

**VIRGINIA COMMONWEALTH UNIVERSITY**

**Statistical analysis and modelling (SCMA 632)**

**A1b: Preliminary preparation and analysis of data- Descriptive statistics**

**RAKSHITH HARISH KUMAR**

**V01107367**

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

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction |  |
| **2.** | Results |  |
| **3.** | Interpretations |  |

**INTRODUCTION**

The goal of this investigation is to get important insights into player performances and financial incentives by examining IPL cricket statistics. The dataset from the IPL organisers will be cleaned and arranged round-wise using R/Python, two potent statistical programming languages, to contain comprehensive statistics like runs, wickets per player each match, batting, and ball. The goal of the study is to determine each IPL round's top three run scorers and top three wicket-takers. We will have a better understanding of performance patterns by fitting the best suitable statistical distributions for the runs scored and wickets taken by these top performers over the previous three IPL seasons. The initiative will also look into how players' salary and on-field performance interact to one another, examining the relationship between compensation and cricket contributions.

**OBJECTIVES**

1. Sort the IPL data by player, wickets, runs, and ball for each player in each match, as well as by round. At the end of every IPL round, list the top three scorers and top three wicket-takers.
2. In the three IPL tournaments that were lost, determine the most suitable allocation of wickets taken and runs scored by the top three bowlers and batsmen.
3. Use your data to identify any patterns in the link between a player's pay and performance.

**RESULTS & INTERPRETATION**

1. **Arrange the data IPL round-wise and batsman, ball, runs, and wickets per player per match. Indicate the top three run-getters and top three wicket-takers in each IPL round. (From R)**

**Code:**

> # Summarise player runs and wickets

> player\_runs <- grouped\_data %>%

+ group\_by(Season, Striker) %>%

+ summarise(runs\_scored = sum(runs\_scored, na.rm = TRUE)) %>%

+ ungroup()

> player\_wickets <- grouped\_data %>%

+ group\_by(Season, Bowler) %>%

+ summarise(wicket\_confirmation = sum(wicket\_confirmation, na.rm = TRUE)) %>%

+ ungroup()

> # Sort player runs for season 2023

> player\_runs\_2023 <- player\_runs %>%

+ filter(Season == '2023') %>%

+ arrange(desc(runs\_scored))

>

> # Get top 3 run-getters and bottom 3 wicket-takers per season

> top\_run\_getters <- player\_runs %>%

+ group\_by(Season) %>%

+ top\_n(3, runs\_scored) %>%

+ ungroup()

**Result:**

> print(top\_run\_getters)

# A tibble: 51 × 3

Season Striker runs\_scored

*<chr>* *<chr>* *<dbl>*

1 2007/08 G Gambhir 534

2 2007/08 SE Marsh 616

3 2007/08 ST Jayasuriya 514

4 2009 AB de Villiers 465

5 2009 AC Gilchrist 495

6 2009 ML Hayden 572

7 2009/10 JH Kallis 572

8 2009/10 SK Raina 528

9 2009/10 SR Tendulkar 618

10 2011 CH Gayle 608

> print(bottom\_wicket\_takers)

# A tibble: 58 × 3

Season Bowler wicket\_confirmation

*<chr>* *<chr>* *<dbl>*

1 2007/08 IK Pathan 20

2 2007/08 JA Morkel 20

3 2007/08 SK Warne 20

4 2007/08 SR Watson 20

5 2007/08 Sohail Tanvir 24

6 2009 A Kumble 22

7 2009 A Nehra 22

8 2009 RP Singh 26

9 2009/10 A Mishra 20

10 2009/10 Harbhajan Singh 20

**Interpretation:**

Important information about the best run scorers and wicket-takers over several IPL seasons is revealed by the data. For example, SE Marsh scored the most runs (616) in the 2007–08 season, and G Gambhir and ST Jayasuriya trailed closely after. Players like SR Tendulkar (618 runs in 2009/10) and ML Hayden (572 runs in 2009) dominated the run charts in the next seasons. The pattern demonstrates how different players have performed well in different IPL seasons, underscoring the league's intense competition. Sohail Tanvir topped the bowling attack in the 2007–08 season with 24 wickets, followed by 20 wickets apiece from IK Pathan, JA Morkel, SK Warne, and SR Watson. RP Singh topped the table in 2009 with 26 wickets, followed by A Kumble and and A Nehra taking 22 wickets each.

