VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A1b: Preliminary preparation and analysis of data- Descriptive statistics

YARAMALLA AKANKSHA

V01108249

Date of Submission: 18-06-2024

CONTENTS

INTRODUCTION

This analysis focuses on analyzing IPL cricket data to extract valuable insights into player performances and their financial rewards. Using R/Python, powerful statistical programming languages, the dataset from IPL organizers will be cleaned and organized round-wise to include detailed statistics such as batsman, ball, runs, and wickets per player per match. The analysis aims to identify the top three run-getters and top three wicket-takers in each IPL round. By fitting the most appropriate statistical distributions for the runs scored and wickets taken by these top performers over the last three IPL tournaments, we will gain a deeper understanding of performance patterns. Additionally, the project will investigate the relationship between players' on-field performance and their salaries, exploring how remuneration correlates with cricket contributions.

OBJECTIVES

a) Arrange the data IPL round-wise and batsman, ball, runs, and wickets per player per match. Indicate the top three run-getters and tow three wicket-takers in each IPL round.
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.
c) Find the relationship between a player's performance and the salary he gets in your data.

RESULTS & INTERPRETATION

a) 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:

The data reveals key insights into the top run-getters and wicket-takers in various IPL seasons. For instance, in the 2007/08 season, SE Marsh emerged as the highest run-scorer with 616 runs, followed closely by G Gambhir and ST Jayasuriya. In subsequent seasons, players like ML Hayden (572 runs in 2009) and SR Tendulkar (618 runs in 2009/10) dominated the run charts. The trend shows that different players have excelled in various seasons, highlighting the competitive nature of the IPL.

On the bowling side, Sohail Tanvir led the 2007/08 season with 24 wickets, while other notable bowlers like IK Pathan, JA Morkel, SK Warne, and SR Watson each took 20 wickets. In the 2009 season, RP Singh topped the chart with 26 wickets, with A Kumble and A Nehra taking 22 wickets each.

These statistics underscore the dynamic and unpredictable nature of the IPL, where both seasoned players and emerging talents consistently vie for the top spots in both batting and bowling categories.

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', 'l
ogis', '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("N
ormal", "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 distr
ibution
> best_distribution <- get_best_distribution(Q_de_Kock_runs)
```

Result:

Normal Poisson Exponential

Kolmogorov-Smirnov statistic 0.1694281 0.4391087 0.1129865

Cramer-von Mises statistic 0.6066782 4.7197459 0.1230481

Anderson-Darling statistic 3.6671718 Inf Inf

Goodness-of-fit criteria

Normal Poisson Exponential

Akaike's Information Criterion 764.1790 2091.645 684.3129

Bayesian Information Criterion 769.0406 2094.076 686.7437Interpretation:

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 (0.0805889), Cramer-von Mises (0.1594708), 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:

The correlation coefficient between Salary and Runs scored (0.3061) suggests a moderately positive relationship between a player's salary and their runs scored in IPL matches. This indicates that, on average, players who score more runs tend to receive higher salaries. However, the correlation is not very strong, implying that other factors besides runs scored also influence salary, such as match-winning performances, consistency, and overall impact on the team. Therefore, while runs contribute positively to a player's earnings, they are not the sole determinant, highlighting the multi-dimensional nature of salary determination in professional cricket.