Dr BR Ambedkar National Institute of Technology Jalandhar-144011, Punjab, India.

Department of Computer Science & Engineering

Probability Theory for Data Analytics

(CSPC-309)

Assignment - 1

Submitted to:
Dr. Amritpal Singh
(Assistant Professor)
Department of Computer Science & Engineering

Submitted by:
Abhinandan Gautam
20103002
Section A G1
CSE 3nd year

Title

Analysis of IPL data of different matches from year 2008 to 2019 and infer some useful information from this data using concept of probability distributions.

Dataset

Indian Premier League 2008-2019

https://www.kaggle.com/datasets/nowke9/ipldata

Subject

Distributions (discrete or continuous)

Explaining Dataset

- ## Context

  Indian Premier League (IPL) is a Twenty20 cricket format league in India. It is usually played in April and May every year. The league was founded by Board of Control for Cricket India (BCCI) in 2008. Here this dataset contains data of all IPL matches that took place between year 2008 to 2019.
  This file contains two csv files which store data of all matches from 2008 to 2019. First file is matches.csv which contains match by match data and another file is deliveries.csv which contains ball by ball data. These files contains data till season 11.

- ## Content

  1. Data Till Season 11 (2008-2019)
  2. matches.csv – Match by match data
  3. deliveries.csv – Ball by ball data

- ## Acknowledgements

  1. Data source from 2008-2017 – CrickSheet.org and Manas-Kaggle
  2. Data source from 2018-2019 – IPL T20 official website

