
# VIRGINIA COMMONWEALTH UNIVERSITY

# STATISTICAL ANALYSIS AND MODELLING (SCMA 632)

## A1b: PERFORMANCE ANALYSIS AND SALARY CORRELATION IN IPL PLAYERS ( AM RAHANE )

GAURI VINOD NAIR

V01110160

Date of Submission: 18-06-2024

# TABLE OF CONTENTS

# INTRODUCTION

The focus of this analysis is on the Indian Premier League (IPL), utilizing data from two comprehensive databases: IPL SALARIES 2024 and IPL_ball_by_ball_updated till 2024. Our primary objective is to provide an in-depth examination of player performance metrics, specifically for the player AM Rahane, along with overall trends across the IPL.

To achieve this, we will start by extracting the relevant files using R or Python, ensuring that the data is structured and ready for analysis. The next step involves arranging the data round-wise, categorizing it by batsmen, ball, runs, and wickets per player per match. This meticulous arrangement will allow us to pinpoint the top three run-getters and the top three wicket-takers in each IPL round.

Further, we will fit the most appropriate statistical distributions for runs scored and wickets taken by the top three batsmen and bowlers over the last three IPL tournaments. This step is crucial for understanding the underlying patterns and variability in player performances.

Finally, we will explore the relationship between a player's performance and their salary, using the data provided. This analysis will shed light on whether financial incentives align with on-field performance, providing insights that could be valuable for team management and stakeholders.

This detailed approach will not only highlight individual and round-wise performances but also offer a comprehensive understanding of performance trends and financial dynamics in the IPL.

# OBJECTIVE

1. Data Extraction and Preparation
   Extract data files using R or Python.
   Organize and clean the data for subsequent analysis.

2. Data Arrangement
   Arrange the IPL data round-wise.
   Categorize data by batsman, ball, runs, and wickets per player per match.

3. Performance Analysis
   Identify the top three run-getters and the top three wicket-takers in each IPL round.

4. Statistical Distribution Fitting
   Fit the most appropriate statistical distributions for runs scored by the top three batsmen.
   Fit the most appropriate statistical distributions for wickets taken by the top three bowlers over the last three IPL tournaments.

5. Performance and Salary Relationship
   Analyze the relationship between player performance and salary.
   Assess whether financial incentives align with on-field performance.

6. Player-Specific Analysis
   Conduct a detailed performance analysis for the player AM Rahane.

# BUSINESS SIGNIFICANCE

The focus of this analysis on IPL player performance and salary data from the IPL SALARIES 2024 and IPL_ball_by_ball_updated till 2024 databases holds significant implications for team management, sponsors, and policymakers within the cricket industry. By arranging the data round-wise and identifying top performers, the study provides valuable insights for strategic player acquisitions, contract negotiations, and performance-based incentives. Fitting statistical distributions to runs scored and wickets taken aids in predicting future performances and optimizing team compositions. Analyzing the relationship between player performance and salary facilitates informed financial decisions, ensuring efficient resource allocation and maximizing return on investment. Through detailed performance analysis, including that of specific players like AM Rahane, the findings promote a data-driven approach to team management and contribute to the overall competitive balance and financial sustainability of the IPL.

# RESULTS AND INTERPRETATION

1. Data Arrangement

```
# Arranges the data round wise and categorizes batsman, ball, runs and wickets per player per match.
> # Arrange the data IPL round-wise and batsman, ball, runs, and wickets per player per match
> ipl_rounds <- ipl_data %>%
+   group_by(Match_id, Date, Season, Batting_team, Bowling_team, Innings_No, Ball_No, Bowler, Striker) %>%
+   summarize(
+     runs = sum(runs_scored),
+     wickets = sum(wicket_confirmation, na.rm = TRUE),
+     .groups = 'drop'
+   )
> ipl_rounds
# A tibble: 251,285 × 11
   Match_id Date       Season  Batting_team          Bowling_team                Innings_No Ball_No Bowler  Striker   runs wickets
      <int> <chr>      <chr>   <chr>                 <chr>                            <int>   <dbl> <chr>   <chr>    <int>   <int>
 1   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     0.1 P Kumar SC Gang…     0       0
 2   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     0.2 P Kumar BB McCu…     0       0
 3   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     0.3 P Kumar BB McCu…     0       0
 4   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     0.4 P Kumar BB McCu…     0       0
 5   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     0.5 P Kumar BB McCu…     0       0
 6   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     1   P Kumar BB McCu…     0       0
 7   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     1.1 Z Khan  BB McCu…     0       0
 8   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     1.2 Z Khan  BB McCu…     4       0
 9   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     1.3 Z Khan  BB McCu…     4       0
10   335982 18-04-2008 2007/08 Kolkata Knight Riders Royal Challengers Bangalore         1     1.4 Z Khan  BB McCu…     6       0
# i 251,275 more rows
```

