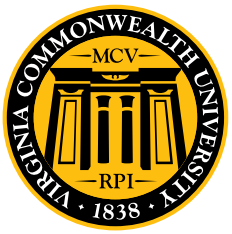



# VIRGINIA COMMONWEALTH UNIVERSITY

## Statistical analysis and modelling (SCMA 632)

### A1b: Analysis of IPL dataset and finding distribution for the player KL Rahul using R and Python

**Cherussery Jayaprakash Lakshmy**

**V01110023**

**Date of Submission: 18-06-2024**

**CONTENTS**


| Sl. No. | Title | Page No. |
|---|---|---|
| **1.** | Introduction | **1** |
| **2.** | Objective | **2** |
| **3.** | Business Significance | **3** |
| **4.** | Results and Interpretations | **4-9** |
| **5.** | Codes | **10-14** |
| **6.** | Recommendations | **15** |

# Introduction

This assignment aims to analyze IPL data from the 2007/08 season to the 2024 season, focusing on player performance and its correlation with player salaries. By examining the top performers in terms of runs and wickets and fitting statistical distributions to their performances, we aim to uncover insights that could be valuable for teams and management.

## Objective

1. **Data Extraction and Arrangement**: Extract IPL data for the seasons and organize it round-wise, detailing batsman, ball, runs, and wickets per player per match.
2. **Performance Analysis**: Identify the top three run-getters and top three wicket-takers in each IPL round.
3. **Statistical Distribution**: Fit the most appropriate statistical distribution for runs scored and wickets taken by the top three batsmen and bowlers in the last three IPL tournaments. Also for the player KL Rahul
4. **Performance-Salary Correlation**: Investigate the relationship between a player's on-field performance and their salary.

## Business Significance

Understanding the performance metrics of players and their correlation with salaries is crucial for IPL franchises. This analysis can help in:

1. **Player Retention and Auction Strategy**: Franchises can make informed decisions on retaining players or bidding for new players during auctions based on performance trends and salary data.
2. **Salary Optimization**: Ensuring that the salaries offered to players are in alignment with their contributions to the team's success, thereby optimizing budget allocations.
3. **Performance Prediction**: Fitting statistical distributions to performance data can aid in predicting future performances, helping teams strategize better.
4. **Talent Scouting and Development**: Identifying consistently high performers can help in scouting and developing young talents, ensuring long-term team success.

## Results and Interpretations

Grouping data by Runs and Wickets

```
[55] player_runs = grouped_data.groupby(['Season', 'Striker'])['runs_scored'].sum().reset_index()
     player_wickets = grouped_data.groupby(['Season', 'Bowler'])['wicket_confirmation'].sum().reset_index()
```

```
player_runs[player_runs['Season']=='2023'].sort_values(by='runs_scored',ascending=False)
```

| | Season | Striker | runs_scored |
|---|---|---|---|
| 2423 | 2023 | Shubman Gill | 890 |
| 2313 | 2023 | F du Plessis | 730 |
| 2311 | 2023 | DP Conway | 672 |
| 2433 | 2023 | V Kohli | 639 |
| 2443 | 2023 | YBK Jaiswal | 625 |
| ... | ... | ... | ... |
| 2404 | 2023 | RP Meredith | 0 |
| 2372 | 2023 | Mohsin Khan | 0 |
| 2307 | 2023 | DG Nalkande | 0 |
| 2429 | 2023 | TU Deshpande | 0 |
| 2324 | 2023 | Harshit Rana | 0 |

177 rows × 3 columns

Interpretation: It can be seen that in the season 2023, Shubman Gill has scored 890 runs with F du Plessis in the second position at 730 runs. These performances underline the contributions of both seasoned players and emerging talents, with Shubman Gill's standout performance being a key highlight of the 2023 IPL season.