- ## Details of matches.csv and deliveries.csv

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               756 non-null    int64
 1   season           756 non-null    int64
 2   city             749 non-null    object
 3   date             756 non-null    object
 4   team1            756 non-null    object
 5   team2            756 non-null    object
 6   toss_winner      756 non-null    object
 7   toss_decision    756 non-null    object
 8   result           756 non-null    object
 9   dl_applied       756 non-null    int64
 10  winner           752 non-null    object
 11  win_by_runs      756 non-null    int64
 12  win_by_wickets   756 non-null    int64
 13  player_of_match  752 non-null    object
 14  venue            756 non-null    object
 15  umpire1          754 non-null    object
 16  umpire2          754 non-null    object
 17  umpire3          119 non-null    object
dtypes: int64(5), object(13)
memory usage: 106.4+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 179078 entries, 0 to 179077
Data columns (total 21 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   match_id          179078 non-null  int64
 1   inning            179078 non-null  int64
 2   batting_team      179078 non-null  object
 3   bowling_team      179078 non-null  object
 4   over              179078 non-null  int64
 5   ball              179078 non-null  int64
 6   batsman           179078 non-null  object
 7   non_striker       179078 non-null  object
 8   bowler            179078 non-null  object
 9   is_super_over     179078 non-null  int64
 10  wide_runs         179078 non-null  int64
 11  bye_runs          179078 non-null  int64