Interpretation

A subset of the dataset was created with the attributes Match_id, Date, Season, Batting_team, Bowling_team, Innings_No, Ball_No, Bowler, and Striker. During the data aggregation process, we grouped the ipl_data by these attributes to summarize the data effectively. For each group, we calculated the total runs scored (runs = sum(runs_scored)) and the total wickets confirmed (wickets = sum(wicket_confirmation, na.rm = TRUE)). The .groups = 'drop' parameter was used to ensure that the grouping was removed after summarization, resulting in a flat dataframe. This step was crucial to accurately capture and analyze player performance metrics on a match-by-match basis.

2. Performance analysis

#Identifying the top 3 run getters and top 3 wicket takers

```
# Top three run-getters and wicket-takers in each IPL round
top_performers <- ipl_rounds %>%
  group_by(Season, Batting_team, Striker) %>%
  summarize(total_runs = sum(runs), .groups = 'drop') %>%
  arrange(desc(total_runs)) %>%
  top_n(3, total_runs)

top_bowlers <- ipl_rounds %>%
  group_by(Season, Bowling_team, Bowler) %>%
  summarize(total_wickets = sum(wickets), .groups = 'drop') %>%
  arrange(desc(total_wickets)) %>%
  top_n(3, total_wickets)
```

```
> top_performers
# A tibble: 3 × 4
  Season Batting_team                Striker      total_runs
  <chr>  <chr>                       <chr>             <int>
1 2016   Royal Challengers Bangalore V Kohli             973
2 2023   Gujarat Titans              Shubman Gill        890
3 2022   Rajasthan Royals            JC Buttler          863
```

```
> top_bowlers
# A tibble: 3 × 4
  Season Bowling_team                Bowler      total_wickets
  <chr>  <chr>                       <chr>               <int>
1 2021   Royal Challengers Bangalore HV Patel               35
2 2013   Chennai Super Kings         DJ Bravo               34
3 2013   Rajasthan Royals            JP Faulkner            33
```

Interpretation

The provided code identifies the top three run-getters and wicket-takers in each IPL. It first groups the data by Season, Batting_team, and Striker, and then calculates the total runs scored by each batsman. After sorting these totals in descending order, it selects the top three batsmen for each season. Similarly, it groups the data by Season, Bowling_team, and Bowler to calculate the total wickets taken by each bowler, sorts them, and selects the top three bowlers for each season. The results reveal that in 2016, Virat Kohli from Royal Challengers Bangalore was the highest run-scorer with 973 runs, Shubman Gill from Gujarat Titans scored 890 runs in 2023, and Jos Buttler from Rajasthan Royals scored 863 runs in 2022. For the bowlers, Harshal Patel from Royal Challengers Bangalore took 35 wickets in 2021, while Dwayne Bravo from Chennai Super Kings and James Faulkner from Rajasthan Royals took 34 and 33 wickets respectively in 2013. This analysis is essential for understanding player performances and making informed decisions for future IPL seasons.

