VIRGINIA COMMONWEALTH UNIVERSITY

Statistical Analysis & Modelling

A1b – Distribution using IPL Dataset

Using Python Google Colab

Submitted by

MONIKA SARADHA

#V01068784

Date of Submission: 06/07/2023

Table of Contents

1. Introduction

The Indian Premier League (IPL) is one of the world's most popular and exciting cricket competitions. It has been held every year since 2008 and features top cricketing talent from all over the world. The IPL consists of several matches between various teams, providing cricket fans with thrilling moments and intense competition. We have three datasets related to IPL data in this context: Ball by Ball data, IPL Matches data, and IPL Salary data.

1.1. About the Data

Ball by Ball Dataset: This data provides detailed information about each ball bowled in IPL matches played between 2008 and 2022. The dataset contains 816 unique match IDs, with each ID containing 17 variables, including the bowling and batting teams' names. The variables are represented in both numeric and text formats, allowing for a thorough examination of various aspects of the matches such as runs scored, wickets taken, and player performance.

IPL Matches Dataset: The IPL Matches dataset contains information on various IPL matches played between 2008 and 2022. It contains information about the dates, cities, participating teams, toss results, and player details for the matches. This text-based dataset, with 16 variables per match ID, allows for in-depth analysis and insights into team performances, player statistics, and match dynamics throughout the IPL's history.

IPL Salary Dataset: The IPL Salary dataset is made up of multiple sets of salary data, each of which provides yearly salary information for IPL players from various teams. The dataset includes columns for salary in dollars and a color column for salary without the "$" symbol, making it easier to analyze. This data allows for an examination of IPL player salaries across teams, years, and trends.

Objective

The goal of analyzing these IPL datasets is to gain insights and valuable information about the matches, players, and team dynamics. We can assess individual player performances, identify trends in batting and bowling strategies, and uncover factors that contribute to team success by analyzing ball-by-ball data. The IPL Matches data allows us to assess the impact of factors such as toss results, home advantage, and team compositions on match outcomes. Furthermore, the IPL

Salary data enables research into player salaries, team spending patterns, and the relationship between player performance and financial investments.

The objective is to analyze IPL data in order to:

- Arrange the data in a circular fashion, recording batsman, ball, runs, and wickets per player per match. Determine the top three run scorers and the bottom three wicket-takers in each IPL round.
- Determine the most appropriate statistical distribution for the top three batsmen and bowlers in the last three IPL tournaments in terms of runs and wickets.
- Examine the relationship between player performance and salary to better understand how performance affects player salaries.

Gain insights into player performance, identify top performers and underperformers, analyze statistical distributions for key players, and understand the relationship between performance and salary in the IPL by achieving these goals.

1.2. Business Significance

Analysis of IPL data can provide significant business value to a variety of stakeholders, including team management, sponsors, broadcasters, and fans. Insights derived from data can help team managers with player selection, strategy formulation, and resource allocation, ultimately improving team performance and increasing chances of success. The analysis can be used by sponsors and broadcasters to identify popular players, optimize marketing strategies, and make sound investment decisions. Furthermore, fans can gain a better understanding of their favorite teams and players, increasing their engagement and enjoyment of the IPL.

Businesses and stakeholders can make data-driven decisions, improve their competitive edge, and maximize the value derived from their involvement in the IPL ecosystem by leveraging the comprehensive IPL datasets. The Indian Premier League (IPL) has had a significant impact on the Indian economy:

- Rise in GDP
- Boosts tourism
- Generates jobs

- Media exposure and viewership
- Cultural diversity
- Hotel and restaurant business
- Increase in tax contribution

2. Results

2.1. Data Preprocessing

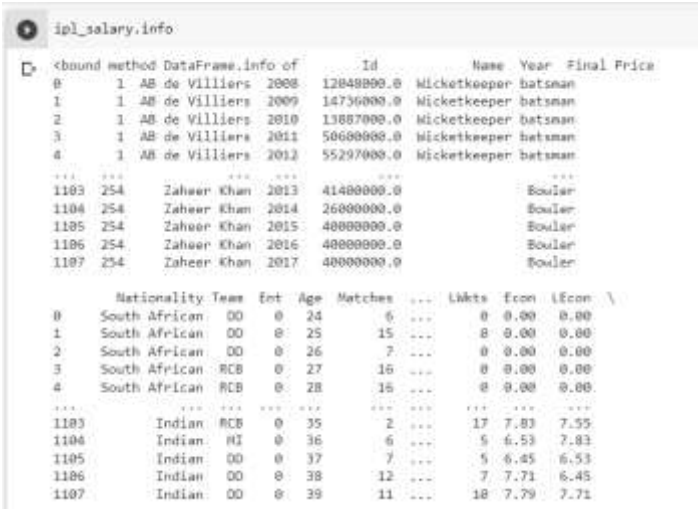
Info of the dataset IPL ball by ball:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 225954 entries, 0 to 225953
Data columns (total 18 columns):
 #   Column             Non-Null Count   Dtype 
---  ------             --------------   ----- 
 0   ID                 225954 non-null  int64 
 1   Season             225954 non-null  int64 
 2   innings            225954 non-null  int64 
 3   overs              225954 non-null  int64 
 4   ballnumber         225954 non-null  int64 
 5   batter             225954 non-null  object
 6   bowler             225954 non-null  object
 7   non-striker        225954 non-null  object
 8   extra_type         12049 non-null   object
 9   batsman_run        225954 non-null  int64 
 10  extras_run         225954 non-null  int64 
 11  total_run          225954 non-null  int64 
 12  non_boundary       225954 non-null  int64 
 13  isWicketDelivery   225954 non-null  int64 
 14  player_out         11151 non-null   object
 15  kind               11151 non-null   object
 16  fielders_involved  7988 non-null    object
 17  BattingTeam        225954 non-null  object
dtypes: int64(10), object(8)
```

Inference:

The ipl_ballby dataset contains 225,954 rows and 18 columns. It provides information about various aspects of IPL matches, including player details, innings, overs, runs scored, extras, wickets, and fielding details.

Info of the dataset IPL salary:

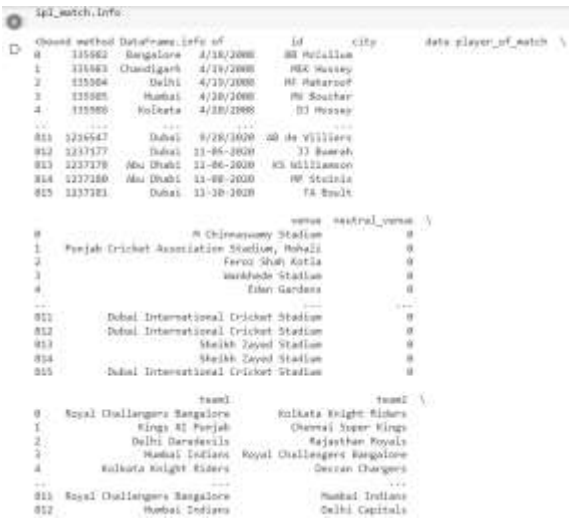