## top three run-getters and top three wicket-takers in each IPL round.

```
top_run_getters = player_runs.groupby('Season').apply(lambda x: x.nlargest(3, 'runs_scored')).reset_index(drop=True)
bottom_wicket_takers = player_wickets.groupby('Season').apply(lambda x: x.nlargest(3, 'wicket_confirmation')).reset_index(drop=True)
print("Top Three Run Getters:")
print(top_run_getters)
print("Top Three Wicket Takers:")
print(bottom_wicket_takers)
```

```
Top Three Run Getters:
     Season          Striker  runs_scored
0   2007/08         SE Marsh          616
1   2007/08        G Gambhir          534
2   2007/08     ST Jayasuriya          514
3      2009        ML Hayden          572
4      2009     AC Gilchrist          495
5      2009    AB de Villiers          465
6   2009/10      SR Tendulkar          618
7   2009/10         JH Kallis          572
8   2009/10          SK Raina          528
9      2011         CH Gayle          608
10     2011          V Kohli          557
11     2011      SR Tendulkar          553
12     2012         CH Gayle          733
13     2012        G Gambhir          590
14     2012         S Dhawan          569
15     2013        MEK Hussey          733
16     2013         CH Gayle          720
17     2013          V Kohli          639
```

```
50     2024  B Sai Sudharsan          418
Top Three Wicket Takers:
     Season           Bowler  wicket_confirmation
0   2007/08    Sohail Tanvir                   24
1   2007/08        IK Pathan                   20
2   2007/08        JA Morkel                   20
3      2009         RP Singh                   26
4      2009         A Kumble                   22
5      2009          A Nehra                   22
6   2009/10          PP Ojha                   22
7   2009/10         A Mishra                   20
8   2009/10  Harbhajan Singh                   20
9      2011       SL Malinga                   30
10     2011         MM Patel                   22
11     2011        S Aravind                   22
12     2012         M Morkel                   30
13     2012        SP Narine                   29
14     2012       SL Malinga                   25
15     2013         DJ Bravo                   34
16     2013      JP Faulkner                   33
17     2013   R Vinay Kumar                   27
18     2014        MM Sharma                   26
19     2014        SP Narine                   22
20     2014          B Kumar                   21
21     2015         DJ Bravo                   28
22     2015       SL Malinga                   26
23     2015          A Nehra                   25
24     2016          B Kumar                   24
25     2016        SR Watson                   23
26     2016        YS Chahal                   22
27     2017          B Kumar                   28
```

Interpretation: The early years were dominated by well-established players like Shaun Marsh, Sachin Tendulkar, and Lasith Malinga, while recent years have seen a mix of experienced and young players like Virat Kohli, David Warner, and Shubman Gill making significant impacts.

# Fitting the most appropriate distribution for runs scored and wickets taken by the top three batsmen and bowlers in the lost three IPL tournaments.

```
**********
year: 2024  Batsman: RD Gaikwad
p value for alpha = 2.599259711013304e-20
p value for beta = 0.02041902689492403
p value for betaprime = 0.01950376359866901
p value for burr12 = 0.46882020698395865
p value for crystalball = 0.24953646987270484
p value for dgamma = 0.1570743843120962
p value for dweibull = 0.20046582403736823
p value for erlang = 1.893799588395604e-06
p value for exponnorm = 0.4644304230917985
p value for f = 1.3560920695663998e-07
p value for fatiguelife = 1.304427037367869e-14
p value for gamma = 0.005830868576003678
p value for gengamma = 0.015331622187826133
p value for gumbel_l = 0.05546236480086464
p value for johnsonsb = 4.646964117947127e-13
p value for kappa4 = 0.006363220770325362
p value for lognorm = 1.1719355665219537e-16
p value for nct = 0.5881570496217812
p value for norm = 0.24953651809309751
p value for norminvgauss = 0.5538573365183158
p value for powernorm = 0.1788753268739085
p value for rice = 0.1828753716397571
p value for recipinvgauss = 0.06459275668874309
p value for t = 0.2494021485911212
p value for trapz = 7.476391685388162e-13
p value for truncnorm = 0.24173236832621992

Best fitting distribution: nct
Best p value: 0.5881570496217812
Parameters for the best fit: (5.718048022849898, 9.399490726283615, -54.25277343780452, 8.497060689079994)
```

```
**********
year: 2024  Bowler: HV Patel
p value for alpha = 0.0002993252328930706
p value for beta = 2.7775719087765 89e-19
p value for betaprime = 1.7052883875145053e-30
p value for burr12 = 5.427998338605459e-15
p value for crystalball = 1.1109118198587684e-05
p value for dgamma = 4.375428528574276e-05
p value for dweibull = 1.8553295107771936e-05
p value for erlang = 5.473635282991912e-24
p value for exponnorm = 0.0002813279943461815
p value for f = 1.9012983291282487e-09
p value for fatiguelife = 1.9734428958773156e-05
p value for gamma = 1.470787431589663e-16
p value for gengamma = 1.4345058849022962e-16
p value for gumbel_l = 4.541523588271283e-05
p value for johnsonsb = 2.827201329331457e-51
p value for kappa4 = 9.177530010006471e-23
p value for lognorm = 5.2162358572043325e-22
p value for nct = 0.0001960277304576293
p value for norm = 1.1109124960635979e-05
p value for norminvgauss = 3.8111964780204675e-05
p value for powernorm = 3.2186417463058256e-05
p value for rice = 3.354567282896195e-05
p value for recipinvgauss = 5.05058721389515e-12
p value for t = 9.451105792399515e-05
p value for trapz = 1.0447243016629734e-51
p value for truncnorm = 0.0002182292327632623

Best fitting distribution: alpha
Best p value: 0.0002993252328930706
Parameters for the best fit: (5.200800514990576, -4.106246473111661, 27.580368990504883)
```