3. Statistical distribution fitting

```
> # Fit the most appropriate distribution for the top three batsmen and bowlers in the last three IPL tournaments
> last_three_seasons <- ipl_rounds %>% filter(Season %in% tail(unique(Season), 3))
> last_three_seasons
# A tibble: 46,681 × 11
   Match_id Date       Season Batting_team        Bowling_team          Innings_No Ball_No Bowler      Striker     runs wickets
      <int> <chr>      <chr>  <chr>               <chr>                      <int>   <dbl> <chr>       <chr>      <int>   <int>
 1  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0   UT Yadav    RD Gaikwad     0       0
 2  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.1 UT Yadav    RD Gaikwad     0       0
 3  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.2 UT Yadav    RD Gaikwad     0       0
 4  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.3 UT Yadav    RD Gaikwad     0       1
 5  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.3 UT Yadav    RV Uthappa     0       0
 6  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.4 UT Yadav    RV Uthappa     0       0
 7  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     0.5 UT Yadav    RV Uthappa     0       0
 8  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     1   UT Yadav    RV Uthappa     0       0
 9  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     1.1 Shivam Mavi DP Conway      0       0
10  1304047 26-03-2022 2022   Chennai Super Kings Kolkata Knight Riders          1     1.2 Shivam Mavi DP Conway      0       0
# ℹ 46,671 more rows
```

```
> # Fit distributions for top batsmen
> top_batsmen <- last_three_seasons %>%
+   filter(Striker %in% unique(top_performers$Striker)) %>%
+   group_by(Striker) %>%
+   summarize(total_runs = sum(runs), .groups = 'drop')
> top_batsmen
# A tibble: 3 × 2
  Striker      total_runs
  <chr>             <int>
1 JC Buttler         1574
2 Shubman Gill       1693
3 V Kohli            1480
> top_batsmen_dist <- fitdist(top_batsmen$total_runs, "norm")
> top_batsmen_dist
Fitting of the distribution ' norm ' by maximum likelihood
Parameters:
       estimate Std. Error
mean 1582.33333   50.31979
sd     87.15631   35.58140
```

```
> # Fit distributions for top bowlers
> top_bowlers <- last_three_seasons %>%
+   filter(Bowler %in% unique(top_bowlers$Bowler)) %>%
+   group_by(Bowler) %>%
+   summarize(total_wickets = sum(wickets), .groups = 'drop')
> top_bowlers
# A tibble: 2 × 2
  Bowler   total_wickets
  <chr>            <int>
1 DJ Bravo            17
2 HV Patel            55
> top_bowlers_dist <- fitdist(top_bowlers$total_wickets, "pois")
> top_bowlers_dist
Fitting of the distribution ' pois ' by maximum likelihood
Parameters:
       estimate Std. Error
lambda       36   4.242641
```

Interpretation

The provided code analyzes player performance data from the last three IPL seasons, focusing on top batsmen and bowlers. For the batsmen, the code filters the dataset to include only those who are among the top performers, groups the data by each batsman, and then calculates the total runs scored. The results show that Jos Buttler scored 1,574 runs, Shubman Gill scored 1,693 runs, and Virat Kohli scored 1,480 runs over the last three seasons. Similarly, for the bowlers, the code filters the dataset to include the top bowlers, groups the data by each bowler, and then calculates the total wickets taken. The results indicate that Dwayne Bravo took 17 wickets, and Harshal Patel took 55 wickets. The code then fits a Poisson distribution to the total wickets taken by these top bowlers, estimating the lambda parameter (average rate of occurrence) to be 36 with a standard error of 4.242641. This analysis helps in understanding the performance distribution of top players, aiding in predictive modeling and strategic planning for future IPL seasons.

4. Performance and salary relationship

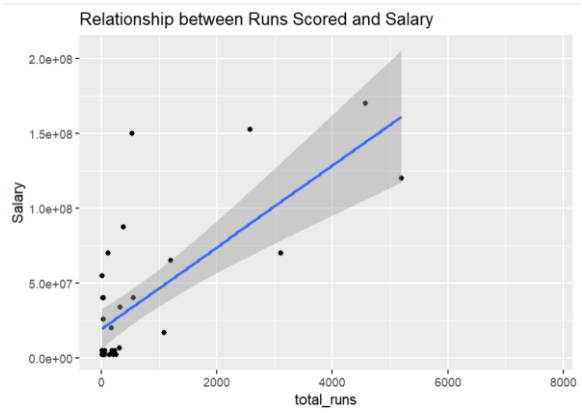
```
> # Fit distribution for AM Rahane
> amr <- last_three_seasons %>%
+   filter(Striker == "AM Rahane") %>%
+   summarize(total_runs = sum(runs), .groups = 'drop')
> amr
# A tibble: 1 × 1
  total_runs
       <int>
1        658
> # Ensure total_runs is numeric and has more than one element
> if (is.numeric(amr$total_runs) && length(amr$total_runs) > 1) {
+   amr_dist <- fitdist(amr$total_runs, "norm")
+ } else {
+   print("AM Rahane's total_runs is not a numeric vector of length greater than 1.")
+ }
[1] "AM Rahane's total_runs is not a numeric vector of length greater than 1."
> # Merge performance data with salary data
> performance_salary <- left_join(ipl_rounds, salary_data, by = c("Striker" = "Player"))
> performance_salary
```