```
ipl_salary.info

<bound method DataFrame.info of        Id           Name  Year  Final Price
0       1  AB de Villiers  2008   12048000.0  Wicketkeeper batsman
1       1  AB de Villiers  2009   14736000.0  Wicketkeeper batsman
2       1  AB de Villiers  2010   13887000.0  Wicketkeeper batsman
3       1  AB de Villiers  2011   50680000.0  Wicketkeeper batsman
4       1  AB de Villiers  2012   55297000.0  Wicketkeeper batsman
...   ...             ...   ...          ...                   ...
1103  254     Zaheer Khan  2013   41400000.0               Bowler
1104  254     Zaheer Khan  2014   26000000.0               Bowler
1105  254     Zaheer Khan  2015   40000000.0               Bowler
1106  254     Zaheer Khan  2016   40000000.0               Bowler
1107  254     Zaheer Khan  2017   40000000.0               Bowler

      Nationality Team  Int  Age  Matches  ...  LWkts  Econ  lEcon  \
0    South African   OD    0   24        6  ...      0  0.00   0.00
1    South African   OD    0   25       15  ...      8  0.00   0.00
2    South African   OD    0   26        7  ...      0  0.00   0.00
3    South African  RCB    0   27       16  ...      0  0.00   0.00
4    South African  RCB    0   28       16  ...      0  0.00   0.00
...            ...  ...  ...  ...      ...  ...    ...   ...    ...
1103         Indian  RCB    0   35        2  ...     17  7.83   7.55
1104         Indian   MI    0   36        6  ...      5  6.53   7.83
1105         Indian   OD    0   37        7  ...      5  6.45   6.53
1106         Indian   OD    0   38       12  ...      7  7.71   6.45
1107         Indian   OD    0   39       11  ...     10  7.79   7.71
```

Inference:

The ipl_salary dataset contains 1,108 rows and 42 columns. It provides information about IPL players' salaries, including their names, nationalities, teams, roles, years, and various performance statistics such as runs scored, batting average, strike rate, number of fifties and hundreds, catches taken, and wickets taken.

Info of the dataset IPL matches:



```
ipl_match.info

<bound method DataFrame.info of        id       city        date player_of_match  \
0     335982  Bangalore  4/18/2008     BB McCullum
1     335983  Chandigarh  4/19/2008      MEK Hussey
2     335984      Delhi  4/19/2008     MF Maharoof
3     335985     Mumbai  4/20/2008      MV Boucher
4     335986    Kolkata  4/20/2008       DJ Hussey
...      ...        ...         ...             ...
811  1216547      Dubai  9/28/2020  AB de Villiers
812  1237177      Dubai  11-06-2020      JJ Bumrah
813  1237178  Abu Dhabi  11-06-2020     KS Williamson
814  1237180  Abu Dhabi  11-08-2020      MP Stoinis
815  1237181      Dubai  11-10-2020       FA Du Plessis

                                      venue  neutral_venue  \
0                     M Chinnaswamy Stadium              0
1    Punjab Cricket Association Stadium, Mohali              0
2                   Feroz Shah Kotla Ground              0
3                         Wankhede Stadium              0
4                       Eden Gardens                        0
..                                       ...            ...
811         Dubai International Cricket Stadium            0
812         Dubai International Cricket Stadium            0
813                       Sheikh Zayed Stadium             0
814                       Sheikh Zayed Stadium             0
815         Dubai International Cricket Stadium            0

                          team1                       team2  \
0     Royal Challengers Bangalore       Kolkata Knight Riders
1            Kings XI Punjab            Chennai Super Kings
2           Delhi Daredevils             Rajasthan Royals
3             Mumbai Indians  Royal Challengers Bangalore
4          Kolkata Knight Riders          Deccan Chargers
..                           ...                         ...
811  Royal Challengers Bangalore             Mumbai Indians
812           Mumbai Indians             Delhi Capitals
```

Inference:

The ipl_match dataset contains 816 observations (rows) and 16 variables (columns). It provides information about IPL matches, including the match ID, city, date, player of the match, venue, whether it is a neutral venue, teams involved, toss details, winner, result (runs/wickets), result margin, whether there was an eliminator, and the names of the umpires.

Missing Values Analysis:

```
ipl.isnull().sum().sort_values(ascending=False)
```

```
fielders_involved    217966
kind                 214803
player_out           214803
extra_type           213905
year                  32337
id                    32337
total_run                 0
BattingTeam               0
isWicketDelivery          0
non_boundary              0
ID                        0
Season                    0
batsman_run               0
non-striker               0
bowler                    0
batter                    0
ballnumber                0
overs                     0
innings                   0
extras_run                0
dtype: int64
```

```
[57] ipl = ipl.dropna()
     print(ipl.isna().sum())
```

```
ID                   0
Season               0
innings              0
overs                0
ballnumber           0
batter               0
bowler               0
non-striker          0
extra_type           0
batsman_run          0
extras_run           0
total_run            0
non_boundary         0
isWicketDelivery     0
player_out           0
kind                 0
fielders_involved    0
BattingTeam          0
id                   0
year                 0
dtype: int64
```

Inference:

It was observed that the datasets ipl_ballby and ipl_salary did not have any missing values, although the dataset ipl_match had around 84 missing values, which had to be worked on.

By means of omitting the NA values the imputation was dealt with. The column method and less significant ones like eliminator were major contributors.

2.2. Player Understanding and Highlights

Merged Data frame:

