Project on   
Programming of Data Science



**Declaration**

By including this statement, we the authors of this work, verify that:

* We hold a copy of this assignment that we can produce if the original is lost or damaged.
* We hereby certify that no part of this assignment/product has been copied from any other student’s work or from any other source except where due acknowledgement is made in the assignment.
* No part of this assignment/product has been written/produced for us by another person except where such collaboration has been authorised by the subject lecturer/tutor concerned.
* We are aware that this work may be reproduced and submitted to plagiarism detection software programs for the purpose of detecting possible plagiarism (which may retain a copy on its database for future plagiarism checking).
* We hereby certify that we have read and understand what the School of Computing, Engineering and Mathematics defines as minor and substantial breaches of misconduct as outlined in the learning guide for this unit.

Garima Mahajan

19514664

Programming of Data Science

4.1 Validating the Strike Rate

# # Creating the data frame (cric.df)

# # The data frame command is used for the formulation of the required dataframe.

# # The dataframe (cric.df ) takes seven variables namely: Batsman, Hows Out, Runs, Balls, 4s, 6s, SR.

cric.df <- data.frame(  
 Batsman = c("KL Rahul", "M Vijay", "CA Pujara", "V Kohli (c)",   
 "AM Rahane", "RG Sharma", "RR Pant +", "R Ashwin",   
 "I Sharma", "Mohammed Shami", "JJ Bumrah"),  
 `How Out` = c("c Finch b Hazlewood", "c +Paine b Starc", "run out (Cummins)",  
 "c Khawaja b Cummins", "c Handscomb b Hazlewood", "c Harris b Lyon",  
 "c +Paine b Lyon", "c Handscomb b Cummins", "b Starc",  
 "C +Paine b Hazlewood", "not out"),  
 Runs = c(2, 11, 123, 3, 13, 37, 25, 25, 4, 6, 0),  
 Balls = c(8, 22, 246, 16, 31, 61, 38, 76, 20, 10, 0),  
 `4s` = c(0, 1, 7, 0, 0, 2, 2, 1, 1, 1, 0),  
 `6s` = c(0, 0, 2, 0, 1, 3, 1, 0, 0, 0, 0),  
 `SR` = c(25.00, 50.00, 50.00, 18.75, 41.93, 60.65, 65.78, 32.89, 20.00, 60.00, NA)  
)

print(cric.df)

## Batsman How.Out Runs Balls X4s X6s SR  
## 1 KL Rahul c Finch b Hazlewood 2 8 0 0 25.00  
## 2 M Vijay c +Paine b Starc 11 22 1 0 50.00  
## 3 CA Pujara run out (Cummins) 123 246 7 2 50.00  
## 4 V Kohli (c) c Khawaja b Cummins 3 16 0 0 18.75  
## 5 AM Rahane c Handscomb b Hazlewood 13 31 0 1 41.93  
## 6 RG Sharma c Harris b Lyon 37 61 2 3 60.65  
## 7 RR Pant + c +Paine b Lyon 25 38 2 1 65.78  
## 8 R Ashwin c Handscomb b Cummins 25 76 1 0 32.89  
## 9 I Sharma b Starc 4 20 1 0 20.00  
## 10 Mohammed Shami C +Paine b Hazlewood 6 10 1 0 60.00  
## 11 JJ Bumrah not out 0 0 0 0 NA

# # the formula for calculating the strike rate is the ratio of runs and balls multiplied by hunderd.

cric.df$SR\_New <- (cric.df$Runs/cric.df$Balls)\*100

# # calculating the error

cric.df$error <- cric.df$SR\_New - cric.df$SR

# # rounding the error

cric.df$error <- round(cric.df$error, digits = 2)  
print(cric.df)

## Batsman How.Out Runs Balls X4s X6s SR  
## 1 KL Rahul c Finch b Hazlewood 2 8 0 0 25.00  
## 2 M Vijay c +Paine b Starc 11 22 1 0 50.00  
## 3 CA Pujara run out (Cummins) 123 246 7 2 50.00  
## 4 V Kohli (c) c Khawaja b Cummins 3 16 0 0 18.75  
## 5 AM Rahane c Handscomb b Hazlewood 13 31 0 1 41.93  
## 6 RG Sharma c Harris b Lyon 37 61 2 3 60.65  
## 7 RR Pant + c +Paine b Lyon 25 38 2 1 65.78  
## 8 R Ashwin c Handscomb b Cummins 25 76 1 0 32.89  
## 9 I Sharma b Starc 4 20 1 0 20.00  
## 10 Mohammed Shami C +Paine b Hazlewood 6 10 1 0 60.00  
## 11 JJ Bumrah not out 0 0 0 0 NA  
## SR\_New error  
## 1 25.00000 0.00  
## 2 50.00000 0.00  
## 3 50.00000 0.00  
## 4 18.75000 0.00  
## 5 41.93548 0.01  
## 6 60.65574 0.01  
## 7 65.78947 0.01  
## 8 32.89474 0.00  
## 9 20.00000 0.00  
## 10 60.00000 0.00  
## 11 NaN NaN