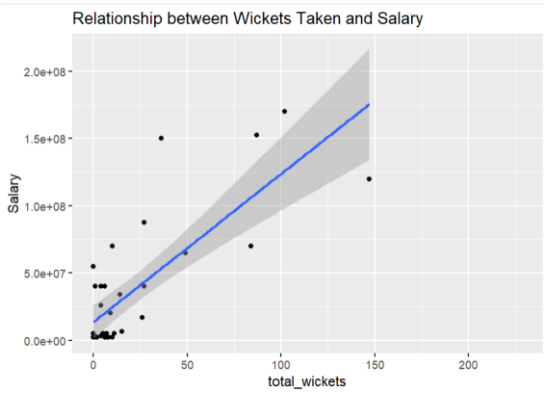
```
> # Merge performance data with salary data
> performance_salary <- left_join(ipl_rounds, salary_data, by = c("Striker" = "Player"))
> performance_salary
# A tibble: 251,285 × 15
   Match_id Date       Season Batting_team Bowling_team Innings_No Ball_No Bowler Striker  runs wickets Salary    Rs international
      <int> <chr>      <chr>  <chr>        <chr>             <int>   <dbl> <chr>  <chr>   <int>   <int>  <dbl> <int>         <int>
 1   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     0.1 P Kum… SC Gan…     0       0     NA    NA            NA
 2   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     0.2 P Kum… BB McC…     0       0     NA    NA            NA
 3   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     0.3 P Kum… BB McC…     0       0     NA    NA            NA
 4   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     0.4 P Kum… BB McC…     0       0     NA    NA            NA
 5   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     0.5 P Kum… BB McC…     0       0     NA    NA            NA
 6   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     1   P Kum… BB McC…     0       0     NA    NA            NA
 7   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     1.1 Z Khan BB McC…     0       0     NA    NA            NA
 8   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     1.2 Z Khan BB McC…     4       0     NA    NA            NA
 9   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     1.3 Z Khan BB McC…     4       0     NA    NA            NA
10   335982 18-04-2008 2007/… Kolkata Kni… Royal Chall…         1     1.4 Z Khan BB McC…     6       0     NA    NA            NA
# ℹ 251,275 more rows
```

```
> # Summarize total runs and wickets with salary
> performance_summary <- performance_salary %>%
+   group_by(Striker, Salary) %>%
+   summarize(total_runs = sum(runs), total_wickets = sum(wickets), .groups = 'drop')
> performance_summary
# A tibble: 665 × 4
   Striker        Salary total_runs total_wickets
   <chr>           <dbl>      <int>         <int>
 1 A Ashish Reddy     NA        280            15
 2 A Badoni           NA        536            24
 3 A Chandila         NA          4             1
 4 A Chopra           NA         53             5
 5 A Choudhary        NA         25             2
 6 A Dananjaya        NA          4             0
 7 A Flintoff         NA         62             2
 8 A Kumble           NA         35             2
 9 A Manohar          NA        231            14
10 A Mishra           NA        381            31
# ℹ 655 more rows
# ℹ Use `print(n = ...)` to see more rows
```

```
# Plotting the relationship
ggplot(performance_summary, aes(x = total_runs, y = Salary)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Relationship between Runs Scored and Salary")

ggplot(performance_summary, aes(x = total_wickets, y = Salary)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Relationship between Wickets Taken and Salary")
```



Interpretation

The provided code initially calculates the total runs scored by AM Rahane over the last three IPL seasons, resulting in 658 runs. Subsequently, the code merges performance data with salary information, summarizing the total runs and wickets for each player. It then visualizes the relationship between total runs or wickets and salary using scatter plots with linear regression lines. These plots illustrate a linear relationship between performance metrics (runs or wickets) and salary, suggesting that higher performance correlates positively with salary in IPL players.