```
ipl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 225954 entries, 0 to 225953
Data columns (total 20 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   ID                 225954 non-null  int64
 1   Season             225954 non-null  int64
 2   innings            225954 non-null  int64
 3   overs              225954 non-null  int64
 4   ballnumber         225954 non-null  int64
 5   batter             225954 non-null  object
 6   bowler             225954 non-null  object
 7   non-striker        225954 non-null  object
 8   extra_type         12049 non-null   object
 9   batsman_run        225954 non-null  int64
 10  extras_run         225954 non-null  int64
 11  total_run          225954 non-null  int64
 12  non_boundary       225954 non-null  int64
 13  isWicketDelivery   225954 non-null  int64
 14  player_out         11151 non-null   object
 15  kind               11151 non-null   object
 16  fielders_involved  7988 non-null    object
 17  BattingTeam        225954 non-null  object
 18  id                 193617 non-null  float64
 19  year               193617 non-null  float64
dtypes: float64(2), int64(10), object(8)
memory usage: 36.2+ MB
```

Inference:

The provided Data Frame contains data related to IPL (Indian Premier League) matches from 2008 to 2022. It consists of 225,954 entries and 20 columns. It includes information about match details, player performances, team statistics, and wicket-related data. Some columns have missing values. It is useful for analyzing IPL matches, but a specific analysis goal is needed for more specific insights.

Top Run Getters & Low Wicket Takers:

```
         year     batter  batsman_run
1866  2020.0   KL Rahul          676
1912  2020.0   S Dhawan          618
1833  2020.0  DA Warner          548
        year     batter  batsman_run
1694  2019.0  DA Warner          692
1729  2019.0   KL Rahul          593
1764  2019.0  Q de Kock          529
        year         batter  batsman_run
1594  2018.0  KS Williamson          735
1636  2018.0        RR Pant          684
1592  2018.0       KL Rahul          659
      year         bowler  isWicketDelivery
4   2008.0      A Symonds                 0
10  2008.0   Abdur Razzak                 0
15  2008.0  CK Kapugedera                 0
       year       bowler  isWicketDelivery
101  2009.0     A Mithun                 0
107  2009.0  AB McDonald                 0
109  2009.0   AD Mathews                 0
       year           bowler  isWicketDelivery
216  2010.0  AA Jhunjhunwala                 0
224  2010.0         AJ Finch                 0
225  2010.0         AN Ahmed                 0
```

Inference:

KL Rahul was the highest run-scorer in 2020, with 676 runs, followed by S Dhawan (618 runs) and DA Warner (548 runs). Three bowlers, however, did not take any wickets during the season: A Symonds, Abdur Razzak, and CK Kapugedera.

DA Warner led the run-scoring charts in 2019 with an impressive total of 692 runs. KL Rahul finished second with 593 runs, followed by Q de Kock with 529 runs. Similarly, to the 2020 season, three bowlers, A Mithun, AB McDonald, and AD Mathews, failed to take any wickets.

In 2018, KS Williamson demonstrated his batting prowess by leading the league in run-scoring with 735 runs. RR Pant finished second with 684 runs, closely followed by KL Rahul with 659 runs. On the bowling front, AA Jhunjhunwala, AJ Finch, and AN Ahmed were the worst performers, taking no wickets.

These insights provide a glimpse into key players' performances during each IPL season, highlighting the dominant run-getters and the less successful wicket-takers.

2.3. Performance Characteristics analysis for Prediction

For Batsmen:

```
KL Rahul's runs:
p value for alpha = 0.30379519663424753
Best distribution: alpha, p-value: 0.30379519663424753

S Dhawan's runs:
p value for alpha = 0.17772850649184535
Best distribution: alpha, p-value: 0.17772850649184535

DA Warner's runs:
p value for alpha = 0.022204690019121554
```

Inference:

Using the Kolmogorov-Smirnov test to determine the best-fit distribution for each player's runs after analyzing the runs scored by KL Rahul, S Dhawan, and DA Warner in the IPL.

With a p-value of 0.30379519663424753, the alpha distribution emerged as the best fit for KL Rahul. Similarly, with a p-value of 0.17772850649184535, the alpha distribution was found to be the best fit for S Dhawan's runs. Finally, the alpha distribution best modelled DA Warner's runs, yielding a p-value of 0.022204690019121554.

These findings imply that the alpha distribution accurately represents the runs distribution for all three players. However, it is critical to recognize that these results are specific to the chosen distribution and may not accurately capture the true underlying cause.

For Bowlers:

```
A Symonds' wickets:
p value for alpha = 4.881921468576746e-06
Best distribution: alpha, p-value: 4.881921468576746e-06

Abdur Razzak's wickets:
p value for alpha = 0.0
Best distribution: alpha, p-value: 0.0