 12  legbye_runs       179078 non-null  int64
 13  noball_runs       179078 non-null  int64
 14  penalty_runs      179078 non-null  int64
 15  batsman_runs      179078 non-null  int64
 16  extra_runs        179078 non-null  int64
 17  total_runs        179078 non-null  int64
 18  player_dismissed  8834 non-null    object
 19  dismissal_kind    8834 non-null    object
 20  fielder           6448 non-null    object
dtypes: int64(13), object(8)
memory usage: 28.7+ MB
```

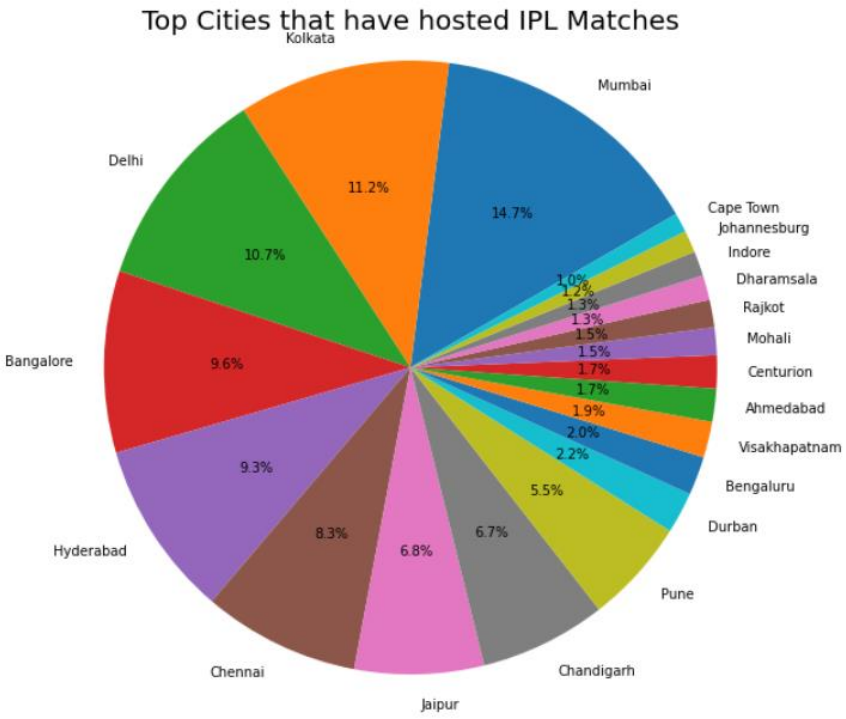
Significance Of Problem

Using the data of previous matches of IPL we can make analysis about different stats of venue, players and other things. Using these stats we can predict the probability of some event in different situations.

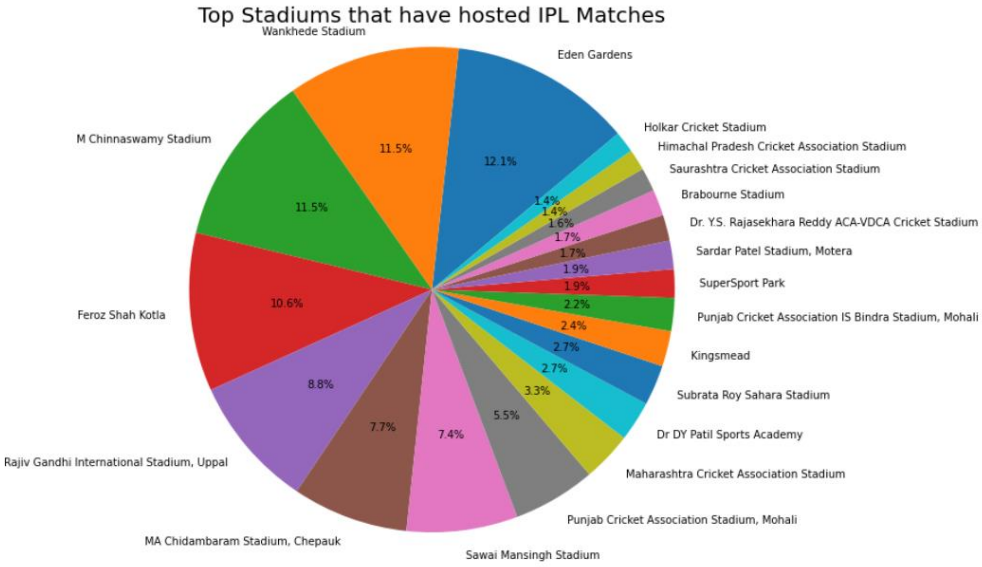
1. batting performance moving average
