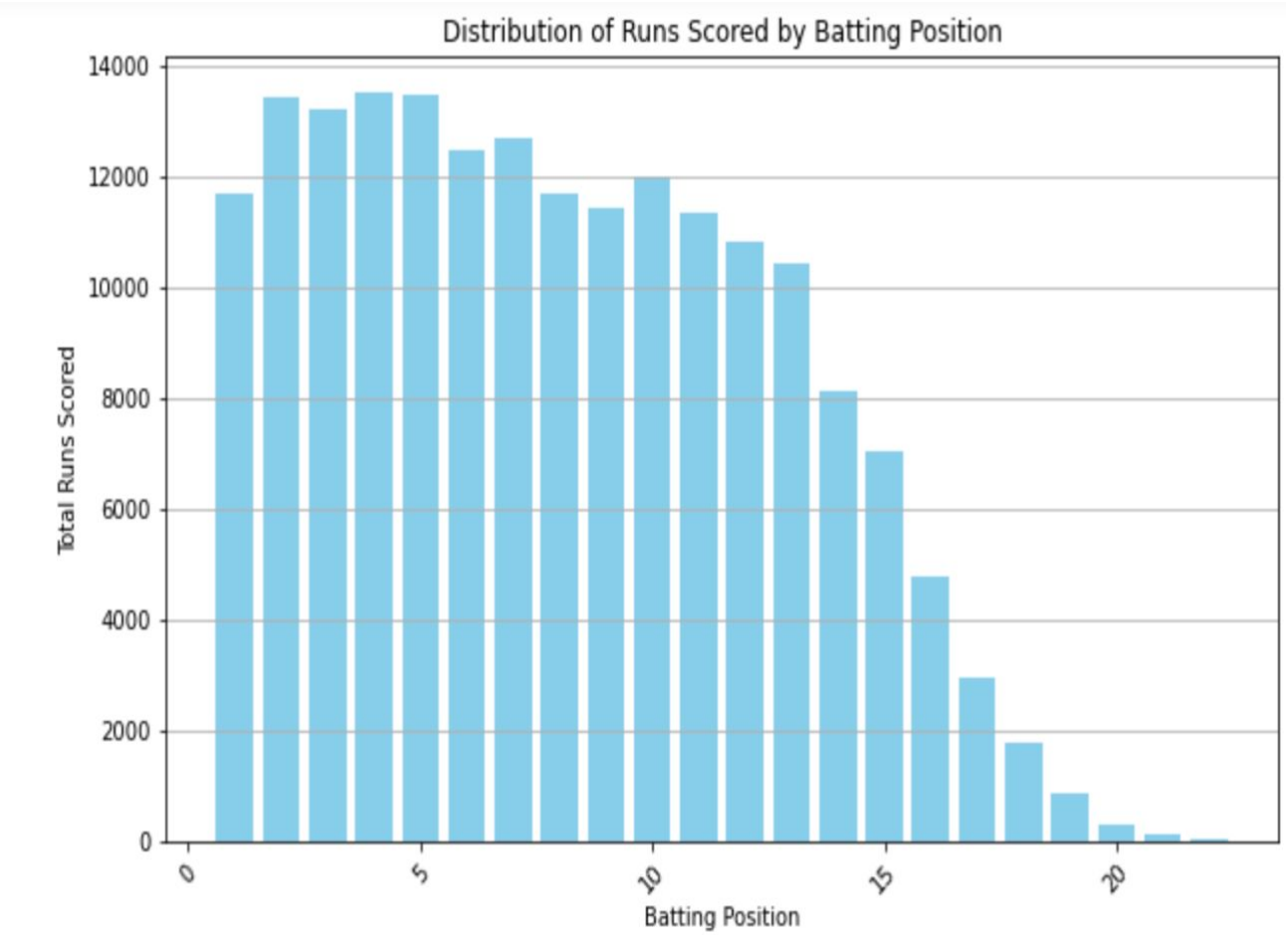
- **9.Explore the relationship between the type of dismissal_kind and the fielder involved.**