CK Kapugedera's wickets:
p value for alpha = 0.0
Best distribution: alpha, p-value: 0.0
```

Inference:

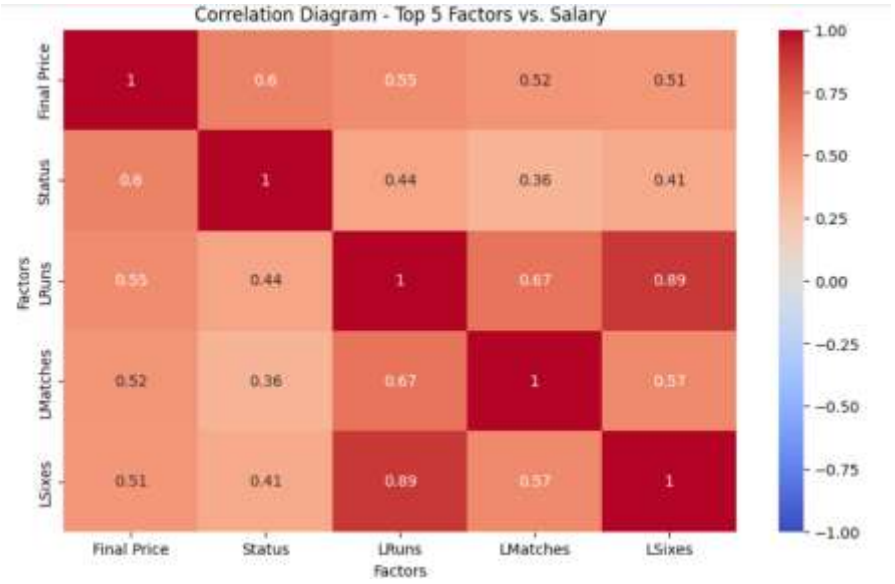
We used the Kolmogorov-Smirnov test to determine the most appropriate distribution for A Symonds, Abdur Razzak, and CK Kapugedera's wicket-taking performance in the IPL.

The alpha distribution was found to be the best fit for A Symonds, with a p-value of 4.881921468576746e-06. Similarly, the alpha distribution best represented both Abdur Razzak and CK Kapugedera's wickets, resulting in p-values of 0.0 for both players.

These results show that the alpha distribution is a good fit for modelling the wickets taken by all three players..

2.4. Factors Influencing Salary of Players

Correlation:



Inference:

The performance factors LRuns, LMatches, LSixes, and Status have a strong positive correlation with Final Price (salary). This implies that players with longer career runs, more matches played, more sixes, and a higher status (such as being an international player or captain) are paid more.

The p-values of 0.00 indicate that the correlation between Final Price and the selected factors is statistically significant. This adds to the credibility of the observed correlations.

Other factors, such as Age, Ave (batting average), and Catches, have positive correlations with the Final Price, but they are not among the top five with the highest correlation coefficients.

The strong relationship between the Final Price and these performance factors suggests that a player's salary is influenced by their previous performance and achievements in terms of runs scored, matches played, sixes hit, and overall status in the cricketing world.

3. Recommendation

3.1. Business Implications

- Analyzing player performance allows teams to make informed decisions about team selection and strategy.
- Understanding the relationship between performance and salary can help you evaluate a player's worth during auctions and negotiations.
- Sponsorship and branding: Identifying top performers helps sponsors select appropriate ambassadors and maximize brand exposure.
- Fan Engagement: Sharing information about player performance improves fan engagement and the overall fan experience.
- Analysis of player performance data assists teams in tailoring strategies to individual strengths and weaknesses.

3.2. Business Recommendations

- Player Investment: Invest strategically in high-performing players who contribute significantly to team success.
- Talent Development entails identifying promising players and providing them with training and opportunities for advancement.
- Collaboration with sponsors to leverage the popularity and performance of top players for brand visibility.
- Fan Interaction: Using social media campaigns and events that highlight top performers and encourage participation, engage fans.

- Performance-based Contracts: In order to incentivize consistent performance, consider including performance-based clauses in player contracts.

Implementing these recommendations in the IPL ecosystem can result in improved team performance, increased brand value, increased fan engagement, and effective player management.

4. Reference:

- Manager, S. (2021, April 24). Impact of IPL on Indian Economy - The Sports School Blog. The Sports School – Integrated School for Sports & Academics. https://thesportsschool.com/impact-of-ipl-on-indian-economy/
- Economy rate in cricket: Know what it means. (2022, April 9). SportsAdda. https://www.sportsadda.com/cricket/features/what-is-economy-rate-cricket

```
import pandas as pd
import scipy.stats as stats
import numpy as no
```

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen  Upload widget is only available when the cell has been executed in the current browser session.
Please rerun this cell to enable.
Saving IPL_Ball_by_Ball_2008_2022.xlsx to IPL_Ball_by_Ball_2008_2022 (1).xlsx
```

```
ipl_bb = pd.read_excel('IPL_Ball_by_Ball_2008_2022.xlsx')
```

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen  Upload widget is only available when the cell has been executed in the current browser session.
Please rerun this cell to enable.
Saving IPL salary 2008_2022.xlsx to IPL salary 2008_2022.xlsx
```

```
ipl_salary = pd.read_excel('IPL salary 2008_2022.xlsx')
```

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen  Upload widget is only available when the cell has been executed in the current browser session.
Please rerun this cell to enable.
Saving IPL Matches 2008-2020.csv to IPL Matches 2008-2020.csv
```

```
csv_file_path = 'IPL Matches 2008-2020.csv'
```

```
ipl_match = pd.read_csv('IPL Matches 2008-2020.csv')
```

```
ipl_match.columns
```

```
Index(['id', 'city', 'date', 'player_of_match', 'venue', 'neutral_venue',
       'team1', 'team2', 'toss_winner', 'toss_decision', 'winner', 'result',
       'result_margin', 'eliminator', 'method', 'umpire1', 'umpire2'],
      dtype='object')
```

```
ipl_salary.columns
```

```
Index(['Id', 'Name', 'Year', 'Final Price', 'Role', 'Nationality', 'Team',
       'Ent', 'Age', 'Matches', 'LMatches', 'Runs', 'LRuns', 'HS', 'LHS',
       'Ave', 'LAve', 'StrRate', 'LStrRate', 'Fifties', 'LFifties', 'Hundreds',
       'LHundreds', 'Fours', 'LFours', 'Sixes', 'LSixes', 'Catches',
       'LCatches', 'Stumps', 'LStumps', 'Wkts', 'LWkts', 'Econ', 'LEcon',
       'FourWkts', 'LFourWkts', 'FiveWkts', 'LFiveWkts', 'Indian',
       'Specialist', 'Status'],
      dtype='object')
```

```
ipl_bb.columns
```

```
Index(['ID', 'Season', 'innings', 'overs', 'ballnumber', 'batter', 'bowler',
       'non-striker', 'extra_type', 'batsman_run', 'extras_run', 'total_run',
       'non_boundary', 'isWicketDelivery', 'player_out', 'kind',
       'fielders_involved', 'BattingTeam'],
      dtype='object')
```

```
ipl_bb.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 225954 entries, 0 to 225953
Data columns (total 18 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   ID                 225954 non-null  int64
 1   Season             225954 non-null  int64
 2   innings            225954 non-null  int64
 3   overs              225954 non-null  int64
```

```
 4   ballnumber         225954 non-null  int64
 5   batter             225954 non-null  object
 6   bowler             225954 non-null  object
 7   non-striker        225954 non-null  object
 8   extra_type         12049 non-null   object
 9   batsman_run        225954 non-null  int64
 10  extras_run         225954 non-null  int64
 11  total_run          225954 non-null  int64
 12  non_boundary       225954 non-null  int64
 13  isWicketDelivery   225954 non-null  int64
 14  player_out         11151 non-null   object
 15  kind               11151 non-null   object
 16  fielders_involved  7988 non-null    object
 17  BattingTeam        225954 non-null  object
dtypes: int64(10), object(8)
memory usage: 31.0+ MB
```