2. score forecasting
3. gaining insights into fitness
4. performance of player against different opposition
5. player contribution to wins and losses for making strategic decisions on team composition
6. field mapping, player tracking, ball tracking, player shot analysis
7. other aspects involved in how the ball is delivered, its angle, spin, velocity, and trajectory.

Analysis

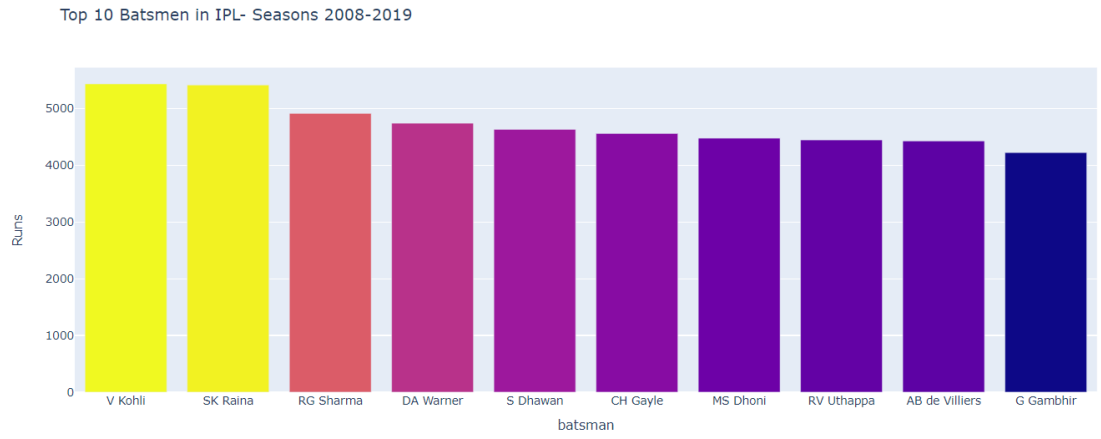
- List of the Top 20 Cities where the most number of matches have been played.



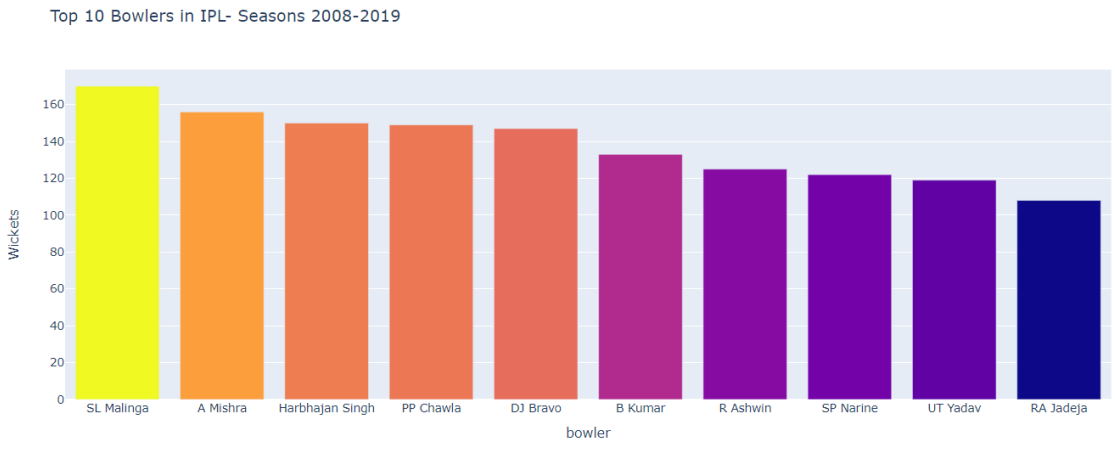
- List of the Top 20 venues where the most number of IPL matches have been played.



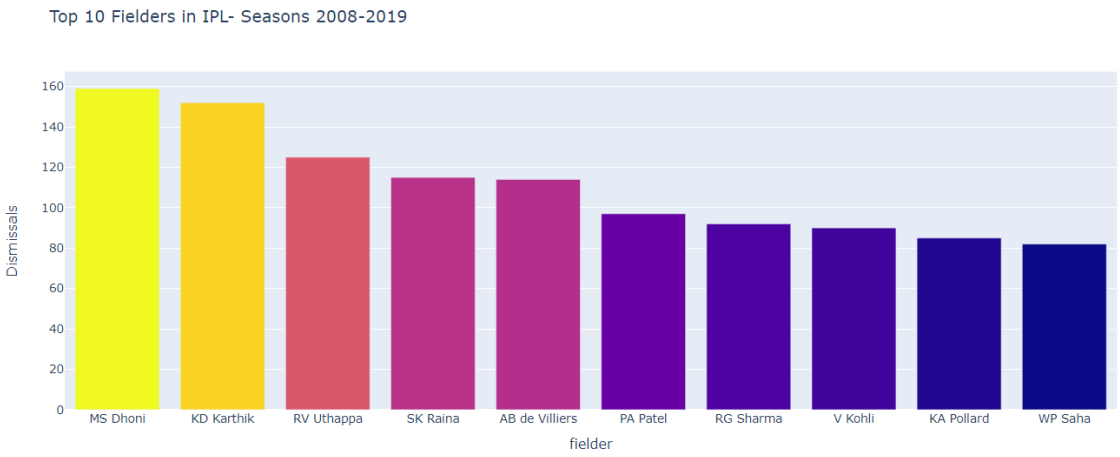
- List of Top 10 Scoring Batsman



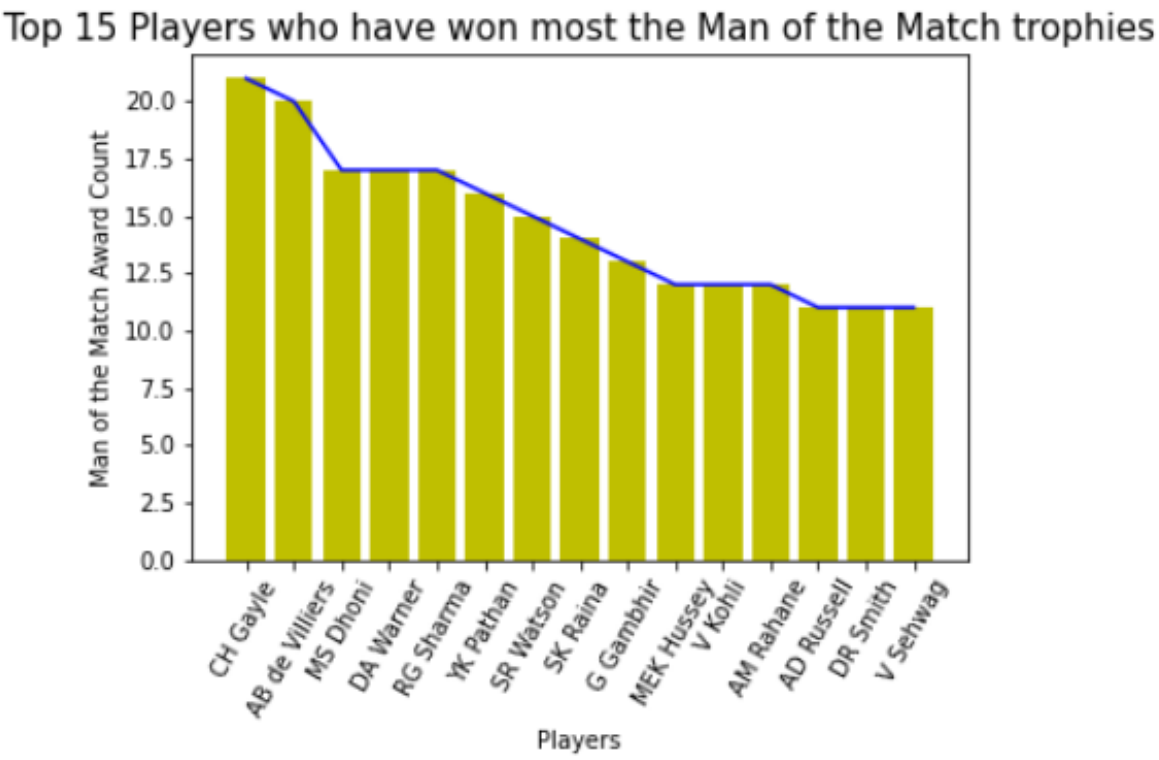
- List of the top 10 Bowlers with highest number of wickets



