VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A2b: Multiple regression analysis and diagnostics of data

PUSHPAK DEVAKI

V01108254

Date of Submission: 23-06-2024

CONTENTS

INTRODUCTION

This analysis focuses on analysing 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) To establish the relationship between the player’s performance and payment.
b) Analyse the relationship between salary and performance over the last three years.

RESULTS & INTERPRETATION

a) Using IPL data, establish the relationship between the player's performance and payment he receives and discuss your findings. Analyze the Relationship Between Salary and Performance Over the Last Three Years (Regression Analysis)

Code:

```
# Group Data

grouped_data <- df_ipl %>%

  group_by(Season, `Innings.No`, Striker, Bowler) %>%

  summarise(runs_scored = sum(runs_scored, na.rm = TRUE),

            wicket_confirmation = sum(wicket_confirmation, na.rm = TRUE))

# Subset Data for a Specific Season

df_merged <- df_merged %>% filter(Season %in% c('2022'))

# Linear Regression on Wickets with stats

X <- df_merged %>% select(wicket_confirmation)

y <- df_merged$Rs
```

```
set.seed(42)

train_index <- createDataPartition(y, p = 0.8, list = FALSE)

X_train <- X[train_index, ]

y_train <- y[train_index]

X_test <- X[-train_index, ]

y_test <- y[-train_index]

X_train_sm <- cbind(1, as.matrix(X_train))

model_sm <- lm(y_train ~ X_train_sm - 1)

# Print summary of the linear regression model

summary(model_sm)
```

Result:

```
# Print summary of the linear regression model
> summary(model_sm)

Call:
lm(formula = y_train ~ X_train_sm - 1)

Residuals:
    Min      1Q  Median      3Q     Max 
-514.11 -217.37  -49.73  139.58  881.50 

Coefficients:
                              Estimate Std. Error t value Pr(>|t|)  
X_train_sm                     196.581     97.106   2.024   0.0522 .
X_train_smwicket_confirmation   21.096      8.659   2.436   0.0212 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 357.2 on 29 degrees of freedom
Multiple R-squared:  0.5795,	Adjusted R-squared:  0.5505 
F-statistic: 19.98 on 2 and 29 DF,  p-value: 3.507e-06
```

Python Code:

```
import pandas as pd

from sklearn.linear_model import LinearRegression

from sklearn.metrics import r2_score, mean_absolute_percentage_error

X = df_merged[['runs_scored']] # Independent variable(s)

y = df_merged['Rs'] # Dependent variable

# Split the data into training and test sets (80% for training, 20% for testing)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Create a LinearRegression model
model = LinearRegression()
# Fit the model on the training data
model.fit(X_train, y_train)
```

Result:

```
OLS Regression Results
==============================================================================
Dep. Variable:                     Rs   R-squared:                       0.080
Model:                            OLS   Adj. R-squared:                  0.075
Method:                 Least Squares   F-statistic:                     15.83
Date:                Sun, 23 Jun 2024   Prob (F-statistic):           0.000100
Time:                        17:10:02   Log-Likelihood:                -1379.8
No. Observations:                 183   AIC:                             2764.
Df Residuals:                     181   BIC:                             2770.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
               coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       430.8473     46.111      9.344      0.000     339.864     521.831
runs_scored   0.6895      0.173      3.979      0.000       0.348       1.031
==============================================================================
Omnibus:                       15.690   Durbin-Watson:                   2.100
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               18.057
Skew:                           0.764   Prob(JB):                     0.000120
Kurtosis:                       2.823   Cond. No.                         363.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Interpretation:

From the P values above that are 0.0552 and 0.0212 we can understand that both the factors ae significant. The coefficients indicate that for each unit increase in "X_train_sm," the dependent variable "y_train" increases by approximately 196.581 units. The residual standard error of 357.2 indicates the typical distance between the observed values and the model's predictions. The overall model is significant (p-value = 3.507e-06).