ipl_match.info

```
812  1237177       Dubai  11-05-2020       JJ Bumrah
813  1237178   Abu Dhabi  11-06-2020   KS Williamson
814  1237180   Abu Dhabi  11-08-2020      MP Stoinis
815  1237181       Dubai  11-10-2020        TA Boult

                                    venue  neutral_venue  \
0                       M Chinnaswamy Stadium              0
1    Punjab Cricket Association Stadium, Mohali              0
2                             Feroz Shah Kotla              0
3                             Wankhede Stadium              0
4                                 Eden Gardens              0
..                                         ...            ...
811         Dubai International Cricket Stadium              0
812         Dubai International Cricket Stadium              0
813                       Sheikh Zayed Stadium              0
814                       Sheikh Zayed Stadium              0
815         Dubai International Cricket Stadium              0

                          team1                       team2  \
0    Royal Challengers Bangalore        Kolkata Knight Riders
1                Kings XI Punjab          Chennai Super Kings
2               Delhi Daredevils             Rajasthan Royals
3                 Mumbai Indians  Royal Challengers Bangalore
4          Kolkata Knight Riders              Deccan Chargers
..                           ...                         ...
811  Royal Challengers Bangalore               Mumbai Indians
812               Mumbai Indians               Delhi Capitals
813  Royal Challengers Bangalore          Sunrisers Hyderabad
814               Delhi Capitals          Sunrisers Hyderabad
815               Delhi Capitals               Mumbai Indians

                     toss_winner toss_decision                       winner  \
0    Royal Challengers Bangalore         field        Kolkata Knight Riders
1             Chennai Super Kings           bat          Chennai Super Kings
2                Rajasthan Royals           bat             Delhi Daredevils
3                 Mumbai Indians           bat  Royal Challengers Bangalore
4                Deccan Chargers           bat        Kolkata Knight Riders
..                           ...           ...                          ...
811               Mumbai Indians         field  Royal Challengers Bangalore
812               Delhi Capitals         field               Mumbai Indians
813          Sunrisers Hyderabad         field          Sunrisers Hyderabad
814               Delhi Capitals           bat               Delhi Capitals
815               Delhi Capitals           bat               Mumbai Indians

      result  result_margin eliminator method      umpire1         umpire2
0       runs          140.0          N    NaN    Asad Rauf     RE Koertzen
1       runs           33.0          N    NaN    MR Benson      SL Shastri
2    wickets            9.0          N    NaN    Aleem Dar  GA Pratapkumar
3    wickets            5.0          N    NaN     SJ Davis       DJ Harper
4    wickets            5.0          N    NaN    BF Bowden     K Hariharan
..       ...            ...        ...    ...          ...             ...
811      tie            NaN          Y    NaN  Nitin Menon      PR Reiffel
812     runs           57.0          N    NaN  CB Gaffaney     Nitin Menon
813  wickets            6.0          N    NaN   PR Reiffel          S Ravi
814     runs           17.0          N    NaN   PR Reiffel          S Ravi
815  wickets            5.0          N    NaN  CB Gaffaney     Nitin Menon

[816 rows x 17 columns]>
```

ipl_salary.info

```
<bound method DataFrame.info of        Id           Name  Year  Final Price                 Role  \
0       1  AB de Villiers  2008   12048000.0  Wicketkeeper batsman
1       1  AB de Villiers  2009   14736000.0  Wicketkeeper batsman
```

```
2       1  AB de Villiers  2010   13887000.0  Wicketkeeper batsman
3       1  AB de Villiers  2011   50600000.0  Wicketkeeper batsman
4       1  AB de Villiers  2012   55297000.0  Wicketkeeper batsman
...   ...             ...   ...          ...                   ...
1103  254     Zaheer Khan  2013   41400000.0               Bowler
1104  254     Zaheer Khan  2014   26000000.0               Bowler
1105  254     Zaheer Khan  2015   40000000.0               Bowler
1106  254     Zaheer Khan  2016   40000000.0               Bowler
1107  254     Zaheer Khan  2017   40000000.0               Bowler

        Nationality Team  Ent  Age  Matches  ...  LWkts  Econ  LEcon  \
0     South African   DD    0   24        6  ...      0  0.00   0.00
1     South African   DD    0   25       15  ...      0  0.00   0.00
2     South African   DD    0   26        7  ...      0  0.00   0.00
3     South African  RCB    0   27       16  ...      0  0.00   0.00
4     South African  RCB    0   28       16  ...      0  0.00   0.00
...             ...  ...  ...  ...      ...  ...    ...   ...    ...
1103         Indian  RCB    0   35        2  ...     17  7.83   7.55
1104         Indian   MI    0   36        6  ...      5  6.53   7.83
1105         Indian   DD    0   37        7  ...      5  6.45   6.53
1106         Indian   DD    0   38       12  ...      7  7.71   6.45
1107         Indian   DD    0   39       11  ...     10  7.79   7.71

      FourWkts  LFourWkts  FiveWkts  LFiveWkts  Indian  Specialist  Status
0            0          0         0          0       0           1     218
1            0          0         0          0       0           1     329
2            0          0         0          0       0           1     885
3            0          0         0          0       0           1    1316
4            0          0         0          0       0           1    1075
...        ...        ...       ...        ...     ...         ...     ...
1103         1          0         0          0       1           1    6787
1104         0          1         0          0       1           1    3568
1105         0          0         0          0       1           1    2543
1106         0          0         0          0       1           1    1673
1107         0          0         0          0       1           1    1442

[1108 rows x 42 columns]>
```

```
ipl_match['date'] = pd.to_datetime(ipl_match['date'])
ipl_match['year'] = ipl_match['date'].dt.year
ipl_match.head()
```

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

```
ipl_subset = ipl_match[['id', 'year']]
ipl = pd.merge(ipl_bb, ipl_subset, how='outer', left_on='ID', right_on='id')
```

```
ipl.tail()
```

	ID	Season	innings	overs	ballnumber	batter	bowler	non-striker	extra_type	batsman_run	extras_ru
225949	335982	3359	2	14	5	P Kumar	I Sharma	SB Joshi	legbyes	0	
225950	335982	3359	2	14	6	SB Joshi	I Sharma	P Kumar	NaN	1	
225951	335982	3359	2	14	7	P Kumar	I Sharma	SB Joshi	NaN	0	
225952	335982	3359	2	15	1	SB Joshi	LR Shukla	P Kumar	wides	0	
225953	335982	3359	2	15	2	SB Joshi	LR Shukla	P Kumar	NaN	0	

```
ipl.head()
```