# # It is seen that the error is observed in row 5, 6 and 7.

4.2 Ball Outcome

# # Declaring the state variable as list

state = list (balls = 0, runs = 0, fours = 0, sixes = 0, extras = 0, out = FALSE)

# # formulating oneBall function with multiple if-else statements.

# # The variable state and outcome has been used as arguments to function: oneBall

oneBall <- function(state, outcome){  
 if (outcome == "0 runs"){  
 state$balls = state$balls + 1  
 } else if (outcome == "1 run"){  
 state$balls = state$balls + 1  
 state$runs = state$runs + 1  
 } else if (outcome == "2 runs"){  
 state$balls = state$balls + 1  
 state$runs = state$runs + 2  
 } else if (outcome == "4 runs"){  
 state$balls = state$balls + 1  
 state$runs = state$runs + 4  
 state$fours = state$fours + 1  
 } else if (outcome == "6 runs"){  
 state$balls = state$balls + 1  
 state$runs = state$runs + 6  
 state$sixes = state$sixes + 1  
 } else if (outcome == "Wide"){  
 state$extras = state$extras + 1  
 } else if (outcome == "No ball"){  
 state$extras = state$extras + 1  
 } else if (outcome == "Out"){   
 state$out=TRUE  
 state$balls = state$balls + 1  
 }  
 print(state)  
}  
  
oneBall(state, "6 runs")

## $balls  
## [1] 1  
##   
## $runs  
## [1] 6  
##   
## $fours  
## [1] 0  
##   
## $sixes  
## [1] 1  
##   
## $extras  
## [1] 0  
##   
## $out  
## [1] FALSE

4.3 Batsman’s score

# # Formulating oneBatsman function by using similar control statements but by taking only one argument i.e. state.

state = list(balls = 0, runs = 0, fours = 0, sixes = 0, extras = 0, out = FALSE)

oneBatsman <- function(state){  
 repeat{  
 outcome = sample(c("0 runs", "1 runs", "2 runs", "4 runs", "6 runs", "Wide", "No Ball", "Out"), size = 1)  
   
state = oneBall(state, outcome)  
 if (outcome == "Out") {  
 return(state) #the function returns the updated state  
 print(state) #the function also prints the updated states  
 break #the repeat loop is broken when outcome is out  
 }  
 }  
}

oneBatsman(state)

## $balls  
## [1] 2  
##   
## $runs  
## [1] 4  
##   
## $fours  
## [1] 1  
##   
## $sixes  
## [1] 0  
##   
## $extras  
## [1] 0  
##   
## $out  
## [1] TRUE

## $balls  
## [1] 2  
##   
## $runs  
## [1] 4  
##   
## $fours  
## [1] 1  
##   
## $sixes  
## [1] 0  
##   
## $extras  
## [1] 0  
##   
## $out  
## [1] TRUE

4.4 Team Score

# # creating a simulation for 10 batsman and storing the data into the dataframe d

d = NULL  
for (player in 1:10) {  
 state = list(balls = 0, runs = 0, fours = 0, sixes = 0, extras = 0, out = FALSE)  
 u\_state = oneBatsman(state)  
 u\_state = data.frame(u\_state)  
 u\_state$player = player  
 d = rbind(d, u\_state) #append the data to new database  
}

print(d)

## balls runs fours sixes extras out player  
## 1 2 4 1 0 0 TRUE 1  
## 2 1 0 0 0 0 TRUE 2  
## 3 1 0 0 0 0 TRUE 3  
## 4 1 0 0 0 0 TRUE 4  
## 5 4 18 0 3 1 TRUE 5  
## 6 3 6 0 1 0 TRUE 6  
## 7 25 70 4 6 4 TRUE 7  
## 8 6 16 2 1 0 TRUE 8  
## 9 1 0 0 0 1 TRUE 9  
## 10 3 4 1 0 1 TRUE 10

# # getting total runs by adding runs and extras

`total runs` = sum(d$runs)+sum(d$extras)  
  
print(`total runs`)

## [1] 125