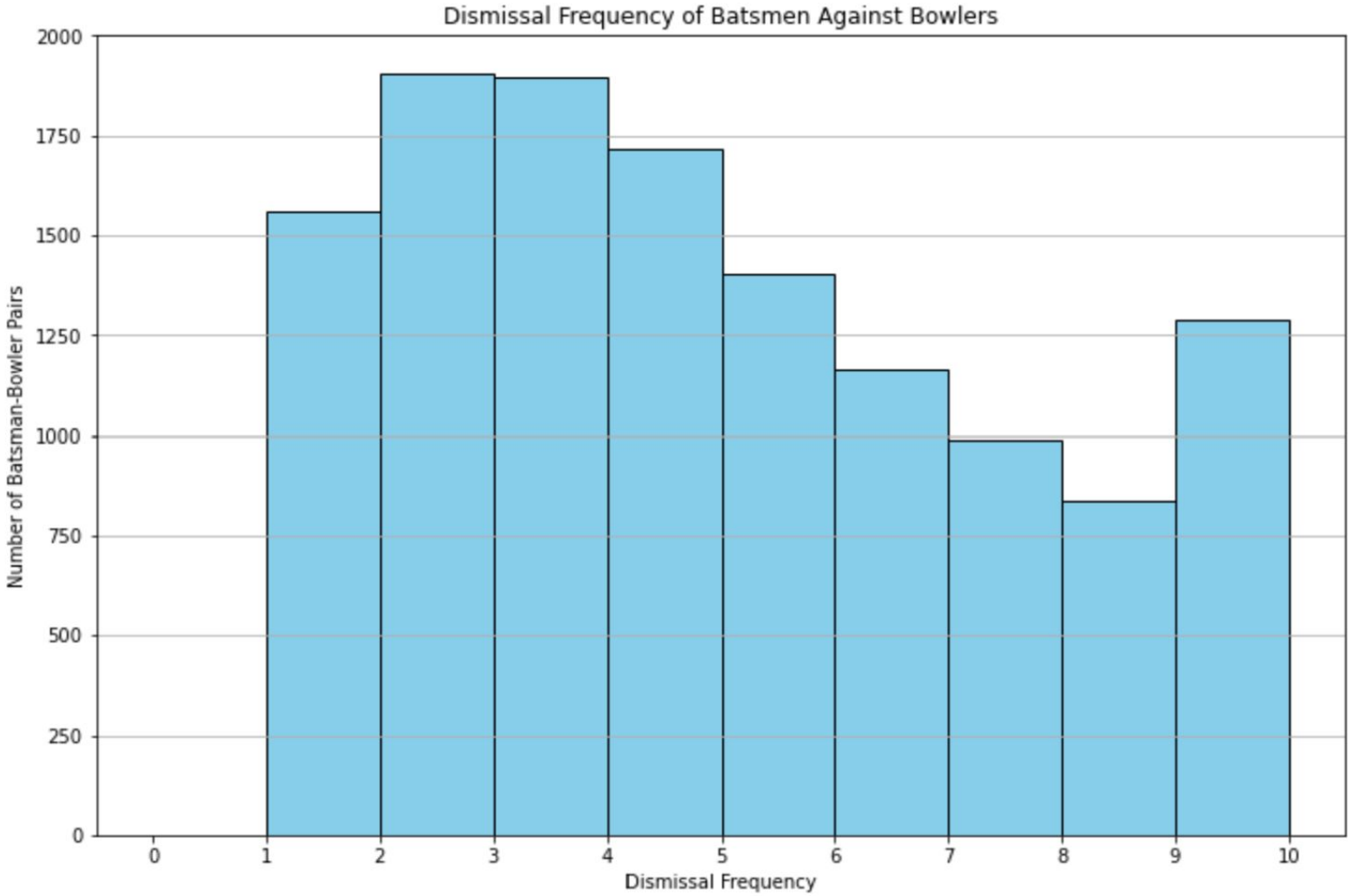
- **10.Merge datasets based on match ID**
  Compare the bowling strike rate (average balls bowled per wicket) of different bowlers across different seasons.
  Are there any noticeable trends or changes in bowling effectiveness over time?