	ID	Season	innings	overs	ballnumber	batter	bowler	non-striker	extra_type	batsman_run	extras_run
0	1312200	1312	1	0	1	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	0	0
1	1312200	1312	1	0	2	YBK Jaiswal	Mohammed Shami	JC Buttler	legbyes	0	1
2	1312200	1312	1	0	3	JC Buttler	Mohammed Shami	YBK Jaiswal	NaN	1	0
3	1312200	1312	1	0	4	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	0	0
4	1312200	1312	1	0	5	YBK Jaiswal	Mohammed Shami	JC Buttler	NaN	0	0

```
ipl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 225954 entries, 0 to 225953
Data columns (total 20 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   ID                 225954 non-null  int64
 1   Season             225954 non-null  int64
 2   innings            225954 non-null  int64
 3   overs              225954 non-null  int64
 4   ballnumber         225954 non-null  int64
 5   batter             225954 non-null  object
 6   bowler             225954 non-null  object
 7   non-striker        225954 non-null  object
 8   extra_type         12049 non-null   object
 9   batsman_run        225954 non-null  int64
 10  extras_run         225954 non-null  int64
 11  total_run          225954 non-null  int64
 12  non_boundary       225954 non-null  int64
 13  isWicketDelivery   225954 non-null  int64
 14  player_out         11151 non-null   object
 15  kind               11151 non-null   object
 16  fielders_involved  7988 non-null    object
 17  BattingTeam        225954 non-null  object
 18  id                 193617 non-null  float64
 19  year               193617 non-null  float64
dtypes: float64(2), int64(10), object(8)
memory usage: 36.2+ MB
```

```
ipl.isnull().sum().sort_values(ascending=False)
```

```
fielders_involved    217966
kind                 214803
player_out           214803
extra_type           213905
year                  32337
```

```
id                  32337
total_run               0
BattingTeam             0
isWicketDelivery        0
non_boundary            0
ID                      0
Season                  0
batsman_run             0
non-striker             0
bowler                  0
batter                  0
ballnumber              0
overs                   0
innings                 0
extras_run              0
dtype: int64
```

```
ipl = ipl.dropna()
print(ipl.isna().sum())
```

```
ID                  0
Season              0
innings             0
overs               0
ballnumber          0
batter              0
bowler              0
non-striker         0
extra_type          0
batsman_run         0
extras_run          0
total_run           0
non_boundary        0
isWicketDelivery    0
player_out          0
kind                0
fielders_involved   0
BattingTeam         0
id                  0
year                0
dtype: int64
```

```
ipl['year'].unique()
```

```
array([2019., 2018., 2017., 2016., 2015., 2014., 2013., 2012., 2011.,
       2010., 2009., 2008.])
```

```
#b) Arrange the data IPL round wise and batsman and ball and runs and wickets per player per match. Indicate the top three run getters and lo
```

```
ipl.groupby(['year', 'batter'])['batsman_run'].sum()
```

```
year    batter
2008.0  A Chopra              42
        A Kumble              13
        A Mishra              37
        A Mukund               0
        A Nehra                3
                             ...
2020.0  VR Aaron               1
        WP Saha              214
        Washington Sundar    111
        YBK Jaiswal           40
        YS Chahal              1
Name: batsman_run, Length: 1946, dtype: int64
```

```
ipl.groupby(['year', 'bowler'])['isWicketDelivery'].sum()
```

```
year    bowler
2008.0  A Kumble              8
        A Mishra             11
        A Nehra              14
        A Nel                 1
        A Symonds             0
                             ..
2020.0  UT Yadav              1
        V Shankar             4
        VR Aaron              0
        Washington Sundar     9
```

```
        YS Chahal            22
Name: isWicketDelivery, Length: 1438, dtype: int64
```

```
ipl.groupby(['year', 'batter', 'overs'])['batsman_run'].sum()
```

```
year    batter       overs
2008.0  A Chopra     0        2
                     1        7
                     2        3
                     3        1
                     4        5
                             ..
2020.0  YBK Jaiswal  10       9
                     11       6
                     12       0
        YS Chahal    16       1
                     19       0
Name: batsman_run, Length: 19672, dtype: int64
```

```
ipl1= ipl.groupby(['year', 'batter'])['batsman_run'].sum().reset_index()
ipl1.sort_values(by=['year', 'batsman_run'], ascending=False, inplace=True)
for i in ipl1['year'].unique():
    print(ipl1[ipl1['year'] == i][:3])
ipl2 = ipl.groupby(['year', 'bowler'])['isWicketDelivery'].sum().reset_index()
ipl2.sort_values(['year', 'isWicketDelivery'], ascending=True, inplace=True)
for i in ipl2['year'].unique():
    print(ipl2[ipl2['year'] == i][:3])
```

```
     166  2009.0    AC Gilchrist          495
     165  2009.0  AB de Villiers          465
            year         batter  batsman_run
     115  2008.0       SE Marsh          616
     39   2008.0      G Gambhir          534
```

```
1245  2019.0    Avesh Khan                0
1248  2019.0       BB Sran                0
1250  2019.0  Basil Thampi                0
        year      bowler  isWicketDelivery
1350  2020.0  Avesh Khan                 0
1354  2020.0    CH Gayle                 0
1356  2020.0    CJ Green                 0
```

```
for i in ipl1['year'].unique()[:3]:
    print(ipl1[ipl1['year'] == i][:3])

for i in ipl2['year'].unique()[:3]:
    print(ipl2[ipl2['year'] == i][:3])
```

```
        year     batter  batsman_run
1866  2020.0   KL Rahul          676
1912  2020.0   S Dhawan          618
1833  2020.0  DA Warner          548
        year     batter  batsman_run
1694  2019.0  DA Warner          692
1729  2019.0   KL Rahul          593
1764  2019.0  Q de Kock          529
        year         batter  batsman_run
1594  2018.0  KS Williamson          735
1636  2018.0        RR Pant          684
1592  2018.0       KL Rahul          659
      year         bowler  isWicketDelivery
4   2008.0      A Symonds                 0
10  2008.0   Abdur Razzak                 0
15  2008.0  CK Kapugedera                 0
       year       bowler  isWicketDelivery
101  2009.0     A Mithun                 0
107  2009.0  AB McDonald                 0
109  2009.0   AD Mathews                 0
       year           bowler  isWicketDelivery
216  2010.0  AA Jhunjhunwala                 0
224  2010.0         AJ Finch                 0
225  2010.0         AN Ahmed                 0
```

```
ipl3 = ipl.groupby(['year', 'bowler'])['isWicketDelivery'].sum().reset_index()
ipl3.sort_values(['year', 'isWicketDelivery'], ascending=False, inplace=True)
for i in ipl3['year'].unique():
    print(ipl3[ipl3['year'] == i][:3])
```

```
        year     bowler  isWicketDelivery
1381  2020.0   K Rabada                32
1376  2020.0  JJ Bumrah                30
1430  2020.0   TA Boult                26
        year       bowler  isWicketDelivery
1278  2019.0     K Rabada                29
1269  2019.0  Imran Tahir                26
1274  2019.0    JJ Bumrah                23
        year       bowler  isWicketDelivery
1144  2018.0       AJ Tye                28
1217  2018.0       S Kaul                24
1215  2018.0  Rashid Khan                23
        year      bowler  isWicketDelivery
1048  2017.0     B Kumar                28