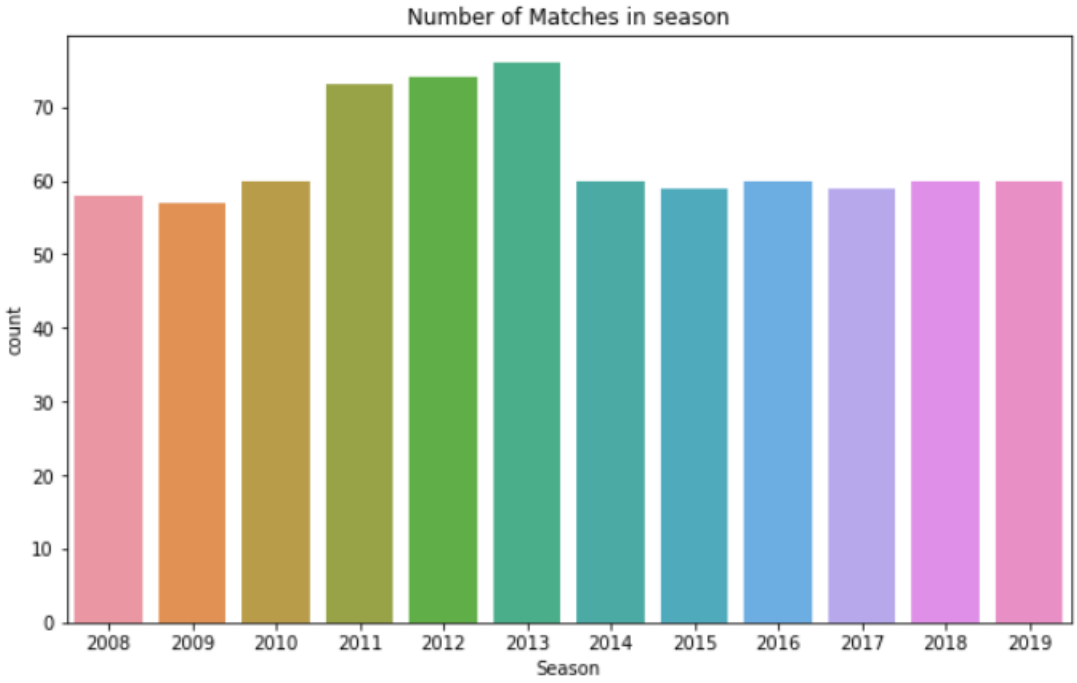
- List of the top 10 fielders



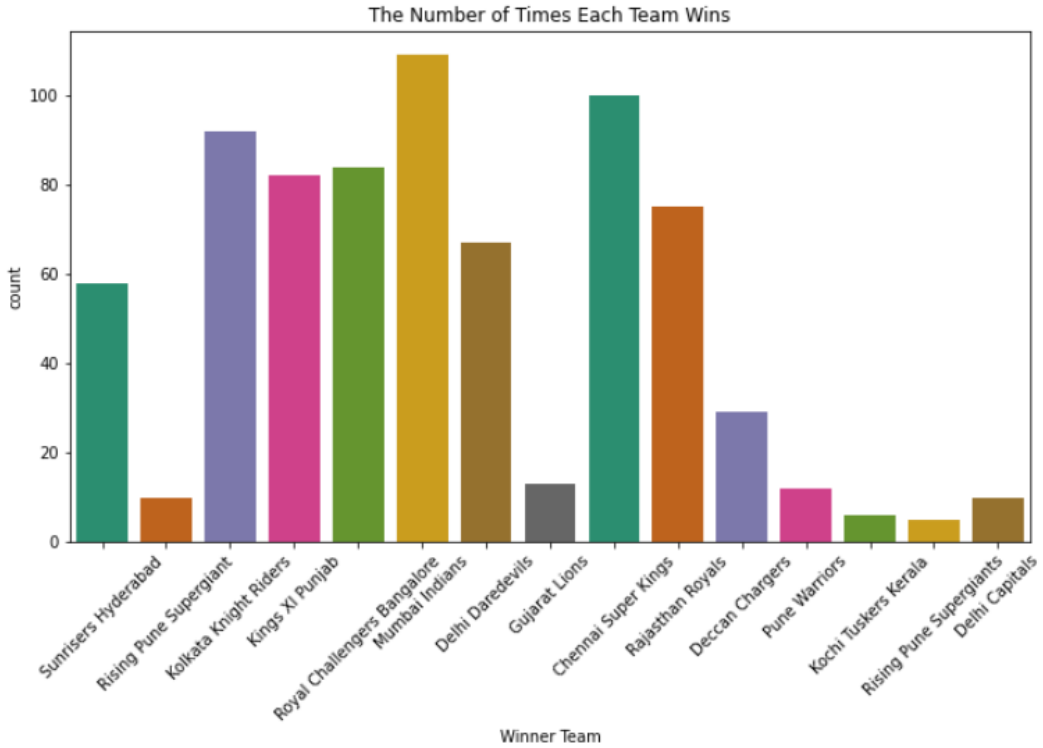
- List of the Players who have achieved highest number of 'Man of the Match Awards



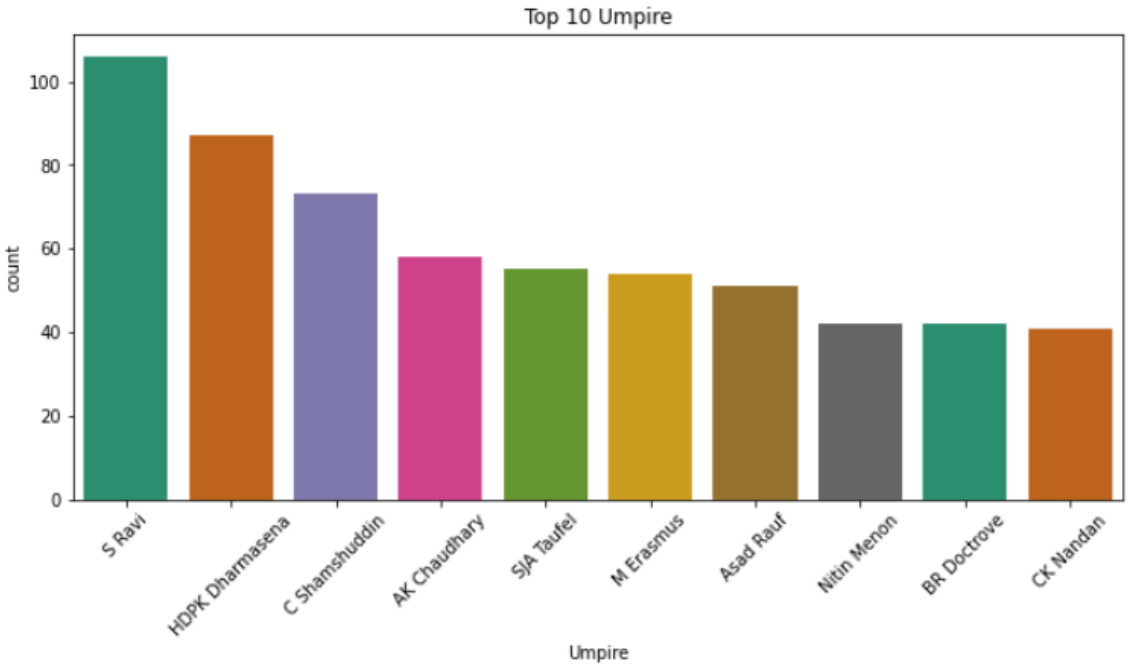
- Number of matches in each season



- Number of times each team win



- List of top ten umpires

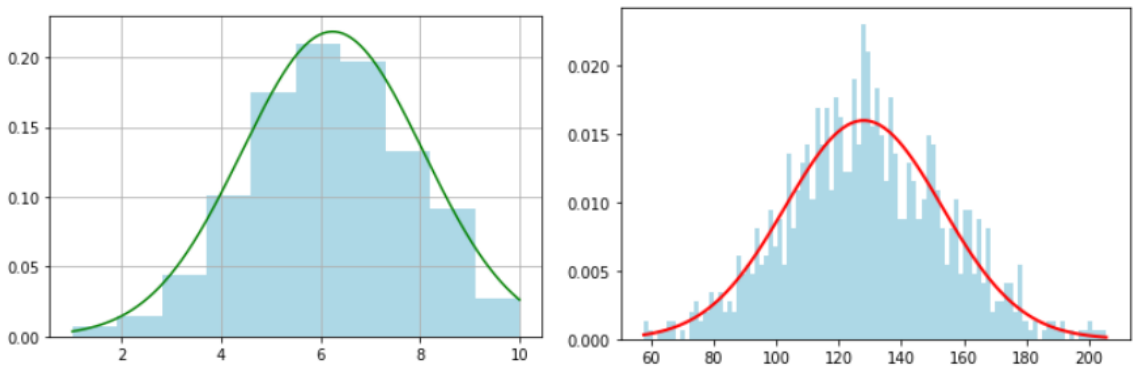


- <u>Normal Distribution</u>

  Normal distribution is a continuous probability distribution that describes many natural datasets. It is also known as bell curve or Gaussian distribution. We see many natural examples that are closer to a normal distribution.

  - Heights of people
  - Shoe sizes
  - Lap duration in a car race

  In a perfect normal distribution, we can see 50% symmetry about the center. Also, the centre is - mean = mode = median



Python Code