## Distribution for KL Rahul

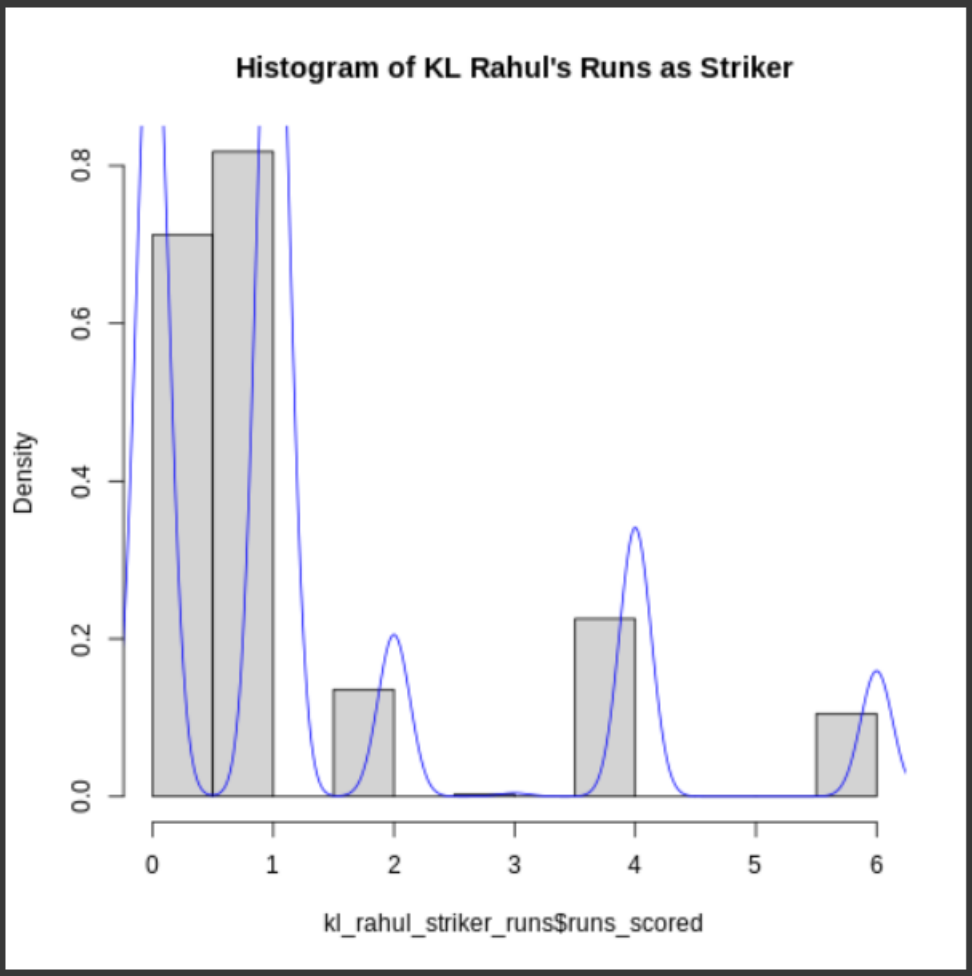
```
[49] print("Distribution for KL Rahul")
     get_best_distribution(runs[runs["Striker"] == "KL Rahul"]["runs_scored"])
```

```
Distribution for KL Rahul
p value for alpha = 3.439822697019343e-50
p value for beta = 0.30051910420099104
p value for betaprime = 0.3083252430394989
p value for burr12 = 0.46187713102710526
p value for crystalball = 0.02169172684247168
p value for dgamma = 0.06770258558041672
p value for dweibull = 0.10186919378179622
p value for erlang = 0.5713953642722212
p value for exponnorm = 0.2160721375507493
p value for f = 3.271576641222778e-23
p value for fatiguelife = 0.412197583971466
p value for gamma = 0.5713982751559553
p value for gengamma = 0.16010152392031302
p value for gumbel_l = 0.0016806774551022254
p value for johnsonsb = 0.9402453631468569
p value for kappa4 = 1.3895397566735892e-07
p value for lognorm = 9.796218603186654e-32
p value for nct = 0.20349727522799924
p value for norm = 0.021691727067096336
p value for norminvgauss = 0.3817037858972116
p value for powernorm = 0.02664556549931103
p value for rice = 0.027062729391117993
p value for recipinvgauss = 0.4426895366659932
p value for t = 0.0216940881910511
p value for trapz = 1.8532732379092856e-35
p value for truncnorm = 0.6753901355264902

Best fitting distribution: johnsonsb
Best p value: 0.9402453631468569
```

```
Parameters for the best fit: (0.9331207997896902, 0.7776389044559282, -2.345202857963142, 143.0833194837059)
('johnsonsb',
 0.9402453631468569,
 (0.9331207997896902,
  0.7776389044559282,
  -2.345202857963142,
  143.0833194837059))
```