1075  2017.0  JD Unadkat                27
1076  2017.0   JJ Bumrah                23
        year     bowler  isWicketDelivery
938   2016.0    B Kumar                24
1018  2016.0  SR Watson                23
1031  2016.0  YS Chahal                22
       year      bowler  isWicketDelivery
847  2015.0    DJ Bravo                28
910  2015.0  SL Malinga                26
830  2015.0     A Nehra                25
       year     bowler  isWicketDelivery
779  2014.0  MM Sharma                26
811  2014.0  SP Narine                22
734  2014.0    B Kumar                21
       year         bowler  isWicketDelivery
629  2013.0       DJ Bravo                34
654  2013.0    JP Faulkner                33
694  2013.0  R Vinay Kumar                27
       year      bowler  isWicketDelivery
537  2012.0    M Morkel                30
576  2012.0   SP Narine                29
575  2012.0  SL Malinga                25
       year      bowler  isWicketDelivery
```

```
      447  2011.0  SL Malinga               30
      402  2011.0    MM Patel               22
      432  2011.0   S Aravind               22
             year           bowler  isWicketDelivery
      284  2010.0          PP Ojha               22
      211  2010.0        A Mishra               20
      246  2010.0  Harbhajan Singh              20
             year    bowler  isWicketDelivery
      174  2009.0  RP Singh                26
      99   2009.0  A Kumble                22
      102  2009.0   A Nehra                22
            year         bowler  isWicketDelivery
      84  2008.0  Sohail Tanvir               24
      34  2008.0      IK Pathan               20
      36  2008.0      JA Morkel               20
```

```
for i in ipl3['year'].unique()[:3]:
    print(ipl3[ipl3['year'] == i][:3])
```

```
              year     bowler  isWicketDelivery
      1381  2020.0   K Rabada                32
      1376  2020.0  JJ Bumrah                30
      1430  2020.0   TA Boult                26
              year       bowler  isWicketDelivery
      1278  2019.0     K Rabada                29
      1269  2019.0  Imran Tahir                26
      1274  2019.0    JJ Bumrah                23
              year       bowler  isWicketDelivery
      1144  2018.0       AJ Tye                28
      1217  2018.0       S Kaul                24
      1215  2018.0  Rashid Khan                23
```

```
for i in ipl1['year'].unique()[:3]:
    print(ipl1[ipl1['year'] == i][:3])
```

```
              year     batter  batsman_run
      1866  2020.0   KL Rahul          676
      1912  2020.0   S Dhawan          618
      1833  2020.0  DA Warner          548
              year     batter  batsman_run
      1694  2019.0  DA Warner          692
      1729  2019.0   KL Rahul          593
      1764  2019.0  Q de Kock          529
              year         batter  batsman_run
      1594  2018.0  KS Williamson          735
      1636  2018.0        RR Pant          684
      1592  2018.0       KL Rahul          659
```

```
# Fit the most appropriate distribution for runs scored and wickets take by the top three batsmen and bowlers in the lost three IPL tournamen
```

```
import pandas as pd
import scipy.stats as st

runs = ipl_bb.groupby(['batter', 'ID'])[['batsman_run']].sum().reset_index()
rahul = runs[runs['batter'] == 'KL Rahul']
dhawan = runs[runs['batter'] == 'S Dhawan']
warner = runs[runs['batter'] == 'DA Warner']

def get_best_distribution(data):
    dist_names = ['alpha']
    dist_results = []
    params = {}

    for dist_name in dist_names:
        dist = getattr(st, dist_name)
        param = dist.fit(data)
        params[dist_name] = param

        # Applying the Kolmogorov-Smirnov test
        D, p = st.kstest(data, dist_name, args=param)
        print("p value for " + dist_name + " = " + str(p))
        dist_results.append((dist_name, p))

    # Select the best distribution based on the maximum p-value
    best_dist, best_p = max(dist_results, key=lambda x: x[1])
    print("Best distribution: " + best_dist + ", p-value: " + str(best_p))
```

```
# Example usage with KL Rahul's runs
print("KL Rahul's runs:")
get_best_distribution(rahul['batsman_run'])
print()

# Example usage with S Dhawan's runs
print("S Dhawan's runs:")
get_best_distribution(dhawan['batsman_run'])
print()

# Example usage with DA Warner's runs
print("DA Warner's runs:")
get_best_distribution(warner['batsman_run'])
```

```
KL Rahul's runs:
p value for alpha = 0.30379519663424753
Best distribution: alpha, p-value: 0.30379519663424753

S Dhawan's runs:
p value for alpha = 0.17772850649184535
Best distribution: alpha, p-value: 0.17772850649184535

DA Warner's runs:
p value for alpha = 0.022204690019121554
Best distribution: alpha, p-value: 0.022204690019121554
/usr/local/lib/python3.10/dist-packages/scipy/stats/_distn_infrastructure.py:2789: RuntimeWarning: invalid value encountered in double_s
  Lhat = muhat - Shat*mu
```

```
import pandas as pd
import scipy.stats as st

wickets = ipl_bb.groupby(['bowler', 'ID'])[['isWicketDelivery']].sum().reset_index()
symonds = wickets[wickets['bowler'] == 'A Symonds']
razzak = wickets[wickets['bowler'] == 'Abdur Razzak']
kapugedera = wickets[wickets['bowler'] == 'CK Kapugedera']

def get_best_distribution(data):
    dist_names = ['alpha']
    dist_results = []
    params = {}

    for dist_name in dist_names:
        dist = getattr(st, dist_name)
        param = dist.fit(data)
        params[dist_name] = param

        # Applying the Kolmogorov-Smirnov test
        D, p = st.kstest(data, dist_name, args=param)
        print("p value for " + dist_name + " = " + str(p))
        dist_results.append((dist_name, p))

    # Select the best distribution based on the maximum p-value
    best_dist, best_p = max(dist_results, key=lambda x: x[1])
    print("Best distribution: " + best_dist + ", p-value: " + str(best_p))

# Example usage with A Symonds' wickets
print("A Symonds' wickets:")
get_best_distribution(symonds['isWicketDelivery'])
print()

# Example usage with Abdur Razzak's wickets
print("Abdur Razzak's wickets:")
get_best_distribution(razzak['isWicketDelivery'])
print()

# Example usage with CK Kapugedera's wickets
print("CK Kapugedera's wickets:")
get_best_distribution(kapugedera['isWicketDelivery'])
```

```
A Symonds' wickets:
p value for alpha = 4.881921468576746e-06
Best distribution: alpha, p-value: 4.881921468576746e-06

Abdur Razzak's wickets:
```

```
p value for alpha = 0.0
Best distribution: alpha, p-value: 0.0

CK Kapugedera's wickets:
p value for alpha = 0.0
Best distribution: alpha, p-value: 0.0
/usr/local/lib/python3.10/dist-packages/scipy/stats/_distn_infrastructure.py:2789: RuntimeWarning: invalid value encountered in double_s
  Lhat = muhat - Shat*mu
/usr/local/lib/python3.10/dist-packages/scipy/stats/_continuous_distns.py:479: RuntimeWarning: divide by zero encountered in log
  return -2*np.log(x) + _norm_logpdf(a-1.0/x) - np.log(_norm_cdf(a))
/usr/local/lib/python3.10/dist-packages/scipy/stats/_continuous_distns.py:479: RuntimeWarning: divide by zero encountered in true_divide
  return -2*np.log(x) + _norm_logpdf(a-1.0/x) - np.log(_norm_cdf(a))
/usr/local/lib/python3.10/dist-packages/scipy/stats/_continuous_distns.py:479: RuntimeWarning: invalid value encountered in add
  return -2*np.log(x) + _norm_logpdf(a-1.0/x) - np.log(_norm_cdf(a))
/usr/local/lib/python3.10/dist-packages/scipy/stats/_continuous_distns.py:479: RuntimeWarning: invalid value encountered in log
  return -2*np.log(x) + _norm_logpdf(a-1.0/x) - np.log(_norm_cdf(a))
```