```
#Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from scipy import stats


#Reading Data
matches_data = pd.read_csv("matches.csv")
deliveries_data = pd.read_csv("deliveries.csv")


matches_data.head(2)
matches_data.info()
matches_data.describe()
matches_data.columns


deliveries_data.head(3)
deliveries_data.info()
deliveries_data.describe()
deliveries_data.columns


#List of the Top 20 Cities where the most number of matches have been p
layed.
city_counts=matches_data.groupby('city').apply(lambda x:x['city'].count
()).reset_index(name='Match Counts')
top_cities_order=city_counts.sort_values(by='Match Counts',ascending=Fa
lse)
top_cities=top_cities_order[:20]
print('Top 15 Cities with the maximum number of Matches Played:\n',top_
cities)
plt.figure(figsize=(9,9))
plt.pie(top_cities['Match Counts'],labels=top_cities['city'],autopct='%
1.1f%%', startangle=30)
plt.axis('equal')
plt.title('Top Cities that have hosted IPL Matches',size=20)


#List of the Top 20 venues where the most number of IPL matches have be
en played.
venue_counts=matches_data.groupby('venue').apply(lambda x:x['venue'].co
unt()).reset_index(name='Match Counts')
```

```
top_venues_order=venue_counts.sort_values(by='Match Counts',ascending=F
alse)
top_venues=top_venues_order[:20]
print('Top 20 Stadiums with the maximum number of Matches Played:\n',to
p_venues)
plt.figure(figsize=(9,9))
plt.pie(top_venues['Match Counts'],labels=top_venues['venue'],autopct='
%1.1f%%', startangle=40)
plt.axis('equal')
plt.title('Top Stadiums that have hosted IPL Matches',size=20)
```

```
#List of Top 10 Scoring Batsman
batting_tot=deliveries_data.groupby('batsman').apply(lambda x:np.sum(x[
'batsman_runs'])).reset_index(name='Runs')
batting_sorted=batting_tot.sort_values(by='Runs',ascending=False)
top_batsmen=batting_sorted[:10]
print('The Top 10 Batsmen in thr Tournament are:\n',top_batsmen)
fig = px.bar(top_batsmen, x='batsman', y='Runs',
             hover_data=['batsman'], color='Runs',title='Top 10 Batsmen
 in IPL- Seasons 2008-2019')
fig.show()
```

```
#List of the top 10 Bowlers with highest number of wickets
bowling_wickets=deliveries_data[deliveries_data['dismissal_kind']!='run
 out']
bowling_tot=bowling_wickets.groupby('bowler').apply(lambda x:x['dismiss
al_kind'].dropna()).reset_index(name='Wickets')
bowling_wick_count=bowling_tot.groupby('bowler').count().reset_index()
bowling_top=bowling_wick_count.sort_values(by='Wickets',ascending=False
)
top_bowlers=bowling_top.loc[:,['bowler','Wickets']][0:10]
print('The Top Wicket Takers in the Tournament are:\n',top_bowlers)
fig = px.bar(top_bowlers, x='bowler', y='Wickets',
             hover_data=['bowler'], color='Wickets',title='Top 10 Bowle
rs in IPL- Seasons 2008-2019')
fig.show()
```

```
#Creating a list of the best fielders- Considering Catch,Run Out and St
umpings
fielder_list=deliveries_data.groupby('fielder').apply(lambda x:x).dropn
a().reset_index()
fielder_list_count=fielder_list.groupby('fielder').count()
fielder_list_counts=fielder_list_count['dismissal_kind'].reset_index(na
me='Dismissals')
```

```
fielder_list_max=fielder_list_counts.sort_values(by='Dismissals',ascend
ing=False)
top_fielders=fielder_list_max[0:10]
print('The Best Fielders(and WicketKeepers) in the Torunament are:\n',t
op_fielders)

fig = px.bar(top_fielders, x='fielder', y='Dismissals',
             hover_data=['fielder'], color='Dismissals',title='Top 10 F
ielders in IPL- Seasons 2008-2019')
fig.show()