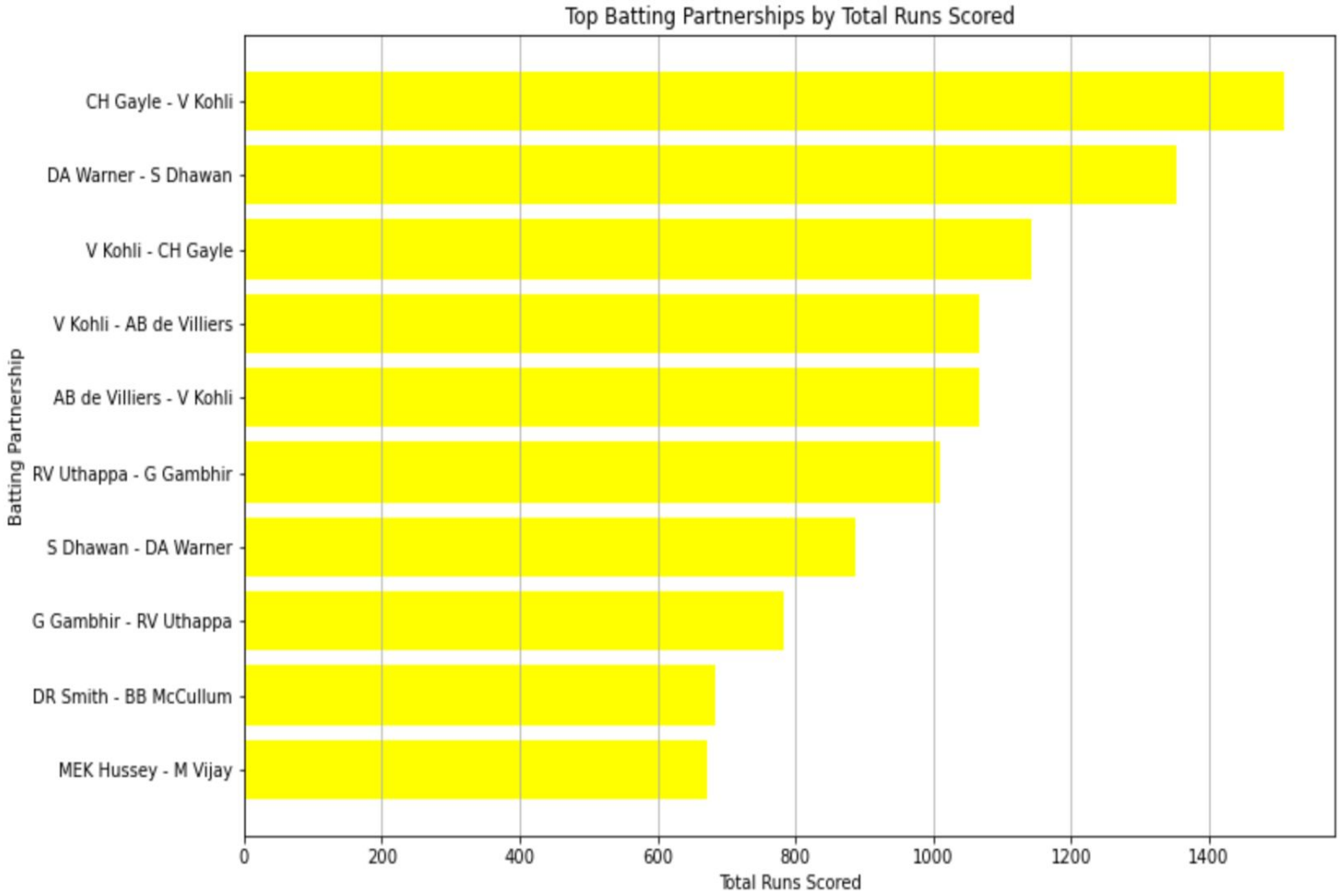
- **11. Analyze the distribution of runs scored by batsman across different batting positions (e.g., opening batsmen, middle-order batsmen). Do certain batting positions tend to score more runs?**



- **12. Explore the relationship between the batsman and the bowler in terms of dismissal frequency.**
  Do certain batsmen struggle against specific bowlers?



- **13. Analyze the runs scored by different pairs of batsman and non_striker to identify successful partnerships.**
  Are there specific partnerships that consistently contribute significantly to the team's total score?



- **14.Compare the strike rate (runs scored per 100 balls) of individual batsmen against different bowling-teams.**
  Do certain batsmen perform better against specific teams?

	batsman	bowling_team	ball	batsman_runs	strike_rate
0	A Ashish Reddy	Chennai Super Kings	25	45	180.000000
1	A Ashish Reddy	Delhi Daredevils	24	36	150.000000
2	A Ashish Reddy	Kings XI Punjab	23	37	160.869565
3	A Ashish Reddy	Kolkata Knight Riders	14	17	121.428571
4	A Ashish Reddy	Mumbai Indians	25	27	108.000000

Inferences and Conclusion

- Team batting second has more chances of winning a match. Team batting second won 53 out of 100 matches played
- Chances of a match getting tied is just 1% . This means only 1 match gets tied out of 100 matches
- Chances of a match getting no result is just 0.5%. This means only 1 out of 200 matches played will have no result
- There is 20% chance of a close match where a team batting first just win by a margin of less than 10 runs

Thanks :-

Preeti Tikku
Shivam Kumar
Shashank