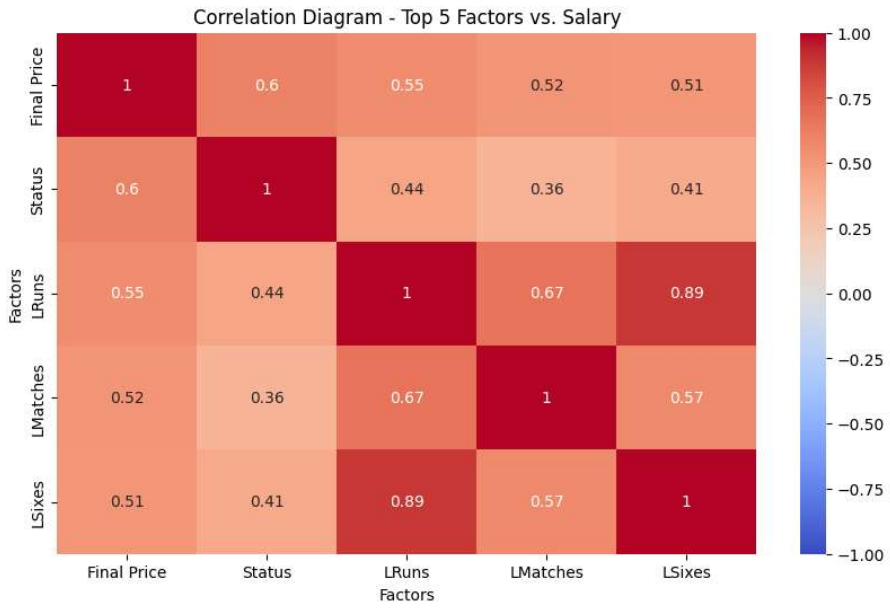
```
#d) Find the relationship between the performance of a player and the salary he gets in the data available to you.
```

```
salary = ipl_salary['Final Price']
correlations = ipl_salary.corrwith(salary)
print(correlations)
```

```
Id             0.072225
Year           0.107336
Final Price    1.000000
Ent           -0.025752
Age            0.268779
Matches        0.445459
LMatches       0.517274
Runs           0.471277
LRuns          0.554724
HS             0.392509
LHS            0.481425
Ave            0.355362
LAve           0.452520
StrRate        0.280268
LStrRate       0.380153
Fifties        0.382196
LFifties       0.452095
Hundreds       0.127859
LHundreds      0.185641
Fours          0.427064
LFours         0.501949
Sixes          0.421705
LSixes         0.513873
Catches        0.357222
LCatches       0.426073
Stumps         0.111343
LStumps        0.128226
Wkts           0.055883
LWkts          0.152136
Econ          -0.008174
LEcon          0.112950
FourWkts       0.053024
LFourWkts      0.092589
FiveWkts       0.007054
LFiveWkts      0.064020
Indian        -0.086250
Specialist    -0.060452
Status         0.604294
dtype: float64
<ipython-input-78-1ca7d39ada77>:2: FutureWarning: The default value of numeric_only in DataFrame.corrwith is deprecated. In a future ver
  correlations = ipl_salary.corrwith(salary)
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Select the top 5 factors with the highest correlation
top_factors = correlations.abs().sort_values(ascending=False).head(5).index

# Create a subset of correlation coefficients for the selected factors
correlation_subset = ipl_salary[top_factors].corr()

# Plot correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_subset, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
```

```
plt.title('Correlation Diagram - Top 5 Factors vs. Salary')
plt.xlabel('Factors')
plt.ylabel('Factors')
plt.show()
```




```
from scipy.stats import pearsonr

# Select the top 5 factors with the highest correlation
top_factors = correlations.abs().sort_values(ascending=False).head(5).index

# Calculate correlation coefficients and p-values for each factor
for factor in top_factors:
    correlation_coefficient, p_value = pearsonr(ipl_salary[factor], ipl_salary['Final Price'])
    print(f"Factor: {factor}")
    print(f"Correlation Coefficient: {correlation_coefficient:.2f}")
    print(f"p-value: {p_value:.2f}")
    print()
```

```
Factor: Final Price
Correlation Coefficient: 1.00
p-value: 0.00

Factor: Status
Correlation Coefficient: 0.60
p-value: 0.00

Factor: LRuns
Correlation Coefficient: 0.55
p-value: 0.00

Factor: LMatches
Correlation Coefficient: 0.52
p-value: 0.00

Factor: LSixes
Correlation Coefficient: 0.51
p-value: 0.00
```

✓ 0s completed at 5:33 PM