USING R:

```
Shapiro-Wilk Normality Test:

        Shapiro-Wilk normality test

data:  kl_rahul_striker_runs$runs_scored
W = 0.72957, p-value < 2.2e-16


Skewness: 1.564706
Kurtosis: 1.558586

Parameters of Normal Distribution for KL Rahul's Runs as Striker:
    mean       sd
1.315789 1.631166

Kolmogorov-Smirnov Test for Normality:

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  kl_rahul_striker_runs$runs_scored
D = 0.34207, p-value < 2.2e-16
alternative hypothesis: two-sided
```




Interpretation:

**Best Fitting Distribution**: Johnson SB

**Best p-value**: 0.9402

**Parameters for Johnson SB**: (0.9331, 0.7776, -2.3452, 143.0833)

The statistical analysis of KL Rahul's run-scoring data indicates that the Johnson SB distribution provides the best fit, with a high p-value of 0.9402, suggesting a good match to the observed data. The parameters for the Johnson SB distribution describe its shape and scale, capturing the variability and distribution of KL Rahul's runs effectively. This finding implies that the Johnson SB distribution can be used to model and predict KL Rahul's future performance with a high degree of confidence.


Interpretation: The positive correlation coefficient of 0.3061 indicates that runs scored and player salaries are positively associated in IPL cricket. This statistical insight underscores the value placed on batting performance in the IPL, guiding strategic decisions aimed at building competitive and successful teams.

## Codes

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
ipl_bbb = pd.read_csv('/content/IPL_ball_by_ball_updated till 2024
(1).csv',low_memory=False)
ipl_salary = pd.read_excel('/content/IPL SALARIES 2024 (1).xlsx')
ipl_salary.head(2)
ipl_bbb.tail()
grouped_data = ipl_bbb.groupby(['Season', 'Innings No',
'Striker','Bowler']).agg({'runs_scored': sum,
'wicket_confirmation':sum}).reset_index()
```

## Grouping data by Runs and Wickets

```
player_runs = grouped_data.groupby(['Season',
'Striker'])['runs_scored'].sum().reset_index()
player_runs[player_runs['Season']=='2023'].sort_values(by='runs_scored'
,ascending=False)
```

## top three run-getters and top three wicket-takers in each IPL round.

```
top_run_getters = player_runs.groupby('Season').apply(lambda x:
x.nlargest(3, 'runs_scored')).reset_index(drop=True)
bottom_wicket_takers = player_wickets.groupby('Season').apply(lambda x:
x.nlargest(3, 'wicket_confirmation')).reset_index(drop=True)
print("Top Three Run Getters:")
print(top_run_getters)
print("Top Three Wicket Takers:")
print(bottom_wicket_takers)
ipl_year_id = pd.DataFrame(columns=["id", "year"])
ipl_year_id["id"] = ipl_bbb["Match id"]
ipl_year_id["year"] = pd.to_datetime(ipl_bbb["Date"],
dayfirst=True).dt.year
#create a copy of ipl_bbbc dataframe
ipl_bbbc= ipl_bbb.copy()
ipl_bbbc['year'] = pd.to_datetime(ipl_bbb["Date"],
dayfirst=True).dt.year
ipl_bbbc[["Match id", "year",
"runs_scored","wicket_confirmation","Bowler",'Striker']].head()
```

## Fitting the most appropriate distribution for runs scored and wickets taken by the top three batsmen and bowlers in the lost three IPL tournaments.