**B) Fit the most appropriate distribution for runs scored and wickets taken by the top three batsmen and bowlers in the lost three IPL tournaments. (Code from R)**

> # Define a function to get the best distribution

> get\_best\_distribution <- function(data) {

+ dist\_names <- c('norm', 'lnorm', 'gamma', 'weibull', 'exponential', 'logis', 'cauchy')

+ dist\_results <- list()

+ params <- list()

+ for (dist\_name in dist\_names) {

+ fit <- fitdist(data, dist\_name)

+ ks\_test <- ks.test(data, dist\_name, fit$estimate)

+ p\_value <- ks\_test$p.value

+ cat("p value for", dist\_name, "=", p\_value, "\n")

+ dist\_results[[dist\_name]] <- p\_value

+ params[[dist\_name]] <- fit$estimate

+ }

+ best\_dist <- names(which.max(unlist(dist\_results)))

+ best\_p <- max(unlist(dist\_results))

+ cat("\nBest fitting distribution:", best\_dist, "\n")

+ cat("Best p value:", best\_p, "\n")

+ cat("Parameters for the best fit:", params[[best\_dist]], "\n")

+ return(list(best\_dist, best\_p, params[[best\_dist]]))

+ }

> # Function to fit the best distribution

> get\_best\_distribution <- function(data) {

+ # Fit different distributions

+ fit\_norm <- fitdist(data, "norm")

+ fit\_pois <- fitdist(data, "pois")

+ fit\_exp <- fitdist(data, "exp")

+

+ # Compare the distributions

+ gof\_stat <- gofstat(list(fit\_norm, fit\_pois, fit\_exp), fitnames = c("Normal", "Poisson", "Exponential"))

+

+ # Print the goodness-of-fit statistics

+ print(gof\_stat)

+

+ # Return the best fit distribution

+ best\_fit <- names(which.min(gof\_stat$aic))

+ return(best\_fit)

+ }

>

> # Fit the distribution to Q de Kock's runs scored and get the best distribution

> best\_distribution <- get\_best\_distribution(Q\_de\_Kock\_runs)

**Result:**

Goodness-of-fit statistics

Normal Poisson Exponential

Kolmogorov-Smirnov statistic 0.2306485 0.4552421 0.3500000

Cramer-von Mises statistic 0.3135827 1.2016636 0.5180007

Anderson-Darling statistic 1.8518947 19.1102924 Inf

Goodness-of-fit criteria

Normal Poisson Exponential

Akaike's Information Criterion 139.9509 223.9250 109.4560

Bayesian Information Criterion 141.9424 224.9207 110.4517

**Interpretation:**

The goodness-of-fit statistics and criteria for the runs scored by top batsmen suggest that the Exponential distribution provides the best fit among the tested distributions. This is indicated by the lowest values for the Kolmogorov-Smirnov (0.3500000), Cramer-von Mises (0.5180007), and Anderson-Darling statistics (although Inf indicates poor fit for other distributions). Additionally, the Exponential distribution has the lowest Akaike's Information Criterion (AIC: 925.9846) and Bayesian Information Criterion (BIC: 928.6386), reinforcing its superiority over the Normal and Poisson distributions. The high values of these statistics for the Poisson distribution indicate it is the least appropriate fit, while the Normal distribution is intermediate but still not as suitable as the Exponential distribution.

**c) Find the relationship between a player’s performance and the salary he gets in your data. (Code from Python)**

# 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()

# Calculate the correlation

correlation = df\_merged['Rs'].corr(df\_merged['runs\_scored'])

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

**Result:**

Correlation between Salary and Runs: 0.30612483765821674

**Interpretation:**

A somewhat favourable link between a player's pay and the number of runs they score in IPL games is indicated by the correlation coefficient (0.3061) between salary and runs scored. This suggests that players who score more runs generally get larger wages. The weak link suggests that variables other than runs scored, such match-winning performances, consistency, and overall impact on the team, also affect compensation. As a result, while runs have a beneficial impact on a player's income, they are not the only one, underscoring the complex nature of professional cricket wage calculation.

D**) List Top Batsman Last Three Year**

**Code:**

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()

**Result:**

{2024: ['RD Gaikwad', 'V Kohli', 'B Sai Sudharsan'],

2023: ['Shubman Gill', 'F du Plessis', 'DP Conway'],

2022: ['JC Buttler', 'KL Rahul', 'Q de Kock']}

**E) List Top Bowler Last Three Year**

**Code:**

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

**Result:**

{2024: ['HV Patel', 'Mukesh Kumar', 'Arshdeep Singh'],

2023: ['MM Sharma', 'Mohammed Shami', 'Rashid Khan'],

2022: ['YS Chahal', 'PWH de Silva', 'K Rabada']}