#List of the Players who have achieved highest number of "Man of the Ma
tch Awards"
motm=matches_data.groupby('player_of_match').apply(lambda x:x['player_o
f_match'].count()).reset_index(name='Man of the Match Awards')
motm_sort=motm.sort_values(by='Man of the Match Awards',ascending=False
)
motm_top=motm_sort[0:15]
plt.plot(motm_top['player_of_match'],motm_top['Man of the Match Awards'
],color='b')
plt.bar(motm_top['player_of_match'],motm_top['Man of the Match Awards']
,color='y')
plt.xlabel('Players')
plt.ylabel('Man of the Match Award Count')
plt.title('Top 15 Players who have won most the Man of the Match trophi
es',size=15)
plt.xticks(rotation=60)


#Number of matches in each season
plt.subplots(figsize=(10,6))
sns.countplot(matches_data['season'], data=matches_data)
plt.xlabel("Season")
plt.title("Number of Matches in season")
plt.show()


#Number of times each team wins
plt.subplots(figsize=(11,6))
sns.countplot(matches_data['winner'], data=matches_data, orient='h', pa
lette="Dark2")
plt.xticks(rotation=45)
plt.xlabel("Winner Team")
plt.title('The Number of Times Each Team Wins')
plt.show()
```

```
#List of top 10 Umpire
umpire_df = pd.melt(matches_data, id_vars=['id'], value_vars=['umpire1'
, 'umpire2'])
umpire_df["value"].value_counts()
plt.subplots(figsize=(11,5))
sns.countplot("value", data=umpire_df, order=umpire_df["value"].value_c
ounts().index[:10], palette="Dark2")
plt.xticks(rotation=45)
plt.title("Top 10 Umpire")
plt.xlabel("Umpire")


win_by_wickets_data = matches_data[matches_data.win_by_wickets > 0].win
_by_wickets
# Get mean (mu) and std (sigma)
win_by_wickets_mean, win_by_wickets_std = win_by_wickets_data.mean(), w
in_by_wickets_data.std()

# Plot histogram (normalized) - LIGHT-BLUE
win_by_wickets_data.hist(color='lightblue', weights = np.zeros_like(win
_by_wickets_data) + 1.0 / win_by_wickets_data.count())


# Normal distribution for random points between 1 to 10 with mean, std.
random_data = np.arange(1, 10, 0.001)
plt.plot(random_data, stats.norm.pdf(random_data, win_by_wickets_mean,
win_by_wickets_std), color='green')
mu, sigma = 128, 25 # From the above example
highest_scores = np.random.normal(mu, sigma, 1000) # Random 1000 values
count, bins, _ = plt.hist(highest_scores, 100, density=True, stacked=Tr
ue, color='lightblue') # plot 100 points
plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) * np.exp( - (bins - mu)**
2 / (2 * sigma**2) ), linewidth = 2, color = 'r') # Plot the PD
```

Link of Colab Notebook

https://colab.research.google.com/drive/1XvajTWFcqKJDRw0114FL7a-VSJbRSJNm#scrollTo=l06fEZ7UNdlW

Summary

In this project, dataset of IPL T-20 matches from 2008 to 2019 is used for performing some analysis on it and get some useful information out of that which can be used for various purpose.

I performed different type of analysis using python code, such as finding players with max run, players with maximum wickets, best fielders on basis of catch & run out, players with maximum number of "Man Of Match" title. Using these stats we can make a new team which has maximum probability of wining next IPL tournament.

Similarly, I performed analysis on basis of location i.e. top cities where most of IPL matches had been played and top stadium where maximum number of matches took place. Using this data we can find the probability that in which stadium of which city match should held so that maximum of people able to attend match.

I also find the data of win by wickets and win by runs. I apply normal distribution of win by wickets data and plot normal distribution curve. Using this data we can find probability of wining a match if we have given number of wickets. Similarly we can perform same analysis on win by runs data and can find probability that how much run a team should score in order to maximize the chances of wining.

We can perform a lot more analysis on this data and can find useful information from it. Dataset link is provided above in file and my work link is also provided in this file. If anyone want to further improve they can do so.