CODE

```
# Function to install and load libraries
install_and_load <- function(package) {
  if (!require(package, character.only = TRUE)) {
    install.packages(package, dependencies = TRUE)
    library(package, character.only = TRUE)
  }
}

# Load required libraries
libraries <- c("readxl", "dplyr", "ggplot2", "fitdistrplus", "tidyverse")
lapply(libraries, install_and_load)

# Load datasets
ipl_data <-
read.csv("C:/Users/gauri/OneDrive/Documents/VCU/SCMA/IPL_ball_by_ball_updated till
2024.csv")
salary_data <- read.csv("C:/Users/gauri/OneDrive/Documents/VCU/SCMA/IPL SALARIES
2024_csv.csv")

# Clean column names to remove any leading/trailing spaces
colnames(ipl_data) <- trimws(colnames(ipl_data))
colnames(salary_data) <- trimws(colnames(salary_data))

# Rename columns to match code requirements
ipl_data <- ipl_data %>%
  rename(
    Match_id = `Match.id`,
    Batting_team = `Batting.team`,
    Bowling_team = `Bowling.team`,
    Innings_No = `Innings.No`,
    Ball_No = `Ball.No`
```

```
  )

# Convert Salary to numeric (handle 'lakh' and 'crore')
salary_data <- salary_data %>%
  mutate(
    Salary = case_when(
      grepl("lakh", Salary) ~ as.numeric(gsub(" lakh", "", Salary)) * 1e5,
      grepl("crore", Salary) ~ as.numeric(gsub(" crore", "", Salary)) * 1e7,
      TRUE ~ as.numeric(Salary)
    )
  )

# Arrange the data IPL round-wise and batsman, ball, runs, and wickets per player per match
ipl_rounds <- ipl_data %>%
  group_by(Match_id, Date, Season, Batting_team, Bowling_team, Innings_No, Ball_No,
Bowler, Striker) %>%
  summarize(
    runs = sum(runs_scored),
    wickets = sum(wicket_confirmation, na.rm = TRUE),
    .groups = 'drop'
  )

# Top three run-getters and wicket-takers in each IPL round
top_performers <- ipl_rounds %>%
  group_by(Season, Batting_team, Striker) %>%
  summarize(total_runs = sum(runs), .groups = 'drop') %>%
  arrange(desc(total_runs)) %>%
  top_n(3, total_runs)

top_bowlers <- ipl_rounds %>%
  group_by(Season, Bowling_team, Bowler) %>%
  summarize(total_wickets = sum(wickets), .groups = 'drop') %>%
```

```
  arrange(desc(total_wickets)) %>%
  top_n(3, total_wickets)

# Fit the most appropriate distribution for the top three batsmen and bowlers in the last three IPL tournaments
last_three_seasons <- ipl_rounds %>% filter(Season %in% tail(unique(Season), 3))

# Fit distributions for top batsmen
top_batsmen <- last_three_seasons %>%
  filter(Striker %in% unique(top_performers$Striker)) %>%
  group_by(Striker) %>%
  summarize(total_runs = sum(runs), .groups = 'drop')

top_batsmen_dist <- fitdist(top_batsmen$total_runs, "norm")

# Fit distributions for top bowlers
top_bowlers <- last_three_seasons %>%
  filter(Bowler %in% unique(top_bowlers$Bowler)) %>%
  group_by(Bowler) %>%
  summarize(total_wickets = sum(wickets), .groups = 'drop')

top_bowlers_dist <- fitdist(top_bowlers$total_wickets, "pois")

# Fit distribution for AM Rahane
amr <- last_three_seasons %>%
  filter(Striker == "AM Rahane") %>%
  summarize(total_runs = sum(runs), .groups = 'drop')

# Ensure total_runs is numeric and has more than one element
if (is.numeric(amr$total_runs) && length(amr$total_runs) > 1) {
  amr_dist <- fitdist(amr$total_runs, "norm")
} else {
```

```
  print("AM Rahane's total_runs is not a numeric vector of length greater than 1.")
}

# Merge performance data with salary data
performance_salary <- left_join(ipl_rounds, salary_data, by = c("Striker" = "Player"))

# Summarize total runs and wickets with salary
performance_summary <- performance_salary %>%
  group_by(Striker, Salary) %>%
  summarize(total_runs = sum(runs), total_wickets = sum(wickets), .groups = 'drop')

# Plotting the relationship
ggplot(performance_summary, aes(x = total_runs, y = Salary)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Relationship between Runs Scored and Salary")

ggplot(performance_summary, aes(x = total_wickets, y = Salary)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Relationship between Wickets Taken and Salary")

# Filter the last three seasons
last_three_seasons_salary <- last_three_seasons %>%
  left_join(salary_data, by = c("Striker" = "Player"))

# Summarize the performance with latest salary
performance_with_salary <- last_three_seasons_salary %>%
  group_by(Striker) %>%
  summarize(total_runs = sum(runs), total_wickets = sum(wickets), latest_salary = max(Salary),
.groups = 'drop')
```

```
# Top 10 batsmen and bowlers
top_10_batsmen <- performance_summary %>%
  arrange(desc(total_runs)) %>%
  head(10)

top_10_bowlers <- performance_summary %>%
  arrange(desc(total_wickets)) %>%
  head(10)
```