```
import scipy.stats as st

def get_best_distribution(data):
    dist_names = ['alpha','beta','betaprime','burr12','crystalball',
                  'dgamma','dweibull','erlang','exponnorm','f','fatigue
life',
                  'gamma','gengamma','gumbel_l','johnsonsb','kappa4',
                  'lognorm','nct','norm','norminvgauss','powernorm','ri
ce',
                  'recipinvgauss','t','trapz','truncnorm']
    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 fitted distribution
    best_dist, best_p = (max(dist_results, key=lambda item: item[1]))
    # store the name of the best fit and its p value
    print("\nBest fitting distribution: "+str(best_dist))
    print("Best p value: "+ str(best_p))
    print("Parameters for the best fit: "+ str(params[best_dist]))
    return best_dist, best_p, params[best_dist]
total_run_each_year = ipl_bbbc.groupby(["year",
"Striker"])["runs_scored"].sum().reset_index()
print(total_run_each_year)
```

```
list_top_batsman_last_three_year = {}
for i in total_run_each_year["year"].unique()[:3]:
    list_top_batsman_last_three_year[i] =
total_run_each_year[total_run_each_year.year ==
i][:3]["Striker"].unique().tolist()

list_top_batsman_last_three_year
```

```
import warnings
warnings.filterwarnings('ignore')
runs = ipl_bbbc.groupby(['Striker','Match
id'])[['runs_scored']].sum().reset_index()

for key in list_top_batsman_last_three_year:
    for Striker in list_top_batsman_last_three_year[key]:
        print("************************")
        print("year:", key, " Batsman:", Striker)
        get_best_distribution(runs[runs["Striker"] ==
Striker]["runs_scored"])
        print("\n\n")
```

## Distribution for KL Rahul

```
print("Distribution for KL Rahul")
get_best_distribution(runs[runs["Striker"] == "KL
Rahul"]["runs_scored"])
total_wicket_each_year = ipl_bbbc.groupby(["year",
"Bowler"])["wicket_confirmation"].sum().reset_index()
total_wicket_each_year.sort_values(["year", "wicket_confirmation"],
ascending=False, inplace=True)
print(total_wicket_each_year)
list_top_bowler_last_three_year = {}
for i in total_wicket_each_year["year"].unique()[:3]:
    list_top_bowler_last_three_year[i] =
total_wicket_each_year[total_wicket_each_year.year ==
i][:3]["Bowler"].unique().tolist()
list_top_bowler_last_three_year
```

```
import warnings
warnings.filterwarnings('ignore')
wickets = ipl_bbbc.groupby(['Bowler','Match
id'])[['wicket_confirmation']].sum().reset_index()

for key in list_top_bowler_last_three_year:
    for bowler in list_top_bowler_last_three_year[key]:
        print("************************")
        print("year:", key, " Bowler:", bowler)
        get_best_distribution(wickets[wickets["Bowler"] ==
bowler]["wicket_confirmation"])
        print("\n\n")
R2024 =total_run_each_year[total_run_each_year['year']==2024]
!pip install fuzzywuzzy
from fuzzywuzzy import process

# Convert to DataFrame
```

```
df_salary = ipl_salary.copy()
df_runs = R2024.copy()

# Function to match names
def match_names(name, names_list):
    match, score = process.extractOne(name, names_list)
    return match if score >= 80 else None  # Use a threshold score of
80

# Create a new column in df_salary with matched names from df_runs
df_salary['Matched_Player'] = df_salary['Player'].apply(lambda x:
match_names(x, df_runs['Striker'].tolist()))

# Merge the DataFrames on the matched names
df_merged = pd.merge(df_salary, df_runs, left_on='Matched_Player',
right_on='Striker')
df_merged.info()
```

## Relationship between a player's performance and the salary he gets

```
# Calculate the correlation
correlation = df_merged['Rs'].corr(df_merged['runs_scored'])

print("Correlation between Salary and Runs:", correlation)
```

## Recommendations

- **Data-Driven Decision Making**: Franchises should leverage detailed performance and salary data analytics to make strategic decisions during player auctions and team compositions.
- **Invest in Consistent Performers**: Focus on retaining and investing in players who consistently perform well, as identified by the top run-getters and wicket-takers analysis.
- **Budget Allocation**: Align player salaries with performance metrics to ensure fair and strategic budget allocation, avoiding overpayment or underpayment issues.
- **Advanced Analytics**: Utilize advanced statistical models and machine learning techniques to predict player performances and optimize team strategies.
- **Holistic Player Evaluation**: Consider not just historical performance data but also factors like player fitness, age, and potential for future performance when making salary decisions